

ESP32-C5

Technical Reference Manual Pre-release v0.1

PRELIMINARY



ESPRESSIF

About This Document

The **ESP32-C5 Technical Reference Manual** is targeted at developers working on low level software projects that use the ESP32-C5 SoC. It describes the hardware modules listed below for the ESP32-C5 SoC and other products in ESP32-C5 series. The modules detailed in this document provide an overview, list of features, hardware architecture details, any necessary programming procedures, as well as register descriptions.

Navigation in This Document

Here are some tips on navigation through this extensive document:

- [Release Status at a Glance](#) on the very next page is a minimal list of all chapters from where you can directly jump to a specific chapter.
- Use the **Bookmarks** on the side bar to jump to any specific chapters or sections from anywhere in the document. Note this PDF document is configured to automatically display **Bookmarks** when open, which is necessary for an extensive document like this one. However, some PDF viewers or browsers ignore this setting, so if you don't see the **Bookmarks** by default, try one or more of the following methods:
 - Install a PDF Reader Extension for your browser;
 - Download this document, and view it with your local PDF viewer;
 - Set your PDF viewer to always automatically display the **Bookmarks** on the left side bar when open.
- Use the native **Navigation** function of your PDF viewer to navigate through the documents. Most PDF viewers support to go **Up**, **Down**, **Previous**, **Next**, **Back**, **Forward** and **Page** with buttons, menu, or hot keys.
- You can also use the built-in **GoBack** button on the upper right corner on each and every page to go back to the previous place before you click a link within the document. Note this feature may only work with some Acrobat-specific PDF viewers (for example, Acrobat Reader and Adobe DC) and browsers with built-in Acrobat-specific PDF viewers or extensions (for example, Firefox).

Release Status at a Glance

Note that this manual is still work in progress. See our release progress below:

No.	Chapter	Progress
Part I. Microprocessor and Master		
1	High-Performance CPU [to be added later]	0%
2	RISC-V Trace Encoder (TRACE)	Published
3	Low-Power CPU	Published
4	GDMA Controller (GDMA)	Published
Part II. Memory Organization		
5	System and Memory	Published
6	eFuse Controller (eFuse) [to be added later]	80%
Part III. System Component		
7	GPIO Matrix and IO MUX	Published
8	Reset and Clock	Published
9	Chip Boot Control	Published
10	Interrupt Matrix [to be added later]	80%
11	Event Task Matrix (ETM)	Published
12	Low-power Management [to be added later]	9%
13	System Timer	Published
14	Timer Group (TIMG)	Published
15	Watchdog Timers (WDT)	Published
16	RTC Timer	Published
17	Permission Control (PMS)	Published
18	System Registers [to be added later]	9%
19	Debug Assistant	Published
20	Power Supply Detector	Published
Part IV. Cryptography/Security Component		
21	AES Accelerator (AES)	Published
22	ECC Accelerator (ECC)	Published
23	HMAC Accelerator (HMAC)	Published
24	RSA Accelerator (RSA)	Published
25	SHA Accelerator (SHA)	Published
26	Digital Signature Algorithm (DSA)	Published
27	Elliptic Curve Digital Signature Algorithm (ECDSA)	Published
28	External Memory Encryption and Decryption (XTS_AES)	Published
29	Random Number Generator (RNG)	Published
Part V. Connectivity Interface		
30	UART Controller (UART)	Published
31	SPI Controller [to be added later]	73%
32	I2C Controller (I2C)	Published
33	I2S Controller (I2S)	Published
34	Pulse Count Controller (PCNT) [to be added later]	80%
35	USB Serial/JTAG Controller	Published

No.	Chapter	Progress
36	Controller Area Network Flexible Data-Rate [to be added later]	39%
37	SDIO Slave Controller (SDIO)	Published
38	LED PWM Controller (LEDC)	Published
39	Motor Control PWM (MCPWM)	Published
40	Remote Control Peripheral (RMT)	Published
41	Parallel IO Controller (PARLIO)	Published
42	BitScrambler	Published
Part VI. Analog Signal Processing		
43	Temperature Sensor	Published
44	ADC Controller	Published
45	Analog Voltage Comparator	Published

Note:

Check the link or the QR code to make sure that you use the latest version of this document:
https://www.espressif.com/documentation/esp32-c5_technical_reference_manual_en.pdf



Contents

I	Microprocessor and Master	33
1	RISC-V Trace Encoder (TRACE)	34
1.1	Terminology	34
1.2	Introduction	35
1.3	Features	36
1.4	Architectural Overview	37
1.5	Functional Description	37
1.5.1	Synchronization	37
1.5.2	Address Mode	38
1.5.3	Optional Sideband Signals	38
1.5.4	Filtering	39
1.5.5	Anchor Tag	39
1.5.6	Memory Writing Mode	40
1.5.7	Automatic Restart	40
1.6	Encoder Output Packets	40
1.6.1	Header	40
1.6.2	Index	41
1.6.3	Payload	41
1.6.3.1	Format 3 Packets	41
1.6.3.2	Format 2 Packets	43
1.6.3.3	Format 1 Packets	44
1.7	Interrupt	45
1.8	Programming Procedures	46
1.8.1	Encoder Option Configuration	46
1.8.2	Filter Configuration	47
1.8.3	Enable Encoder	48
1.8.4	Disable Encoder	49
1.8.5	Notify	49
1.8.6	Decode Data Packets	50
1.8.7	AHB Configuration	50
1.8.8	Software Retention	51
1.9	Register Summary	52
1.10	Registers	53
2	Low-Power CPU	64
2.1	Features	64
2.2	Configuration and Status Registers (CSRs)	65
2.2.1	Register Summary	65
2.2.2	Registers	66
2.3	Interrupts and Exceptions	73
2.3.1	Interrupts	74

2.3.2	Interrupt Handling	74
2.3.3	Exceptions	74
2.4	Debugging	75
2.4.1	Features	75
2.4.2	Functional Description	75
2.4.3	Register Summary	75
2.4.4	Registers	76
2.5	Hardware Trigger	78
2.5.1	Features	78
2.5.2	Functional Description	78
2.5.3	Trigger Execution Flow	79
2.5.4	Register Summary	79
2.5.5	Registers	79
2.6	Performance Counter	82
2.7	System Access	83
2.7.1	Memory Access	83
2.7.2	Peripheral Access	83
2.8	Event Task Matrix Feature	83
2.9	Sleep and Wake-Up Process	84
2.9.1	Features	84
2.9.2	Process	84
2.9.3	Wake-Up Sources	86
2.10	Register Summary	86
2.11	Registers	86
3	GDMA Controller (GDMA)	88
3.1	Overview	88
3.2	Features	88
3.3	Architecture	89
3.4	Functional Description	90
3.4.1	Linked List	90
3.4.2	Peripheral-to-Memory and Memory-to-Peripheral Data Transfer	91
3.4.3	Memory-to-Memory Data Transfer	92
3.4.4	Enabling GDMA	92
3.4.5	Linked List Reading Process	93
3.4.6	EOF	93
3.4.7	Accessing Memory	94
3.4.8	Arbitration	94
3.5	Event Task Matrix Feature	95
3.6	Interrupts	96
3.7	Programming Procedures	97
3.7.1	Programming Procedures for GDMA's Transmit Channel	97
3.7.2	Programming Procedures for GDMA's Receive Channel	97
3.7.3	Programming Procedures for Memory-to-Memory Transfer	98
3.7.4	Programming Procedures for Channel Priority and Weight	98
3.8	Register Summary	99

3.9	Registers	106
II	Memory Organization	134
4	System and Memory	135
4.1	Overview	135
4.2	Features	135
4.3	Functional Description	136
4.3.1	Address Mapping	136
4.3.2	Internal Memory	138
4.3.2.1	ROM	138
4.3.2.2	HP SRAM	139
4.3.2.3	LP SRAM	139
4.3.3	External Memory	140
4.3.3.1	External Memory Address Mapping	140
4.3.3.2	Cache	140
4.3.3.3	Cache Operations	140
4.3.4	GDMA Address Space	142
4.3.5	Modules/Peripherals Address Mapping	143
III	System Component	146
5	GPIO Matrix and IO MUX	147
5.1	Overview	147
5.2	Features	147
5.2.1	HP GPIO Matrix and HP IO MUX	147
5.2.2	LP GPIO Matrix and LP IO MUX	148
5.3	Architectural Overview	148
5.4	Peripheral Input via GPIO Matrix	150
5.4.1	Overview	150
5.4.2	Signal Synchronization	151
5.4.3	GPIO Filter	151
5.4.4	Glitch Filter	152
5.4.5	Simple GPIO Input	152
5.4.6	GPIO Wakeup	152
5.4.6.1	HP GPIO Wakeup	152
5.4.6.2	LP GPIO Wakeup	153
5.4.7	Programming Procedure	153
5.4.7.1	HP GPIO Matrix	153
5.4.7.2	LP GPIO Matrix	155
5.5	Peripheral Output via GPIO Matrix	155
5.5.1	Overview	155
5.5.2	Simple GPIO Output	155
5.5.3	Sigma Delta Modulated Output (SDM)	156
5.5.3.1	Functional Description	156
5.5.3.2	SDM Configuration	156

5.5.4	Programming Procedure	157
5.5.4.1	HP GPIO Matrix	157
5.5.4.2	LP GPIO Matrix	158
5.6	Direct Input and Output via IO MUX	158
5.6.1	Overview	158
5.6.2	Functional Description	158
5.6.2.1	HP IO MUX	158
5.6.2.2	LP IO MUX	158
5.7	Analog Functions	159
5.7.1	Overview	159
5.7.2	Analog Functions	159
5.8	Pin Functions in Light-sleep	159
5.9	Pin Hold Feature	160
5.10	Hysteresis Characteristics	161
5.11	Power Supplies and Management of GPIO Pins	162
5.11.1	Power Supplies of GPIO Pins	162
5.11.2	Power Supply Management	162
5.12	HP Peripheral Signal List	162
5.13	HP IO MUX Function List	167
5.14	LP IO MUX Function List	169
5.15	GPIO Pin Analog Function List	170
5.16	Event Task Matrix Function	170
5.17	Interrupts	172
5.18	Register Summary	172
5.18.1	HP GPIO Matrix Register Summary	172
5.18.2	HP IO MUX Register Summary	176
5.18.3	GPIO EXT Register Summary	177
5.18.4	LP GPIO Matrix Register Summary	178
5.18.5	LP IO MUX Register Summary	179
5.19	Registers	179
5.19.1	HP GPIO Matrix Registers	179
5.19.2	HP IO MUX Registers	195
5.19.3	GPIO EXT Registers	198
5.19.4	LP GPIO Matrix Registers	214
5.19.5	LP IO MUX Registers	223
6	Reset and Clock	227
6.1	Reset	227
6.1.1	Overview	227
6.1.2	Architectural Overview	227
6.1.3	Features	227
6.1.4	Functional Description	228
6.1.5	Peripheral Reset	229
6.2	Clock	229
6.2.1	Overview	229
6.2.2	Architectural Overview	230

6.2.3	Features	230
6.2.4	Functional Description	231
6.2.4.1	HP System Clock	231
6.2.4.2	LP System Clock	232
6.2.4.3	Peripheral Clocks	233
6.2.4.4	PMU Control of High-Performance System Clock Gating	236
6.3	Programming Procedures	238
6.3.1	HP System Clock Configuration	238
6.3.2	LP System Clock Configuration	239
6.3.3	Peripheral Clock Reset and Configuration	239
6.4	Register Summary	241
6.4.1	PCR Register Summary	241
6.4.2	LP System Clock Register Summary	243
6.5	Registers	245
6.5.1	PCR Registers	245
6.5.2	LP System Clock Registers	298
7	Chip Boot Control	310
7.1	Overview	310
7.2	Functional Description	310
7.2.1	Default Configuration	311
7.2.2	Chip Boot Mode Control	311
7.2.3	SDIO Sampling and Driving Clock Edge Control	313
7.2.4	ROM Messages Printing Control	313
7.2.5	JTAG Signal Source Control	314
7.2.6	Crystal Frequency Selection	315
8	Event Task Matrix (ETM)	316
8.1	Overview	316
8.2	Features	316
8.3	Functional Description	316
8.3.1	Architecture	316
8.3.2	Events	317
8.3.3	Tasks	320
8.3.4	Event and Task Status	324
8.3.5	Timing Considerations	324
8.3.6	Channel Control	325
8.4	Register Summary	327
8.5	Registers	331
9	System Timer	426
9.1	Overview	426
9.2	Features	426
9.3	System Timer Structure	427
9.4	Clock Source Selection	427
9.5	Functional Description	428

9.5.1	Counter	428
9.5.2	Comparator and Alarm	429
9.5.3	Event Task Matrix	430
9.5.4	Synchronization Operation	430
9.6	Interrupts	431
9.7	Programming Procedure	431
9.7.1	Read Current Count Value	431
9.7.2	Configure a One-Time Alarm in Target Mode	432
9.7.3	Configure Periodic Alarms in Period Mode	432
9.7.4	Update After Light-sleep	432
9.8	Register Summary	434
9.9	Registers	436
10	Timer Group (TIMG)	451
10.1	Overview	451
10.2	Feature List	451
10.3	Architectural Overview	452
10.4	Functional Description	452
10.4.1	16-bit Prescaler and Clock Selection	452
10.4.2	54-bit Time-base Counter	453
10.4.3	Alarm Generation	453
10.4.4	Timer Reload	454
10.4.5	Frequency Calculation of Slow Clock for Timer Group 0	455
10.5	Event Task Matrix Feature	455
10.6	Interrupts	456
10.7	Programming Procedures	457
10.7.1	Timer as a Simple Clock	457
10.7.2	Timer as One-shot Alarm	457
10.7.3	Timer as Periodic Alarm by APB	457
10.7.4	Timer as Periodic Alarm by ETM	458
10.7.5	Frequency Calculation of Slow Clock	459
10.8	Register Summary	460
10.9	Registers	461
11	Watchdog Timers (WDT)	474
11.1	Overview	474
11.2	Digital Watchdog Timers	475
11.2.1	Features	475
11.2.2	Functional Description	476
11.2.2.1	Clock Source and 32-Bit Counter	476
11.2.2.2	Stages and Timeout Actions	477
11.2.2.3	Write Protection	478
11.2.2.4	Flash Boot Protection	478
11.3	Super Watchdog	478
11.3.1	Features	478
11.3.2	Super Watchdog Controller	479

11.3.2.1	Structure	479
11.3.2.2	Workflow	479
11.4	Interrupts	480
11.5	Register Summary	480
11.6	Registers	480
12	RTC Timer	488
12.1	Introduction	488
12.2	Feature List	488
12.3	Functional Description	488
12.4	Event Task Matrix Feature	490
12.5	Interrupts	490
12.6	Register Summary	492
12.7	Registers	493
13	Permission Control (PMS)	502
13.1	Overview	502
13.2	Introduction to APM	503
13.3	Features	504
13.4	TEE and REE Terminology	505
13.5	Functional Description	505
13.5.1	TEE Controller Functional Description	505
13.5.2	APM Controller Functional Description	506
13.5.2.1	Architecture	506
13.5.2.2	SYS_APM Controller	508
13.5.2.2.1	Address Ranges	508
13.5.2.2.2	Access Permissions of Address Ranges	509
13.5.2.3	PERI_APM Controller	510
13.6	Illegal Access and Interrupts	511
13.6.1	SYS_APM Controller	511
13.6.2	PERI_APM Controller	512
13.7	Programming Procedure	512
13.8	Register Summary	513
13.8.1	HP_APM_REG	513
13.8.2	LP_APM_REG	516
13.8.3	LP_APMO_REG	517
13.8.4	CPU_APM_REG	519
13.8.5	TEE_REG	520
13.8.6	LP_TEE_REG	522
13.9	Registers	524
13.9.1	HP_APM_REG	524
13.9.2	LP_APM_REG	537
13.9.3	LP_APMO_REG	543
13.9.4	CPU_APM_REG	547
13.9.5	TEE_REG	556
13.9.6	LP_TEE_REG	558

14	Debug Assistant	562
14.1	Overview	562
14.2	Features	562
14.3	Functional Description	562
14.3.1	Region Read/Write Monitoring	562
14.3.2	SP Monitoring	563
14.3.3	PC Logging	563
14.3.4	CPU/DMA Bus Access Logging	563
14.4	Interrupts	563
14.5	Programming Procedures	564
14.5.1	Region Monitoring and SP Monitoring Configuration	564
14.5.2	PC Logging Configuration	565
14.5.3	CPU/DMA Bus Access Logging Configuration	566
14.5.3.1	Bus Access Logging 1 Configuration	566
14.5.3.2	Bus Access Logging 2 Configuration	569
14.6	Register Summary	573
14.6.1	Bus Logging Configuration Register Summary	573
14.6.2	Summary of Other Registers	574
14.7	Registers	576
14.7.1	Bus Logging Configuration Registers	576
14.7.2	Other Registers	585
15	Power Supply Detector	600
15.1	Overview	600
15.2	Features	600
15.3	Functional Description	600
15.3.1	Architecture	600
15.3.2	Brown-out Detector	601
15.3.3	Voltage Glitch Detectors	602
15.4	Interrupts	603
15.5	Register Summary	605
15.6	Registers	606
IV	Cryptography/Security Component	613
16	AES Accelerator (AES)	614
16.1	Introduction	614
16.2	Features	614
16.3	Clock and Reset	614
16.4	AES Working Modes	615
16.5	Typical AES Working Mode	616
16.5.1	Key, Plaintext, and Ciphertext	616
16.5.2	Endianness	617
16.5.3	Operation Process	619
16.6	DMA-AES Working Mode	620
16.6.1	Key, Plaintext, and Ciphertext	620

16.6.2	Endianness	621
16.6.3	Standard Incrementing Function	622
16.6.4	Block Number	622
16.6.5	Initialization Vector	622
16.6.6	Block Operation Process	622
16.7	Anti-Attack Pseudo-Round Function	623
16.8	Memory Summary	624
16.9	Register Summary	625
16.10	Registers	626
17	ECC Accelerator (ECC)	632
17.1	Overview	632
17.2	Feature List	632
17.3	ECC Basics	632
17.3.1	Elliptic Curve and Points on the Curves	632
17.3.2	Affine Coordinates and Jacobian Coordinates	633
17.3.3	Memory Blocks	633
17.3.4	Data and Data Block	633
17.3.5	Writing Data	634
17.3.6	Reading Data	634
17.3.7	Standard Calculation and Jacobian Calculation	634
17.4	Function Description	634
17.4.1	Curve Mode	634
17.4.2	Working Modes	635
17.4.2.1	Affine Point Multiplication (Affine Point Multi)	635
17.4.2.2	Affine Point Verification (Affine Point Verif)	636
17.4.2.3	Affine Point Verification + Affine Point Multiplication (Affine Point Verif + Multi)	636
17.4.2.4	Jacobian Point Multiplication (Jacobian Point Multi)	636
17.4.2.5	Point Addition (Point Add)	637
17.4.2.6	Jacobian Point Verification (Jacobian Point Verif)	637
17.4.2.7	Affine Point Verification + Jacobian Point Multiplication (Affine Point Verif + Jacobian Point Multi)	637
17.4.2.8	Mod Addition (Mod Add)	638
17.4.2.9	Mod Subtraction (Mod Sub)	638
17.4.2.10	Mod Multiplication (Mod Multi)	638
17.4.2.11	Mod Division (Mod Div)	639
17.4.3	Enhancing Anti-Attack Performance	639
17.4.4	Clock	639
17.4.5	Reset	640
17.5	Interrupts	641
17.6	Programming Procedures	641
17.7	Register Summary	642
17.8	Registers	643
18	HMAC Accelerator (HMAC)	648
18.1	Overview	648

18.2	Feature List	648
18.3	Functional Description	648
18.3.1	Upstream Mode	649
18.3.2	Downstream Mode - JTAG Enable Feature	649
18.3.3	Downstream Mode - Digital Signature Algorithm and Key Derivation Feature	650
18.4	HMAC eFuse Configuration	650
18.5	HMAC Process (Detailed)	651
18.5.1	Enable HMAC module	651
18.5.2	Configure HMAC keys and key purposes	651
18.5.3	Downstream mode process	652
18.5.4	Upstream mode process	652
18.6	HMAC Algorithm Details	653
18.6.1	Padding Bits	653
18.6.2	HMAC Algorithm Structure	654
18.6.3	Register Summary	655
18.6.4	Registers	656
19	RSA Accelerator (RSA)	662
19.1	Introduction	662
19.2	Features	662
19.3	Functional Description	662
19.3.1	Large-Number Modular Exponentiation	663
19.3.2	Large-Number Modular Multiplication	664
19.3.3	Large-Number Multiplication	665
19.3.4	Options for Additional Acceleration	665
19.4	Interrupts	667
19.5	Memory Summary	668
19.6	Register Summary	668
19.7	Registers	669
20	SHA Accelerator (SHA)	674
20.1	Introduction	674
20.2	Features	674
20.3	Working Modes	674
20.4	Function Description	675
20.4.1	Preprocessing	675
20.4.1.1	Padding the Message	675
20.4.1.2	Parsing the Message	675
20.4.1.3	Setting the Initial Hash Value	676
20.4.2	Hash Operation	676
20.4.2.1	Typical SHA Mode Process	676
20.4.2.2	DMA-SHA Mode Process	678
20.4.3	Message Digest	678
20.4.4	Interrupt	679
20.5	Register Summary	680
20.6	Registers	681

21	Digital Signature Algorithm (DSA)	685
21.1	Overview	685
21.2	Features	685
21.3	Functional Description	685
21.3.1	Overview	685
21.3.2	Private Key Operands	686
21.3.3	Software Prerequisites	686
21.3.4	DSA Operation at the Hardware Level	688
21.3.5	DSA Operation at the Software Level	688
21.4	Memory Summary	691
21.5	Register Summary	692
21.6	Registers	693
22	Elliptic Curve Digital Signature Algorithm (ECDSA)	696
22.1	Introduction	696
22.2	Features	696
22.3	ECDSA Basics	696
22.3.1	Domain Parameters	696
22.3.2	Key Generation	697
22.3.3	Signature Generation	697
22.3.4	Signature Verification	698
22.4	Functional Description	698
22.4.1	ECDSA Working Modes	698
22.4.2	Data and Data Block	700
22.4.2.1	Writing Data	700
22.4.2.2	Reading Data	700
22.4.2.3	Padding the Message	700
22.4.2.4	Parsing the Message	701
22.4.3	Security Features	701
22.4.3.1	High Anti-Attack Performance	701
22.4.3.2	Dynamic Access Permission	701
22.4.3.3	Hardware Occupation	702
22.5	Programming Procedures	702
22.5.1	ECDSA Process	702
22.5.1.1	IDLE Stage	703
22.5.1.2	PREP Stage	704
22.5.1.3	LOAD Stage	704
22.5.1.4	PROC Stage	704
22.5.1.5	GAIN Stage	704
22.5.1.6	POST Stage	705
22.5.1.7	ECDSA SHA Interface	705
22.5.2	Clock	706
22.5.3	Reset	707
22.6	Interrupts	707
22.7	Memory Blocks	709
22.8	Register Summary	710

22.9	Registers	711
23	External Memory Encryption and Decryption (XTS_AES)	718
23.1	Overview	718
23.2	Features	718
23.3	Module Structure	718
23.4	Functional Description	719
23.4.1	XTS Algorithm	720
23.4.2	Key	720
23.4.3	Target Memory Space	720
23.4.4	Data Writing	721
23.4.5	Manual Encryption Block	722
23.4.6	Auto Encryption Block	722
23.4.7	Auto Decryption Block	723
23.5	Software Process	723
23.6	Anti-DPA	724
23.6.1	Clock Anti-DPA Function	724
23.6.2	Pseudo-round Anti-DPA Function	725
23.7	Register Summary	727
23.8	Registers	728
24	Random Number Generator (RNG)	734
24.1	Introduction	734
24.2	Features	734
24.3	Functional Description	734
24.4	Programming Procedure	735
24.5	Register Summary	737
24.6	Registers	738
V	Connectivity Interface	739
25	UART Controller (UART)	740
25.1	Overview	740
25.2	Features	740
25.3	UART Structure	741
25.4	Functional Description	742
25.4.1	Clock and Reset	742
25.4.2	UART FIFO	743
25.4.3	Baud Rate Generation and Detection	743
25.4.3.1	Baud Rate Generation	743
25.4.3.2	Baud Rate Detection	744
25.4.4	UART Data Frame	745
25.4.5	AT_CMD Character Structure	746
25.4.6	RS485	746
25.4.6.1	Driver Control	746
25.4.6.2	Turnaround Delay	747

25.4.6.3	Bus Snooping	747
25.4.7	IrDA	747
25.4.8	Wakeup	748
25.4.9	Flow Control	749
25.4.9.1	Hardware Flow Control	749
25.4.9.2	Software Flow Control	751
25.4.10	GDMA Mode	751
25.4.11	Interrupts	752
25.5	Programming Procedures	754
25.5.1	Register Type	754
25.5.2	Detailed Steps	755
25.5.2.1	Initializing UART n	755
25.5.2.2	Configuring UART n Communication	755
25.5.2.3	Enabling UART n	756
25.6	Register Summary	757
25.6.1	UART Register Summary	757
25.6.2	LP UART Register Summary	758
25.6.3	UHCI Register Summary	759
25.7	Registers	761
25.7.1	UART Registers	761
25.7.2	LP UART Registers	784
25.7.3	UHCI Registers	804
26	I2C Controller (I2C)	826
26.1	Overview	826
26.2	Feature List	826
26.3	Architecture Overview	827
26.4	Functional Description	829
26.4.1	Clock Configuration	829
26.4.2	SCL and SDA Noise Filtering	830
26.4.3	SCL Clock Stretching	830
26.4.4	Generating SCL Pulses in Idle State	831
26.4.5	Synchronization	831
26.4.6	Open-Drain Output	832
26.4.7	Timing Parameters Configuration	833
26.4.8	Timeout Control	835
26.4.9	Command Configuration	835
26.4.10	TX/RX RAM Data Storage	836
26.4.11	Data Conversion	837
26.4.12	Addressing Mode	837
26.4.13	$R/\overline{W}$ Bit Check in 10-bit Addressing Mode	838
26.4.14	Start the I2C Controller	838
26.5	Functional Differences Between LP_I2C and I2C	838
26.6	Interrupts	839
26.7	Programming Procedures	840
26.7.1	I2C _{master} Writes to I2C _{slave} with a 7-bit Address in One Command Sequence	840

26.7.1.1	Introduction	840
26.7.1.2	Configuration Example	841
26.7.2	I2C _{master} Writes to I2C _{slave} with a 10-bit Address in One Command Sequence	842
26.7.2.1	Introduction	842
26.7.2.2	Configuration Example	842
26.7.3	I2C _{master} Writes to I2C _{slave} with Two 7-bit Addresses in One Command Sequence	843
26.7.3.1	Introduction	844
26.7.3.2	Configuration Example	844
26.7.4	I2C _{master} Writes to I2C _{slave} with a 7-bit Address in Multiple Command Sequences	845
26.7.4.1	Introduction	846
26.7.4.2	Configuration Example	847
26.7.5	I2C _{master} Reads I2C _{slave} with a 7-bit Address in One Command Sequence	848
26.7.5.1	Introduction	848
26.7.5.2	Configuration Example	849
26.7.6	I2C _{master} Reads I2C _{slave} with a 10-bit Address in One Command Sequence	850
26.7.6.1	Introduction	850
26.7.6.2	Configuration Example	850
26.7.7	I2C _{master} Reads I2C _{slave} with Two 7-bit Addresses in One Command Sequence	852
26.7.7.1	Introduction	852
26.7.7.2	Configuration Example	853
26.7.8	I2C _{master} Reads I2C _{slave} with a 7-bit Address in Multiple Command Sequences	854
26.7.8.1	Introduction	855
26.7.8.2	Configuration Example	856
26.8	Register Summary	858
26.8.1	I2C Register Summary	858
26.8.2	LP_I2C Register Summary	859
26.9	Registers	861
26.9.1	I2C Registers	861
26.9.2	LP_I2C Registers	884
27	I2S Controller (I2S)	907
27.1	Overview	907
27.2	Terminology	907
27.3	Features	908
27.4	System Architecture	909
27.5	Supported Audio Standards	911
27.5.1	TDM Philips Standard	912
27.5.2	TDM MSB Alignment Standard	912
27.5.3	TDM PCM Standard	913
27.5.4	PDM Standard	913
27.6	I2S TX/RX Clock	913
27.7	I2S Reset	915
27.8	I2S Master/Slave Mode	916
27.8.1	Master/Slave TX Mode	916
27.8.2	Master/Slave RX Mode	916
27.9	Transmitting Data	917

27.9.1	Data Format Control	917
27.9.1.1	Bit Width Control of Channel Valid Data	917
27.9.1.2	Endian Control of Channel Valid Data	918
27.9.1.3	A-law/ μ -law Compression and Decompression	918
27.9.1.4	Bit Width Control of Channel TX Data	919
27.9.1.5	Bit Order Control of Channel Data	919
27.9.2	Channel Mode Control	920
27.9.2.1	TDM TX Mode	920
27.9.2.2	PDM TX Mode	921
27.9.3	Synchronous Counter	924
27.10	Receiving Data	924
27.10.1	Channel Mode Control	924
27.10.1.1	TDM RX Mode	925
27.10.1.2	PDM RX Mode	925
27.10.2	Data Format Control	925
27.10.2.1	Bit Order Control of Channel Data	925
27.10.2.2	Bit Width Control of Channel Storage (Valid) Data	926
27.10.2.3	Bit Width Control of Channel RX Data	926
27.10.2.4	Endian Control of Channel Storage Data	926
27.10.2.5	A-law/ μ -law Compression and Decompression	926
27.11	Event Task Matrix Feature	927
27.12	I2S Interrupts	928
27.13	Software Configuration Process	928
27.13.1	Configure I2S as TX Mode	928
27.13.2	Configure I2S as RX Mode	929
27.14	Register Summary	930
27.15	Registers	931
28	USB Serial/JTAG Controller	950
28.1	Overview	950
28.2	Feature List	950
28.3	Functional Description	952
28.3.1	CDC-ACM USB Interface Functional Description	952
28.3.2	CDC-ACM Firmware Interface Functional Description	953
28.3.3	USB-to-JTAG Interface: JTAG Command Processor	954
28.3.4	USB-to-JTAG Interface: CMD_REP Usage Example	955
28.3.5	USB-to-JTAG Interface: Response Capture Unit	956
28.3.6	USB-to-JTAG Interface: Control Transfer Requests	956
28.4	Interrupts	957
28.5	Programming Procedures	958
28.6	Register Summary	960
28.7	Registers	962
29	SDIO Slave Controller (SDIO)	987
29.1	Overview	987
29.2	Features	987

29.3	Architecture Overview	987
29.4	Standards Compliance	988
29.5	Functional Description	988
29.5.1	Physical Bus	988
29.5.2	Supported Commands	989
29.5.3	I/O Function 0 Address Space	989
29.5.4	I/O Function 1/2 Address Space Map	992
29.5.4.1	Accessing SLC HOST Register Space	992
29.5.4.2	Transferring Incremental-Address Packets	992
29.5.4.3	Transferring Fixed-Address Packets	993
29.5.5	DMA	993
29.5.5.1	Linked List	993
29.5.5.2	Write-Back of Linked List	995
29.5.5.3	Data Padding and Discarding	996
29.5.6	SDIO Bus Timing	997
29.6	Interrupt	998
29.6.1	Host Interrupt	998
29.6.2	Slave Interrupt	998
29.7	Packet Sending and Receiving Procedure	999
29.7.1	Sending Packets to SDIO Host	999
29.7.2	Receiving Packets from SDIO Host	1001
29.8	Register Summary	1004
29.8.1	HINF Register Summary	1004
29.8.2	SLC Register Summary	1004
29.8.3	SLC Host Register Summary	1005
29.9	Registers	1007
29.9.1	HINF Registers	1007
29.9.2	SLC Registers	1012
29.9.3	SLC Host Registers	1034
30	LED PWM Controller (LEDC)	1044
30.1	Overview	1044
30.2	Feature List	1044
30.3	Architectural Overview	1044
30.4	Functional Description	1045
30.4.1	Timers	1045
30.4.1.1	Clock Source	1046
30.4.1.2	Clock Divider Configuration	1046
30.4.1.3	20-Bit Counter	1047
30.4.2	PWM Generators	1048
30.4.3	Duty Cycle Fading	1049
30.4.3.1	Linear Duty Cycle Fading	1050
30.4.3.2	Gamma Curve Fading	1050
30.4.3.3	Suspend and Resume Duty Cycle Fading	1051
30.5	Event Task Matrix Feature	1052
30.6	Interrupts	1053

30.7	Programming Procedures	1053
30.8	Memory Blocks	1055
30.9	Register Summary	1056
30.10	Registers	1058
31	Motor Control PWM (MCPWM)	1070
31.1	Overview	1070
31.2	Features	1070
31.3	Modules	1073
31.3.1	Overview	1073
31.3.1.1	Timer Module	1073
31.3.1.2	Operator Module	1073
31.3.1.3	Capture Module	1074
31.3.1.4	ETM Module	1074
31.3.2	PWM Timer Module	1075
31.3.2.1	Configurations of the PWM Timer Module	1075
31.3.2.2	PWM Timer's Working Modes and Timing Event Generation	1075
31.3.2.3	Shadow Register of PWM Timer	1081
31.3.2.4	PWM Timer Synchronization and Phase Locking	1081
31.3.3	PWM Operator Module	1081
31.3.3.1	PWM Generator Module	1083
31.3.3.2	Dead Time Generator Module	1095
31.3.3.3	PWM Carrier Module	1100
31.3.3.4	Fault Detection Module	1102
31.3.4	Capture Module	1103
31.3.4.1	Introduction	1103
31.3.4.2	Capture Timer	1104
31.3.4.3	Capture Channel	1104
31.3.5	ETM Module	1104
31.3.5.1	Overview	1104
31.3.5.2	MCPWM-Related ETM Events	1104
31.3.5.3	MCPWM-Related ETM Tasks	1105
31.4	Interrupts	1106
31.5	Register Summary	1107
31.6	Registers	1110
32	Remote Control Peripheral (RMT)	1157
32.1	Overview	1157
32.2	Features	1157
32.3	Functional Description	1158
32.3.1	RMT Architecture	1158
32.3.2	RAM	1159
32.3.2.1	Structure of RAM	1159
32.3.2.2	Use of RAM	1159
32.3.2.3	RAM Access	1160
32.3.3	Clock	1160

32.3.4	Transmitter	1161
32.3.4.1	Normal TX Mode	1161
32.3.4.2	Wrap TX Mode	1161
32.3.4.3	TX Modulation	1161
32.3.4.4	Continuous TX Mode	1162
32.3.4.5	Simultaneous TX Mode	1162
32.3.5	Receiver	1162
32.3.5.1	Normal RX Mode	1162
32.3.5.2	Wrap RX Mode	1163
32.3.5.3	RX Filtering	1163
32.3.5.4	RX Demodulation	1163
32.4	Configuration Update	1164
32.5	Interrupts	1164
32.6	Register Summary	1166
32.7	Registers	1168
33	Parallel IO Controller (PARLIO)	1183
33.1	Introduction	1183
33.2	Glossary	1183
33.3	Features	1183
33.4	Architectural Overview	1185
33.5	Functional Description	1185
33.5.1	Clock Generator	1185
33.5.2	Clock and Reset Restriction	1186
33.5.3	Master-Slave Mode	1187
33.5.4	Receive Modes of the RX Unit	1188
33.5.4.1	Level Enable Mode	1189
33.5.4.2	Pulse Enable Mode	1189
33.5.4.3	Software Enable Mode	1190
33.5.5	RX Unit GDMA SUC EOF Generation	1190
33.5.6	RX Unit Timeout	1191
33.5.7	Unlimited Length Data Transmission of TX Unit	1191
33.5.8	TX Chip Select Function	1191
33.5.9	Output Clock Gating of TX Unit	1191
33.5.10	Bus Idle Value of TX Unit	1192
33.5.11	Data Transfer in a Single Frame	1192
33.5.12	Bit Reversal in One Byte	1192
33.6	Interrupts	1193
33.7	Programming Procedures	1193
33.7.1	Data Receiving Operation Process	1193
33.7.2	Data Transmitting Operation Process	1194
33.8	Application Examples	1195
33.8.1	Co-working with SPI	1195
33.8.2	Co-working with I2S	1196
33.9	Register Summary	1198
33.10	Registers	1199

34 BitScrambler	1210
34.1 Introduction	1210
34.2 Feature List	1211
34.3 Application Examples	1211
34.4 Architectural Overview	1212
34.4.1 Data Path	1212
34.4.2 Control Path	1214
34.5 Functional Description	1214
34.5.1 Instructions	1214
34.5.2 Configuration Registers	1221
34.6 Programming Procedures	1223
34.7 Register Summary	1225
34.8 Registers	1226
 VI Analog Signal Processing	 1238
 35 Temperature Sensor	 1239
35.1 Overview	1239
35.2 Features	1239
35.3 Architecture	1239
35.4 Functional Description	1240
35.4.1 Temperature Sensor Power Up	1240
35.4.2 Temperature Sensor Clock	1240
35.4.3 Automatic Temperature Monitoring Modes	1240
35.4.4 Temperature Measurement Range and Offset	1241
35.4.5 Data Conversion	1241
35.5 Programming Procedure	1241
35.6 Interrupts	1242
35.7 Event Task Matrix Feature	1242
35.8 Register Summary	1244
35.9 Registers	1245
 36 ADC Controller	 1249
36.1 Overview	1249
36.2 Terminology	1249
36.3 Feature List	1249
36.4 Architectural Overview	1250
36.5 Functional Description	1251
36.5.1 ADC Power Up	1251
36.5.2 ADC Channels	1251
36.5.3 ADC Clock	1251
36.5.4 One-Shot Sampling Mode	1252
36.5.5 Multi-Channel Sampling Mode	1252
36.5.6 ADC Conversion and Attenuation	1252
36.5.7 DIG ADC FSM	1253
36.5.8 Pattern Table	1254

36.5.9	ADC Filters	1255
36.5.10	Threshold Monitors	1256
36.5.11	GDMA Support	1256
36.6	Event Task Matrix Feature	1257
36.7	Interrupts	1257
36.8	Programming Procedure	1258
36.8.1	Configuring One-shot Sampling Mode	1258
36.8.2	Configuring Multi-Channel Sampling Mode	1258
36.9	Register Summary	1259
36.10	Registers	1260
37	Analog Voltage Comparator	1272
37.1	Introduction	1272
37.2	Feature List	1272
37.3	Architectural Overview	1272
37.4	Functional Description	1273
37.5	Event Task Matrix Feature	1274
37.6	Interrupts	1274
37.7	Programming Procedures	1275
VII	Appendix	1276
	Related Documentation and Resources	1277
	Glossary	1278
	Abbreviations for Peripherals	1278
	Abbreviations Related to Registers	1278
	Access Types for Registers	1280
	Programming Reserved Register Field	1282
	Introduction	1282
	Programming Reserved Register Field	1282
	Interrupt Configuration Registers	1283
	Revision History	1284

List of Tables

1.3-1	Trace Encoder Parameters	36
1.5-1	Optional sideband signals	38
1.6-1	Header Format	41
1.6-2	Index Format	41
1.6-3	Packet format 3 subformat 0	41
1.6-4	Packet format 3 subformat 1	42
1.6-5	Packet format 3 subformat 3	42
1.6-6	Packet format 2	43
1.6-7	Packet format 1 with address	44
1.6-8	Packet format 1 without address	45
1.7-1	TRACE's Internal Interrupt Sources	46
1.8-1	Debug module trigger support (the action field of the mcontrol register)	47
1.8-2	Trace Status	49
2.3-1	LP CPU Exception Causes	74
2.6-1	Performance Counter	82
2.9-1	Wake Sources	86
3.4-1	GDMA Selecting Peripherals via Register Configuration	92
4.3-1	Memory Address Mapping	137
4.3-2	Module/Peripheral Address Mapping	143
5.4-1	HP GPIO Wakeup Signal Trigger and Clear Conditions	152
5.4-2	LP GPIO Wakeup Signal Trigger and Clear Conditions	153
5.8-1	Bit Used to Control IO MUX Functions in Light-sleep Mode	159
5.12-1	Peripheral Signals via HP GPIO Matrix	163
5.13-1	HP IO MUX Pin Functions	167
5.14-1	LP IO MUX Pin Functions	169
5.15-1	GPIO Pin Analog Functions	170
5.17-1	HP IO MUX and HP GPIO Matrix's Internal Interrupt Sources	172
5.17-2	LP IO MUX and LP GPIO Matrix's Internal Interrupt Sources	172
6.1-1	Reset Source	228
6.2-1	Crystal Frequency	231
6.2-2	CPU_CLK Clock Source	231
6.2-3	Frequency of CPU_CLK, AHB_CLK and HP_ROOT_CLK	232
6.2-4	Derived HP Clock Source	234
6.2-5	HP Clocks Used by Each Peripheral	234
6.2-6	Derived LP Clock Source	235
6.2-7	LP Clocks Used by Each Peripheral	235
6.2-8	Mapping between controlling the clock gating of read/write registers and related PMU registers	237
6.2-9	Mapping between controlling the functional clock gating of high-performance system peripherals and related PMU registers	238

7.2-1	Default Configuration of Strapping Pins	311
7.2-2	Boot Mode Control	311
7.2-3	SDIO Input Sampling Edge/Output Driving Edge Control	313
7.2-4	UART0 ROM Message Printing Control	314
7.2-5	USB Serial/JTAG ROM Message Printing Control	314
7.2-6	JTAG Signal Source Control	315
7.2-7	Crystal Frequency Selection	315
8.3-1	Selectable Events for ETM Channel n	317
8.3-2	Mappable Tasks for ETM Channel n	321
9.5-1	UNIT n Configuration Bits	428
9.5-2	Trigger Point	430
9.5-3	Synchronization Operation for Configuration Registers	430
9.6-1	System Timer's Internal Interrupt Sources	431
10.4-1	Alarm Generation When Up-Down Counter Increments	454
10.4-2	Alarm Generation When Up-Down Counter Decrements	454
10.6-1	TIMG n 's Internal Interrupt Sources	456
11.2-1	Timeout Actions	477
12.3-1	Configure RTC Timer to Log the Occurrence Time of Specific Events	489
12.3-2	Target Time Configuration	489
12.5-1	RTC Timer's Internal Interrupt Sources	490
13.1-1	Management Areas by PMP and AMP	502
13.4-2	Comparison Between TEE and REE	505
13.5-1	Master Access Source from HP System	506
13.5-2	APM Controllers Configuration Registers	508
14.5-1	HP CPU Packet Format	567
14.5-2	LP CPU Packet Format	567
14.5-3	DMA Packet Format	567
14.5-4	LOST Packet Format	568
14.5-5	DMA Access Source	568
14.5-6	DMA_0 Packet Format	571
14.5-7	DMA LOST Packet Format	571
15.3-1	Configuration for Detecting Power Pins	603
15.4-1	Power Supply Detector's Internal Interrupt Source	603
16.4-1	AES Accelerator Working Mode	615
16.4-2	Key Length and Encryption/Decryption	615
16.5-1	Working Status under Typical AES Working Mode	616
16.5-2	Text Endianness Type for Typical AES	617
16.5-3	Key Endianness Type for AES-128 Encryption and Decryption	618
16.5-4	Key Endianness Type for AES-256 Encryption and Decryption	618
16.6-1	Block Cipher Mode	620

16.6-2	Working Status under DMA-AES Working mode	620
16.6-3	TEXT-PADDING	621
16.6-4	Text Endianness for DMA-AES	621
17.3-1	ECC Accelerator Memory Blocks	633
17.4-1	ECC Accelerator Curve Mode Selection	635
17.4-2	Working Modes of ECC Accelerator	635
18.4-1	HMAC Purposes and Configuration Value	651
19.3-1	Acceleration Performance	667
19.4-1	RSA's Internal Interrupt Source	667
20.3-1	SHA Accelerator Working Mode	675
20.3-2	SHA Hash Algorithm Selection	675
20.4-1	The Storage and Length of Message Digest from Different Algorithms	679
20.4-2	SHA's Internal Interrupt Sources	679
22.4-1	ECDSA Working Mode	699
22.4-2	ECDSA Elliptic Curves Selection	699
22.4-3	ECDSA SHA Algorithm	699
22.4-4	ECDSA Working Status	699
22.7-1	ECDSA Memory Blocks	709
23.4-1	<i>Key</i> Generated Based on <i>Key_A</i>	720
23.4-2	Mapping Between Offsets and Registers	721
23.6-1	Configuration of XTS_AES Pseudo-round Anti-DPA	726
25.2-1	UART and LP UART Feature Comparison	740
25.4-1	UART_CHAR_WAKEUP Mode Configuration	749
26.3-1	I2C Timing Parameters (Cited from Table 5 in The I2C-bus specification Version 2.1)	829
26.4-1	I2C Synchronous Registers	832
27.4-1	I2S Signal Description	911
27.9-1	Bit Width of Channel Valid Data	918
27.9-2	Endian of Channel Valid Data	918
27.9-3	Data-Fetching Control in PDM Mode	921
27.9-4	I2S Channel Control for Raw PDM Data	922
27.9-5	I2S PCM-to-PDM Data Output	922
27.10-1	Channel Storage Data Width	926
27.10-2	Channel Storage Data Endian	926
28.3-1	Standard CDC-ACM Control Requests	952
28.3-2	CDC-ACM Settings with RTS and DTR	953
28.3-3	Commands of a Nibble	955
28.3-4	USB-to-JTAG Control Requests	957
28.3-5	JTAG Capability Descriptors	957
28.5-1	Reset SoC into Download Mode	959

28.5-2	Reset SoC into Booting from flash	959
29.5-1	SDIO Slave CCCR Configuration	990
29.5-2	SDIO Slave FBR Configuration	991
30.4-1	Commonly-used Frequencies and Resolutions	1048
30.8-1	LEDC Memory Blocks	1055
31.3-1	Functions of Each Submodule in the Operator Module	1074
31.3-2	Timing Events Used in PWM Generator	1084
31.3-3	Timing Events Priority When PWM Timer Increments	1084
31.3-4	Timing Events Priority when PWM Timer Decrements	1085
31.3-5	Dead Time Generator Switches Control Fields	1096
31.3-6	Typical Dead Time Generator Operating Modes	1096
31.3-7	MCPWM-Related ETM Events	1104
31.3-8	ETM Related Tasks	1105
32.4-1	Configuration Update	1164
33.5-1	Operations to Reset AHB Clock Domain with Clock Restrictions	1186
33.5-2	Requirements for TX Unit Operating as Slave with Clock Restrictions	1188
33.5-3	Requirements for RX Unit Operating as Slave with Clock Restrictions	1188
33.6-1	PARLIO's Internal Interrupt Sources	1193
34.5-1	Instruction Structure	1215
34.5-2	Effective CTL_MUX_SEL_n values when CTL_MUX_REL is 1	1216
34.5-3	CTL_MUX_SEL Values	1216
34.5-4	Instruction Opcode	1217
34.5-5	LOOP Opcode	1218
34.5-6	ADD Opcode	1219
34.5-7	IF Opcode	1219
34.5-8	IFN Opcode	1219
34.5-9	LDCTD Opcode	1220
34.5-10	LDCTI Opcode	1220
34.5-11	ADDCTI Opcode	1221
34.5-12	Settings for EOF modes	1222
35.4-1	Temperature Measurement Range and Offset	1241
36.5-1	SAR ADC Channels	1251
37.4-1	Mapping Between PAD and GPIO	1273
37.6-1	Analog Voltage Comparator's Internal Interrupt Sources	1274
37.7-4	Configuration of ENA/RAW/ST Registers	1283

List of Figures

1.0-1	Trace Encoder Overview	34
1.2-1	Trace Flow Overview	35
1.6-1	Trace packet Format	40
2.0-1	LP CPU Overview	64
2.9-1	Wake-Up and Sleep Flow of LP CPU	84
3.1-1	Modules that Share GDMA Channels	88
3.3-1	GDMA controller Architecture	89
3.4-1	Structure of a Linked List	90
3.4-2	Relationship among Linked Lists	93
4.3-1	System Structure and Address Mapping	137
4.3-2	ROM-Cache Structure	139
4.3-3	Cache Structure	140
4.3-4	Modules/peripherals that can work with GDMA	142
5.3-1	Architecture of HP GPIO Matrix, HP IO MUX, LP GPIO Matrix, and LP IO MUX	149
5.3-2	Internal Structure of a Pad	150
5.4-1	GPIO Input Synchronized on Rising Edge or on Falling Edge of HP IO MUX Operating Clock	151
5.4-2	GPIO Filter Timing of GPIO Input Signals	152
5.4-3	Glitch Filter Timing Example	154
5.10-1	Example of Level Flip on the Chip Pad — Hysteresis Function Not Enabled	161
5.10-2	Example of Level Flip on the Chip Pad — Hysteresis Function Enabled	161
6.1-1	Reset Types	227
6.2-1	System Clock	230
6.3-1	Clock Configuration Example	239
7.2-1	Chip Boot Flow	312
8.3-1	ETM Channel Architecture	317
8.3-2	Event Task Matrix Clock Architecture	324
9.3-1	System Timer Structure	427
9.5-1	System Timer Alarm Generation	428
10.1-1	Timer Group Overview	451
10.3-1	Timer Group Architecture	452
11.1-1	Watchdog Timers Overview	474
11.2-1	Digital Watchdog Timers in ESP32-C5	476
11.3-1	Super Watchdog Controller Structure	479
13.1-1	PMP-APM Management Relation	503
13.5-1	APM Controller Architecture	507
15.3-1	Architecture of Power Supply Detector	601

15.3-2	Brown-out Reset Workflow	602
15.3-3	Structure of Voltage Glitch Detectors	603
17.4-1	clk_ecc Diagram	640
17.4-2	clk_apb_ecc Diagram	640
18.6-1	HMAC SHA-256 Padding Diagram	654
18.6-2	HMAC Structure Schematic Diagram	654
21.3-1	Software Preparations and Hardware Working Process	687
22.5-1	ECDSA Process	703
22.5-2	clk_ecdsa Diagram	706
22.5-3	clk_apb_ecdsa Diagram	706
23.3-1	Architecture of the External Memory Encryption and Decryption	719
24.3-1	Noise Source	735
25.3-1	UART Structure	741
25.4-1	UART Controllers Division	744
25.4-2	The Timing Diagram of Weak UART Signals Along Negative Edges	744
25.4-3	Structure of UART Data Frame	745
25.4-4	AT_CMD Character Structure	746
25.4-5	Driver Control Diagram in RS485 Mode	747
25.4-6	The Timing Diagram of Encoding and Decoding in SIR mode	748
25.4-7	IrDA Encoding and Decoding Diagram	748
25.4-8	Hardware Flow Control Diagram	750
25.4-9	Connection between Hardware Flow Control Signals	750
25.4-10	Data Transfer in GDMA Mode	752
25.5-1	UART Programming Procedures	755
26.3-1	I2C Master Architecture	827
26.3-2	I2C Slave Architecture	827
26.3-3	I2C Protocol Timing (Cited from Fig.31 in The I2C-bus specification Version 2.1)	828
26.4-1	I2C Timing Diagram	833
26.4-2	Structure of I2C Command Registers	835
26.7-1	I2C _{master} Writing to I2C _{slave} with a 7-bit Address	840
26.7-2	I2C _{master} Writing to a Slave with a 10-bit Address	842
26.7-3	I2C _{master} Writing to I2C _{slave} with Two 7-bit Addresses	844
26.7-4	I2C _{master} Writing to I2C _{slave} with a 7-bit Address in Multiple Sequences	846
26.7-5	I2C _{master} Reading I2C _{slave} with a 7-bit Address	848
26.7-6	I2C _{master} Reading I2C _{slave} with a 10-bit Address	850
26.7-7	I2C _{master} Reading N Bytes of Data from addrM of I2C _{slave} with a 7-bit Address	852
26.7-8	I2C _{master} Reading I2C _{slave} with a 7-bit Address in Segments	855
27.4-1	ESP32-C5 I2S System Diagram	910
27.5-1	TDM Philips Standard Timing Diagram	912
27.5-2	TDM MSB Alignment Standard Timing Diagram	912

27.5-3	TDM PCM Standard Timing Diagram	913
27.5-4	PDM Standard Timing Diagram	913
27.6-1	I2S Clock Generator	914
27.9-1	TX Data Format Control	920
27.9-2	TDM Channel Control	921
27.9-3	PDM Channel Control Example	923
28.2-1	USB Serial/JTAG High Level Diagram	951
28.2-2	USB Serial/JTAG Block Diagram	952
29.3-1	SDIO Slave Block Diagram	988
29.5-1	CMD52 Content	989
29.5-2	CMD53 Content	989
29.5-3	Function 0 Address Space	989
29.5-4	Function 1/2 Address Space Map	992
29.5-5	DMA Linked List Descriptor Structure of the SDIO Slave	993
29.5-6	DMA Linked List of the SDIO Slave	995
29.5-7	Data Flow of Sending Incremental-address Packets From Host to Slave	996
29.5-8	Sampling Timing Diagram	997
29.5-9	Output Timing Diagram	997
29.7-1	Procedure of Slave Sending Packets to Host	1000
29.7-2	Procedure of Slave Receiving Packets from Host	1002
29.7-3	Loading Receiving Buffer	1003
30.3-1	LED PWM Architecture	1045
30.3-2	Timer and PWM Generator Block Diagram	1045
30.4-1	Frequency Division When LEDC_CLK_DIV is a Non-Integer Value	1047
30.4-2	Relationship Between Counter And Resolution	1047
30.4-3	LED PWM Output Signal Diagram	1049
30.4-4	Output Signal of Linear Duty Cycle Fading	1050
30.4-5	Output Signal of Gamma Curve Fading	1051
31.2-1	MCPWM Module Overview	1071
31.3-1	Timer Module	1073
31.3-2	Operator Module	1073
31.3-3	Capture Module	1074
31.3-4	ETM Module	1074
31.3-5	Count-Up Mode Waveform	1076
31.3-6	Count-Down Mode Waveforms	1076
31.3-7	Count-Up-Down Mode Waveforms, Count-Down at Synchronization Event	1076
31.3-8	Count-Up-Down Mode Waveforms, Count-Up at Synchronization Event	1077
31.3-9	UTEP and UTEZ Generation in Count-Up Mode	1078
31.3-10	DTEP and DTEZ Generation in Count-Down Mode	1079
31.3-11	DTEP and UTEZ Generation in Count-Up-Down Mode	1079
31.3-12	Block Diagram of A PWM Operator	1082
31.3-13	Symmetrical Waveform in Count-Up-Down Mode	1086

31.3-14	Count-Up, Single Edge Asymmetric Waveform, with Independent Modulation on PWMxA and PWMxB — Active High	1087
31.3-15	Count-Up, Pulse Placement Asymmetric Waveform with Independent Modulation on PWMxA	1088
31.3-16	Count-Up-Down, Dual Edge Symmetric Waveform, with Independent Modulation on PWMxA and PWMxB — Active High	1089
31.3-17	Count-Up-Down, Dual Edge Symmetric Waveform, with Independent Modulation on PWMxA and PWMxB — Complementary	1090
31.3-18	Count-Up-Down, Fault or Synchronization Events, with Same Modulation on PWMxA and PWMxB	1091
31.3-19	Example of an NCI Software-Force Event on PWMxA	1093
31.3-20	Example of a CNTU Software-Force Event on PWMxB	1094
31.3-21	Options for Setting up the Dead Time Generator Module	1096
31.3-22	Active High Complementary (AHC) Dead Time Waveforms	1097
31.3-23	Active Low Complementary (ALC) Dead Time Waveforms	1098
31.3-24	Active High (AH) Dead Time Waveforms	1098
31.3-25	Active Low (AL) Dead Time Waveforms	1099
31.3-26	Example of Waveforms Showing PWM Carrier Action	1100
31.3-27	Example of the First Pulse and the Subsequent Sustaining Pulses of the PWM Carrier Submodule	1101
31.3-28	Possible Duty Cycle Settings for Sustaining Pulses in the PWM Carrier Submodule	1102
32.3-1	RMT Architecture	1158
32.3-2	Format of Pulse Code in RAM	1159
33.4-1	PARLIO Architecture	1185
33.5-1	PARLIO Clock Generation	1186
33.5-2	Positive Waveform	1188
33.5-3	Negative Waveform	1188
33.5-4	Sub-Modes of Level Enable Mode for RX Unit	1189
33.5-5	Sub-Modes of Pulse Enable Mode for RX Unit	1190
33.5-6	Sub-Mode of Software Enable Mode for RX Unit	1190
34.1-1	BitScrambler System Context Diagram	1211
34.4-1	BitScrambler Data Path Diagram	1212
34.4-2	BitScrambler Control Path Diagram	1214
35.3-1	Temperature Sensor Architecture	1240
36.4-1	SAR ADC Architecture	1250
36.5-1	SAR ADC Clock Structure	1251
36.5-2	DIG ADC FSM Block Diagram	1253
36.5-3	APB_SARADC_SAR_PATT_TAB1_REG Contains Patterns 0 - 3	1254
36.5-4	APB_SARADC_SAR_PATT_TAB2_REG Contains Patterns 4 - 7	1254
36.5-5	Pattern Structure	1254
36.5-6	cmd0 configuration	1255
36.5-7	cmd1 Configuration	1255
36.5-8	DMA Data Format	1256
37.3-1	Analog Voltage Comparator Architecture	1272

Part I

Microprocessor and Master

This part covers the essential processing elements of the system, diving into the microprocessor architecture with both high-performance and low-power CPUs. Details include trace encoding for debugging purposes and controllers for Direct Memory Access (DMA).

Chapter 1

RISC-V Trace Encoder (TRACE)

The high-performance CPU (HP CPU) of ESP32-C5 supports the instruction trace interface through the trace encoder. The trace encoder connects to HP CPU's instruction trace interface, compresses the information into smaller packets, and then stores the packets into internal SRAM.

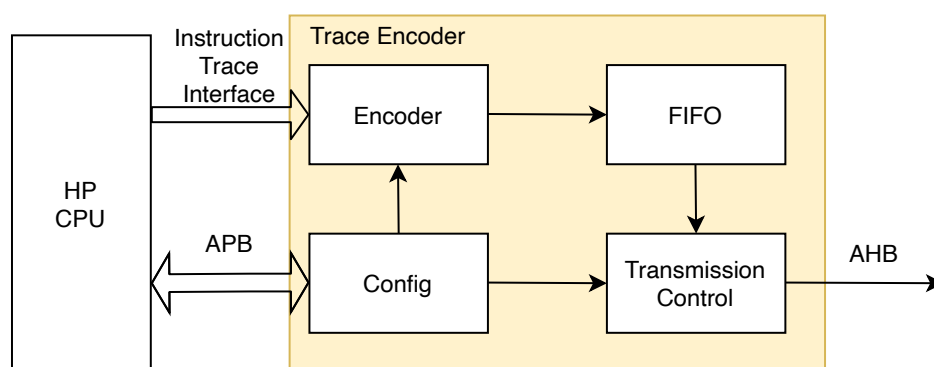


Figure 1.0-1. Trace Encoder Overview

1.1 Terminology

To better illustrate the functions of the RISC-V Trace Encoder, the following terms are used in this chapter.

hart	RISC-V hardware thread
branch	an instruction which conditionally changes the execution flow
uninferable discontinuity (updiscon)	a program counter change that can not be inferred from the program binary alone
delta	a change in the program counter that is other than the difference between two instructions placed consecutively in memory
trap	the transfer of control to a trap handler caused by either an exception or an interrupt
qualification	an instruction that meets the filtering criteria passes the qualification, and will be traced
te_inst	the name of the packet type emitted by the encoder
retire	the final stage of executing an instruction, when the machine state is updated
EPC	exception program counter

1.2 Introduction

In complex systems, understanding program execution flow is not straightforward. This may be due to a number of factors, for example, interactions with other cores, peripherals, real-time events, poor implementations, or some combination of all of the above.

It is hard to use a debugger to monitor the program execution flow of a running system in real time, as this is intrusive and might affect the running state. However, it is important to provide the visibility of program execution.

That is where instruction trace comes in, which provides a trace of the program execution. It works by tracking execution from a known start address and sending messages about the address deltas taken by the program. These deltas are typically introduced by jump, call, return, and branch type instructions, although interrupts and exceptions are also types of deltas.

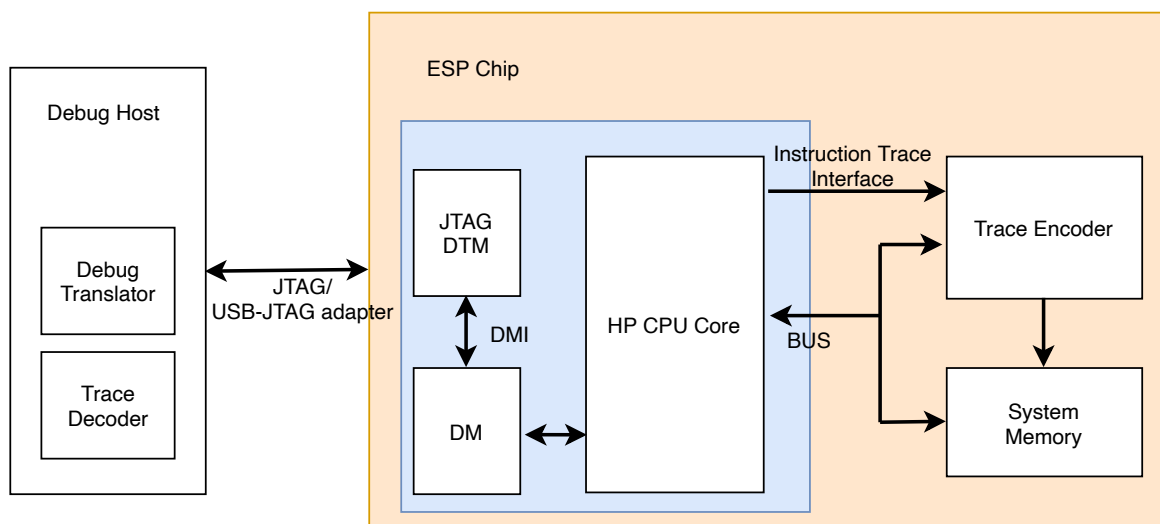


Figure 1.2-1. Trace Flow Overview

Figure 1.2-1 shows the instruction delta trace flow.

- The HP CPU core provides an instruction trace interface that outputs the instruction information executed by the HP CPU. Such information includes instruction address, instruction type, etc. For more details about ESP32-C5 HP CPU's instruction trace interface, please refer to Chapter 1 [High-Performance CPU \[to be added later\]](#).
- The trace encoder collects the relevant instruction trace information from HP CPU's instruction trace interface, compresses the information into lower bandwidth packets, and then stores the packets in system memory.
- The debugger (debug host) can dump the trace packets from the system memory via JTAG or USB Serial/JTAG, and use a decoder to decompress and reconstruct the program execution flow. The trace decoder, usually software on an external PC, takes in the trace packets and reconstructs the program instruction flow with the program binary that runs on the originating hart. This decoding step can be done offline or in real time while the hart is executing.

1.3 Features

- Compatible with [Efficient Trace for RISC-V Version 2.0](#). See Table 1.3-1 for the implemented parameters
- Support for delta address mode and full address mode
- Support for a filter unit
- Support for notifying an instruction address via debug trigger or filter unit
- Support for the following sideband signals to control trace data flow:
 - Support for debugging trigger to start or stop encoder
 - When the hart is halted, the encoder can report the last packet and then stop
 - When the hart is reset, the encoder can report the last packet and then stop
 - Support for stalling the hart when trace FIFO is almost full
- Arbitrary address range of the trace memory size
- Configurable synchronization modes:
 - Synchronization counter counts by packet
 - Synchronization counter counts by cycle
 - Synchronization counter can be disabled
- Trace lost status to indicate packet loss
- Automatic restart after packet loss
- Memory writing in the loop or non-loop mode
- Support two interrupts:
 - Triggered when the packet size exceeds the configured memory space
 - Triggered when a packet is lost
- FIFO (128 × 8 bits) to buffer packets
- AHB burst transmission with configurable burst length

Table 1.3-1. Trace Encoder Parameters

Parameter Name	Value	Description
arch_p	0	Initial version
bpred_size_p	0	Branch prediction mode is not supported
cache_size_p	0	Jump target cache mode is not supported
call_counter_size_p	0	Implicit return mode is not supported
ctype_width_p	0	Width of the ctype bus
context_width_p	0	Width of context bus
ecause_width_p	6	Width of exception cause
ecause_choice_p	0	Multiple choice is not supported
f0s_width_p	0	Format 0 packets are not supported
filter_context_p	0	Filtering on context is not supported

Parameter Name	Value	Description
filter_excint_p	1	Filtering on exception cause or interrupt is supported
filter_privilege_p	1	Filtering on privilege is supported
filter_tval_p	1	Filtering on trap value is supported
iaddress_lsb_p	1	Compressed instructions are supported
iaddress_width_p	32	The instruction bus is 32-bit
iretire_width_p	1	Width of the iretire bus
ilastsize_width_p	0	Width of the ilastsize
itype_width_p	3	Width of the itype bus
nocontext_p	1	Exclude context from te_inst packets
notime_p	1	Exclude time from te_inst packets
privilege_width_p	1	Only machine and user mode are supported
retires_p	1	Maximum number of instructions that can be retired per block
return_stack_size_p	0	Implicit return mode is not supported
sijump_p	0	Sequentially inferable jump mode is not supported
taken_branches_p	1	Only one instruction retired per cycle
impdef_width_p	0	Not implemented

For detailed descriptions of the above parameters, please refer to the [Efficient Trace for RISC-V Version 2.0 > Chapter Parameters and Discovery](#).

1.4 Architectural Overview

This chapter mainly introduces the implementation details of ESP32-C5's trace encoder.

As shown in Figure 1.0-1, the trace encoder contains an encoder, a FIFO, a register configuration module, and a transmission control module.

The encoder receives the HP CPU's instruction information via the instruction trace interface, compresses it into different packets, and writes it to the internal FIFO.

The transmission control module writes the data in the FIFO to the internal SRAM through the AHB bus.

The FIFO is 128 deep and 8-bit wide. When the memory bandwidth is insufficient, the FIFO may overflow, resulting in packet loss. When a packet is lost, the encoder will send a packet to indicate this event.

Thereafter, the encoder will stop working till FIFO gets empty. An interrupt will also be generated to indicate this event if it has been enabled.

1.5 Functional Description

1.5.1 Synchronization

In order to make the trace robust there must be regular synchronization points within the trace.

Synchronization is accomplished by sending a full valued instruction address. When the synchronization counter value reaches the value of the [TRACE_RESYNC_PROLONGED](#) field of the [TRACE_RESYNC_PROLONGED_REG](#) register, the encoder will send a synchronization packet (format 3 subformat 0, see Section 1.6.3.1).

The synchronization counter is configured via [TRACE_RESYNC_MODE](#):

- 0: Disable the synchronization counter
- 1: Invalid. No effect
- 2: Synchronization counter counts by packet
- 3: Synchronization counter counts by cycle

You can adjust the trace bandwidth by increasing the value of [TRACE_RESYNC_PROLONGED](#) to reduce the frequency of sending synchronization packets, thereby reducing the bandwidth occupied by packets.

1.5.2 Address Mode

ESP32-C5 supports two address modes: delta address mode and full address mode.

- Delta address mode (default): In delta address mode, addresses are encoded as the difference between the actual address of the current instruction and the actual address of the instruction reported in the previous packet that contained an address. This differential encoding requires fewer bits than the full address, and thus results in more efficient trace compression.
- Full address mode: In full address mode, all addresses in the trace are encoded as absolute addresses instead of in differential form. This kind of encoding is always less efficient, but it can be a useful debugging aid for software decoder developers. This mode is enabled by setting [TRACE_FULL_ADDRESS](#).

1.5.3 Optional Sideband Signals

The Efficient Trace for RISC-V Version 2.0 > Chapter Optional Sideband Signals has provided some optional sideband signals for encoder control. ESP32-C5 has implemented the following functions:

Table 1.5-1. Optional sideband signals

Signal	Function
trigger[2:0]	• A pulse on bit 0 will cause the encoder to start tracing, and continue until further notice, subject to other filtering criteria also being met. • A pulse on bit 1 will cause the encoder to stop tracing until further notice. • A pulse on bit 2 will cause the encoder to report a specific address. This function is enabled by setting TRACE_DM_TRIGGER_ENA. The trigger signal is asserted upon a trigger match. For example, if the action field of mcontrol is 2, while the trigger matched it will assert trigger bit0; if the action is 3, while the trigger matched it will assert trigger bit1. For details, please refer to Table 1.8-1.
halted	Hart is halted. Upon assertion, the encoder will output a packet to report the address of the last instruction retired before halting, followed by a support packet to indicate that tracing has stopped. Upon deassertion, the encoder will start tracing again, commencing with a synchronization packet. This function is enabled by setting TRACE_HALT_ENA.

Signal	Function
reset	Hart is in reset. Behavior is as described above for halted.
stall	Stall request to hart. Some applications may require lossless trace, which can be achieved by using this signal to stall the hart if the trace encoder is unable to output a trace packet (internal FIFO almost full). This function is enabled by setting TRACE_STALL_ENA .

1.5.4 Filtering

Filtering provides a mechanism to control the criteria for the encoder to produce a trace. For example, it may be desirable to trace:

- When the instruction address is within a particular range
- Starting from one instruction address and continue until a second instruction address
- For one or more specified privilege levels
- Exception and/or interrupt handlers for specified exception causes

The filtering unit has a filter and a comparator unit.

Each comparator unit is actually a pair of comparators (Primary and Secondary, or P, S) allowing a bounded range to be matched with a signal unit if required, and offers:

- Input selected from instruction address (iaddress) and trap value (tval)
- A range of arithmetic options (<, >, =, !=, etc.) independently selectable for each comparator
- Secondary match value may be used as a mask for the primary comparator
- The two comparators can be combined in several ways: P , $P\&\&S$, $!(P\&\&S)$, latch (set on P clear on S)
- Each comparator can also be used to explicitly report a particular instruction address

Each filter can specify filtering against instruction from the hart, and offers:

- 1 run-time selectable comparator units to match
- Selection for specific privilege level (priv) and exception cause (ecause) as inputs
- Select matching for interrupt

1.5.5 Anchor Tag

Since the length of data packets is variable, in order to identify boundaries between data packets when packed packets are written to memory, the ESP32-C5 encoder inserts zero bytes between data packets:

- The maximum packet length is 13 bytes, so a sequence of at least 14 zero bytes cannot occur within a normal packet. Therefore, the first non-zero byte seen after a sequence of at least 14 zero bytes must be the first byte of a packet.
- Every time when 128 packets are transmitted, the encoder writes 14 zero bytes to the memory partition boundary as anchor tags.

1.5.6 Memory Writing Mode

When writing the trace memory, the size of the trace packets might exceed the capacity of the memory. In this case, you can choose whether to wrap around the trace memory or not by configuring the memory writing mode:

- Loop mode: When the size of the trace packets exceeds the capacity of the trace memory (namely when `TRACE_MEM_CURRENT_ADDR_REG` reaches the value of `TRACE_MEM_END_ADDR_REG`), the trace memory is wrapped around, so that the encoder loops back to the memory's starting address `TRACE_MEM_START_ADDR_REG`, and old data in the memory will be overwritten by new data.
- Non-loop mode: When the size of the trace packets exceeds the capacity of the trace memory, the trace memory is not wrapped around. The encoder stops at `TRACE_MEM_END_ADDR_REG`, and old data will be retained.

1.5.7 Automatic Restart

When packets are lost due to FIFO overflow, the encoder will stop working and need to be resumed by software manually or restarted by hardware automatically. If the `TRACE_RESTART_ENA` bit of `TRACE_TRIGGER_REG` is set, once the FIFO is empty, the encoder can automatically be restarted.

If the automatic restart feature is enabled, the encoder will be restarted in any case. Therefore, to disable the encoder, the automatic restart feature must be disabled first by clearing the `TRACE_RESTART_ENA` bit of the `TRACE_TRIGGER_REG` register.

1.6 Encoder Output Packets

This section mainly introduces ESP32-C5 trace encoder output packet format.



Figure 1.6-1. Trace packet Format

A packet includes header, index, and payload. Header, index and payload are transmitted sequentially in bit stream form, from the fields listed at the top of tables below to the fields listed at the bottom. If a field consists of multiple bits, then the least significant bit is transmitted first.

1.6.1 Header

Header is 1-byte long. The format of header is shown in Table 1.6-1.

Table 1.6-1. Header Format

Field	Bit Width	Description	Value
length	5	Length of whole packet	4 ~ 13
flow	2	Reserved	0
timestamp	1	Reserved	0

1.6.2 Index

Index has 2 bytes. The format of index is shown in Table 1.6-2.

Table 1.6-2. Index Format

Field	Bit Width	Description	Value
index	16	The index of each packet	0~65536

1.6.3 Payload

The length of payload ranges from 1 byte to 10 bytes.

1.6.3.1 Format 3 Packets

Format 3 packets are used for synchronization, and report supporting information. There are 4 subformats defined in the specification. ESP32-C5 only supports 3 of them.

Format 3 Subformat 0 - Synchronization

This packet contains all the information the decoder needs to fully identify an instruction. It is sent for the first traced instruction (unless that instruction also happens to be the first one in an exception handler), and when synchronization has been scheduled by the expiry of the synchronization timer. The payload length is 5 bytes.

Table 1.6-3. Packet format 3 subformat 0

Field name	Bit Width	Description
format	2	11 (sync): Synchronization
subformat	2	00 (start): Start of tracing, or resync
branch	1	Set to 0 if the address points to a branch instruction, and the branch was taken. Set to 1 if the instruction is not a branch or if the branch is not taken.
privilege	1	The privilege level of the reported instruction
address	31	Full instruction address. The address must be left shifted 1 bit in order to recreate the original byte address.
sign_extend	3	Reserved

Format 3 Subformat 1 - Exception

This packet also contains all the information the decoder needs to fully identify an instruction. It is sent following an exception or interrupt, and includes the cause, the 'trap value' (for exceptions), and the address

of the trap handler or of the exception itself. The length is variable: if the interrupt field is 1, the tvalepc field is omitted and the length is 6 bytes, otherwise the length is 10 bytes.

Table 1.6-4. Packet format 3 subformat 1

Field name	Bit Width	Description
format	2	11 (sync): Synchronization
subformat	2	01 (exception): Exception cause and trap handler address
branch	1	Set to 0 if the address points to a branch instruction, and the branch was taken. Set to 1 if the instruction is not a branch or if the branch is not taken.
privilege	1	The privilege level of the reported instruction
ecause	6	Exception cause
interrupt	1	Interrupt
theadr	1	When set to 1, the address field points to the trap handler address. When set to 0, the address field points to the EPC (exception program counter) for an exception at the target for an updiscon, and is undefined for other exceptions and interrupts.
address	31	Full instruction address. The value of this field must be left shifted 1 bit in order to recreate the original byte address.
tvalepc	32	Value from appropriate utval/stval/mtval CSR (control/status register). Omitted if the interrupt is 1
sign_extend	3	Reserved

Format 3 Subformat 3 - Support

This packet provides supporting information to aid the decoder. It is issued when the trace is ended, filter match is failed, or a packet is lost. The length is 2 bytes.

Table 1.6-5. Packet format 3 subformat 3

Field name	Bit Width	Description
format	2	11 (sync): Synchronization
subformat	2	11 (support): Supporting information for the decoder
enable	1	Indicates if the encoder is enabled
encoder_mode	1	Always 0. Only supported by branch trace
qual_status	2	Indicates qualification status: • 00 (no_change): No change to filter qualification • 01 (ended_rep): Qualification ended, preceding instruction sent explicitly to indicate last qualification instruction • 10 (trace lost): One or more packets lost • 11 (ended_upd): Qualification ended, preceding te_inst would have been sent anyway due to an updiscon, even if wasn't the last qualified instruction

Field name	Bit Width	Description
ioptions	6	Indicates optional modes: - sequentially inferred jumps (Bits 5): When set to 1, the targets of sequentially inferable jumps will not be reported. (This feature is not implemented, so bit 5 should always be 0) - branch prediction (Bit 4): When set to 1, the branch prediction is enabled. (This feature is not implemented, so bit 4 should always be 0) - jump target cache (Bit 3): When set to 1, the jump target cache is enabled. (This feature is not implemented, so bit 3 should always be 0) - full address (Bit 2): 0: delta address mode 1: full address mode - implicit exception (Bit 1): Exclude address from format 3 subformat 1 packet if trap vector can be determined from ecause. (This feature is not implemented, so bit 1 should always be 0) - implicit return (Bit 0): When set to 1, function return addresses will not be reported. (This feature is not implemented, bit 0 should always be 0)
sign_extend	2	Reserved

1.6.3.2 Format 2 Packets

This packet contains only an instruction address, and is used when the address of an instruction must be reported (for example if the instruction is the target for an updiscon, or is the last instruction before exception), and there is no reported branch information.

The length is variable if delta address mode is enabled, ranging from 5 bytes to 8 bytes. The address field can be 8/16/24/32 bits, inferred from the packet length. For example, if the header length is n bytes, the address bits are $n - 4$ bytes. A sign bit is required to extend the address field to 32 bits.

Table 1.6-6. Packet format 2

Field name	Bit Width	Description
format	2	10 (addr-only): No branch information
notify	1	If the value of this bit is different from the MSB of address, it indicates that this packet is reporting an instruction that is not the target of an uninferable discontinuity because a notification was requested via trigger[2] or a filter matched.
updiscon	1	If the value of this bit is different from notify, it indicates that this packet is reporting the instruction following an uninferable discontinuity and is also the instruction before an exception, privilege change or resync.

Field name	Bit Width	Description
zero_extend	4	Reserved
address	Variable	Delta instruction address. These bits can be 8/16/24/32.

1.6.3.3 Format 1 Packets

This packet includes branch information, and is used when either the branch information must be reported (for example because the branch map is full), or when the address of instruction must be reported, and there has been at least one branch since the previous packet.

The packet length is variable if delta address mode is enabled, ranging from 5 bytes to 12 bytes. The address field can be 8/16/24/32 bits, inferred from the packet length. For example:

- The header length is n bytes
- The value of the format field is 1
- The value of the branch field is:
 - 0: the packet is format 1 without address
 - 1: branch_map is 1 bit wide, zero_ext is 6 bits wide, and address is $n - 5$ bits wide
 - 2 ~ 3: branch_map is 3 bits wide, zero_ext is 4 bits wide, and address is $n - 5$ bits wide
 - 4 ~ 7: branch_map is 7 bits wide, zero_ext is 0 bits wide, and address is $n - 5$ bits wide
 - 8 ~ 15: branch_map is 15 bits wide, zero_ext is 0 bits wide, and address is $n - 6$ bits wide
 - 16 ~ 31: branch_map is 31 bits wide, zero_ext is 0 bits wide, and address is $n - 8$ bits wide

Format 1 - address, branch_map

The length is variable.

Table 1.6-7. Packet format 1 with address

Field name	Bit Width	Description
format	2	01: Includes branch information
branches	5	Number of valid bits branch_map. The number of bits of branch_map is determined as follows: • 0: (cannot occur for this format) • 1: 1 bit • 2 ~ 3: 3 bits • 4 ~ 7: 7 bits • 8 ~ 15: 15 bits • 16 ~ 31: 31 bits For example if branches = 12, branch_map is 15-bit long, and the 12 LSBs are valid.

Field name	Bit Width	Description
branch_map	Variable	An array of bits indicating whether branches are taken or not. Bit 0 represents the oldest branch instruction executed. For each bit: • 0: branch taken • 1: branch not taken The field Bits is variable and determined by the branches field.
notify	1	If the value of this bit is different from the MSB of address, it indicates that this packet is reporting an instruction that is not the target of an uninferable discontinuity because a notification was requested via trigger[2] or a filter matched.
updiscon	1	If the value of this bit is different from notify, it indicates that this packet is reporting the instruction following an uninferable discontinuity and is also the instruction before an exception, privilege change or resync.
zero_ext	Variable	The length of this field is determined by branches as follows: • 1: 6 bits • 2 ~ 3: 4 bits • 4 ~ 31: 0 bits
address	Variable	Delta instruction address. These bits can be 8/16/24/32

Format 1 - no address, branch_map

The length is 5 bytes.

Table 1.6-8. Packet format 1 without address

Field name	Bit Width	Description
format	2	01: includes branch information
branches	5	Number of valid bits in branch_map. The length of branch_map is 31 bits. Only 0 valid.
branch_map	31	An array of bits indicating whether branches are taken or not. Bit 0 represents the oldest branch instruction executed. For each bit: • 0: branch taken • 1: branch not taken
sign_extend	2	Reserved

1.7 Interrupt

ESP32-C5's TRACE (i.e., the trace encoder of HP CPU) can generate the TRACE_INTR interrupt signal that will be sent to the [Interrupt Matrix](#).

There are several internal interrupt sources from TRACE that can generate the TRACE_INTR interrupt signal. The interrupt sources from TRACE are listed in [Table 1.7-1](#) with their trigger conditions and the resulted interrupt signal in [Table 1.7-1](#).

Table 1.7-1. TRACE's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
TRACE_MEM_FULL_INTR	Triggered when the packet size exceeds the capacity of the trace memory, namely when TRACE_MEM_CURRENT_ADDR_REG reaches the value of TRACE_MEM_END_ADDR_REG . If necessary, this interrupt can be enabled to notify the HP CPU for processing, such as applying for a new memory space again	TRACE_INTR
TRACE_FIFO_OVERFLOW_INTR	Triggered when the internal FIFO overflows and one or more packets have been lost.	TRACE_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section 1.9 [Register Summary](#).

1.8 Programming Procedures

1.8.1 Encoder Option Configuration

By default, trace packets are sent to memory with deltas address until the trace FIFO overflows or the trace memory becomes full. There are some optional configuration options that can change the trace behavior.

- Full address mode
 - The encoder uses delta address mode by default. Users can set [TRACE_FULL_ADDRESS](#) to enable full address mode for debugging.
- Restart after packet loss
 - Once the trace packet is lost due to FIFO overflow, the encoder will automatically stop if [TRACE_RESTART_ENA](#) is set. When the FIFO is empty, the encoder will restart to report packets.
- Stall CPU while FIFO is almost full
 - If [TRACE_STALL_ENA](#) is set, when the FIFO is almost full, a stall request will be asserted to the CPU, and the CPU will be stalled until the request is deasserted.
- Stop tracing when the CPU is halted

If [TRACE_HALT_ENA](#) is set, when the CPU is halted, the encoder will output a packet to report the address of the last instruction before halting, followed by a support packet to indicate that tracing has stopped. Upon the deassertion of the halted signal, the encoder will start tracing again, commencing with a synchronization packet.

- Stop tracing when the CPU is in reset

The encoder will be reset separately, if [TRACE_RESET_ENA](#) is set. When the CPU is in reset, the encoder will output a packet to report the address of the last instruction before reset, followed by a support packet to indicate that tracing has stopped. Once the CPU exits from reset, the encoder will start tracing again, commencing with a synchronization packet.

- Using trigger outputs from the Debug Module

The debug module of the HP CPU has a trigger unit. This defines a match control register (mcontrol) containing a 4-bit action field, and reserves codes 2 ~ 4 of this field for trace use. ESP32-C5 has implemented this feature. The values of the action field are listed in Table 1.8-1.

Table 1.8-1. Debug module trigger support (the action field of the mcontrol register)

Value	Description
2	Trace-on: start trace
3	Trace-off: end trace
4	Trace-notify: notify the encoder to report an address

For details about the debug module trigger unit, please refer to Chapter 1 *High-Performance CPU [to be added later]*.

1.8.2 Filter Configuration

The filter unit described in Section 1.5.4 can be configured as follows:

1. Select one of the following match modes:

- Set [TRACE_MATCH_PRIVILEGE](#) to enable privilege match mode:
 - Configure [TRACE_MATCH_CHOICE_PRIVILEGE](#) to specify the privilege level.
- Set [TRACE_MATCH_ECAUSE](#) to enable ecause match mode:
 - Configure [TRACE_MATCH_CHOICE_ECAUSE](#) to specify the ecause value
- Set [TRACE_MATCH_INTERRUPT](#) to enable interrupt match mode:
 - Configure [TRACE_MATCH_VALUE_INTERRUPT](#) to specify interrupt level
- Set [TRACE_MATCH_COMP](#) to enable comparator match mode:
 - Configure the primary comparator:
 - * Configure [TRACE_P_INPUT](#) to choose the input
 - * Configure [TRACE_P_FUNCTION](#) to select the comparator function
 - * Configure [TRACE_FILTER_P_COMPARATOR_MATCH_REG](#) to define match value
 - Configure the secondary comparator if needed:
 - * Configure [TRACE_S_INPUT](#) to choose the input
 - * Configure [TRACE_S_FUNCTION](#) to select the comparator function
 - * Configure [TRACE_FILTER_S_COMPARATOR_MATCH_REG](#) to define match value

- Choose which comparator to match via [TRACE_MATCH_MODE](#):
 - * 0: only the primary comparator matches, and the secondary comparator has no effect
 - * 1: both the primary comparator and the secondary comparator match ($P \& S$)
 - * 2: neither the primary comparator nor secondary comparator match ($\neg(P \& S)$)
 - * 3: start filtering when primary matches up until secondary matches

If [TRACE_P_NOTIFY](#) or [TRACE_S_NOTIFY](#) is set, the comparator will report the matched address as a notification, the instructions will not be filtered. For example:

- If the [TRACE_MATCH_MODE](#) is 0, and [TRACE_P_NOTIFY](#) is set, then all instructions will be traced, and the matched primary comparator will be notified. For details about notify, please refer to Section 1.8.5.
- If the [TRACE_MATCH_MODE](#) is 0, and [TRACE_S_NOTIFY](#) is set, the instruction will be filtered by the primary comparator, and the instruction that matches the secondary comparator will be notified.

2. Set [TRACE_FILTER_EN](#) to enable filter unit.

1.8.3 Enable Encoder

- Configure the address space for the trace memory via [TRACE_MEM_START_ADDR_REG](#) and [TRACE_MEM_END_ADDR_REG](#). The encoder can access 0x40800000 ~ 0x4085FFFF
- Update the value of [TRACE_MEM_CURRENT_ADDR_REG](#) to the value of [TRACE_MEM_START_ADDR_REG](#) by setting [TRACE_MEM_CURRENT_ADDR_UPDATE](#)
- (Optional) Configure the memory writing mode via the [TRACE_MEM_LOOP](#) bit of [TRACE_TRIGGER_REG](#)
 - 0: Non-loop mode
 - 1: Loop mode (default)
- Configure the synchronization mode via the [TRACE_RESYNC_MODE](#) bit of [TRACE_RESYNC_PROLONGED_REG](#)
 - 0: Disable the synchronization counter
 - 1: Invalid. No effect
 - 2: Synchronization counter counts by packet
 - 3: Synchronization counter counts by cycle
- (Optional) Configures the threshold for the synchronization counter (default value is 128) via [TRACE_RESYNC_PROLONGED_REG](#)
- (Optional) Enable Interrupt
 - Set the corresponding bit of [TRACE_INTR_ENA_REG](#) to enable the corresponding interrupt
 - Set the corresponding bit of [TRACE_INTR_CLR_REG](#) to clear the corresponding interrupt
 - Read [TRACE_INTR_RAW_REG](#) to know which interrupt occurs

- (Optional) Enable automatic restart by setting the [TRACE_RESTART_ENA](#) bit of [TRACE_TRIGGER_REG](#). This function is enabled by default
- There are 2 ways to enable the trace encoder:
 - Setting the [TRACE_TRIGGER_ON](#) bit of [TRACE_TRIGGER_REG](#)
 - Through the Debug module by configuring Trace-on trigger action to start the encoder. When the trigger is matched and the action is 2, it will start the encoder. For trigger configurations, please refer to the RISC-V Debug Specification.

Once the encoder is enabled, it will keep tracing the HP CPU's instruction trace interface and writing packets to the trace memory.

1.8.4 Disable Encoder

There are two ways to stop producing trace packets:

1. Set [TRACE_TRIGGER_OFF](#) of [TRACE_TRIGGER_REG](#) to end the encoder
2. Through the Debug module by configuring Trace-off trigger action to stop the encoder. When the trigger is matched and the action is 3, it will disable the encoder. For trigger configurations, please refer to the RISC-V Debug Specification.

Due to the internal FIFO, the packets are not written to the memory immediately after the end of the trace. It is recommended to query the [TRACE_FIFO_STATUS_REG](#) to confirm that the data is all written to memory.

Table 1.8-2. Trace Status

Field	Description
TRACE_FIFO_EMPTY	If 1 indicates that the FIFO is empty, all data have been written to memory
TRACE_WORK_STATUS	Encoder's work status: • 0: idle state, the encoder is not started • 1: the encoder is normally working that output packets • 2: the encoder is not outputting packets, the CPU is halted or in reset • 3: the encoder is not outputting packets, it occurs while a packet is lost, and is waiting for FIFO empty to restart.

1.8.5 Notify

Users may want to report a special address, even if it is not the target of an uninferable discontinuity. There are two ways to notify and report a special address:

1. The debug trigger unit matched, and the action is 4
2. The filter comparator matched

When a notification is requested, the encoder will report a format 2 or format 1 with an address packet, determined by the `branch_map`. If `branch_map` is 0, report format 2, otherwise report format 1. The notify field

shows whether the packet is reporting a notification address. If it is the target of an uninferable discontinuity, even if it is a notification, the notify field will not be asserted, otherwise the decoder cannot reconstruct the instruction stream correctly.

1.8.6 Decode Data Packets

- Find the first address to decode
 - Read the [TRACE_MEM_FULL_INTR_RAW](#) bit of the [TRACE_INTR_RAW_REG](#) register to know if the trace memory is full
 - * if read 0, read the trace packets from [TRACE_MEM_START_ADDR_REG](#)
 - * if read 1, and the loop mode is enabled, then the old trace packets are overwritten. In this case, read the [TRACE_MEM_CURRENT_ADDR_REG](#) to know the last writing address, and use this address as the first address to decode
- Use the decoder to decode data packets
 - The decoder reads all data packets starting from the first address, and reconstructs the data stream with the binary file
 - As mentioned in [1.6](#), the encoder writes 14 zero bytes to the memory partition boundary every time when 128 packets are transmitted. Given this fact, the first non-zero byte after 14 zero bytes should be the header of a new packet

1.8.7 AHB Configuration

ESP32-C5 encoder supports the AHB bus. There are some optional configurations to control the transmission bandwidth of the encoder.

If there are many DMA masters, users can increase the bandwidth of Trace by setting AHB burst [TRACE_AHB_CONFIG_REG](#).

- The [TRACE_HBURST](#) supports the following configurations:
 - 0x0: SINGLE
 - 0x1: INCR (length not defined)
 - 0x2: INCR4
 - 0x4: INCR8
 - Others: reserved

Burst transfer means that once arbitration is successful, multiple words can be transferred continuously. For SINGLE, the length is 1; for INCR, the length is undefined; for INCR4, the length is 4; for INCR8, the length is 8. The longer the burst length, the better the bandwidth.

- When configured as INCR transfer, since INCR is a burst of indeterminate length, in order to avoid Trace occupying the bus for a long time, it will end the INCR transfer after the transfer length up to [TRACE_MAX_INCR](#), and restart bus arbitration.

This is the configuration used to adjust the bandwidth. When the bandwidth is sufficient, it's not required to do the configuration, and users can use the default configuration.

1.8.8 Software Retention

The `TRACE_WORK_STATUS` represents the state of the encoder. When the encoder is not in the IDLE state, it must backup and restore the configuration when the chip enters sleep.

1.9 Register Summary

The addresses in this section are relative to RISC-V Trace Encoder base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

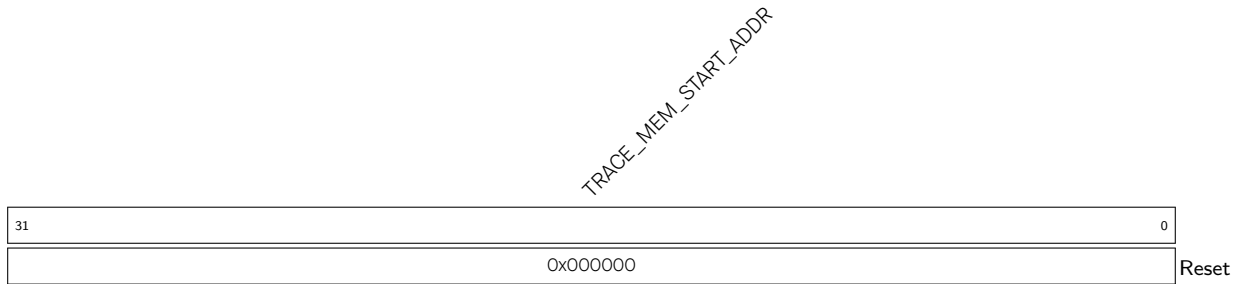
Name	Description	Address	Access
Memory configuration registers			
TRACE_MEM_START_ADDR_REG	Memory start address	0x0000	R/W
TRACE_MEM_END_ADDR_REG	Memory end address	0x0004	R/W
TRACE_MEM_CURRENT_ADDR_REG	Memory current address	0x0008	RO
TRACE_MEM_ADDR_UPDATE_REG	Memory address update	0x000C	WT
Trace FIFO status register			
TRACE_FIFO_STATUS_REG	FIFO status register	0x0010	RO
Interrupt registers			
TRACE_INTR_ENA_REG	Interrupt enable register	0x0014	R/W
TRACE_INTR_RAW_REG	Interrupt raw status register	0x0018	RO
TRACE_INTR_CLR_REG	Interrupt clear register	0x001C	WT
Trace configuration registers			
TRACE_TRIGGER_REG	Trace trigger register	0x0020	varies
TRACE_CONFIG_REG	Trace configuration register	0x0024	R/W
TRACE_FILTER_CONTROL_REG	Filter control register	0x0028	R/W
TRACE_FILTER_MATCH_CONTROL_REG	Filter match control register	0x002C	R/W
TRACE_FILTER_COMPARATOR_CONTROL_REG	Filter comparator match control register	0x0030	R/W
TRACE_FILTER_P_COMPARATOR_MATCH_REG	Primary comparator match value register	0x0034	R/W
TRACE_FILTER_S_COMPARATOR_MATCH_REG	Secondary comparator match value register	0x0038	R/W
TRACE_RESYNC_PROLONGED_REG	Resynchronization configuration register	0x003C	R/W
TRACE_AHB_CONFIG_REG	AHB configuration register	0x0040	R/W
Clock gate control and configuration register			
TRACE_CLOCK_GATE_REG	Clock gating control register	0x0044	R/W
Version register			
TRACE_DATE_REG	Version control register	0x03FC	R/W

1.10 Registers

The addresses in this section are relative to RISC-V Trace Encoder base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

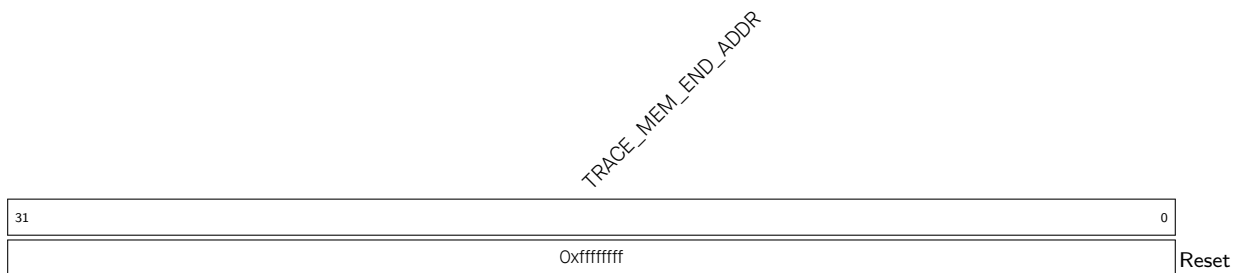
For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 1.1. TRACE_MEM_START_ADDR_REG (0x0000)



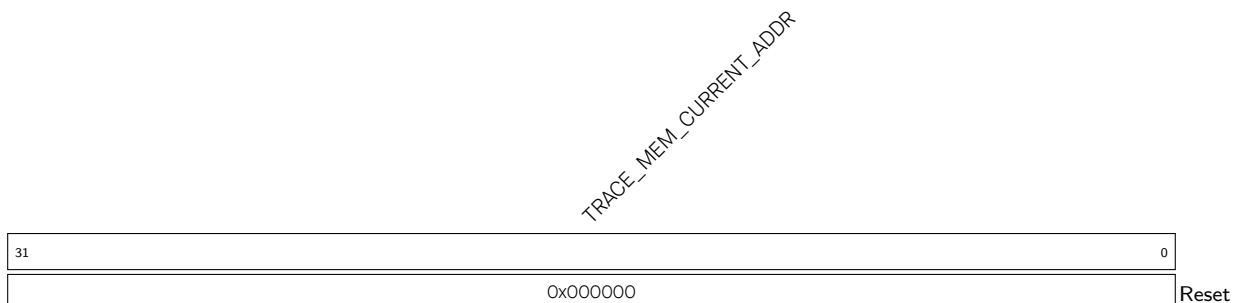
TRACE_MEM_START_ADDR Configures the start address of the trace memory. (R/W)

Register 1.2. TRACE_MEM_END_ADDR_REG (0x0004)



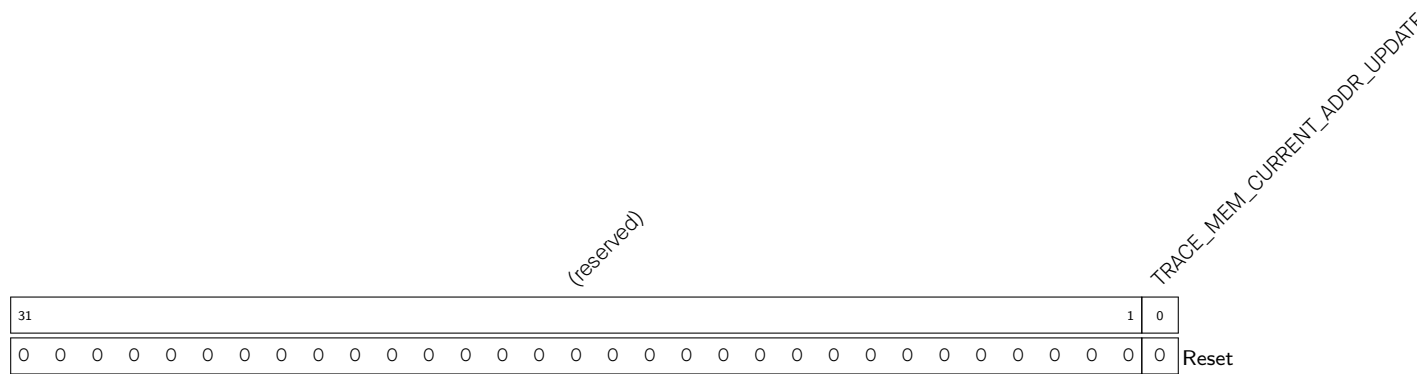
TRACE_MEM_END_ADDR Configures the end address of the trace memory. (R/W)

Register 1.3. TRACE_MEM_CURRENT_ADDR_REG (0x0008)



TRACE_MEM_CURRENT_ADDR Represents the current memory address for writing. (RO)

Register 1.4. TRACE_MEM_ADDR_UPDATE_REG (0x000C)



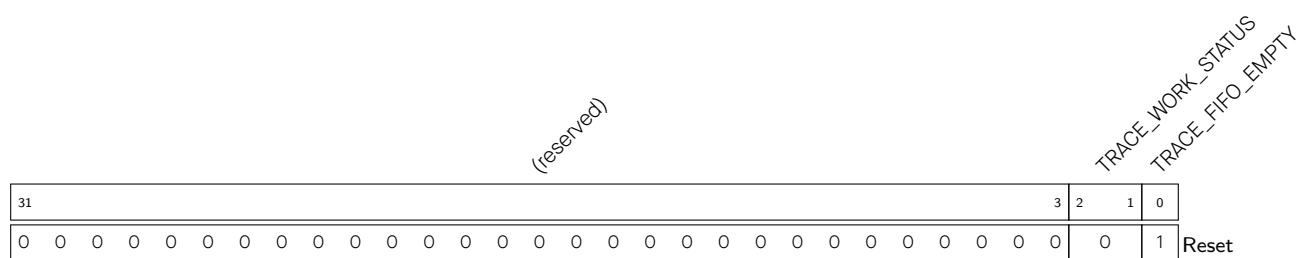
TRACE_MEM_CURRENT_ADDR_UPDATE Configures whether to update the value of TRACE_MEM_CURRENT_ADDR to TRACE_MEM_START_ADDR.

0: Not update

1: Update

(WT)

Register 1.5. TRACE_FIFO_STATUS_REG (0x0010)



TRACE_FIFO_EMPTY Represents whether the FIFO is empty.

1: Empty

0: Not empty (RO)

TRACE_WORK_STATUS Represents the state of the encoder:

0: Idle state

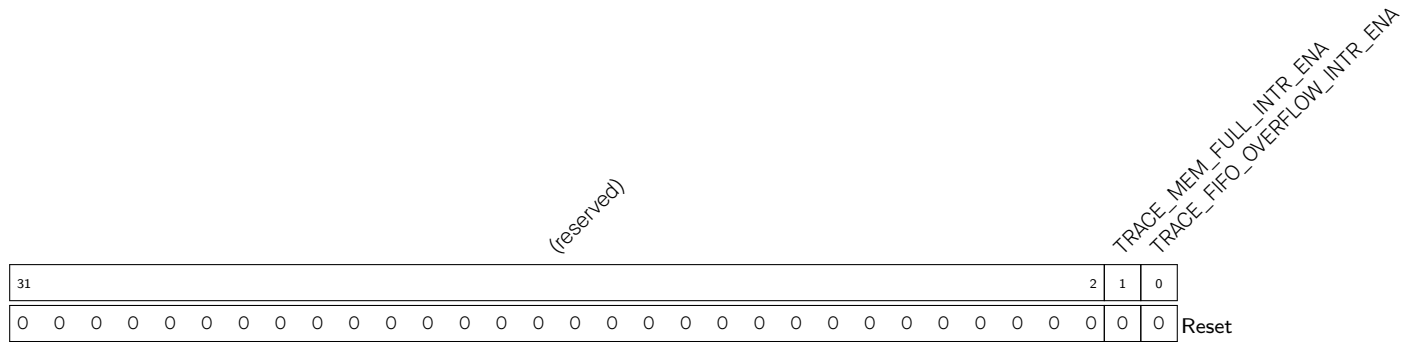
1: Working state

2: Wait state because the hart is halted or in reset

3: Lost state

(RO)

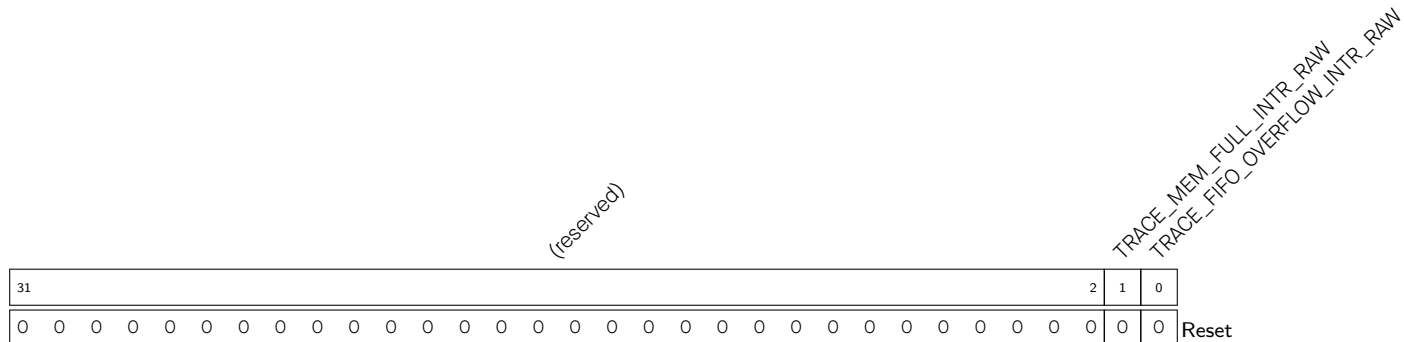
Register 1.6. TRACE_INTR_ENA_REG (0x0014)



TRACE_FIFO_OVERFLOW_INTR_ENA Write 1 to enable TRACE_FIFO_OVERFLOW_INTR. (R/W)

TRACE_MEM_FULL_INTR_ENA Write 1 to enable TRACE_MEM_FULL_INTR. (R/W)

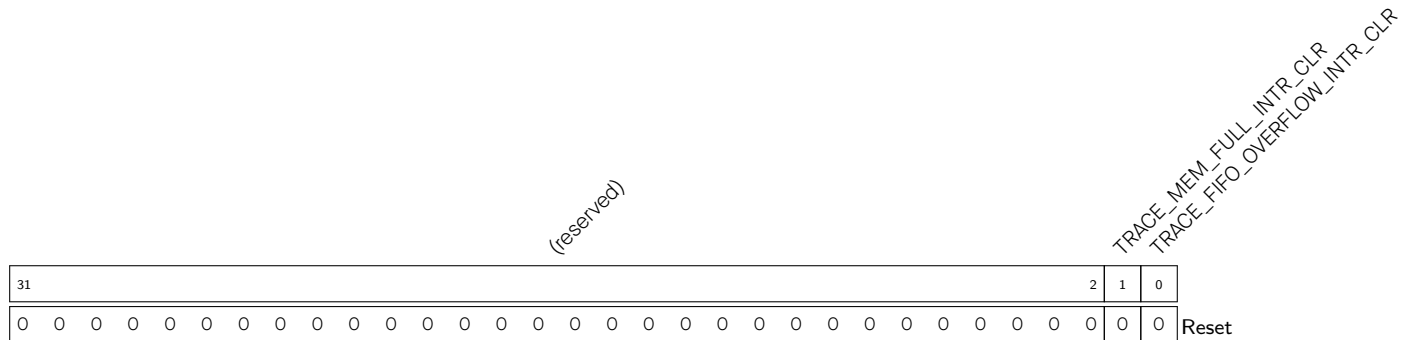
Register 1.7. TRACE_INTR_RAW_REG (0x0018)



TRACE_FIFO_OVERFLOW_INTR_RAW The raw interrupt status of TRACE_FIFO_OVERFLOW_INTR.
(RO)

TRACE_MEM_FULL_INTR_RAW	The raw interrupt status of TRACE_MEM_FULL_INTR. (RO)

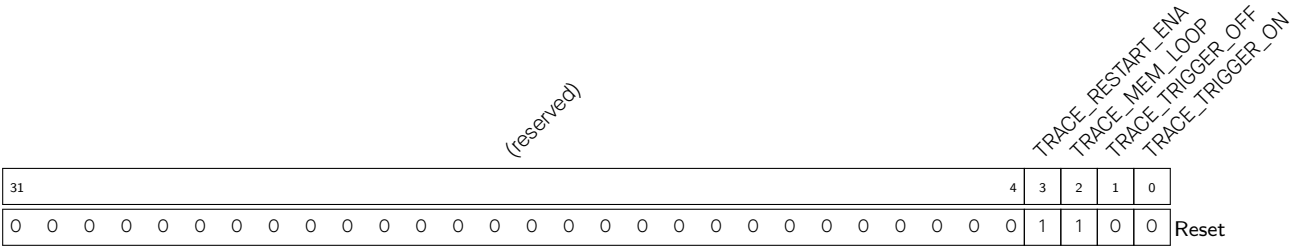
Register 1.8. TRACE_INTR_CLR_REG (0x001C)



TRACE_FIFO_OVERFLOW_INTR_CLR Write 1 to clear TRACE_FIFO_OVERFLOW_INTR. (WT)

TRACE_MEM_FULL_INTR_CLR Write 1 to clear TRACE_MEM_FULL_INTR. (WT)

Register 1.9. TRACE_TRIGGER_REG (0x0020)



TRACE_TRIGGER_ON Configures whether to enable the encoder.

- 0: Invalid
- 1: Enable

(WT)

TRACE_TRIGGER_OFF Configures whether to disable the encoder.

- 0: Invalid
- 1: Disable

(WT)

TRACE_MEM_LOOP Configures the memory writing mode.

- 0: Non-loop mode
- 1: Loop mode

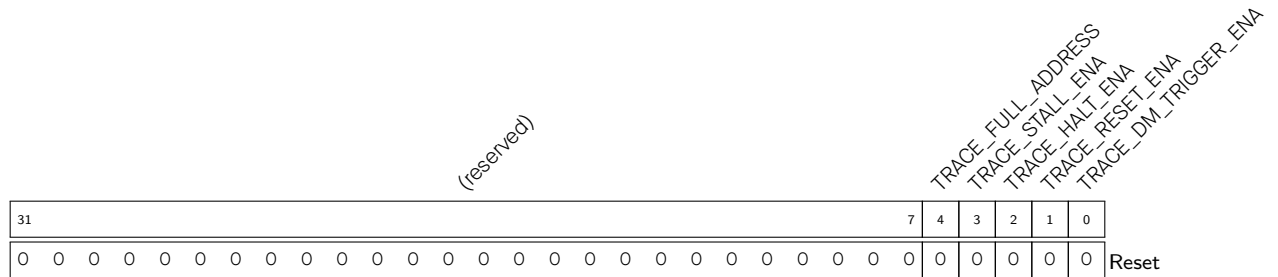
(R/W)

TRACE_RESTART_ENA Configures whether to enable automatic restart.

- 0: Disable
- 1: Enable

(R/W)

Register 1.10. TRACE_CONFIG_REG (0x0024)



TRACE_DM_TRIGGER_ENA Configure whether to enable the trigger signal.

0: Disable

1: Enable

(R/W)

TRACE_RESET_ENA Configures whether to enable the reset signal.

0: Disable

1: Enable

(R/W)

TRACE_HALT_ENA Configures whether to enable the halted signal.

0: Disable

1: Enable

(R/W)

TRACE_STALL_ENA Configures whether to enable the stall signal.

0: Disable

1: Enable

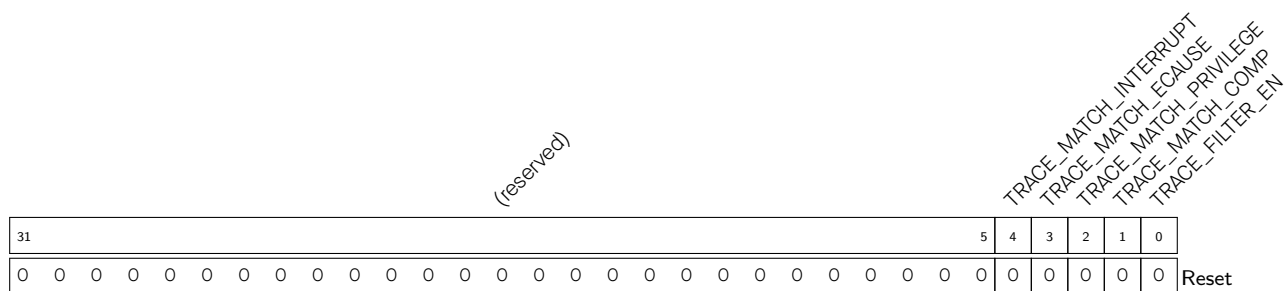
(R/W)

TRACE_FULL_ADDRESS Configure the address mode.

0: Delta address mode

1: Full address mode

Register 1.11. TRACE_FILTER_CONTROL_REG (0x0028)



TRACE_FILTER_EN Configure whether to enable filtering.

0: Disable

1: Enable

(R/W)

TRACE_MATCH_COMP Configures whether to enable the comparator match mode.

0: Disable

1: Enable

(R/W)

TRACE_MATCH_PRIVILEGE Configures whether to enable the privilege match mode.

0: Disable

1: Enable

(R/W)

TRACE_MATCH_ECAUSE Configures whether to enable the ecause match mode.

0: Disable

1: Enable

(R/W)

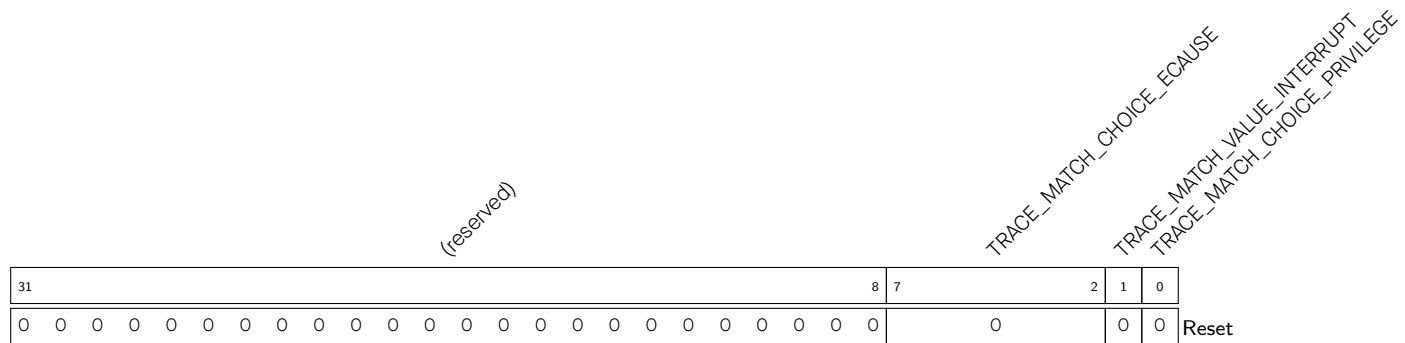
TRACE_MATCH_INTERRUPT Configures whether to enable the interrupt match mode.

0: Disable

1: Enable

(R/W)

Register 1.12. TRACE_FILTER_MATCH_CONTROL_REG (0x002C)



TRACE_MATCH_CHOICE_PRIVILEGE Configures the privilege level for matching.

0: User mode

1: Machine mode

(R/W)

TRACE_MATCH_VALUE_INTERRUPT Configures the interrupt level for matching. Valid only when TRACE_MATCH_INTERRUPT is set.

0: itype=1

1: itype=2

(R/W)

TRACE_MATCH_CHOICE_ECAUSE Configures the ecause code for matching. (R/W)

Register 1.13. TRACE_FILTER_COMPARATOR_CONTROL_REG (0x0030)

(reserved)														TRACE_MATCH_MODE		(reserved)		TRACE_S_NOTIFY		TRACE_S_FUNCTION		(reserved)		TRACE_S_INPUT		(reserved)		TRACE_P_NOTIFY		TRACE_P_FUNCTION		(reserved)		TRACE_P_INPUT	
31														18	17	16	15	14	13	12	10		9	8	7	6	5	4	2		1	0			
0 0 0 0 0 0 0 0 0 0 0 0 0 0														0	0	0	0	0		0	0	0	0	0	0		0	0	Reset						

TRACE_P_INPUT Configures the input of the primary comparator for matching.

0: iaddr

1: tval

(R/W)

TRACE_P_FUNCTION Configures the function for the primary comparator.

0: Equal

1: Not equal

2: Less than

3: Less than or equal

4: Greater than

5: Greater than or equal

Others: Always match

(R/W)

TRACE_P_NOTIFY Configures whether to explicitly report an instruction address matched against the primary comparator.

0: Not report

1: Report

(R/W)

TRACE_S_INPUT Configures the input of the secondary comparator for matching.

0: iaddr

1: tval

(R/W)

TRACE_S_FUNCTION Configures the function of for secondary comparator.

0: Equal

1: Not equal

2: Less than

3: Less than or equal

4: Greater than

5: Greater than or equal

6: The second comparator value is the masked value of the primary comparator

Others: Always match

(R/W)

Continued on the next page...

Register 1.13. TRACE_FILTER_COMPARATOR_CONTROL_REG (0x0030)

Continued from the previous page...

TRACE_S_NOTIFY Configures whether to explicitly report an instruction address matched against the secondary comparator.

0: Not report

1: Report

(R/W)

TRACE_MATCH_MODE Configures the comparator match condition.

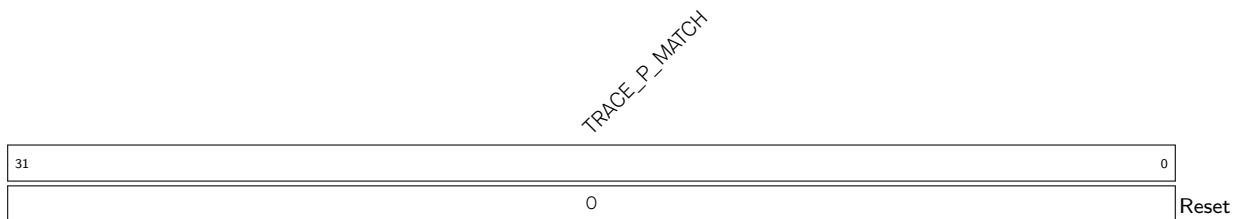
0: Only the primary comparator matches

1: Both primary and secondary comparators match (P&&S)

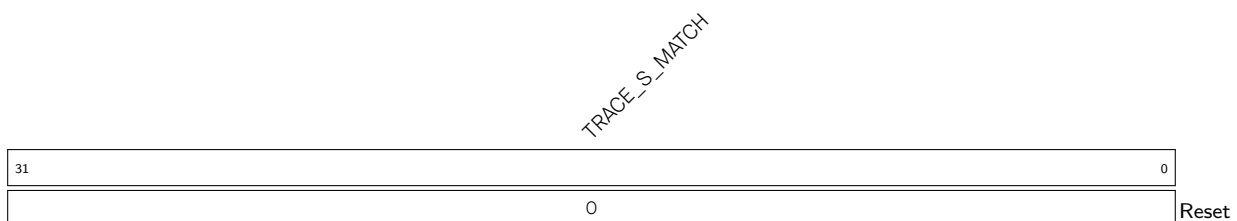
2: Neither primary nor secondary comparator match (!P&&S)

3: Start filtering when the primary comparator matches and stop filtering when the secondary comparator matches

(R/W)

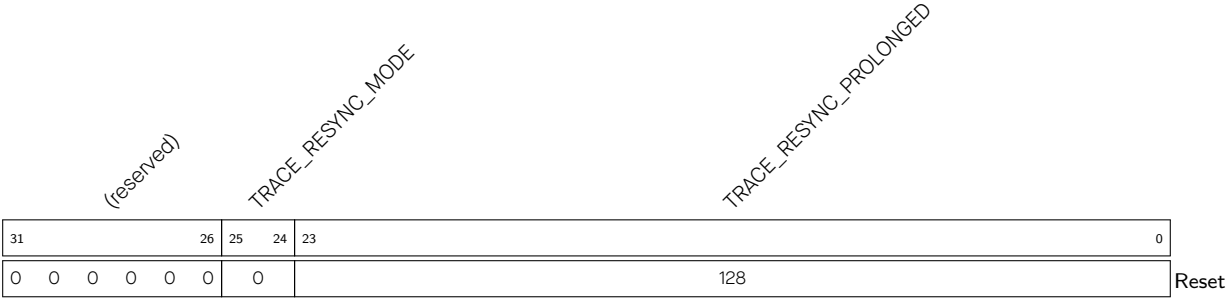
Register 1.14. TRACE_FILTER_P_COMPARATOR_MATCH_REG (0x0034)

TRACE_P_MATCH Configures the match value for the primary comparator. (R/W)

Register 1.15. TRACE_FILTER_S_COMPARATOR_MATCH_REG (0x0038)

TRACE_S_MATCH Configures the match value for the secondary comparator. (R/W)

Register 1.16. TRACE_RESYNC_PROLONGED_REG (0x003C)

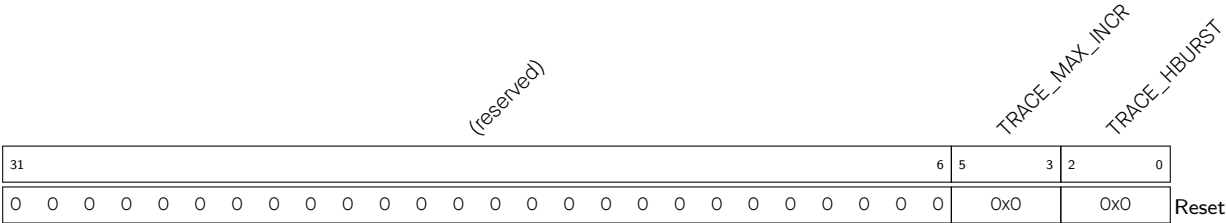


TRACE_RESYNC_PROLONGED Configures the threshold for the synchronization counter. (R/W)

TRACE_RESYNC_MODE Configures the synchronization mode.

- 0: Disable the synchronization counter
- 1: Invalid
- 2: Synchronization counter counts by packet
- 3: Synchronization counter counts by cycle
- (R/W)

Register 1.17. TRACE_AHB_CONFIG_REG (0x0040)



TRACE_HBURST Configures the AHB burst mode.

- 0: SINGLE
- 1: INCR (length not defined)
- 2: INCR4
- 4: INCR8
- Others: Invalid
- (R/W)

TRACE_MAX_INCR Configures the maximum burst length for INCR mode. (R/W)

Register 1.18. TRACE_CLOCK_GATE_REG (0x0044)

(reserved)																																TRACE_CLK_EN		
31																															1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

TRACE_CLK_EN Configures register clock gating. 0: Support clock only when the application writes registers to save power
1: Always force the clock on for registers
This bit doesn't affect register access
(R/W)

Register 1.19. TRACE_DATE_REG (0x03FC)

(reserved)				TRACE_DATE																										
31			28	27																									0	
0	0	0	0	0x2211300																										Reset

TRACE_DATE Version control register. (R/W)

Chapter 2

Low-Power CPU

The ESP32-C5 Low-Power CPU (LP CPU) is a 32-bit processor based upon RISC-V ISA comprising integer (I), multiplication/division (M), atomic (A), and compressed (C) standard extensions. It features ultra-low power consumption and has a 2-stage, in-order, and scalar pipeline. The LP CPU core complex has an interrupt controller (INTC), a debug module (DM), and system bus (SYS BUS) interfaces for memory and peripheral access.

The LP CPU is in sleep mode by default (see Section 2.9). It can stay powered on when the chip enters Deep-sleep mode (see Chapter 4 [Low-power Management \[to be added later\]](#) for details) and can access most peripherals and memories (see Chapter 4 [System and Memory](#) for details). It has two application scenarios:

- Power insensitive scenario: When the High-Performance CPU (HP CPU) is active, the LP CPU can assist the HP CPU with some speed- and efficiency-insensitive controls and computations.
- Power sensitive scenario: When the HP CPU is in the power-down state to save power, the LP CPU can be woken up to handle some external wake-up events.

Figure 2.0-1 shows the resources accessible to the HP CPU and the LP CPU.

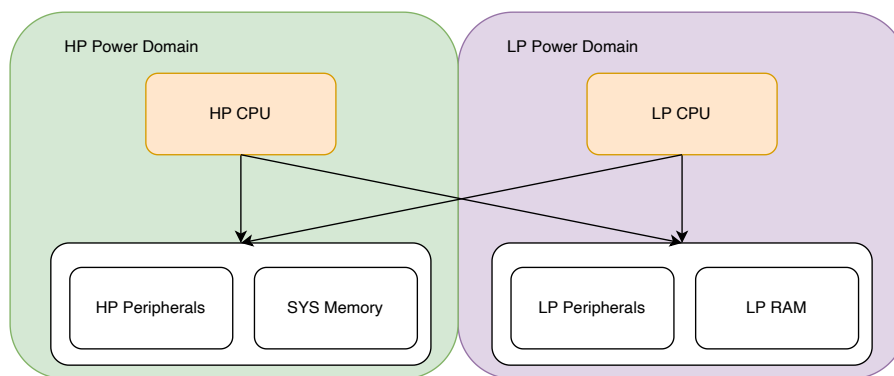


Figure 2.0-1. LP CPU Overview

In the figure above, the HP Power Domain and LP Power Domain can be powered on and off independently. When one domain is powered down, the CPU of the other domain cannot access the resources of that domain. In other words, only when the domain is powered on can its resources be accessed. For more information about power, please refer to the chapter 4 [Low-power Management \[to be added later\]](#).

2.1 Features

The LP CPU has the following features:

- Operating clock frequency up to 20 MHz

- One vector interrupt input
- Debug module compliant with RISC-V External Debug Support Version 0.13 with external debugger support over an industry-standard JTAG/USB port
- Hardware trigger compliant with RISC-V External Debug Support Version 0.13 with up to 2 breakpoints/watchpoints
- 32-bit AHB system bus for peripheral and memory access
- Core performance metric events
- Able to wake up the HP CPU and send an interrupt to it
- Access to HP memory and LP memory
- Access to the entire peripheral address space

2.2 Configuration and Status Registers (CSRs)

2.2.1 Register Summary

Below is a list of CSRs available to the CPU. Except for the custom performance counter CSRs, all the implemented CSRs follow the standard mapping of bit fields as described in the RISC-V Instruction Set Manual, Volume II: Privileged Architecture, Version 1.10. It must be noted that even among the standard CSRs, not all bit fields have been implemented, limited by the subset of features implemented in the CPU. Refer to the next section for a detailed description of the subset of fields implemented under each of these CSRs.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Machine Information CSR			
mhartid	Machine Hart ID	0xF14	RO
Machine Trap Setup CSRs			
mstatus	Machine Mode Status	0x300	R/W
misa ¹	Machine ISA	0x301	R/W
mie	Machine Interrupt Enable	0x304	R/W
mtvec ²	Machine Trap Vector	0x305	R/W
Machine Trap Handling CSRs			
mscratch	Machine Scratch	0x340	R/W
mepc	Machine Trap Program Counter	0x341	R/W
mcause ³	Machine Trap Cause	0x342	R/W
mtval	Machine Trap Value	0x343	R/W
mip	Machine Interrupt Pending	0x344	R/W
Trigger Module CSRs (shared with Debug Mode)			
tselect	Trigger Select Register	0x7A0	R/W
tdata1	Trigger Abstract Data 1	0x7A1	R/W

¹Although [misa](#) is specified as having both read and write access (R/W), its fields are hardwired and thus write has no effect. This is what would be termed WARL (Write Any Read Legal) in RISC-V terminology.

²[mtvec](#) only provides configuration for trap handling in vectored mode with the base address aligned to 256 bytes.

³External interrupt IDs reflected in [mcause](#) include even those IDs which have been reserved by RISC-V standard for core internal sources.

Name	Description	Address	Access
tdata2	Trigger Abstract Data 2	0x7A2	R/W
Debug Mode CSRs			
dcsr	Debug Control and Status	0x7B0	R/W
dpc	Debug PC	0x7B1	R/W
dscratch0	Debug Scratch Register 0	0x7B2	R/W
dscratch1	Debug Scratch Register 1	0x7B3	R/W
Machine Counter/Timer CSRs			
mcycle	Machine Clock Cycle Counter	0xB00	R/W
minstret	Machine Retired Instruction Counter	0xB02	R/W
mhpmcounter$n(n:3-12)$	Machine Performance Monitor Counter	0xB00+ n	R/W
mcycleh	The higher 32 bits of mcycle	0xB80	R/W
minstreth	The higher 32 bits of minstret	0xB82	R/W
mhpmcounter$nh(n:3-12)$	The higher 32 bits of mhpmcounter$n(n:3-12)$	0xB80+ n	R/W
Machine Counter Setup CSR			
mcountinhibit	Machine Counter Control	0x320	R/W

Note that if write, set, or clear operation is attempted on any of the read-only (RO) CSRs indicated in the above table, the CPU will generate an illegal instruction exception.

2.2.2 Registers

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 2.1. mhartid (0xF14)

MHARTID	
31	0
0x00000000	
Reset	

MHARTID Represents Hart ID. The LP CPU hart ID is 0. (RO)

Register 2.2. mstatus (0x300)

(reserved)										reserved				(reserved)				MPP		(reserved)		MPIE		(reserved)		MIE		(reserved)						
31											22	21	20					13	12	11	10			8	7	6			4	3	2			0
0x000										0		0x00				0x0		0x0		0		0x0		0		0x0		Reset						

MIE Write 1 to enable the global machine mode interrupt. (R/W)

MPIE Write 1 to enable the machine previous interrupt (before trap). (R/W)

MPP Configures machine previous privilege mode (before trap).

0x3: Machine mode

Other values: Invalid

Note: Only the lower bit is writable. Any write to the higher bit is ignored as it is directly tied to the lower bit.

(R/W)

Register 2.3. misa (0x301)

MXL		(reserved)				Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1		0x0				0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1

Reset

MXL Machine XLEN = 1 (32-bit). (RO)**Z** Reserved = 0. (RO)**Y** Reserved = 0. (RO)**X** Non-standard extensions present = 0. (RO)**W** Reserved = 0. (RO)**V** Reserved = 0. (RO)**U** User mode implemented = 0. (RO)**T** Reserved = 0. (RO)**S** Supervisor mode implemented = 0. (RO)**R** Reserved = 0. (RO)**Q** Quad-precision floating-point extension = 0. (RO)**P** Reserved = 0. (RO)**O** Reserved = 0. (RO)**N** User-level interrupts supported = 0. (RO)**M** Integer Multiply/Divide extension = 1. (RO)**L** Reserved = 0. (RO)**K** Reserved = 0. (RO)**J** Reserved = 0. (RO)**I** RV32I base ISA = 1. (RO)**H** Hypervisor extension = 0. (RO)**G** Additional standard extensions present = 0. (RO)**F** Single-precision floating-point extension = 0. (RO)**E** RV32E base ISA = 0. (RO)**D** Double-precision floating-point extension = 0. (RO)**C** Compressed Extension = 1. (RO)**B** Reserved = 0. (RO)**A** Atomic Standard Extension = 1. (RO)

Register 2.4. mie (0x304)

(reserved)		(reserved)	
31	30	29	0
0x0	0x0	0x0	
		Reset	

IE Write 1 to enable the interrupt. (R/W)

Register 2.5. mtvec (0x305)

BASE										(reserved)				MODE	
31								8	7	2		1	0		
0x000000								0x00		0x1		Reset			

MODE Represents whether machine mode interrupts are vectored. Only vectored mode **0x1** is available. (RO)

BASE Configures the higher 24 bits of trap vector base address aligned to 256 bytes. (R/W)

Register 2.6. mscratch (0x340)

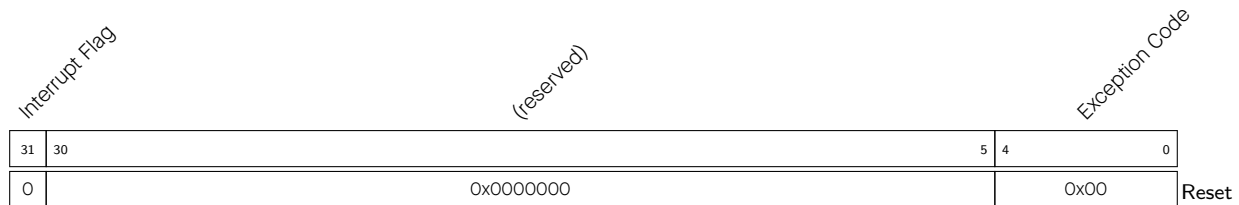
MSCRATCH	
31	0
0x00000000	
Reset	

MSCRATCH Contains machine scratch information for custom use. (R/W)

Register 2.7. mepc (0x341)

MEPC	
31	0
0x00000000	
Reset	

MEPC Configures the machine trap/exception program counter. This is automatically updated with address of the instruction which was about to be executed while CPU encountered the most recent trap. (R/W)

Register 2.8. mcause (0x342)

Exception Code This field is automatically updated with unique ID of the most recent exception or interrupt due to which CPU entered trap. Possible exception IDs are:

0x2: Illegal instruction

0x3: Hardware breakpoint/watchpoint or EBREAK

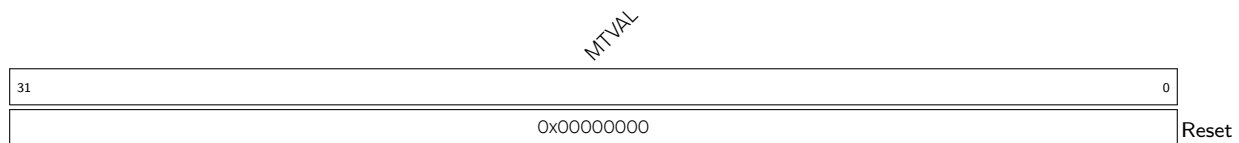
0x6: Misaligned atomic instructions

Note: Exception ID 0x0 (instruction access misaligned) is not present because CPU always masks the lowest bit of the address during instruction fetch.

(R/W)

Interrupt Flag This flag is automatically updated when CPU enters trap. If this is found to be set, it indicates that the latest trap occurred due to an interrupt. For exceptions it remains unset.

(R/W)

Register 2.9. mtval (0x343)

MTVAL Configures machine trap value. This is automatically updated with an exception dependent data which may be useful for handling that exception. Data is to be interpreted depending upon exception IDs:

0x1: Faulting address of instruction

0x2: Faulting instruction opcode

0x5: Faulting data address of load operation

0x7: Faulting data address of store operation

Note: The value of this register is not valid for other exception IDs and interrupts.

(R/W)

Register 2.10. mip (0x344)

(reserved)		(reserved)	
31	30	29	0
0x0	0x0	0x0	
			Reset

IP Configures the pending status of the interrupt.

0: Not pending

1: Pending

(R/W)

Register 2.11. mcycle (0xB00)

mcycle	
31	0
0x0	
Reset	

MCYCLE Configures the lower 32 bits of the clock cycle counter. (R/W)

Register 2.12. minstret (0xB02)

minstret	
31	0
0x0	
Reset	

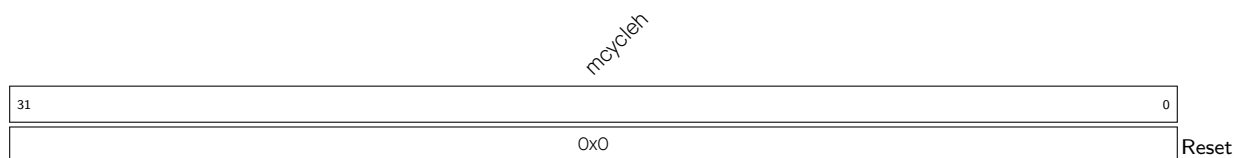
MINSTRET Configures the lower 32 bits of the instruction counter. (R/W)

Register 2.13. mhpmcounter_{*n*} (*n*: 3-12) (0xB00+*n*)

mhpmcounter _{<i>n</i>}	
31	0
0x0	
Reset	

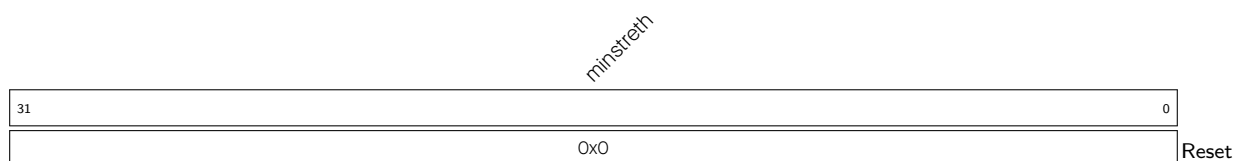
MHPMCOUNTER_{*n*} Configures the lower 32 bits of the performance counter *n*. (R/W)

Register 2.14. mcycleh (0xB80)

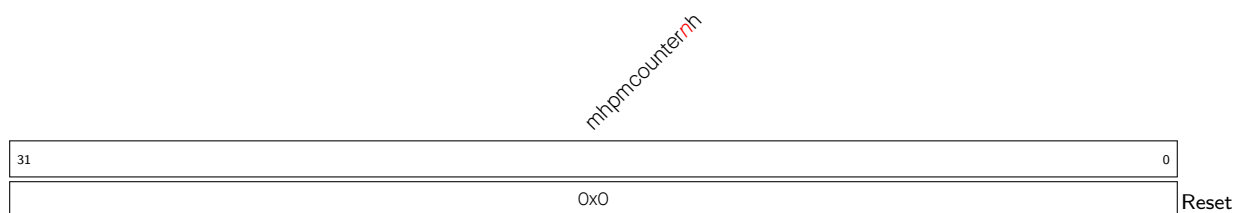


MCYCLEH Configures the higher 32 bits of the clock cycle counter. (R/W)

Register 2.15. minstreth (0xB82)



MINSTRETH Configures the higher 32 bits of the instruction counter. (R/W)

Register 2.16. mhpmpcounter n (n : 3-12)h (0xB80+ n)

MHPMCOUNTER n h Configures the higher 32 bits of the performance counter n . (R/W)

Register 2.17. mcouninhibit (0x320)

HPM			IR	(reserved) CY	
31	3	2	1	0	
0x0			0x0	0x0	0x0
			Reset		

CY Configure whether the clock cycle counter increments.

0: The counter does not count

1: The counter increments

(R/W)

IR Configure whether the instruction counter increments.

0: The counter does not count

1: The counter increments

(R/W)

HPM Configures whether the performance counter $n(n:3-12)$ increments.

0: The counter does not count

1: The counter increments

(R/W)

2.3 Interrupts and Exceptions

The LP CPU handles interrupts and exceptions according to RISC-V Instruction Set Manual, Volume II: Privileged Architecture, Version 1.10. After entering an interrupt/exception handler, the CPU:

- Saves the current program counter (PC) value to the [mepc](#) CSR
- Copies the state of [MIE](#) of [mstatus](#) to [MPIE](#) of [mstatus](#)
- Saves the current privileged mode to [MPP](#) of [mstatus](#)
- Clears [MIE](#) of [mstatus](#)
- Toggles the privileged mode to machine mode (M mode)
- Jumps to the handler address
 - For exceptions, the handler address is the base address of the vector table in the [mtvec](#) CSR.
 - For interrupts, the handler address is $mtvec + 4 * 30$.
- After the mret instruction is executed, the core jumps to the PC saved in the [mepc](#) CSR and restores the value of [MPIE](#) of [mstatus](#) to [MIE](#) of [mstatus](#) and the privileged mode to that configured by [MPP](#).

When the core starts up, the base address of the vector table is initialized to the boot address 0x50000000. After startup, the base address can be changed by writing to the [mtvec](#) CSR. For more information about CSRs, see Section [2.2.1](#).

The core fetches instructions from address 0x50000080 after reset.

2.3.1 Interrupts

The ESP32-C5 LP CPU supports only one interrupt entry, to which all interrupt events jump. The LP CPU supports the following peripheral interrupt sources:

- [Power Management Unit \(PMU\)](#)
- [Low-Power Timer \(RTC_TIMER\)](#)
- [Low-Power UART \(LP_UART\)](#)
- [Low-Power I2C \(LP_I2C\)](#)
- [Low-Power IO MUX \(LP IO MUX\)](#)

For more information on those peripheral interrupts, please refer to the corresponding chapter.

2.3.2 Interrupt Handling

By default, interrupts are disabled globally because the [MIE](#) bit in [mstatus](#) has a reset value of 0. Software must set this bit to enable global interrupts.

1. Enable interrupts
 - To enable interrupts globally, set the [MIE](#) bit of [mstatus](#).
 - To enable Interrupt 30, set the 30th bit of [mie](#) CSR.
2. After interrupts are enabled, the LP CPU can respond to interrupts. It also needs to configure interrupts of the peripherals so that they can send an interrupt signal to the LP CPU.
3. After the interrupt is triggered, the LP CPU jumps to $mtvec + 4 * 30$.
4. After entering the interrupt handler, users need to read [LPPERI_INTERRUPT_SOURCE_REG](#) to get the peripheral that triggered the interrupt and process the interrupt. Note that if the interrupts are triggered by multiple peripherals, the CPU will process them one by one in sequence until none is left. If not all interrupts are processed, the CPU will enter the interrupt handler again.
5. To clear interrupts, just clear the interrupt signal of the peripheral.

2.3.3 Exceptions

The LP CPU supports the RISC-V standard exceptions and can trigger the following exceptions:

Table 2.3-1. LP CPU Exception Causes

Exception ID	Description
2	Illegal instructions
3	Breakpoints (EBREAK)
6	Misaligned atomic instructions

2.4 Debugging

This section describes how to debug and test the LP CPU. Debug support is provided through standard JTAG pins and complies with RISC-V External Debug Support Version 0.13.

For ESP32-C5 system debugging overview, please refer to Chapter 1 [High-Performance CPU \[to be added later\]](#) > Figure 8.

The user interacts with the Debug Host (e.g., laptop), which is running a debugger (e.g., gdb). The debugger communicates with a Debug Translator (e.g., OpenOCD, which may include a hardware driver) to communicate with Debug Transport Hardware (e.g., ESP-Prog adapter). The Debug Transport Hardware connects the Debug Host to the CPU's Debug Transport Module (DTM) through a standard JTAG interface. The DTM provides access to the debug module (DM) using the Debug Module Interface (DMI).

ESP32-C5 features two DTMs interconnected in a daisy-chain configuration. Specifically, TAP0 connects to the HP CPU, and TAP1 is dedicated to the LP CPU.

The HP DM can debug the HP CPU and the LP DM the LP CPU.

The LP CPU implements four registers for core debugging: [dcsr](#), [dpc](#), [dscratch0](#), and [dscratch1](#). All of those registers can only be accessed from debug mode. If software attempts to access them when the LP CPU is not in debug mode, an illegal instruction exception will be triggered.

2.4.1 Features

The Low-Power CPU has the following debugging features:

- Provides necessary information about the implementation to the debugger.
- Allows the CPU core to be halted and resumed.
- CPU core registers (including CSRs) can be read/written by the debugger.
- CPU core can be reset through the debugger.
- CPU can be halted on software breakpoint (planted breakpoint instruction).
- Hardware single-stepping.
- Two hardware triggers (which can be used as breakpoints/watchpoints). See Section 2.5 for details.

2.4.2 Functional Description

The debugging mechanism adheres to the specification RISC-V External Debug Support Version 0.13. For a detailed description of the debugging features, refer to the specification.

According to the specification, a hart can be in the following states: nonexistent, unavail, running, and halted. By default, the LP CPU is in the unavail state. To connect the LP CPU for debugging, users need to clear the state by configuring the [LPPERI_CPU_REG](#) register.

2.4.3 Register Summary

The following table lists the debug CSRs supported for the LP CPU.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
dcsr	Debug Control and Status	0x7B0	R/W
dpc	Debug PC	0x7B1	R/W
dscratch0	Debug Scratch Register 0	0x7B2	R/W
dscratch1	Debug Scratch Register 1	0x7B3	R/W

All debug module registers are implemented in accordance with the specification RISC-V External Debug Support Version 0.13. For more information, refer to the specification.

2.4.4 Registers

The following is a detailed description of the debug CSR supported by the LP CPU.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 2.18. dcsr (0x7B0)

xdebugver				(reserved)								ebreakm				(reserved)				ebreaku				(reserved)				cause				(reserved)				step		prv	
31	28	27										16	15	14	13	12	11		9	8		6	5		3	2	1	0											
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Reset

xdebugver Represents the debug version.

4: External debug support exists
(RO)

ebreakm Configures execution of the EBREAK instruction in machine mode.

0: Trigger an exception with mcause = 3
1: Enter debug mode
(R/W)

ebreaku Configures execution of the EBREAK instruction in user mode.

0: Trigger an exception with mcause = 3 as described in privileged mode
1: Enter debug mode
(R/W)

cause Represents the reason why debug mode was entered. When there are multiple reasons to enter debug mode in a single cycle, the cause with the highest priority number is the one written.

1: An EBREAK instruction was executed. (priority 3)
2: The Trigger Module caused a halt. (priority 4)
3: haltreq was set. (priority 2)
4: The CPU core single stepped because step was set. (priority 1)
Other values: Reserved for future use
(RO)

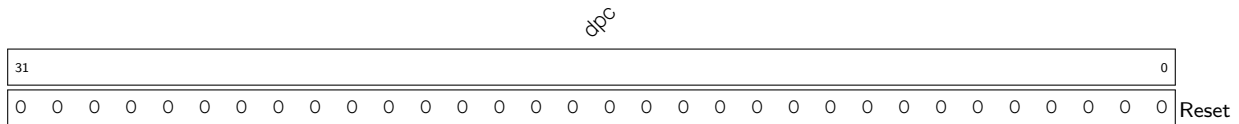
step When set and not in Debug Mode, the core will only execute a single instruction and then enter Debug Mode.

If the instruction does not complete due to an exception, the core will immediately enter Debug Mode before executing the trap handler, with appropriate exception registers set.

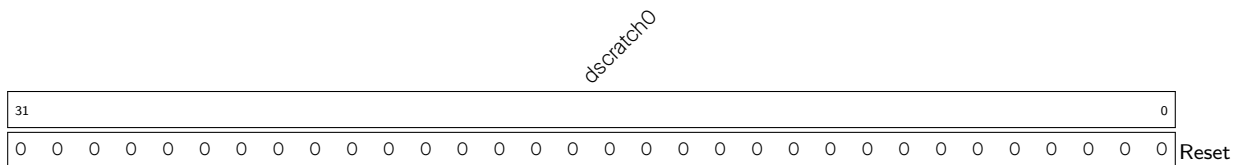
Setting this bit does not mask interrupts. This is a deviation from the RISC-V External Debug Support Specification Version 0.13.

(R/W)

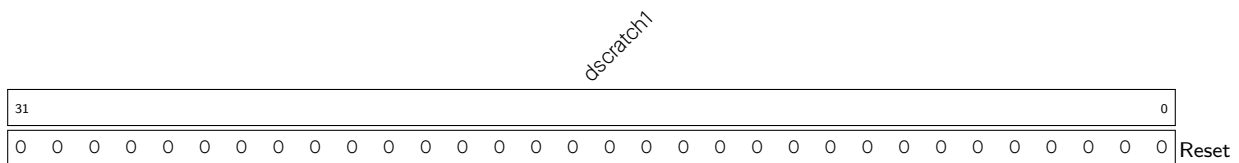
prv Contains the privilege level the core is operating in when debug mode is entered. A debugger can change this value to change the core's privilege level when exiting debug mode. Only **0x3** (machine mode) and **0x0** (user mode) are supported. (RO)

Register 2.19. dpc (0x7B1)

dpc Upon entry to debug mode, dpc is written with the address of the next instruction that will be executed. When resuming, the CPU core's PC is updated to the address stored in dpc. In debug mode, dpc can be modified. This field can be accessed in debug mode. (R/W)

Register 2.20. dscratch0 (0x7B2)

dscratch0 Used by the debug module internally. (R/W)

Register 2.21. dscratch1 (0x7B3)

dscratch1 Used by the debug module internally.(R/W)

2.5 Hardware Trigger

2.5.1 Features

Hardware Trigger module provides breakpoint and watchpoint capability for debugging. It has the following features:

- Two independent trigger units
- Configurable unit to match the address of the program counter
- Able to halt execution and transfer control to the debugger

2.5.2 Functional Description

The hardware trigger module provides three CSRs. See Section 2.5.4 for details. Among these, `tdata1` and `tdata2` are abstract CSRs, which means they are shadow registers for accessing internal registers in the trigger units, one at a time.

To select a specific trigger unit, the corresponding number (0-1) needs to be written to the [tselect](#) CSR. When a valid value is written, the abstract CSRs, [tdata1](#) and [tdata2](#), automatically match the internal registers of the trigger unit. Each trigger unit has two internal registers, namely [mcontrol](#) and [maddress](#), which are mapped to [tdata1](#) and [tdata2](#), respectively.

Writing a value more than 1 to [tselect](#) will set [tselect](#) to 1.

Since software or debugger may need to know the type of the selected trigger to correctly interpret [tdata1](#) and [tdata2](#), the 4-bit field (31-28) of [tdata1](#) encodes the type of the selected trigger. This type field is read-only and always provides a value of 0x2 for every trigger, which stands for support for address and data matching. Hence, it is inferred that [tdata1](#) and [tdata2](#) are to be interpreted as fields of [mcontrol](#) and [maddress](#), respectively. The specification RISC-V External Debug Support Version 0.13 provides information on other possible values, but the trigger module only supports the 0x2 type.

Once a trigger unit has been chosen by writing its index to [tselect](#), it will become possible to configure it by setting the appropriate bits in [mcontrol](#) CSR ([tdata1](#)) and writing the target address to [maddress](#) CSR ([tdata2](#)).

2.5.3 Trigger Execution Flow

When a hart is halted and enters debug mode due to the firing of a trigger ([action](#) = 1):

- [dpc](#) is set to the current PC in the decoding phase
- The cause field in [dcsr](#) is set to 2, which means halt due to trigger

2.5.4 Register Summary

Below is a list of Trigger Module CSRs supported by the CPU. These are only accessible from machine mode.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
tselect	Trigger Select Register	0x7A0	R/W
tdata1	Trigger Abstract Data 1	0x7A1	R/W
mcontrol	tdata1 Shadow Register	0x7A1	R/W
tdata2	Trigger Abstract Data 2	0x7A2	R/W

2.5.5 Registers

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 2.22. [tselect](#) (0x7A0)

tselect																																	
31 0																																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

tselect Configures the index of the selected trigger unit. (R/W)

Register 2.23. tdata1 (0x7A1)

type				dmode	data															
31	28	27	26		0															
0	0	1	0	1	0	x				1				0				4	0	Reset

type Represents the trigger type. This field is reserved since only match type (0x2) triggers are supported. (R0)

dmode This is set to 1 if a trigger is being used by the debugger. This field is reserved since it is only supported in debug mode. (RO)

data Configures the abstract tdata1 content. This will always be interpreted as fields of `mcontrol` since only match type (0x2) triggers are supported. (R/W)

Register 2.24. tdata2 (0x7A2)

tdata2 Configures the abstract tdata2 content. This will always be interpreted as **maddress** since only match type (0x2) triggers are supported. (R/W)

Register 2.25. mcontrol (0x7A1)

(reserved)				dmode	maskmax				hit	select	timing	sizelo	action		chain	match	m	(reserved)			s	u	execute	store	load		
31	28	27	26			21	20	19	18	17	16	15		12	11	10		7	6	5	4	3	2	1	0		
0x2				0	0x0				0	0	0	0	0001		0	0	0	0	0	0	0	0	0	0	0		
Reset																											

Reset

dmode Same as [dmode](#) in [tdata1](#). (RO)

maskmax Represents the maximum NAPOT range.

0: A byte. Only exact match is supported.

Other values: Not supported.

(RO)

hit Not implemented in hardware. This field remains 0. (RO)

select Configures to select between an address match or a data match.

0: Perform a match on the virtual address

1: Perform a match on the data value loaded or stored, or the instruction executed

Note: Only address match is implemented. This field remains 0.

(RO)

timing Configures when the trigger will take action.

0: Take action before the instruction is executed

1: Take action after the instruction is executed

Note: The field remains 0.

(RO)

sizelo Only match of any size is supported. This field remains 0. (RO)

action Configure action of the selected trigger after it is triggered.

0x0: Cause a breakpoint exception

0x1: Enter debug mode (Valid only when dmode = 1)

Note: Only entering debug mode is supported. This field remains 1.

(RO)

CHAIN Not implemented in hardware. This field remains 0. (RO)

match Configures the trigger to perform the matching operation of the lower data/instruction address.

0x0: Exact match. Namely, the address corresponding to a certain byte during the access must exactly match the value of [maddress](#).

0x1: NAPOT match. Namely, at least one byte during the access is in the NAPOT region specified in [maddress](#).

Note: Only exact byte match is supported. This field remains 0.

(R/W)

m Set this field to make the selected trigger operate in machine mode. (RO)

s Set this field to make the selected trigger operate in supervisor mode. Operation in supervisor mode is not supported. This field is always 0. (RO)

Continued on the next page...

Register 2.25. mcontrol (0x7A1)**Continued from the previous page...**

u Set this field to make the selected trigger operate in user mode. Operation in user mode is not supported. This field is always 0. (RO)

execute Configures whether to enable the selected trigger to match the virtual address of instructions.

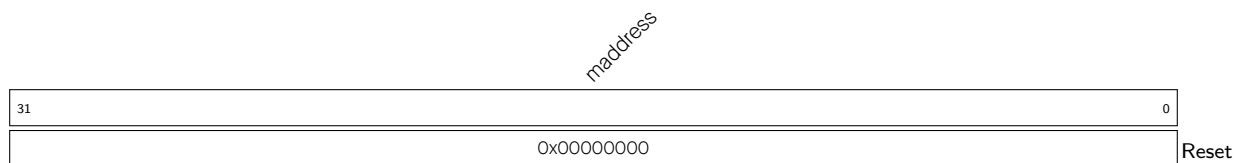
0: Not enable

1: Enable

(R/W)

store Set this field to make the selected trigger match the virtual address of the memory write operation. Not supported by hardware. This field is always 0. (RO)

load Set this field to make the selected trigger match the virtual address of a memory read operation. Not supported by hardware. This field is always 0. (RO)

Register 2.26. maddress (0x7A2)

maddress Configures the address used by the selected trigger when performing match operation. (R/W)

2.6 Performance Counter

The LP CPU implements a clock cycle counter [mcycle\(h\)](#), an instruction counter [minstret\(h\)](#), and 10 event counters [mhpmcounter \$n\$ \(\$n\$:3-12\)](#). The clock cycle counter and instruction counter are always available and each is 64-bit wide. Other performance counters are 40-bit wide each.

By default, all counters are enabled after reset. A counter can be enabled or disabled individually via the corresponding bit in the [mcountinhibit](#) CSR.

As shown in Table 2.6-1, each counter is dedicated to counting a particular event.

Table 2.6-1. Performance Counter

Counter	Counted Event
mcycle	Clock cycles
minstret	The number of instructions
mhpmcounter3	Wait cycles for memory access

Counter	Counted Event
mhpmcounter4	Wait cycles for fetching instructions
mhpmcounter5	The number of memory read operations. An unaligned read is counted as two.
mhpmcounter6	The number of memory write operations. An unaligned write is counted as two.
mhpmcounter7	The number of unconditional jump instructions (jal, jr, jalr)
mhpmcounter8	The number of branch instructions
mhpmcounter9	The number of taken branch instructions
mhpmcounter10	The number of compressed instructions
mhpmcounter11	Wait cycles for multiplication instructions
mhpmcounter12	Wait cycles for division instructions

2.7 System Access

2.7.1 Memory Access

The ESP32-C5 LP CPU can access LP SRAM and HP SRAM. For more information, please refer to Section 4 [System and Memory](#).

- LP SRAM: 16 KB starting from 0x5000_0000 to 0x5000_3FFF, where you can fetch instructions, read data, write data, etc.
- HP SRAM: 384 KB starting from 0x4080_0000 to 0x4085_FFFF, where you can fetch instructions, read data, write data, etc.

Note:

The LP CPU has a high latency to access the HP SRAM, but can access the LP SRAM with no latency.

The LP CPU supports the atomic instruction set. Both the LP CPU and the HP CPU can access memory through atomic instructions, thus achieving atomicity of memory access. For details on the atomic instruction set, please refer to RISC-V Instruction Set Manual Volume I: Unprivileged ISA, Version 2.2.

2.7.2 Peripheral Access

Table 4.3-2 in Chapter 4 [System and Memory](#) lists the peripherals accessible by the LP CPU and their base addresses.

2.8 Event Task Matrix Feature

The LP CPU on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows LP CPU's ETM tasks to be triggered by any peripherals' ETM events, or LP CPU's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to the LP CPU. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

LP CPU can receive the following ETM task:

- ULP_TASK_WAKEUP_CPU: Wakes up the LP CPU.

LP CPU can generate the following ETM events:

- ULP_EVT_ERR_INTR: Indicates that an LP CPU exception occurs.
- ULP_EVT_START_INTR: Indicates that the LP CPU clock is turned on.

2.9 Sleep and Wake-Up Process

2.9.1 Features

- Able to sleep, wake up, and operate independently in the low-power system when the HP CPU is working or sleeping
- Able to actively configure registers to enter the sleep status based on software operating status
- Wake-up events:
 - The HP CPU setting the register `PMU_HP_TRIGGER_LP`
 - The interrupt state of the LP IO
 - ETM events
 - The RTC timer timeout
 - The LP UART receiving a certain number of RX pulses when `LPPERI_LP_UART_WAKEUP_EN` is enabled

2.9.2 Process

The LP CPU is in sleep by default and its wake-up module follows the process below to wake it up for work and make it sleep.

To configure wake-up sources, please refer to Table 2.9-1.

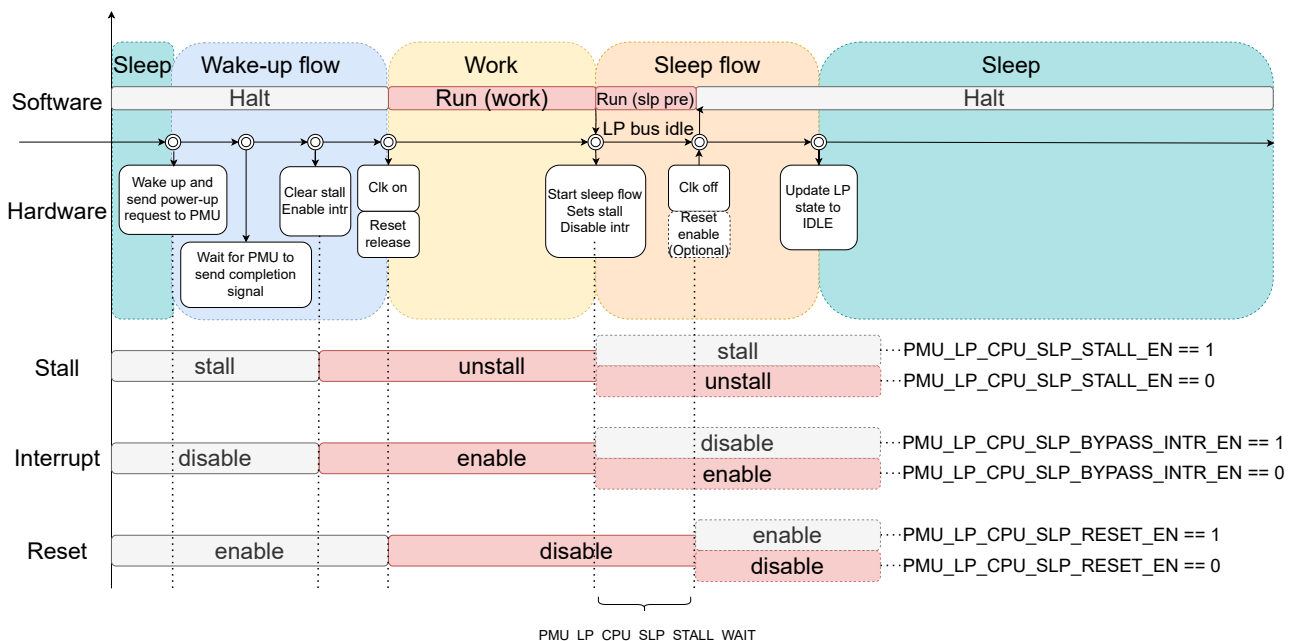


Figure 2.9-1. Wake-Up and Sleep Flow of LP CPU

The first startup of the LP CPU after power-up depends on the wake-up enable and wake-up source configuration by the HP CPU.

- Initialization of the LP CPU
 - Initialize the LP memory.
 - Start the LP CPU. Since the startup of the LP CPU depends on the wake-up process, it is recommended to use the [PMU_HP_TRIGGER_LP](#) register to start the initialization of the LP CPU in the following way:
 - * Set [PMU_LP_CPU_WAKEUP_EN](#) to 0x1
 - * Set [PMU_HP_TRIGGER_LP](#) to 0x1
 - * The LP CPU will go through the wake-up process to start running
- Wake-up process:
 - The wake-up module receives a wake-up signal and sends a power-up request to the PMU.
 - If the current power consumption state (clock, power supply, etc.) meets the requirements of the LP CPU, the PMU will immediately reply with the completion signal. Otherwise, it will adjust the power consumption state before replying with the completion signal.
 - The wake-up module disables the STALL state of the LP CPU and enables interrupt receiving.
 - The wake-up module starts the clock, releases reset (ignore this step if reset is not enabled for sleep), and starts working.
- Sleep process:
 - The LP CPU configures the register [PMU_LP_CPU_SLEEP_REQ](#) to enable the wake-up module to start the sleep process.
 - If [PMU_LP_CPU_SLP_STALL_EN](#) is 1, the wake-up module enables the STALL state of the LP CPU. If it is 0, the module does not enable that state.
 - The wake-up module waits for [PMU_LP_CPU_SLP_STALL_WAIT](#) LP CPU clock cycles, and then turns off the LP CPU clock. If [PMU_LP_CPU_SLP_RESET_EN](#) is 1, the module enables reset of the LP CPU.

2.9.3 Wake-Up Sources

Table 2.9-1. Wake Sources

Register Value ¹	Wake-Up Source	Description
0x1	Register PMU_HP_TRIGGER_LP	The HP CPU sets the register PMU_HP_TRIGGER_LP to wake up the LP CPU, and sets PMU_HP_SW_TRIGGER_INT_CLR to clear this wake-up source.
0x2	LP UART	The LP UART receives a certain number of RX pulses. LPPERI_LP_UART_WAKEUP_EN needs to be enabled. For more information, please refer to Chapter 25 UART Controller (UART) .
0x4	LP IO	This wake-up source uses the LP IO interrupt status register signal. For more information, please refer to Chapter 5 GPIO Matrix and IO MUX .
0x8	ETM	Wake-up sources received from ETM can wake up the LP CPU. For more information, please refer to Chapter 8 Event Task Matrix (ETM) .
0x10	RTC timer	RTC timer target 1 timeout interrupt control. For more information, please refer to Chapter 4 Low-power Management [to be added later] .

¹ Value of the [PMU_LP_CPU_WAKEUP_EN](#) register

2.10 Register Summary

The addresses in this section are relative to Low-Power Peripheral base address provided in Table [4.3-2](#) in Chapter [4 System and Memory](#).

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
LPPERI_CPU_REG	LP CPU Control Register	0x000C	R/W
LPPERI_INTERRUPT_SOURCE_REG	LP CPU Interrupt Status Register	0x0020	RO

2.11 Registers

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

The addresses in this section are relative to Low-Power Peripheral base address provided in Table [4.3-2](#) in Chapter [4 System and Memory](#).

Register 2.27. LPPERI_CPU_REG (0x000C)

Diagram of the LPPER1_LPCORE_DBGM_UNAVAILABLE register. The register is 32 bits wide. Bit 31 is labeled 'LPPER1_LPCORE_DBGM_UNAVAILABLE'. Bits 30-0 are labeled '(reserved)'. The register is shown as a horizontal bar with bit positions 31, 30, and 0 marked at the ends. Below the bar, the bit values are shown: bit 31 is 1, and bits 30-0 are 0. The label 'Reset' is at the bottom right.

LPPERI_LPCORE_DBGM_UNAVALIABLE Configures the LP CPU state.

0: LP CPU can connect to JTAG

1: LP CPU is unavailable and cannot connect to JTAG

(R/W)

Register 2.28. LPPERI_INTERRUPT_SOURCE_REG (0x0020)

Diagram of the LPPERR_LP_INTERRUPT_SOURCE register structure:

- Bits 31 to 6: (reserved)
- Bits 5 to 0: LPPERR_LP_INTERRUPT_SOURCE (Value: 0x0)

LPPERI_LP_INTERRUPT_SOURCE Represents the LP interrupt source.

Bit 5: PMU_LP_INT

Bit 4: Reserved

Bit 3: RTC_TIMER_LP_INT

Bit 2: LP_UART_INT

Bit 1: LP_I2C_INT

Bit 0: LP_IO_INT

(RO)

Chapter 3

GDMA Controller (GDMA)

3.1 Overview

General Direct Memory Access (GDMA) is a feature that allows peripheral-to-memory, memory-to-peripheral, and memory-to-memory data transfer at high speed. The CPU is not involved in the GDMA transfer and therefore is more efficient with less workload.

GDMA has six independent channels: three transmit channels and three receive channels. The GDMA channels are shared and can be assigned to PARLIO, ADC, UHCI, AES, SHA, I2S or general-purpose SPI (GP-SPI) to access internal or external memory.

GDMA uses configurable priority and weight arbitration schemes to manage peripherals' needs for bandwidth.

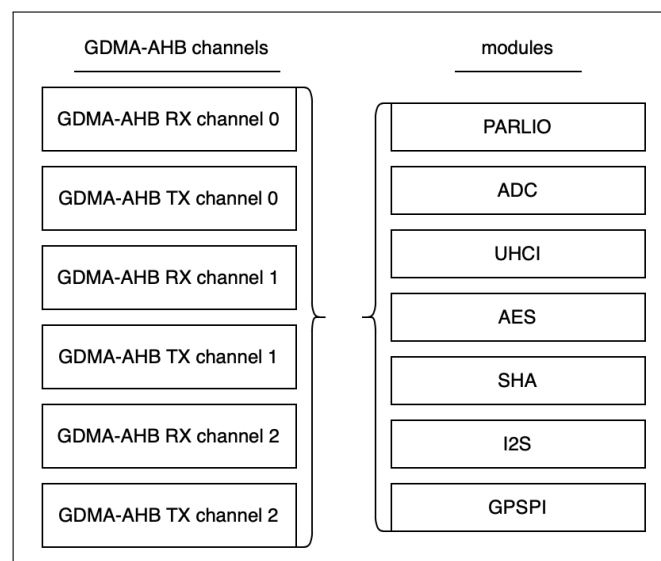


Figure 3.1-1. Modules that Share GDMA Channels

3.2 Features

GDMA has the following features:

- AHB bus architecture
- Programmable length of data to be transferred in bytes
- Access via any address and size
- Alignment:

- Descriptor address: 1-word aligned
- Data address and length: no requirements
- Linked list of descriptors
- INCR burst transfer when accessing memory
- Three transmit channels and three receive channels
- Software-configurable selection of peripheral requesting its service
- Configurable channel priority and weight arbitration
- Support for memory transfer

3.3 Architecture

In ESP32-C5, all modules that need high-speed data transfer support GDMA. The GDMA controller and CPU data bus have access to the same address space in memory. Figure 3.3-1 shows the basic architecture of the GDMA controller.

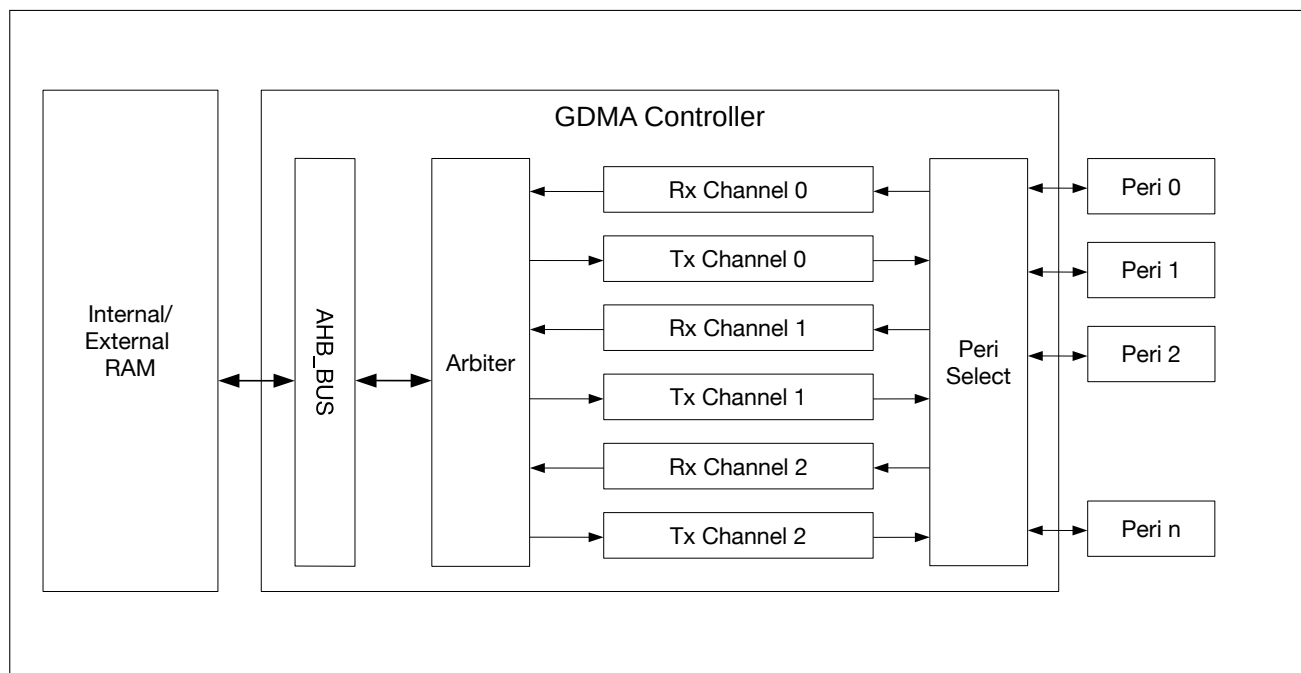


Figure 3.3-1. GDMA controller Architecture

GDMA has six independent channels: three transmit channels and three receive channels. Every channel can be connected to different peripherals. In other words, channels are general-purpose, and shared by peripherals.

GDMA reads data from or writes data to memory via AHB_BUS. Before this, GDMA uses configurable arbitration schemes for channels requesting read or write access. For the available address range of internal and external memory, please see Chapter 4 [System and Memory](#).

Software can use the GDMA through linked lists, which are stored in the internal memory. These linked lists consist of outlink n and inlink n , where n indicates the channel number (ranging from 0 to 2). The GDMA reads

an outlink n (i.e., a linked list of transmit descriptors) from memory and transmits data in the corresponding memory according to the outlink n , or reads an inlink n (i.e., a linked list of receive descriptors) and stores received data into specific address space in the memory according to the inlink n .

3.4 Functional Description

3.4.1 Linked List

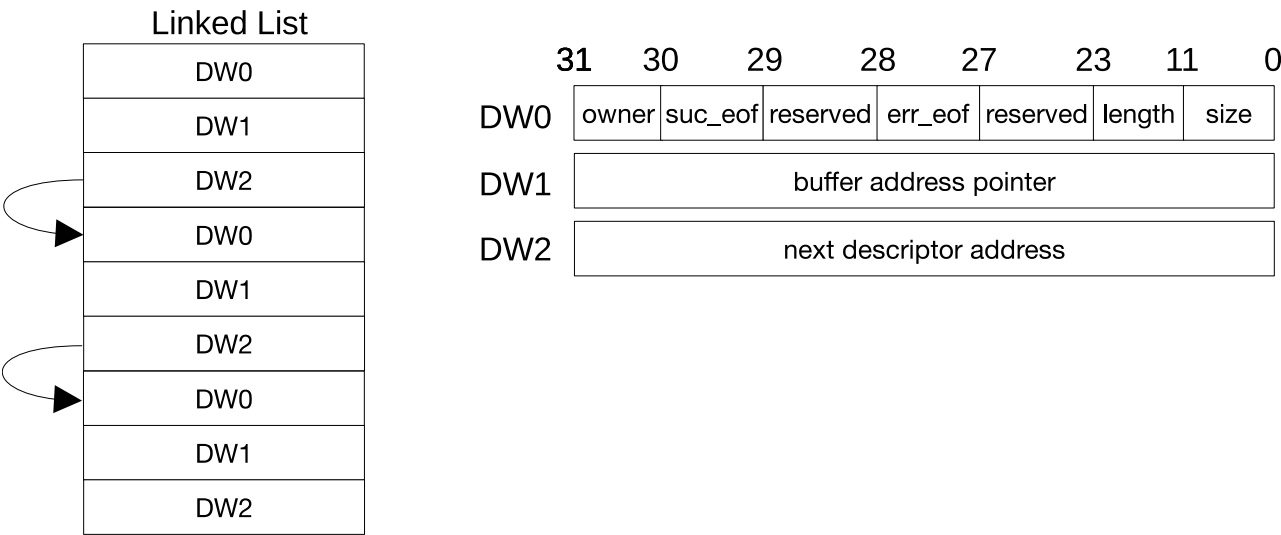


Figure 3.4-1. Structure of a Linked List

Figure 3.4-1 shows the structure of a linked list. An outlink and an inlink have the same structure. A linked list is formed by one or more descriptors, and each descriptor consists of three words. Linked lists should be stored in the memory for the GDMA to be able to use them. The meanings of a descriptor’s fields are as follows:

- owner (DW0) [31]: Specifies who is allowed to access the buffer that this descriptor points to.
 - 0: CPU can access the buffer.
 - 1: The GDMA controller can access the buffer.When the GDMA controller stops using the buffer, this bit in a receive descriptor is automatically cleared by hardware, while this bit in a transmit descriptor can only be automatically cleared by hardware if `AHB_DMA_OUT_AUTO_WRBK_CH $n$` is set to 1. Software can disable automatic clearing by hardware by setting the `AHB_DMA_OUT_LOOP_TEST_CH $n$` or `AHB_DMA_IN_LOOP_TEST_CH $n$` bit. When software loads a linked list, this bit should be set to 1.
 - Note:** `AHB_DMA_OUT` is the prefix of transmit channel registers, and `AHB_DMA_IN` is the prefix of receive channel registers.
- suc_eof (DW0) [30]: Specifies whether the `AHB_DMA_IN_SUC_EOF_CH $n$ _INT` or `AHB_DMA_OUT_EOF_CH $n$ _INT` interrupt will be triggered when the data corresponding to this descriptor has been received or transmitted.
 - 0: No interrupt will be triggered after the current descriptor’s successful transfer;
 - 1: An interrupt will be triggered after the current descriptor’s successful transfer.

For receive descriptors, software needs to clear this bit to 0, and hardware will set it to 1 after receiving data containing the EOF flag.

For transmit descriptors, software needs to set this bit to 1 as needed.

If software configures this bit to 1 in a descriptor, the GDMA will include the EOF flag in the data sent to the corresponding peripheral, indicating to the peripheral that this data segment marks the end of one transfer phase.

- reserved (DWO) [29]: Reserved. The value of this bit does not matter.
- err_eof (DWO) [28]: Specifies whether the received data has errors.
 - 0: The received data does not have errors.
 - 1: The received data has errors.

This bit is used only when UHCI or PARLIO uses the GDMA to receive data. When an error is detected in the received data segment corresponding to a descriptor, this bit in the receive descriptor is set to 1 by hardware.
- reserved (DWO) [27:24]: Reserved.
- length (DWO) [23:12]: Specifies the number of valid bytes in the buffer that this descriptor points to. This field in a transmit descriptor is written by software and indicates how many bytes can be read from the buffer; this field in a receive descriptor is written by hardware automatically and indicates how many valid bytes have been stored in the buffer.
- size (DWO) [11:0]: Specifies the size of the buffer that this descriptor points to.
- buffer address pointer (DW1): Address of the buffer.
- next descriptor address (DW2): Address of the next descriptor. If the current descriptor is the last one, this value is 0. This field can only point to the memory space.

Note:

Please note that addresses of GDMA descriptors should be 4-byte aligned.

If the length of data received is smaller than the size of the buffer, the GDMA controller will not use the available space of the buffer in the next transaction. If the length of data received is larger than the size of the buffer, the GDMA controller will use new descriptors to complete the data transfer after the current transactions.

3.4.2 Peripheral-to-Memory and Memory-to-Peripheral Data Transfer

The GDMA controller can transfer data from memory to peripheral (transmit) and from peripheral to memory (receive). A transmit channel transfers data in the specified memory location to a peripheral's transmitter via an outlink_{*n*}, whereas a receive channel transfers data received by a peripheral to the specified memory location via an inlink_{*n*}.

Every transmit and receive channel can be connected to any peripheral with the GDMA feature. Table 3.4-1 illustrates how to select the peripheral to be connected via registers. “Dummy-*n*” corresponds to register values for memory-to-memory data transfer. When a channel is connected to a peripheral, the rest of the channels cannot be connected to that peripheral.

Table 3.4-1. GDMA Selecting Peripherals via Register Configuration

AHB_DMA_PERI_IN_SEL_CHn AHB_DMA_PERI_OUT_SEL_CHn	Peripheral
0	Dummy-0
1	GP-SPI
2	UHCI
3	I2S
4 ~ 5	Dummy-4 ~ 5
6	AES
7	SHA
8	ADC
9	PARLIO
10 ~ 15	Dummy-10 ~ 15
16 ~ 63	Invalid

3.4.3 Memory-to-Memory Data Transfer

The GDMA controller also allows memory-to-memory data transfer. Such data transfer can be enabled by setting [AHB_DMA_MEM_TRANS_EN_CH \$n\$](#) , which connects the output of transmit channel n to the input of receive channel n . Note that a transmit channel is only connected to the receive channel with the same number (n), and [AHB_DMA_PERI_IN_SEL_CH \$n\$](#) and [AHB_DMA_PERI_OUT_SEL_CH \$n\$](#) should be configured to the same value corresponding to “Dummy”.

3.4.4 Enabling GDMA

The software uses the GDMA controller through linked lists. When the GDMA controller receives data, software loads an inlink, configures the [AHB_DMA_INLINK_ADDR_CH \$n\$](#) field with the address of the first receive descriptor, and sets the [AHB_DMA_INLINK_START_CH \$n\$](#) bit to enable GDMA. When the GDMA controller transmits data, software loads an outlink, prepares data to be transmitted, configures the [AHB_DMA_OUTLINK_ADDR_CH \$n\$](#) field with the address of the first transmit descriptor, and sets the [AHB_DMA_OUTLINK_START_CH \$n\$](#) bit to enable GDMA. The [AHB_DMA_INLINK_START_CH \$n\$](#) bit and [AHB_DMA_OUTLINK_START_CH \$n\$](#) bit are cleared automatically by hardware.

In some cases, you may want to append more descriptors to a DMA transfer that is already started. Naively, it would seem to be possible to do this by clearing the EOF bit of the final descriptor in the existing list and setting its next descriptor address pointer field (DW2) to the first descriptor of the to-be-added list. However, this strategy fails if the existing DMA transfer is almost or entirely finished. Instead, the GDMA controller has specialized logic to make sure a DMA transfer can be continued or restarted: if the transfer is ongoing, the controller will make sure to take the appended descriptors into account; if the transfer has already finished, the controller will restart with the new descriptors. This is implemented by the Restart function.

When using the Restart function, software needs to rewrite the address of the first descriptor in the new list to DW2 of the last descriptor in the loaded list, and set [AHB_DMA_INLINK_RESTART_CH \$n\$](#) bit or [AHB_DMA_OUTLINK_RESTART_CH \$n\$](#) bit (these two bits are cleared automatically by hardware). As shown in Figure 3.4-2, by doing so hardware can obtain the address of the first descriptor in the new list when reading the last descriptor in the loaded list, and then read the new list.

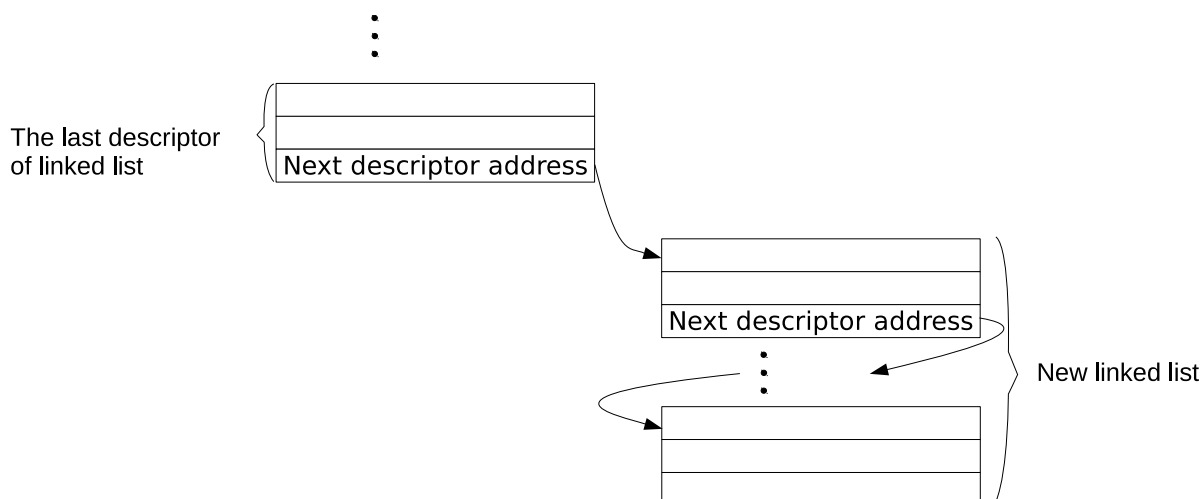


Figure 3.4-2. Relationship among Linked Lists

3.4.5 Linked List Reading Process

Once configured and enabled by software, the GDMA controller starts to read the linked list from memory. The GDMA performs checks on descriptors in the linked list. Only if descriptors pass the checks, the corresponding GDMA channel will start data transfer. If the descriptors fail any of the checks, hardware will trigger descriptor error interrupt (either `AHB_DMA_IN_DSCR_ERR_CH $n$ _INT` or `AHB_DMA_OUT_DSCR_ERR_CH $n$ _INT`), and the channel will halt.

The checks performed on descriptors are:

- Owner bit check when `AHB_DMA_IN_CHECK_OWNER_CH $n$` or `AHB_DMA_OUT_CHECK_OWNER_CH $n$` is set to 1. If the owner bit is 0, the buffer is accessed by the CPU. In this case, the owner bit fails the check. The owner bit will not be checked if `AHB_DMA_IN_CHECK_OWNER_CH $n$` or `AHB_DMA_OUT_CHECK_OWNER_CH $n$` is 0.
- Descriptor address check, which checks if the descriptor address is located in configured memory space for GDMA. If a GDMA descriptor points to `AHB_DMA_ACCESS_INTR_MEM_START_ADDR ~ AHB_DMA_ACCESS_INTR_MEM_END_ADDR`, it passes the check. For details, please refer to Section 3.4.7.

After the software detects a descriptor error interrupt, it must load new descriptors, and enable GDMA by setting `AHB_DMA_OUTLINK_START_CH $n$` or `AHB_DMA_INLINK_START_CH $n$` bit.

3.4.6 EOF

Note: In this chapter, EOF of transmit descriptors refers to `suc_eof` (i.e., bit 30 of DWO), while EOF of receive descriptors refers to both `suc_eof` and `err_eof` (i.e., bit 28 of DWO).

The GDMA controller uses EOF (end of frame) flags to indicate the end of data segment transfer corresponding to a specific descriptor.

For data transmission, the GDMA generates two types of EOF interrupts:

- `AHB_DMA_OUT_EOF_CH $n$ _INT`, generated when the `suc_eof` bit of any descriptor in the linked list is set, and the data corresponding to this descriptor has been transmitted. This interrupt is enabled by setting the `AHB_DMA_OUT_EOF_CH $n$ _INT_ENA` bit.

- `AHB_DMA_OUT_TOTAL_EOF_CHn_INT`, generated when the `suc_eof` bit of the last descriptor in the linked list is set, and the data corresponding to the last descriptor has been transmitted. This interrupt is enabled by setting the `AHB_DMA_OUT_TOTAL_EOF_CHn_INT_ENA` bit.

For data reception, the GDMA also generates two types of EOF interrupts:

- `AHB_DMA_IN_SUC_EOF_CHn_INT`, generated when a data segment with an EOF flag has been received. This interrupt is enabled by setting the `AHB_DMA_IN_SUC_EOF_CHn_INT_ENA` bit.
- (UHCI and PARLIO only) `AHB_DMA_IN_ERR_EOF_CHn_INT`, generated when a data segment corresponding to a descriptor has been received with errors. This interrupt is enabled by setting the `AHB_DMA_IN_ERR_EOF_CHn_INT_ENA` bit, and is valid only when the channel is connected to UHCI or PARLIO.

When detecting an `AHB_DMA_OUT_TOTAL_EOF_CHn_INT` or `AHB_DMA_IN_SUC_EOF_CHn_INT` interrupt, software can read the value of the `AHB_DMA_OUT_EOF_DES_ADDR_CHn` or

`AHB_DMA_IN_SUC_EOF_DES_ADDR_CHn` field, which stores the address of the finished descriptor.

Therefore, the software can tell which descriptors have been used and reclaim them as needed. For RX, the address of the descriptor can also be stored to the `AHB_DMA_IN_ERR_EOF_DES_ADDR_CHn` field when an `AHB_DMA_IN_ERR_EOF_CHn_INT` interrupt is triggered.

3.4.7 Accessing Memory

Any transmit and receive channels of GDMA can access the memory address space configured by `AHB_DMA_ACCESS_INTR_MEM_START_ADDR` and `AHB_DMA_ACCESS_INTR_MEM_END_ADDR`.

To improve data transfer efficiency, GDMA can send data in burst mode. Burst mode is disabled by default, and the burst length can be configured to SINGLE, INCR4, or INCR8 respectively by setting `AHB_DMA_IN_DATA_BURST_MODE_SEL_CHn` and `AHB_DMA_OUT_DATA_BURST_MODE_SEL_CHn` to 0, 1, or 2.

When GDMA accesses the configured memory space, there are no requirements for descriptor field alignment.

3.4.8 Arbitration

To ensure timely response to peripherals running at a high speed with low latency (such as general-purpose SPI), the GDMA controller implements two arbitration schemes, based on channel priority and channel weight respectively.

- Priority arbitration: Each channel can be assigned a priority from 0 ~ 5 (in total 6 levels). The larger the number, the higher the priority. When several channels perform data transfers at the same time, the GDMA would respond to the transfer requests according to priority levels, and the channel with higher priority would get a response more timely.
- Weight arbitration:
 - Each channel is assigned a weight (i.e., the number of tokens) from 0 ~ 15. The GDMA divides the AHB bus clock period into multiple time slots, and in each time slot, the number of transfers performed by a channel is determined by the number of its tokens. Every time a channel performs a data transfer, one of its tokens is spent. If all of its tokens have been spent in a time slot, then this

channel's transfer request will no longer be responded to until the time slot expires, or until tokens of all channels have been spent and the new time slot starts.

- If the number of tokens assigned to a channel is not zero, the GDMA waits for this channel's tokens to be spent even if it does not have data transfer requests, and will not exit from this time slot ahead of expiration. This leads to a waste of bandwidth. Therefore, the GDMA provides weight arbitration optimization. When this feature is enabled, a channel without data transfer requests will not be involved in the arbitration and its tokens will be ignored. When all channels no longer have transfer requests, the GDMA arbiter exits from the current time slot before expiration. This way, the arbitration response is accelerated and the transfer efficiency is improved.

Note:

- If channels have the same weight but different priorities, on condition that the number of bytes to be transferred is the same and not large, channels with higher priorities would finish data transfers first.
- If channels have different priorities and weights, on condition that the number of bytes to be transferred is the same, channels with higher weights would be allocated more bandwidth and finish data transfers first.

3.5 Event Task Matrix Feature

The GDMA controller on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows GDMA's ETM tasks to be triggered by any peripherals' ETM events, or GDMA's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to GDMA. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

GDMA can receive the following ETM tasks:

- GDMA_TASK_IN_START_CH n : Enables the corresponding RX channel n for data transfer.
- GDMA_TASK_OUT_START_CH n : Enables the corresponding TX channel n for data transfer.

Note:

Above ETM tasks can achieve the same functions as CPU configuring [AHB_DMA_INLINK_START_CH \$n\$](#) and [AHB_DMA_OUTLINK_START_CH \$n\$](#) . When [AHB_DMA_IN_ETM_EN_CH \$n\$](#) or [AHB_DMA_OUT_ETM_EN_CH \$n\$](#) is 1, only ETM tasks can be used to configure the transfer direction and enable the corresponding GDMA channel. When [AHB_DMA_IN_ETM_EN_CH \$n\$](#) or [AHB_DMA_OUT_ETM_EN_CH \$n\$](#) is 0, only CPU can be used to enable the corresponding GDMA channel.

GDMA can generate the following ETM events:

- GDMA_EVT_IN_DONE_CH n : Indicates that the data has been received according to the receive descriptor via channel n .
- GDMA_EVT_IN_SUC_EOF_CH n : Indicates that the data corresponding to a receive descriptor has been received via channel n and the EOF bit of this descriptor is 1.
- GDMA_EVT_IN_FIFO_EMPTY_CH n : Indicates that the RX FIFO has become empty.
- GDMA_EVT_IN_FIFO_FULL_CH n : Indicates that the RX FIFO has become full.

- GDMA_EVT_OUT_DONE_CH n : Indicates that the data has been transmitted according to the transmit descriptor via channel n .
- GDMA_EVT_OUT_EOF_CH n : Indicates that the data corresponding to a transmit descriptor has been transmitted or received via channel n and the EOF bit of this descriptor is 1.
- GDMA_EVT_OUT_TOTAL_EOF_CH n : Indicates that the data corresponding to the last transmit descriptors has been sent via transmit channel n and the EOF bit of this descriptor is 1.
- GDMA_EVT_OUT_FIFO_EMPTY_CH n : Indicates that the TX FIFO has become empty.
- GDMA_EVT_OUT_FIFO_FULL_CH n : Indicates that the TX FIFO has become full.

In practical applications, GDMA's ETM events can trigger its own ETM tasks. For example, the GDMA_EVT_OUT_TOTAL_EOF_CH0 event can trigger the GDMA_TASK_IN_START_CH1 task, and in this way trigger a new round of GDMA operations.

3.6 Interrupts

ESP32-C5's GDMA module can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- GDMA_IN_CH0_INTR
- GDMA_IN_CH1_INTR
- GDMA_IN_CH2_INTR
- GDMA_OUT_CH0_INTR
- GDMA_OUT_CH1_INTR
- GDMA_OUT_CH2_INTR

There are several internal interrupt sources from GDMA that can generate the above interrupt signals.

The GDMA_IN_CH n _INTR (n is 0 ~ 2) signals are generated by the following interrupt sources:

- AHB_DMA_IN_DONE_CH n _INT: Triggered when all data corresponding to a receive descriptor has been received via receive channel n .
- AHB_DMA_IN_SUC_EOF_CH n _INT: Triggered when the suc_eof bit in a receive descriptor is 1 and the data corresponding to this receive descriptor has been received via receive channel n .
- AHB_DMA_IN_ERR_EOF_CH n _INT: Triggered when an error is detected in the data segment corresponding to a descriptor received via receive channel n . This interrupt is used only for UHCI (UART0 or UART1) or PARLIO.
- AHB_DMA_IN_DSCR_EMPTY_CH n _INT: Triggered when the size of the buffer pointed by receive descriptors is smaller than the length of data to be received via receive channel n .
- AHB_DMA_IN_DSCR_ERR_CH n _INT: Triggered when a receive descriptor on receive channel n fails any of the two descriptor checks.
- AHB_DMA_INFIFO_OVF_CH n _INT: Triggered when the RX FIFO of GDMA overflows.
- AHB_DMA_INFIFO_UDF_CH n _INT: Triggered when the RX FIFO of GDMA underflows.

The GDMA_OUT_CH n _INTR (n is 0 ~ 2) signals are generated by the following interrupt sources:

- AHB_DMA_OUT_DONE_CH n _INT: Triggered when all data corresponding to a transmit descriptor has been sent via transmit channel n .
- AHB_DMA_OUT_EOF_CH n _INT: Triggered when the suc_eof bit in a transmit descriptor is 1 and data corresponding to this descriptor has been sent via transmit channel n . If [AHB_DMA_OUT_EOF_MODE_CH \$n\$](#) is 0, this interrupt will be triggered when the last byte of data corresponding to this descriptor enters GDMA's transmit channel; if [AHB_DMA_OUT_EOF_MODE_CH \$n\$](#) is 1, this interrupt is triggered when the last byte of data is taken from GDMA's transmit channel.
- AHB_DMA_OUT_DSCR_ERR_CH n _INT: Triggered when a transmit descriptor on transmit channel n fails any of the two descriptor checks.
- AHB_DMA_OUT_TOTAL_EOF_CH n _INT: Triggered when all data corresponding to a linked list (including multiple descriptors) has been sent via transmit channel n .
- AHB_DMA_OUTFIFO_OVF_CH n _INT: Triggered when the TX FIFO of GDMA overflows.
- AHB_DMA_OUTFIFO_UDF_CH n _INT: Triggered when the TX FIFO of GDMA underflows.

3.7 Programming Procedures

The clock gating for GDMA can be configured via [PCR_GDMA_CLK_EN](#), and is enabled by default. GDMA can be reset by configuring [PCR_GDMA_RST_EN](#).

3.7.1 Programming Procedures for GDMA's Transmit Channel

To transmit data, GDMA's transmit channel should be configured by software as follows:

1. Set [AHB_DMA_OUT_RST_CH \$n\$](#) first to 1 and then to 0, to reset the state machine of GDMA's transmit channel and FIFO pointer.
2. Load an outlink, and configure [AHB_DMA_OUTLINK_ADDR_CH \$n\$](#) with address of the first transmit descriptor.
3. Configure [AHB_DMA_PERI_OUT_SEL_CH \$n\$](#) with the value corresponding to the peripheral to be connected, as shown in Table 3.4-1.
4. Set [AHB_DMA_OUTLINK_START_CH \$n\$](#) to enable GDMA's transmit channel for data transfer.
5. Configure and enable the corresponding peripheral. See details in the individual chapter for the corresponding peripheral.
6. Wait for the AHB_DMA_OUT_TOTAL_EOF_CH n _INT interrupt, which indicates the completion of data transfer.

3.7.2 Programming Procedures for GDMA's Receive Channel

To receive data, GDMA's receive channel should be configured by software as follows:

1. Set [AHB_DMA_IN_RST_CH \$n\$](#) first to 1 and then to 0, to reset the state machine of GDMA's receive channel and FIFO pointer.

2. Load an inlink, and configure `AHB_DMA_INLINK_ADDR_CHn` with address of the first receive descriptor.
3. Configure `AHB_DMA_PERI_IN_SEL_CHn` with the value corresponding to the peripheral to be connected, as shown in Table 3.4-1.
4. Set `AHB_DMA_INLINK_START_CHn` to enable GDMA's receive channel for data transfer.
5. Configure and enable the corresponding peripheral. See details in the individual chapter for the corresponding peripheral.

3.7.3 Programming Procedures for Memory-to-Memory Transfer

To transfer data from one memory location to another, GDMA should be configured by software as follows:

1. Set `AHB_DMA_OUT_RST_CHn` first to 1 and then to 0, to reset the state machine of GDMA's transmit channel and FIFO pointer.
2. Set `AHB_DMA_IN_RST_CHn` first to 1 and then to 0, to reset the state machine of GDMA's receive channel and FIFO pointer.
3. Load an outlink, and configure `AHB_DMA_OUTLINK_ADDR_CHn` with address of the first transmit descriptor.
4. Load an inlink, and configure `AHB_DMA_INLINK_ADDR_CHn` with address of the first receive descriptor.
5. Set `AHB_DMA_MEM_TRANS_EN_CHn` to enable memory-to-memory transfer.
6. Set `AHB_DMA_OUTLINK_START_CHn` to enable GDMA's transmit channel for data transfer.
7. Set `AHB_DMA_INLINK_START_CHn` to enable GDMA's receive channel for data transfer.
8. If the `suc_eof` bit is set in a transmit descriptor, an `AHB_DMA_IN_SUC_EOF_CHn_INT` interrupt will be triggered when the data segment corresponding to this descriptor has been transmitted.

3.7.4 Programming Procedures for Channel Priority and Weight

The priority arbitration can be configured as follows:

1. Configure the channel priority for TX and RX respectively via `AHB_DMA_TX_PRI_CHn` and `AHB_DMA_RX_PRI_CHn`.

The weight arbitration can be configured as follows:

1. Configure the time slot via `AHB_DMA_ARB_TIMEOUT`.
2. Configure the number of tokens for TX and RX respectively via `AHB_DMA_TX_CH_ARB_WEIGHT_CHn` and `AHB_DMA_RX_CH_ARB_WEIGHT_CHn`.
3. Enable weight arbitration optimization for TX and RX respectively by clearing `AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CHn` and `AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CHn`.
4. Enable the arbitration by setting `AHB_DMA_WEIGHT_EN`.

3.8 Register Summary

The addresses in this section are relative to GDMA base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Interrupt Registers			
AHB_DMA_IN_INT_RAW_CHO_REG	RX channel 0 raw interrupt status register	0x0000	R/WTC/SS
AHB_DMA_IN_INT_ST_CHO_REG	RX channel 0 masked interrupt status register	0x0004	RO
AHB_DMA_IN_INT_ENA_CHO_REG	RX channel 0 interrupt enable register	0x0008	R/W
AHB_DMA_IN_INT_CLR_CHO_REG	RX channel 0 interrupt clear register	0x000C	WT
AHB_DMA_IN_INT_RAW_CH1_REG	RX channel 1 raw interrupt status register	0x0010	R/WTC/SS
AHB_DMA_IN_INT_ST_CH1_REG	RX channel 1 masked interrupt status register	0x0014	RO
AHB_DMA_IN_INT_ENA_CH1_REG	RX channel 1 interrupt enable register	0x0018	R/W
AHB_DMA_IN_INT_CLR_CH1_REG	RX channel 1 interrupt clear register	0x001C	WT
AHB_DMA_IN_INT_RAW_CH2_REG	RX channel 2 raw interrupt status register	0x0020	R/WTC/SS
AHB_DMA_IN_INT_ST_CH2_REG	RX channel 2 masked interrupt status register	0x0024	RO
AHB_DMA_IN_INT_ENA_CH2_REG	RX channel 2 interrupt enable register	0x0028	R/W
AHB_DMA_IN_INT_CLR_CH2_REG	RX channel 2 interrupt clear register	0x002C	WT
AHB_DMA_OUT_INT_RAW_CHO_REG	TX channel 0 raw interrupt status register	0x0030	R/WTC/SS
AHB_DMA_OUT_INT_ST_CHO_REG	TX channel 0 masked interrupt status register	0x0034	RO
AHB_DMA_OUT_INT_ENA_CHO_REG	TX channel 0 interrupt enable register	0x0038	R/W
AHB_DMA_OUT_INT_CLR_CHO_REG	TX channel 0 interrupt clear register	0x003C	WT
AHB_DMA_OUT_INT_RAW_CH1_REG	TX channel 1 raw interrupt status register	0x0040	R/WTC/SS
AHB_DMA_OUT_INT_ST_CH1_REG	TX channel 1 masked interrupt status register	0x0044	RO
AHB_DMA_OUT_INT_ENA_CH1_REG	TX channel 1 interrupt enable register	0x0048	R/W
AHB_DMA_OUT_INT_CLR_CH1_REG	TX channel 1 interrupt clear register	0x004C	WT
AHB_DMA_OUT_INT_RAW_CH2_REG	TX channel 2 raw interrupt status register	0x0050	R/WTC/SS
AHB_DMA_OUT_INT_ST_CH2_REG	TX channel 2 masked interrupt status register	0x0054	RO
AHB_DMA_OUT_INT_ENA_CH2_REG	TX channel 2 interrupt enable register	0x0058	R/W

Name	Description	Address	Access
AHB_DMA_OUT_INT_CLR_CH2_REG	TX channel 2 interrupt clear register	0x005C	WT
Debug Registers			
AHB_DMA_AHB_TEST_REG	Reserved	0x0060	R/W
Configuration Registers			
AHB_DMA_MISC_CONF_REG	Miscellaneous register	0x0064	R/W
AHB_DMA_IN_CONFO_CHO_REG	Configuration register 0 of RX channel 0	0x0070	R/W
AHB_DMA_IN_CONF1_CHO_REG	Configuration register 1 of RX channel 0	0x0074	R/W
AHB_DMA_IN_POP_CHO_REG	Pop control register of RX channel 0	0x007C	varies
AHB_DMA_IN_LINK_CHO_REG	Linked list descriptor configuration and control register of RX channel 0	0x0080	varies
AHB_DMA_OUT_CONFO_CHO_REG	Configuration register 0 of TX channel 0	0x00D0	R/W
AHB_DMA_OUT_CONF1_CHO_REG	Configuration register 1 of TX channel 0	0x00D4	R/W
AHB_DMA_OUT_PUSH_CHO_REG	Push control register of TX channel 0	0x00DC	varies
AHB_DMA_OUT_LINK_CHO_REG	Linked list descriptor configuration and control register of TX channel 0	0x00E0	varies
AHB_DMA_IN_CONFO_CH1_REG	Configuration register 0 of RX channel 1	0x0130	R/W
AHB_DMA_IN_CONF1_CH1_REG	Configuration register 1 of RX channel 1	0x0134	R/W
AHB_DMA_IN_POP_CH1_REG	Pop control register of RX channel 1	0x013C	varies
AHB_DMA_IN_LINK_CH1_REG	Linked list descriptor configuration and control register of RX channel 1	0x0140	varies
AHB_DMA_OUT_CONFO_CH1_REG	Configuration register 0 of TX channel 1	0x0190	R/W
AHB_DMA_OUT_CONF1_CH1_REG	Configuration register 1 of TX channel 1	0x0194	R/W
AHB_DMA_OUT_PUSH_CH1_REG	Push control register of TX channel 1	0x019C	varies
AHB_DMA_OUT_LINK_CH1_REG	Linked list descriptor configuration and control register of TX channel 1	0x01A0	varies
AHB_DMA_IN_CONFO_CH2_REG	Configuration register 0 of RX channel 2	0x01F0	R/W
AHB_DMA_IN_CONF1_CH2_REG	Configuration register 1 of RX channel 2	0x01F4	R/W
AHB_DMA_IN_POP_CH2_REG	Pop control register of RX channel 2	0x01FC	varies

Name	Description	Address	Access
AHB_DMA_IN_LINK_CH2_REG	Linked list descriptor configuration and control register of RX channel 2	0x0200	varies
AHB_DMA_OUT_CONFO_CH2_REG	Configuration register 0 of TX channel 2	0x0250	R/W
AHB_DMA_OUT_CONF1_CH2_REG	Configuration register 1 of TX channel 2	0x0254	R/W
AHB_DMA_OUT_PUSH_CH2_REG	Push control register of TX channel 2	0x025C	varies
AHB_DMA_OUT_LINK_CH2_REG	Linked list descriptor configuration and control register of TX channel 2	0x0260	varies
AHB_DMA_TX_CH_ARB_WEIGHT_CHO_REG	TX channel 0 arbitration weight configuration register	0x02DC	R/W
AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CHO_REG	TX channel 0 weight arbitration optimization enable register	0x02E0	R/W
AHB_DMA_TX_CH_ARB_WEIGHT_CH1_REG	TX channel 1 arbitration weight configuration register	0x0304	R/W
AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CH1_REG	TX channel 1 weight arbitration optimization enable register	0x0308	R/W
AHB_DMA_TX_CH_ARB_WEIGHT_CH2_REG	TX channel 2 arbitration weight configuration register	0x032C	R/W
AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CH2_REG	TX channel 2 weight arbitration optimization enable register	0x0330	R/W
AHB_DMA_RX_CH_ARB_WEIGHT_CHO_REG	RX channel 0 arbitration weight configuration register	0x0354	R/W
AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CHO_REG	RX channel 0 weight arbitration optimization enable register	0x0358	R/W
AHB_DMA_RX_CH_ARB_WEIGHT_CH1_REG	RX channel 1 arbitration weight configuration register	0x037C	R/W
AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CH1_REG	RX channel 1 weight arbitration optimization enable register	0x0380	R/W
AHB_DMA_RX_CH_ARB_WEIGHT_CH2_REG	RX channel 2 arbitration weight configuration register	0x03A4	R/W
AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CH2_REG	RX channel 2 weight arbitration optimization enable register	0x03A8	R/W
AHB_DMA_IN_LINK_ADDR_CHO_REG	Linked list descriptor configuration register of RX channel 0	0x03AC	R/W
AHB_DMA_IN_LINK_ADDR_CH1_REG	Linked list descriptor configuration register of RX channel 1	0x03B0	R/W

Name	Description	Address	Access
AHB_DMA_IN_LINK_ADDR_CH2_REG	Linked list descriptor configuration register of RX channel 2	0x03B4	R/W
AHB_DMA_OUT_LINK_ADDR_CHO_REG	Linked list descriptor configuration register of TX channel 0	0x03B8	R/W
AHB_DMA_OUT_LINK_ADDR_CH1_REG	Linked list descriptor configuration register of TX channel 1	0x03BC	R/W
AHB_DMA_OUT_LINK_ADDR_CH2_REG	Linked list descriptor configuration register of TX channel 2	0x03C0	R/W
AHB_DMA_INTR_MEM_START_ADDR_REG	Accessible address space start address configuration register	0x03C4	R/W
AHB_DMA_INTR_MEM_END_ADDR_REG	Accessible address space end address configuration register	0x03C8	R/W
AHB_DMA_ARB_TIMEOUT_REG	Weight arbitration timeout configuration register	0x03DC	R/W
AHB_DMA_WEIGHT_EN_REG	Weight arbitration enable register	0x0400	R/W
Version Registers			
AHB_DMA_DATE_REG	Version control register	0x0068	R/W
Status Registers			
AHB_DMA_INFIFO_STATUS_CHO_REG	RX channel 0 FIFO status	0x0078	RO
AHB_DMA_IN_STATE_CHO_REG	RX channel 0 status	0x0084	RO
AHB_DMA_IN_SUC_EOF_DES_ADDR_CHO_REG	Receive descriptor address when EOF occurs on RX channel 0	0x0088	RO
AHB_DMA_IN_ERR_EOF_DES_ADDR_CHO_REG	Receive descriptor address when errors occur on RX channel 0	0x008C	RO
AHB_DMA_IN_DSCR_CHO_REG	Address of the next receive descriptor pointed by the current pre-read receive descriptor on RX channel 0	0x0090	RO
AHB_DMA_IN_DSCR_BFO_CHO_REG	Address of the current pre-read receive descriptor on RX channel 0	0x0094	RO
AHB_DMA_IN_DSCR_BF1_CHO_REG	Address of the previous pre-read receive descriptor on RX channel 0	0x0098	RO

Name	Description	Address	Access
AHB_DMA_OUTFIFO_STATUS_CHO_REG	TX channel 0 FIFO status	0x00D8	RO
AHB_DMA_OUT_STATE_CHO_REG	TX channel 0 status	0x00E4	RO
AHB_DMA_OUT_EOF_DES_ADDR_CHO_REG	Transmit descriptor address when EOF occurs on TX channel 0	0x00E8	RO
AHB_DMA_OUT_EOF_BFR_DES_ADDR_CHO_REG	The last transmit descriptor address when EOF occurs on TX channel 0	0x00EC	RO
AHB_DMA_OUT_DSCR_CHO_REG	Address of the next transmit descriptor pointed by the current pre-read transmit descriptor on TX channel 0	0x00F0	RO
AHB_DMA_OUT_DSCR_BFO_CHO_REG	Address of the current pre-read transmit descriptor on TX channel 0	0x00F4	RO
AHB_DMA_OUT_DSCR_BF1_CHO_REG	Address of the previous pre-read transmit descriptor on TX channel 0	0x00F8	RO
AHB_DMA_INFIFO_STATUS_CH1_REG	RX channel 1 FIFO status	0x0138	RO
AHB_DMA_IN_STATE_CH1_REG	RX channel 1 status	0x0144	RO
AHB_DMA_IN_SUC_EOF_DES_ADDR_CH1_REG	Receive descriptor address when EOF occurs on RX channel 1	0x0148	RO
AHB_DMA_IN_ERR_EOF_DES_ADDR_CH1_REG	Receive descriptor address when errors occur on RX channel 1	0x014C	RO
AHB_DMA_IN_DSCR_CH1_REG	Address of the next receive descriptor pointed by the current pre-read receive descriptor on RX channel 1	0x0150	RO
AHB_DMA_IN_DSCR_BFO_CH1_REG	Address of the current pre-read receive descriptor on RX channel 1	0x0154	RO
AHB_DMA_IN_DSCR_BF1_CH1_REG	Address of the previous pre-read receive descriptor on RX channel 1	0x0158	RO
AHB_DMA_OUTFIFO_STATUS_CH1_REG	TX channel 1 FIFO status	0x0198	RO
AHB_DMA_OUT_STATE_CH1_REG	TX channel 1 status	0x01A4	RO
AHB_DMA_OUT_EOF_DES_ADDR_CH1_REG	Transmit descriptor address when EOF occurs on TX channel 1	0x01A8	RO

Name	Description	Address	Access
AHB_DMA_OUT_EOF_BFR_DES_ADDR_CH1_REG	The last transmit descriptor address when EOF occurs on TX channel 1	0x01AC	RO
AHB_DMA_OUT_DSCR_CH1_REG	Address of the next transmit descriptor pointed by the current pre-read transmit descriptor on TX channel 1	0x01B0	RO
AHB_DMA_OUT_DSCR_BFO_CH1_REG	Address of the current pre-read transmit descriptor on TX channel 1	0x01B4	RO
AHB_DMA_OUT_DSCR_BF1_CH1_REG	Address of the previous pre-read transmit descriptor on TX channel 1	0x01B8	RO
AHB_DMA_INFIFO_STATUS_CH2_REG	RX channel 2 FIFO status	0x01F8	RO
AHB_DMA_IN_STATE_CH2_REG	RX channel 2 status	0x0204	RO
AHB_DMA_IN_SUC_EOF_DES_ADDR_CH2_REG	Receive descriptor address when EOF occurs on RX channel 2	0x0208	RO
AHB_DMA_IN_ERR_EOF_DES_ADDR_CH2_REG	Receive descriptor address when errors occur on RX channel 2	0x020C	RO
AHB_DMA_IN_DSCR_CH2_REG	Address of the next receive descriptor pointed by the current pre-read receive descriptor on RX channel 2	0x0210	RO
AHB_DMA_IN_DSCR_BFO_CH2_REG	Address of the current pre-read receive descriptor on RX channel 2	0x0214	RO
AHB_DMA_IN_DSCR_BF1_CH2_REG	Address of the previous pre-read receive descriptor on RX channel 2	0x0218	RO
AHB_DMA_OUTFIFO_STATUS_CH2_REG	TX channel 2 FIFO status	0x0258	RO
AHB_DMA_OUT_STATE_CH2_REG	TX channel 2 status	0x0264	RO
AHB_DMA_OUT_EOF_DES_ADDR_CH2_REG	Transmit descriptor address when EOF occurs on TX channel 2	0x0268	RO
AHB_DMA_OUT_EOF_BFR_DES_ADDR_CH2_REG	The last transmit descriptor address when EOF occurs on TX channel 2	0x026C	RO

Name	Description	Address	Access
AHB_DMA_OUT_DSCR_CH2_REG	Address of the next transmit descriptor pointed by the current pre-read transmit descriptor on TX channel 2	0x0270	RO
AHB_DMA_OUT_DSCR_BF0_CH2_REG	Address of the current pre-read transmit descriptor on TX channel 2	0x0274	RO
AHB_DMA_OUT_DSCR_BF1_CH2_REG	Address of the previous pre-read transmit descriptor on TX channel 2	0x0278	RO
Priority Registers			
AHB_DMA_IN_PRI_CHO_REG	Priority register of RX channel 0	0x009C	R/W
AHB_DMA_OUT_PRI_CHO_REG	Priority register of TX channel 0	0x00FC	R/W
AHB_DMA_IN_PRI_CH1_REG	Priority register of RX channel 1	0x015C	R/W
AHB_DMA_OUT_PRI_CH1_REG	Priority register of TX channel 1	0x01BC	R/W
AHB_DMA_IN_PRI_CH2_REG	Priority register of RX channel 2	0x021C	R/W
AHB_DMA_OUT_PRI_CH2_REG	Priority register of TX channel 2	0x027C	R/W
Peripheral Selection Registers			
AHB_DMA_IN_PERI_SEL_CHO_REG	Peripheral selection register of RX channel 0	0x00A0	R/W
AHB_DMA_OUT_PERI_SEL_CHO_REG	Peripheral selection register of TX channel 0	0x0100	R/W
AHB_DMA_IN_PERI_SEL_CH1_REG	Peripheral selection register of RX channel 1	0x0160	R/W
AHB_DMA_OUT_PERI_SEL_CH1_REG	Peripheral selection register of TX channel 1	0x01C0	R/W
AHB_DMA_IN_PERI_SEL_CH2_REG	Peripheral selection register of RX channel 2	0x0220	R/W
AHB_DMA_OUT_PERI_SEL_CH2_REG	Peripheral selection register of TX channel 2	0x0280	R/W

3.9 Registers

The addresses in this section are relative to GDMA base address provided in Table 4.3-2 in Chapter 4 [System and Memory](#).

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 3.1. AHB_DMA_IN_INT_RAW_CH n _REG (n : 0-2) (0x0000+0x10* n)

(reserved)																												AHB_DMA_INFIFO_UDF_CH n _INT_RAW AHB_DMA_INFIFO_OVF_CH n _INT_RAW AHB_DMA_IN_DSCR_EMPTY_CH n _INT_RAW AHB_DMA_IN_DSCR_ERR_CH n _INT_RAW AHB_DMA_IN_SUC_EOF_CH n _INT_RAW AHB_DMA_IN_DONE_CH n _INT_RAW							
31																												7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset						

AHB_DMA_IN_DONE_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_DONE_CH n _INT. (R/WTC/SS)

AHB_DMA_IN_SUC_EOF_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_SUC_EOF_CH n _INT. For UHCI, this bit turns to 1 when the last data byte pointed by one receive descriptor has been received and no data error is detected for RX channel n . (R/WTC/SS)

AHB_DMA_IN_ERR_EOF_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_ERR_EOF_CH n _INT. Valid only for UHCI. (R/WTC/SS)

AHB_DMA_IN_DSCR_ERR_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_DSCR_ERR_CH n _INT. (R/WTC/SS)

AHB_DMA_IN_DSCR_EMPTY_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_DSCR_EMPTY_CH n _INT. (R/WTC/SS)

AHB_DMA_IN_FIFO_OVF_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_FIFO_OVF_CH n _INT. (R/WTC/SS)

AHB_DMA_IN_FIFO_UDF_CH n _INT_RAW The raw interrupt status of AHB_DMA_IN_FIFO_UDF_CH n _INT. (R/WTC/SS)

Register 3.2. AHB_DMA_IN_INT_ST_CH n _REG (n : 0-2) (0x0004+0x10* n)

(reserved)																							
31																							
7																							
6																							
5																							
4																							
3																							
2																							
1																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							
0																							

AHB_DMA_IN_DONE_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_DONE_CH n _INT. (RO)

AHB_DMA_IN_SUC_EOF_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_SUC_EOF_CH n _INT. (RO)

AHB_DMA_IN_ERR_EOF_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_ERR_EOF_CH n _INT. (RO)

AHB_DMA_IN_DSCR_ERR_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_DSCR_ERR_CH n _INT. (RO)

AHB_DMA_IN_DSCR_EMPTY_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_DSCR_EMPTY_CH n _INT. (RO)

AHB_DMA_IN_FIFO_OVF_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_FIFO_OVF_CH n _INT. (RO)

AHB_DMA_IN_FIFO_UDF_CH n _INT_ST The masked interrupt status of AHB_DMA_IN_FIFO_UDF_CH n _INT. (RO)

Register 3.3. AHB_DMA_IN_INT_ENA_CH n _REG (n : 0-2) (0x0008+0x10* n)

(reserved)																												AHB_DMA_INFIFO_UDF_CH _n _INT_ENA AHB_DMA_INFIFO_OVF_CH _n _INT_ENA AHB_DMA_IN_DSCR_EMPTY_CH _n _INT_ENA AHB_DMA_IN_DSCR_ERR_CH _n _INT_ENA AHB_DMA_IN_SUC_EOF_CH _n _INT_ENA AHB_DMA_IN_DONE_CH _n _INT_ENA							
31																									7	6	5	4	3	2	1	0	Reset		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								

AHB_DMA_IN_DONE_CH n _INT_ENA Write 1 to enable AHB_DMA_IN_DONE_CH n _INT. (R/W)

AHB_DMA_IN_SUC_EOF_CH n _INT_ENA Write 1 to enable AHB_DMA_IN_SUC_EOF_CH n _INT.
(R/W)

AHB_DMA_IN_ERR_EOF_CH n _INT_ENA Write 1 to enable AHB_DMA_IN_ERR_EOF_CH n _INT.
(R/W)

AHB_DMA_IN_DSCR_ERR_CH n _INT_ENA Write 1 to enable AHB_DMA_IN_DSCR_ERR_CH n _INT.
(R/W)

AHB_DMA_IN_DSCR_EMPTY_CH n _INT_ENA Write 1 to enable AHB_DMA_IN_DSCR_EMPTY_CH n _INT.
(R/W)

AHB_DMA_INFIFO_OVF_CH n _INT_ENA Write 1 to enable AHB_DMA_INFIFO_OVF_CH n _INT. (R/W)

AHB_DMA_INFIFO_UDF_CH n _INT_ENA Write 1 to enable AHB_DMA_INFIFO_UDF_CH n _INT. (R/W)

Register 3.4. AHB_DMA_IN_INT_CLR_CH n _REG (n : 0-2) (0x000C+0x10* n)

(reserved)																												AHB_DMA_INFIFO_UDF_CH _n _INT_CLR AHB_DMA_INFIFO_OVF_CH _n _INT_CLR AHB_DMA_IN_DSCR_EMPTY_CH _n _INT_CLR AHB_DMA_IN_DSCR_ERR_CH _n _INT_CLR AHB_DMA_IN_SUC_EOF_CH _n _INT_CLR AHB_DMA_IN_ERR_EOF_CH _n _INT_CLR AHB_DMA_IN_DONE_CH _n _INT_CLR															
31																												7	6	5	4	3	2	1	0	Reset							
0																												0	0	0	0	0	0	0	0								

AHB_DMA_IN_DONE_CH n _INT_CLR Write 1 to clear AHB_DMA_IN_DONE_CH n _INT. (WT)

AHB_DMA_IN_SUC_EOF_CH n _INT_CLR Write 1 to clear AHB_DMA_IN_SUC_EOF_CH n _INT. (WT)

AHB_DMA_IN_ERR_EOF_CH n _INT_CLR Write 1 to clear AHB_DMA_IN_ERR_EOF_CH n _INT. (WT)

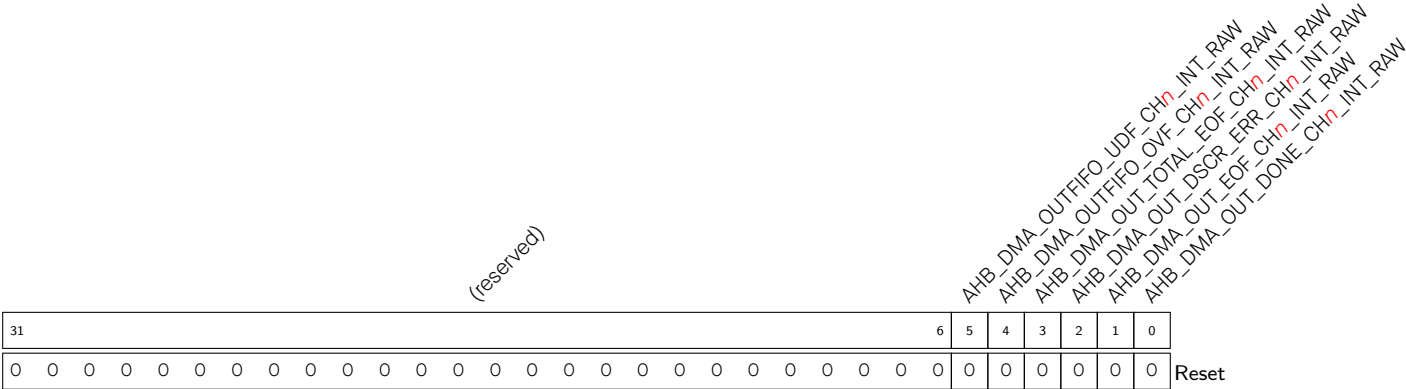
AHB_DMA_IN_DSCR_ERR_CH n _INT_CLR Write 1 to clear AHB_DMA_IN_DSCR_ERR_CH n _INT.
(WT)

AHB_DMA_IN_DSCR_EMPTY_CH n _INT_CLR Write 1 to clear AHB_DMA_IN_DSCR_EMPTY_CH n _INT.
(WT)

AHB_DMA_INFIFO_OVF_CH n _INT_CLR Write 1 to clear AHB_DMA_INFIFO_OVF_CH n _INT. (WT)

AHB_DMA_INFIFO_UDF_CH n _INT_CLR Write 1 to clear AHB_DMA_INFIFO_UDF_CH n _INT. (WT)

Register 3.5. AHB_DMA_OUT_INT_RAW_CHn_REG (n: 0-2) (0x0030+0x10*n)



AHB_DMA_OUT_DONE_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_DONE_CHn_INT. (R/WTC/SS)

AHB_DMA_OUT_EOF_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_EOF_CHn_INT. (R/WTC/SS)

AHB_DMA_OUT_DSCR_ERR_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_DSCR_ERR_CHn_INT. (R/WTC/SS)

AHB_DMA_OUT_TOTAL_EOF_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_TOTAL_EOF_CHn_INT. (R/WTC/SS)

AHB_DMA_OUT_FIFO_OVF_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_FIFO_OVF_CHn_INT. (R/WTC/SS)

AHB_DMA_OUT_FIFO_UDF_CHn_INT_RAW The raw interrupt status of AHB_DMA_OUT_FIFO_UDF_CHn_INT. (R/WTC/SS)

Register 3.6. AHB_DMA_OUT_INT_ST_CH_{*n*}_REG (*n*: 0-2) (0x0034+0x10n*)**

Diagram of the AHB_DMA_OUT_DONE register structure. The register is 32 bits wide, with bits 31-6 reserved. Bits 5-0 are control bits: 5 (Reset), 4 (OUT_DONE), 3 (OUT_DONE_INT_ST), 2 (OUT_DONE_ERR_CH), 1 (OUT_DONE_INT_ST), and 0 (OUT_DONE_ERR_CH).

31	6	5	4	3	2	1	0
(reserved)						Reset	OUT_DONE
(reserved)						OUT_DONE_INT_ST	OUT_DONE_ERR_CH

AHB_DMA_OUT_DONE_CH n _INT_ST The masked interrupt status of AHB_DMA_OUT_DONE_CH n _INT. (RO)

AHB_DMA_OUT_EOF_CH n _INT_ST The masked interrupt status of AHB_DMA_OUT_EOF_CH n _INT.
(RO)

AHB_DMA_OUT_DSCR_ERR_CH <i>n</i> _INT_ST	The masked interrupt status of AHB_DMA_OUT_DSCR_ERR_CH
AHB_DMA_OUT_DSCR_ERR_CH <i>n</i> _INT	AHB_DMA_OUT_DSCR_ERR_CH
AHB_DMA_OUT_DSCR_ERR_CH <i>n</i> _INT	AHB_DMA_OUT_DSCR_ERR_CH

AHB_DMA_OUT_TOTAL_EOF_CH <i>n</i> _INT_ST	The masked interrupt status of AHB_DMA_OUT_TOTAL_EOF_CH <i>n</i> _INT. (RO)

AHB_DMA_OUTFIFO_OVF_CH n _INT_ST The masked interrupt status of AHB_DMA_OUTFIFO_OVF_CH n _INT. (RO)

AHB_DMA_OUTFIFO_UDF_CH n _INT_ST The masked interrupt status of AHB DMA_OUTFIFO_UDF_CH n _INT. (RO)

Register 3.7. AHB_DMA_OUT_INT_ENA_CH n _REG (n : 0-2) (0x0038+0x10* n)

(reserved)

31 6 5 4 3 2 1 0

Reset

AHB_DMA_OUT_DONE_CH n _INT_ENA Write 1 to enable AHB_DMA_OUT_DONE_CH n _INT. (R/W)

AHB_DMA_OUT_EOF_CH_n_INT_ENA Write 1 to enable AHB_DMA_OUT_EOF_CH_n_INT. (R/W)

AHB_DMA_OUT_DSCR_ERR_CHn_INT_ENA Write 1 to enable AHB_DMA_OUT_DSCR_ERR_CHn_INT.
(R/W)

AHB_DMA_OUT_TOTAL_EOF_CH n _INT_ENA Write 1 to enable AHB_DMA_OUT_TOTAL_EOF_CH n _INT.
(R/W)

AHB_DMA_OUTFIFO_OVF_CH**n**_INT_ENA Write 1 to enable AHB_DMA_OUTFIFO_OVF_CH**n**_INT.
(R/W)

AHB_DMA_OUTFIFO_UDF_CH n _INT_ENA Write 1 to enable AHB_DMA_OUTFIFO_UDF_CH n _INT.
(R/W)

Register 3.8. AHB_DMA_OUT_INT_CLR_CH n _REG (n : 0-2) (0x003C+0x10* n)

[illegible]

AHB_DMA_OUT_DONE_CH_n_INT_CLR Write 1 to clear AHB_DMA_OUT_DONE_CH_n_INT. (WT)

AHB DMA OUT EOF CH_n INT CLR Write 1 to clear AHB DMA OUT EOF CH_n INT. (WT)

AHB_DMA_OUT_DSCR_ERR_CH n _INT_CLR Write 1 to clear AHB_DMA_OUT_DSCR_ERR_CH n _INT.
(WT)

AHB_DMA_OUT_TOTAL_EOF_CH n _INT_CLR Write 1 to clear AHB_DMA_OUT_TOTAL_EOF_CH n _INT.
(WT)

AHB_DMA_OUTFIFO_OVF_CH_n_INT_CLR Write 1 to clear AHB_DMA_OUTFIFO_OVF_CH_n_INT. (WT)

AHB_DMA_OUTFIFO_UDF_CH_n_INT_CLR Write 1 to clear AHB_DMA_OUTFIFO_UDF_CH_n_INT. (WT)

Register 3.9. AHB_DMA_AHB_TEST_REG (0x0060)

[illegible]

AHB_DMA_AHB_TESTMODE Reserved. (R/W)

AHB_DMA_AHB_TESTADDR Reserved. (R/W)

Register 3.10. AHB_DMA_MISC_CONF_REG (0x0064)

(reserved)																								AHB_DMA_CLK_EN AHB_DMA_ARB_PRI_DIS (reserved) AHB_DMA_AHBM_RST_INTER							
31																								4	3	2	1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

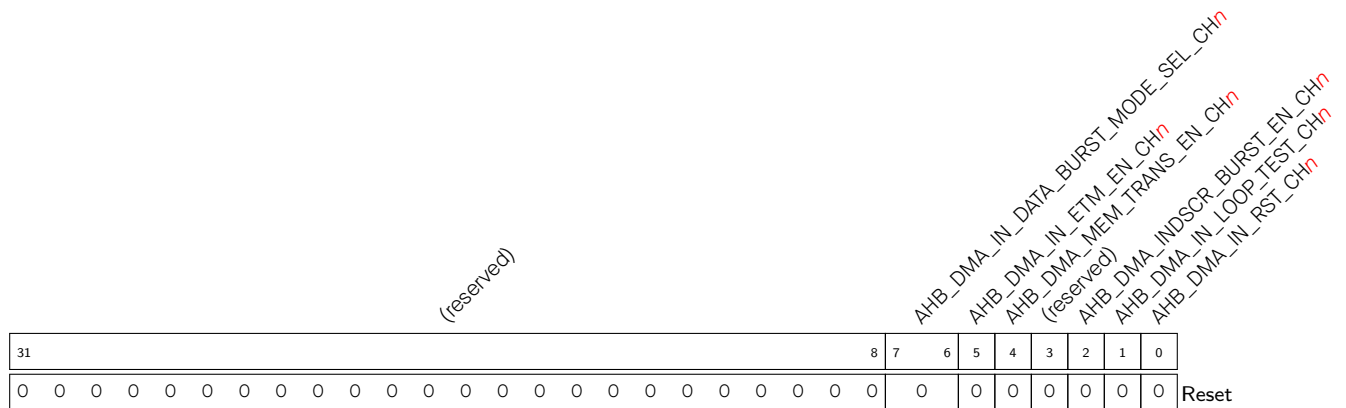
AHB_DMA_AHBM_RST_INTER Write 1 and then 0 to reset the internal AHB FSM. (R/W)

AHB_DMA_ARB_PRI_DIS Configures whether to disable the priority arbitration.

- 0: Enable
 - 1: Disable
- (R/W)

AHB_DMA_CLK_EN Configures AHB DMA clock gating.

- 0: Support clock only when the application writes registers
 - 1: Always force the clock on for registers
- (R/W)

Register 3.11. AHB_DMA_IN_CONFO_CH n _REG (n : 0-2) (0x0070+0xC0* n)

AHB_DMA_IN_RST_CH n Write 1 and then 0 to reset RX channel n FSM and RX FIFO pointer. (R/W)

AHB_DMA_IN_LOOP_TEST_CH n Configures the owner bit value for inlink write-back. (R/W)

AHB_DMA_INDSCR_BURST_EN_CH n Configures whether to enable INCR burst transfer for RX channel n to read descriptors.

0: Disable

1: Enable

(R/W)

AHB_DMA_MEM_TRANS_EN_CH n Configures whether to enable memory-to-memory data transfer.

0: Disable

1: Enable

(R/W)

AHB_DMA_IN_ETM_EN_CH n Configures whether to enable ETM control for RX channel n .

0: Disable

1: Enable

(R/W)

AHB_DMA_IN_DATA_BURST_MODE_SEL_CH n Configures maximum burst length for RX channel n .

0: SINGLE

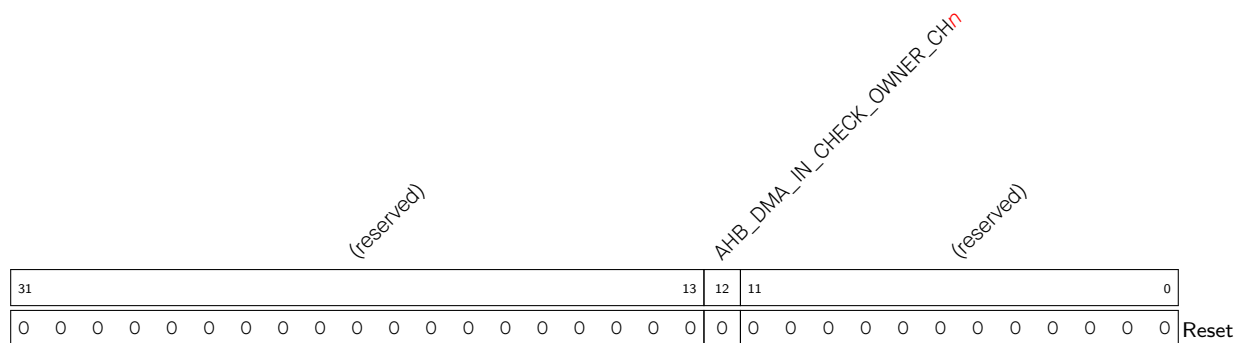
1: INCR4

2: INCR8

3: Reserved

(R/W)

Register 3.12. AHB_DMA_IN_CONF1_CH n _REG (n : 0-2) (0x0074+0xC0* n)



AHB_DMA_IN_CHECK_OWNER_CHn Configures whether to enable owner bit check for RX channel

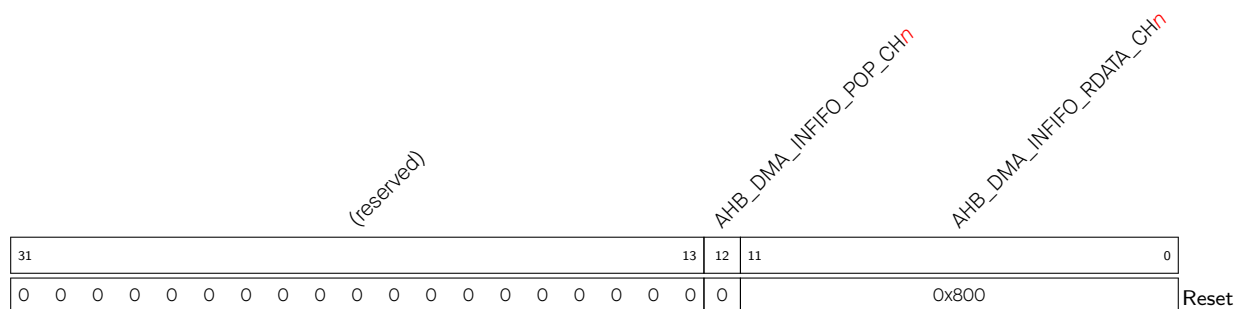
n .

0: Disable

1: Enable

(R/W)

Register 3.13. AHB_DMA_IN_POP_CH n _REG (n : 0-2) (0x007C+0xC0* n)



AHB_DMA_INFIFO_RDATA_CH_n Represents the data popped from AHB DMA RX FIFO. (RO)

AHB_DMA_INFIFO_POP_CH_n Configures whether to pop data from AHB DMA RX FIFO.

0: Invalid. No effect

1: Pop

(WT)

Register 3.14. AHB_DMA_IN_LINK_CH_{*n*}_REG (*n*: 0-2) (0x0080+0xC0**n*)

(reserved)																	5	4	3	2	1	0
																		1	0	0	0	1

Reset

AHB_DMA_INLINK_AUTO_RET_CH*n* Configures whether to return to the current receive descriptor's address when there are some errors in current receiving data.

0: Not return

1: Return

(R/W)

AHB_DMA_INLINK_STOP_CH*n* Configures whether to stop RX channel *n* from receiving data.

0: Invalid. No effect

1: Stop

(WT)

AHB_DMA_INLINK_START_CH*n* Configures whether to enable RX channel *n* for data transfer.

0: Disable

1: Enable

(WT)

AHB_DMA_INLINK_RESTART_CH n Configures whether to restart RX channel n for AHB DMA transfer.

0: Invalid. No effect

1: Restart

(WT)

AHB_DMA_INLINK_PARK_CH n Represents the status of the receive descriptor's FSM.

0: Running

1: Idle

(RO)

Register 3.15. AHB_DMA_OUT_CONFO_CH n _REG (n : 0-2) (0x00D0+0xC0*(n -1))

(reserved)																				AHB_DMA_OUT_DATA_BURST_MODE_SEL_CH _n						(reserved)						AHB_DMA_OUT_ETM_EN_CH _n						(reserved)						AHB_DMA_OUTDSR_BURST_EN_CH _n						AHB_DMA_OUT_EOF_MODE_CH _n						AHB_DMA_OUT_AUTO_WBACK_CH _n						AHB_DMA_OUT_LOOP_TEST_CH _n						AHB_DMA_OUT_RST_CH _n					
31																				10						9	8	7	6	5	4	3	2	1	0	Reset																																					
0																				0						0	0	0	0	0	1	0	0	0	0																																						

AHB_DMA_OUT_RST_CH n Configures the reset state of TX channel n FSM and TX FIFO pointer.

0: Release reset

1: Reset

(R/W)

AHB_DMA_OUT_LOOP_TEST_CH n Configures the owner bit value for outlink write-back. (R/W)

AHB_DMA_OUT_AUTO_WBACK_CHn Configures whether to enable automatic outlink write-back when all the data in TX FIFO has been transmitted.

0: Disable

1: Enable

(R/W)

AHB_DMA_OUT_EOF_MODE_CH_n Configures when to generate EOF flag.

0: EOF flag for TX channel *n* is generated when data to be transmitted has been pushed into FIFO in AHB DMA.

1: EOF flag for TX channel *n* is generated when data to be transmitted has been popped from FIFO in AHB DMA.

(R/W)

AHB_DMA_OUTDSCR_BURST_EN_CH n Configures whether to enable INCR burst transfer for TX channel n reading descriptors.

0: Disable

1: Enable

(R/W)

AHB_DMA_OUT_ETM_EN_CH n Configures whether to enable ETM control for TX channel n .

0: Disable

1: Enable

(R/W)

AHB_DMA_OUT_DATA_BURST_MODE_SEL_CH n Configures maximum burst length for TX channel n .

0: SINGLE

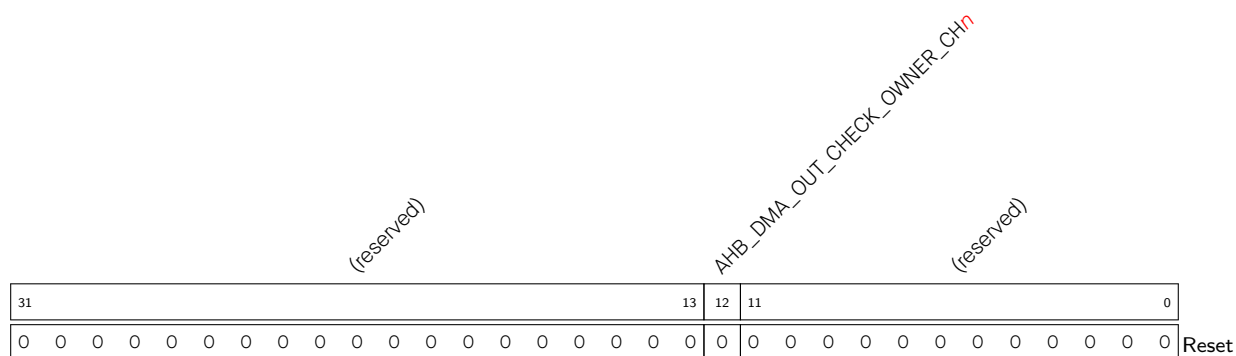
1: INCR4

2: INCR8

3: Reserved

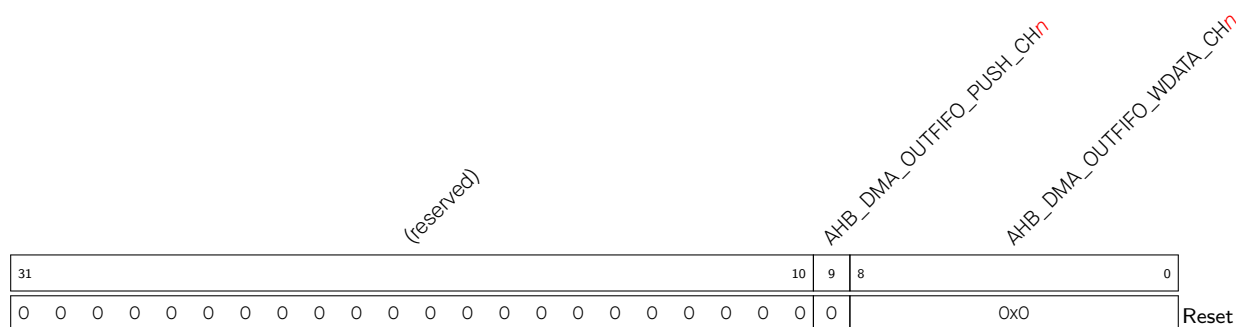
 (R/W)

Register 3.16. AHB_DMA_OUT_CONF1_CH n _REG (n : 0-2) (0x00D4+0xC0* n)



AHB_DMA_OUT_CHECK_OWNER_CH*n* Configures whether to enable owner bit check for TX channel *n*.
0: Disable
1: Enable
(R/W)

Register 3.17. AHB_DMA_OUT_PUSH_CH n _REG (n : 0-2) (0x00DC+0xC0* n)



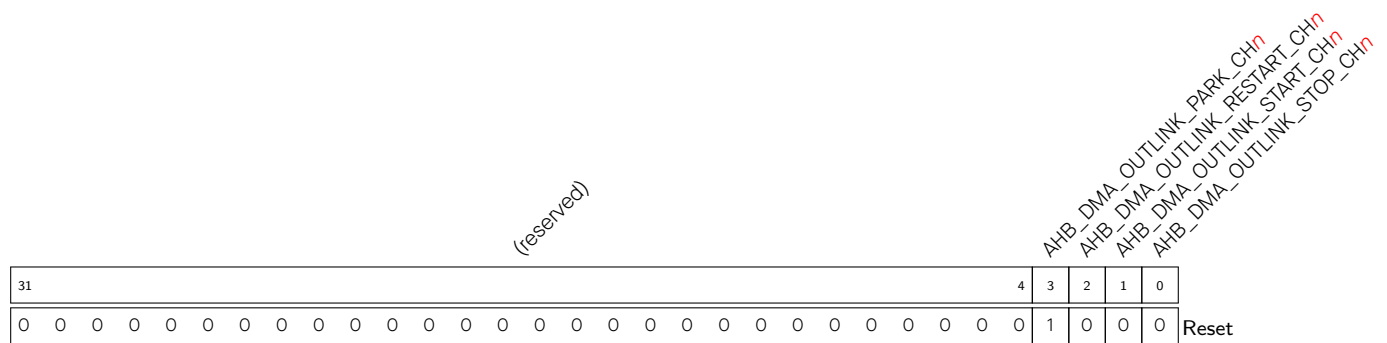
AHB_DMA_OUTFIFO_WDATA_CH n Represents the data that need to be pushed into AHB DMA TX FIFO. (R/W)

AHB_DMA_OUTFIFO_PUSH_CH_n Configures whether to push data into AHB DMA TX FIFO.

- 0: Invalid. No effect
- 1: Push

(WT)

Register 3.18. AHB_DMA_OUT_LINK_CH n _REG (n : 0-2) (0x00E0+0xC0* n)



AHB_DMA_OUTLINK_STOP_CH*n* Configures whether to stop TX channel *n* from transmitting data.

0: Invalid. No effect

1: Stop

(WT)

AHB_DMA_OUTLINK_START_CH*n* Configures whether to enable TX channel *n* for data transfer.

0: Disable

1: Enable

(WT)

AHB_DMA_OUTLINK_RESTART_CH*n* Configures whether to restart TX channel *n* for AHB DMA transfer.

0: Invalid. No effect

1: Restart

(WT)

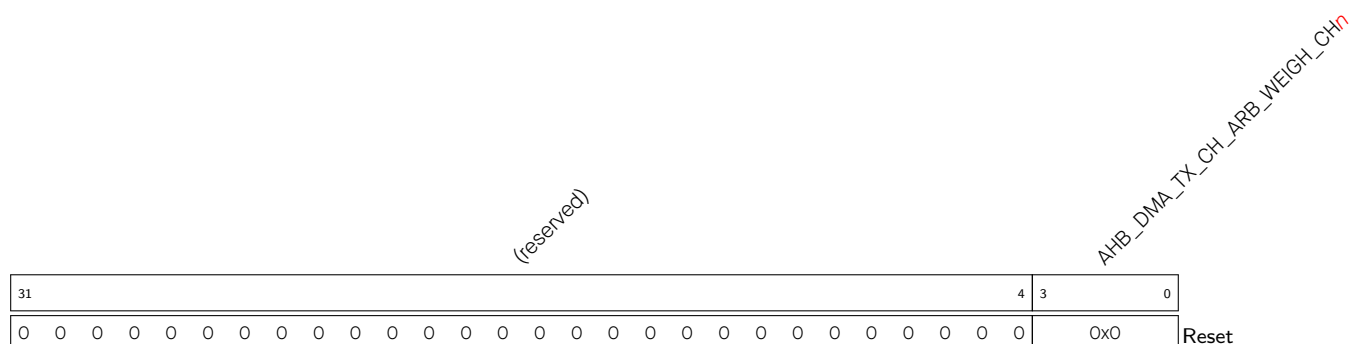
AHB_DMA_OUTLINK_PARK_CH n Represents the status of the transmit descriptor's FSM.

0: Running

1: Idle

(RO)

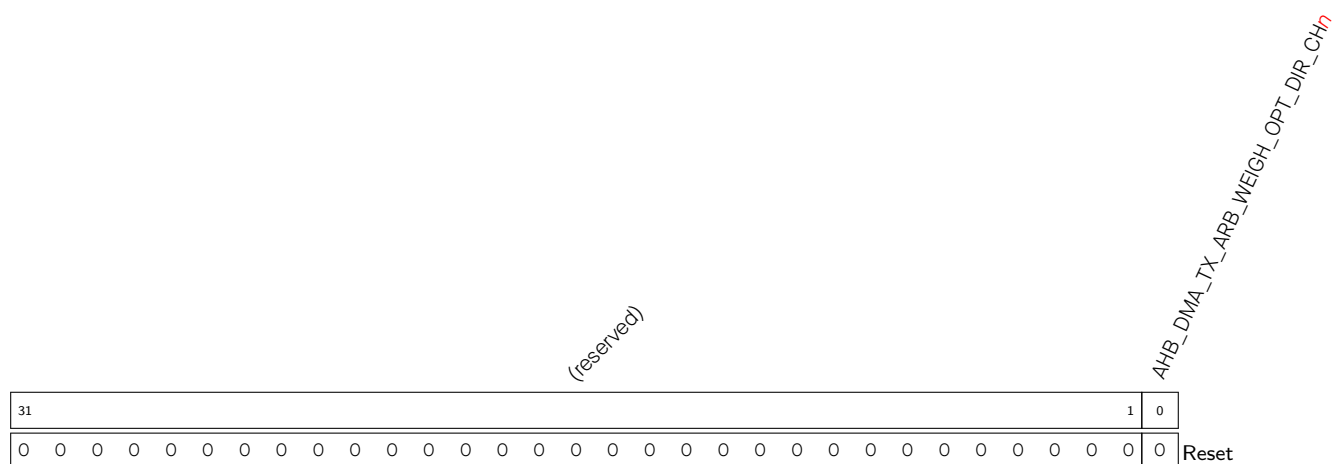
Register 3.19. AHB_DMA_TX_CH_ARB_WEIGHT_CH n _REG (n : 0-2) (0x02DC+0x28* n)



AHB_DMA_TX_CH_ARB_WEIGHT_CH*n* Configures the weight (i.e the number of tokens) of TX channel*n*.

Value range: 0 ~ 15. (R/W)

Register 3.20. AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CH n _REG (n : 0-2) (0x02E0+0x28* n)



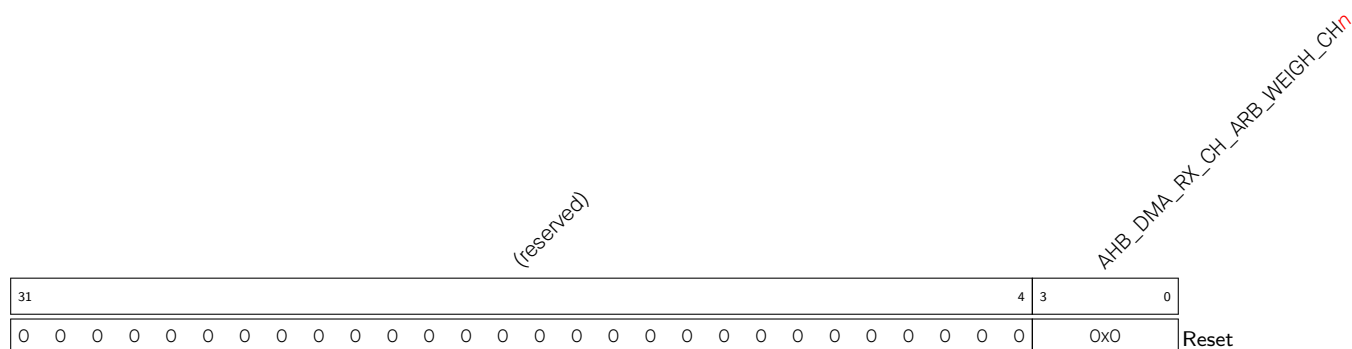
AHB_DMA_TX_ARB_WEIGHT_OPT_DIR_CH n Configures whether to enable weight optimization for TX channel n .

0: Disable

1: Enable

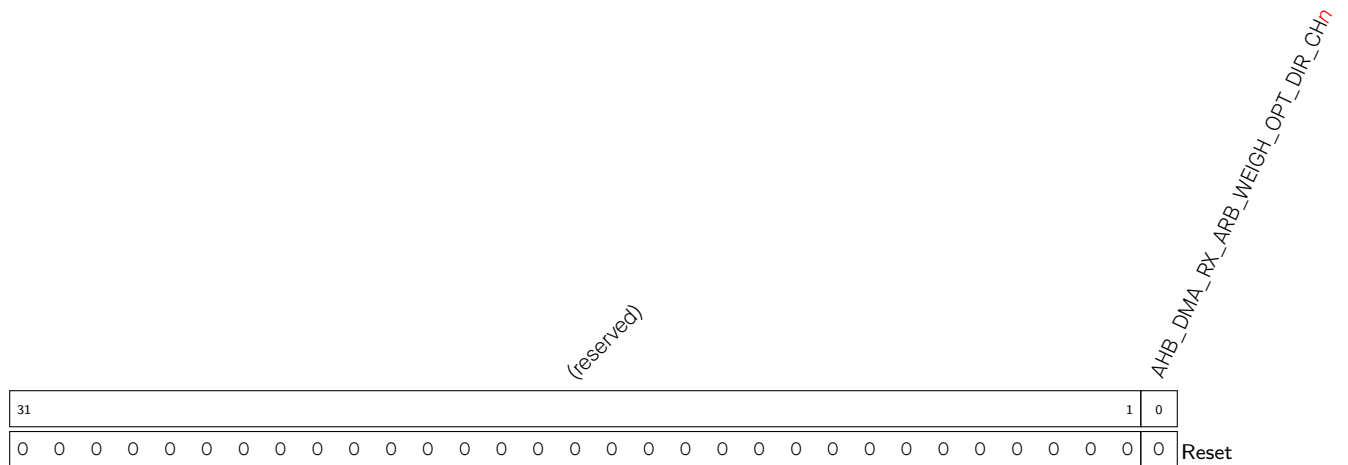
(R/W)

Register 3.21. AHB_DMA_RX_CH_ARB_WEIGHT_CH n _REG (n : 0-2) (0x0354+0x28* n)



AHB_DMA_RX_CH_ARB_WEIGHT_CH*n* Configures the weight (i.e the number of tokens) of RX channel *n*.

Value range: 0 ~ 15. (R/W)

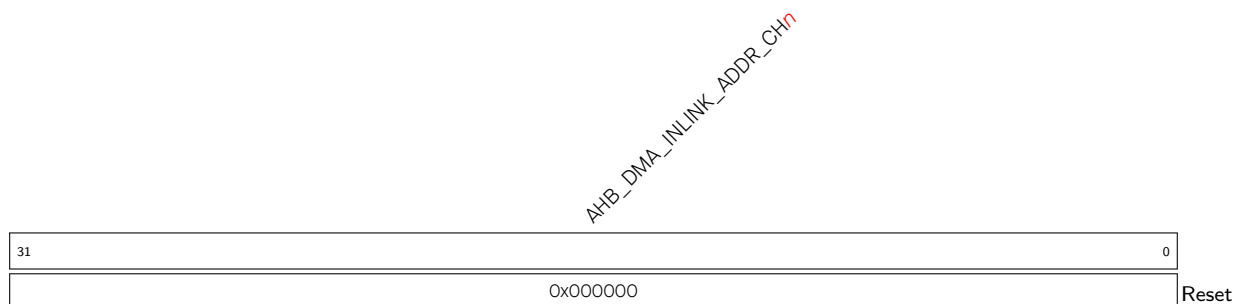
Register 3.22. AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CH n _REG (n : 0-2) (0x0358+0x28* n)

AHB_DMA_RX_ARB_WEIGHT_OPT_DIR_CH n Configures whether to enable weight optimization for RX channel n .

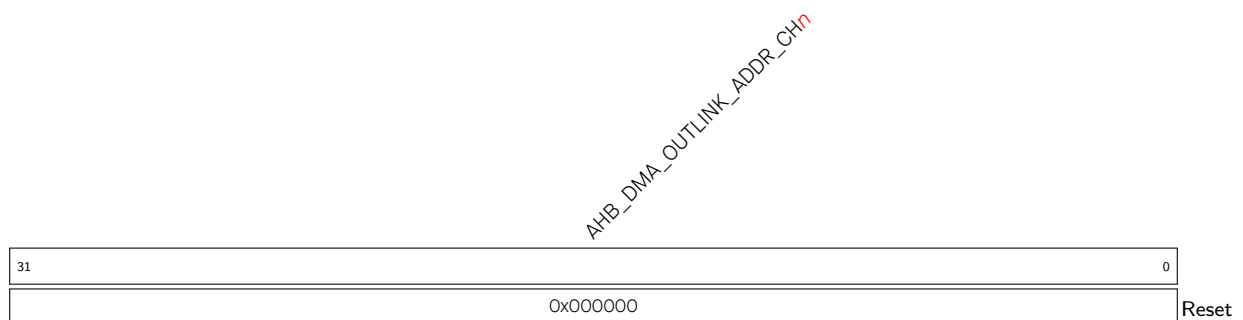
0: Disable

1: Enable

(R/W)

Register 3.23. AHB_DMA_IN_LINK_ADDR_CH n _REG (n : 0-2) (0x03AC+0x4* n)

AHB_DMA_INLINK_ADDR_CH n Represents the first receive descriptor's address. (R/W)

Register 3.24. AHB_DMA_OUT_LINK_ADDR_CH n _REG (n : 0-2) (0x03B8+0x4* n)

AHB_DMA_OUTLINK_ADDR_CH n Represents the first transmit descriptor's address. (R/W)

Register 3.25. AHB_DMA_INTR_MEM_START_ADDR_REG (0x03C4)

AHB_DMA_ACCESS_INTR_MEM_START_ADDR																															
31																															0
0x000000																															
Reset																															

AHB_DMA_ACCESS_INTR_MEM_START_ADDR Configures the start address of accessible address space. (R/W)

Register 3.26. AHB_DMA_INTR_MEM_END_ADDR_REG (0x03C8)

AHB_DMA_ACCESS_INTR_MEM_END_ADDR																															
31																															0
0xffffffff																															
Reset																															

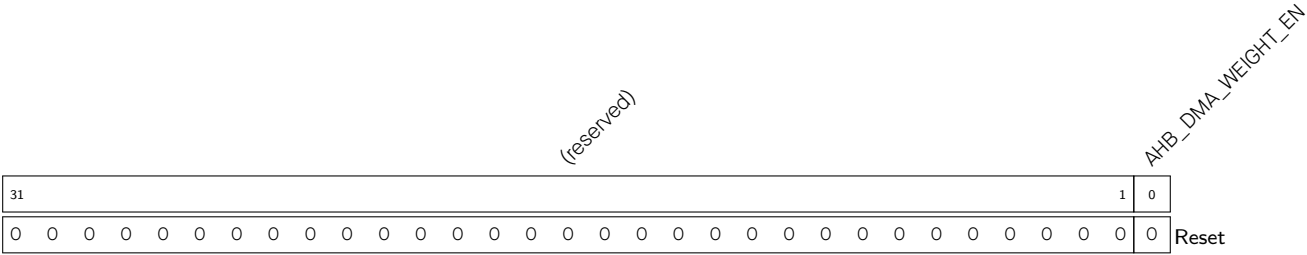
AHB_DMA_ACCESS_INTR_MEM_END_ADDR Configures the end address of accessible address space. (R/W)

Register 3.27. AHB_DMA_ARB_TIMEOUT_REG (0x03DC)

(reserved)																AHB_DMA_ARB_TIMEOUT																																																																																																																																																															
31																16																15																0																																																																																																																															
0																0																0																0																0																0																0																0																0																0x00																Reset															

AHB_DMA_ARB_TIMEOUT Configures the time slot of weight arbitration. Measurement unit: AHB bus clock cycle. (R/W)

Register 3.28. AHB_DMA_WEIGHT_EN_REG (0x0400)



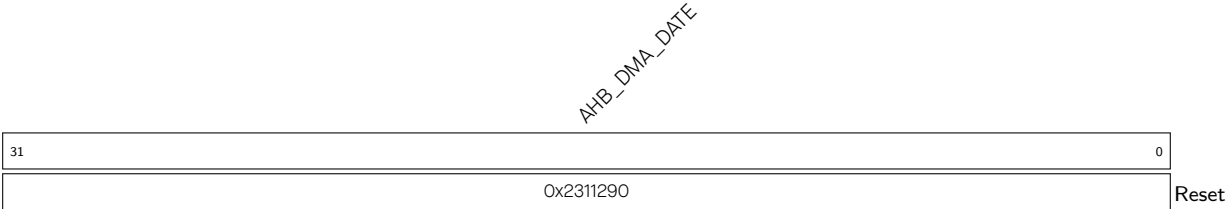
AHB_DMA_WEIGHT_EN Configures whether to enable weight arbitration.

0: Disable

1: Enable

(R/W)

Register 3.29. AHB_DMA_DATE_REG (0x0068)



AHB_DMA_DATE Version control register. (R/W)

Register 3.30. AHB_DMA_INFIFO_STATUS_CH n _REG (n : 0-2) (0x0078+0xC0* n)

(reserved)				AHB_DMA_IN_BUF_HUNGRY_CH _n				AHB_DMA_IN_REMAIN_UNDER_4B_CH _n				AHB_DMA_IN_REMAIN_UNDER_3B_CH _n				AHB_DMA_IN_REMAIN_UNDER_2B_CH _n				AHB_DMA_IN_REMAIN_UNDER_1B_CH _n				(reserved)				AHB_DMA_INFIFO_CNT_CH _n				(reserved)				AHB_DMA_INFIFO_CNT_CH _n						
31	28	27	26	25	24	23	22	15	14	8	7	2	1	0	Reset																											
0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1																

AHB_DMA_INFIFO_FULL_CH_n Represents whether L1 RX FIFO is full.

0: Not Full

1: Full

(RO)

AHB_DMA_INFIFO_EMPTY_CH_n Represents whether L1 RX FIFO is empty.

0: Not empty

1: Empty

(RO)

AHB_DMA_INFIFO_CNT_CH n Represents the number of data bytes in L1 RX FIFO for RX channel n .

(RO)

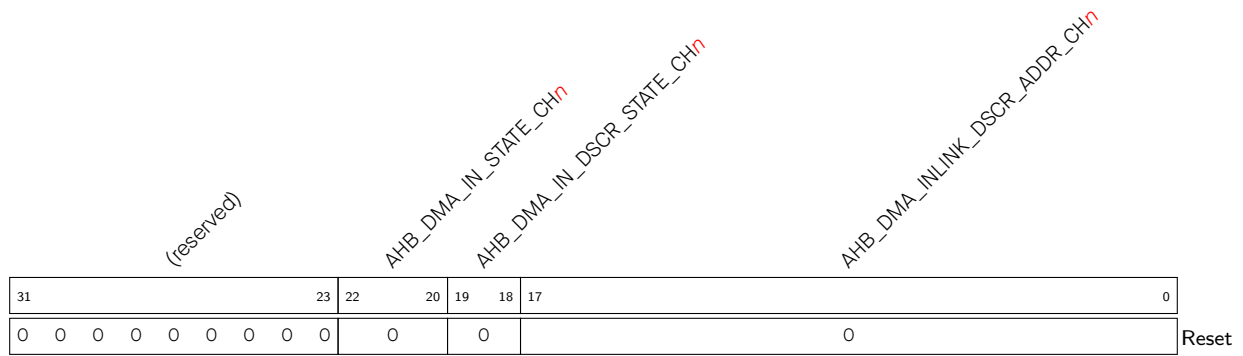
AHB DMA IN REMAIN UNDER 1B CH Reserved. (RO)

AHB_DMA_IN_REMAIN_UNDER_2B_CHn Reserved. (RO)

AHB_DMA_IN_REMAIN_UNDER_3B_CH_n Reserved. (RO)

AHB DMA IN REMAIN UNDER 4B CH₇ Reserved. (RO)

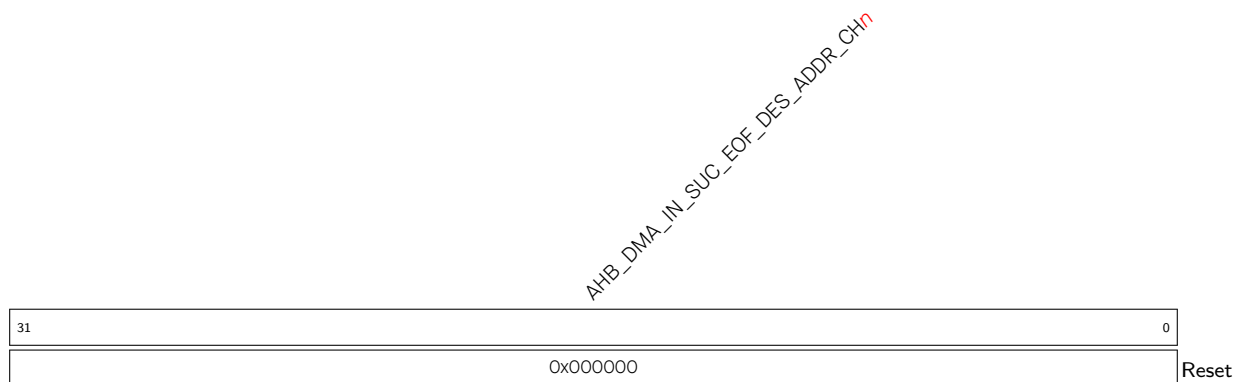
AHB_DMA_IN_BUF_HUNGRY_CH_n Reserved. (RO)

Register 3.31. AHB_DMA_IN_STATE_CH n _REG (n : 0-2) (0x0084+0xC0* n)

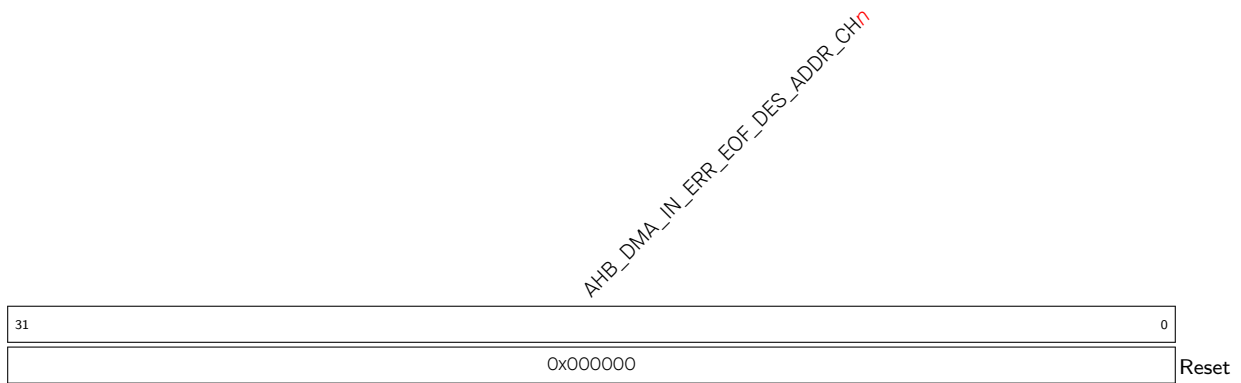
AHB_DMA_INLINK_DSCR_ADDR_CH n Represents the lower 18 bits of the next receive descriptor address that is pre-read (but not processed yet). If the current receive descriptor is the last descriptor, then this field represents the address of the current receive descriptor. (RO)

AHB_DMA_IN_DSCR_STATE_CH n Reserved. (RO)

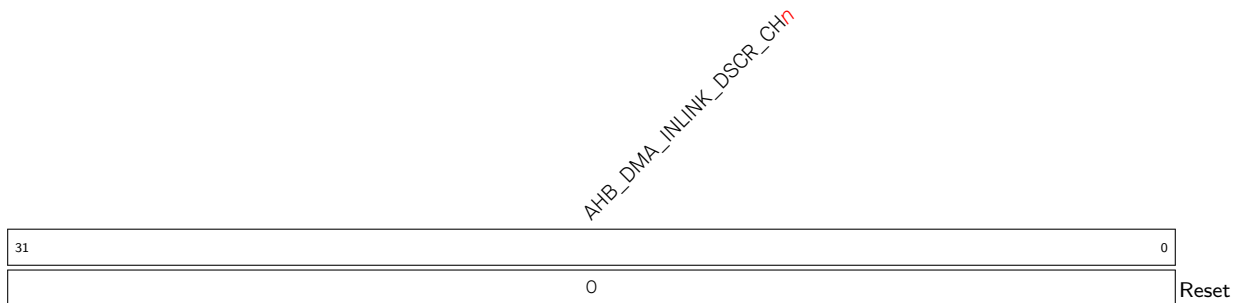
AHB_DMA_IN_STATE_CH n Reserved. (RO)

Register 3.32. AHB_DMA_IN_SUC_EOF_DES_ADDR_CH n _REG (n : 0-2) (0x0088+0xC0* n)

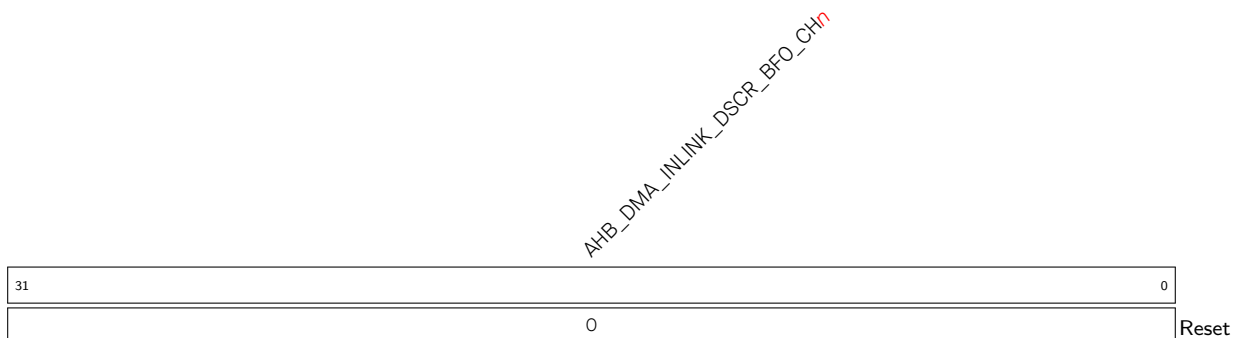
AHB_DMA_IN_SUC_EOF_DES_ADDR_CH n Represents the address of the receive descriptor when the EOF bit in this descriptor is 1. (RO)

Register 3.33. AHB_DMA_IN_ERR_EOF_DES_ADDR_CH_{*n*}_REG (*n*: 0-2) (0x008C+0xC0n*)**

AHB_DMA_IN_ERR_EOF_DES_ADDR_CH_{*n*} Represents the address of the receive descriptor when there are some errors in the currently received data. Valid only for UHCI. (RO)

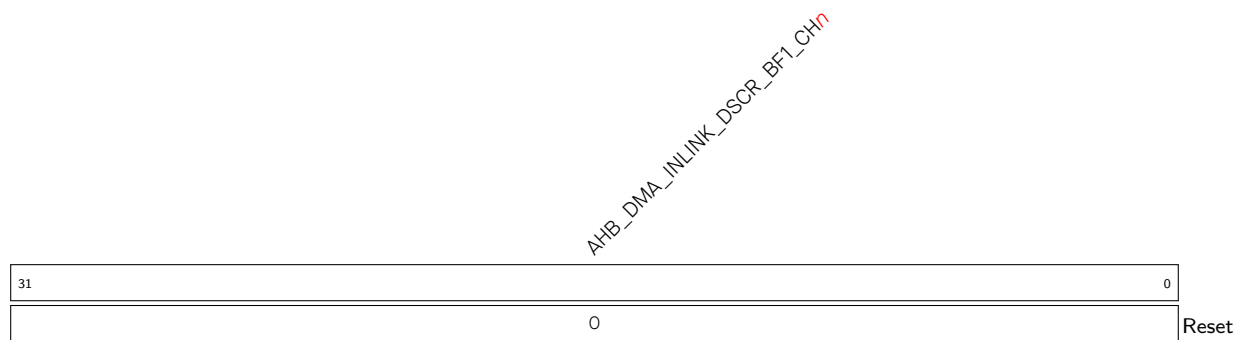
Register 3.34. AHB_DMA_INLINK_DSCR_CH_{*n*}_REG (*n*: 0-2) (0x0090+0xC0n*)**

AHB_DMA_INLINK_DSCR_CH_{*n*} Represents the address of the next receive descriptor x+1 pointed by the current receive descriptor that is pre-read. (RO)

Register 3.35. AHB_DMA_IN_DSCR_BFO_CH_{*n*}_REG (*n*: 0-2) (0x0094+0xC0n*)**

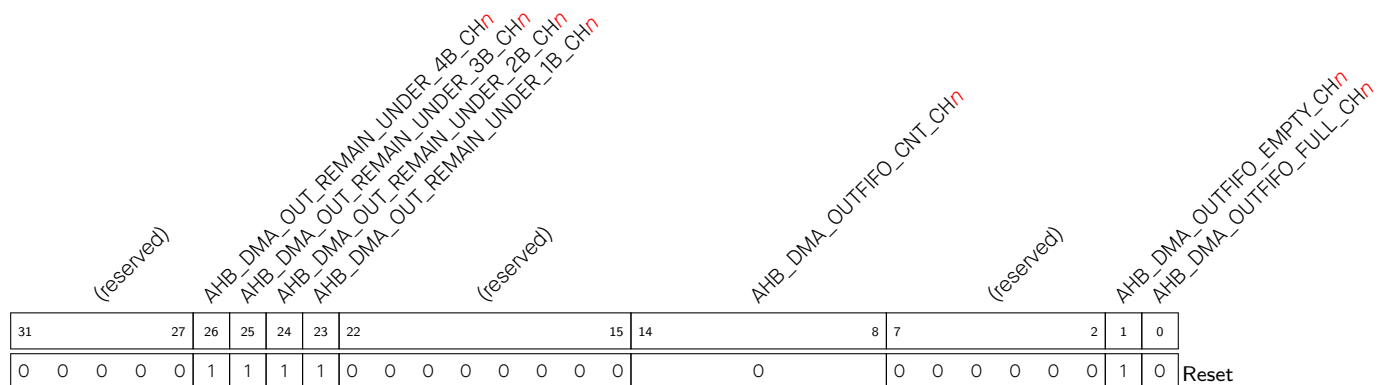
AHB_DMA_INLINK_DSCR_BFO_CH_{*n*} Represents the address of the current receive descriptor x that is pre-read. (RO)

Register 3.36. AHB_DMA_IN_DSCR_BF1_CH n _REG (n : 0-2) (0x0098+0xC0* n)



AHB_DMA_INLINK_DSCR_BF1_CH n Represents the address of the previous receive descriptor x-1 that is pre-read. (RO)

Register 3.37. AHB_DMA_OUTFIFO_STATUS_CH n _REG (n : 0-2) (0x00D8+0xC0* n)



AHB_DMA_OUTFIFO_FULL_CH_n Represents whether L1 TX FIFO is full.

0: Not Full

1: Full

(RO)

AHB_DMA_OUTFIFO_EMPTY_CH_n Represents whether L1 TX FIFO is empty.

0: Not empty

1: Empty

(RO)

AHB_DMA_OUTFIFO_CNT_CH*n* Represents the number of data bytes in L1 TX FIFO for TX channel *n*. (RO)

AHB_DMA_OUT_REMAIN_UNDER_1B_CHn Reserved. (RO)

AHB_DMA_OUT_REMAIN_UNDER_2B_CH_n Reserved. (RO)

AHB_DMA_OUT_REMAIN_UNDER_3B_CH_n Reserved. (RO)

AHB_DMA_OUT_REMAIN_UNDER_4B_CH_n Reserved. (RO)

Register 3.38. AHB_DMA_OUT_STATE_CH_{*n*}_REG (*n*: 0-2) (0x00E4+0xC0n*)**

(reserved)										AHB_DMA_OUT_STATE_CH _n				AHB_DMA_OUT_DSCR_STATE_CH _n				AHB_DMA_OUTLINK_DSCR_ADDR_CH _n																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
31									23	22			20	19	18	17											0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Reset

AHB_DMA_OUTLINK_DSCR_ADDR_CH_{*n*} Represents the lower 18 bits of the next transmit descriptor address that is pre-read (but not processed yet). If the current transmit descriptor is the last descriptor, then this field represents the address of the current transmit descriptor. (RO)

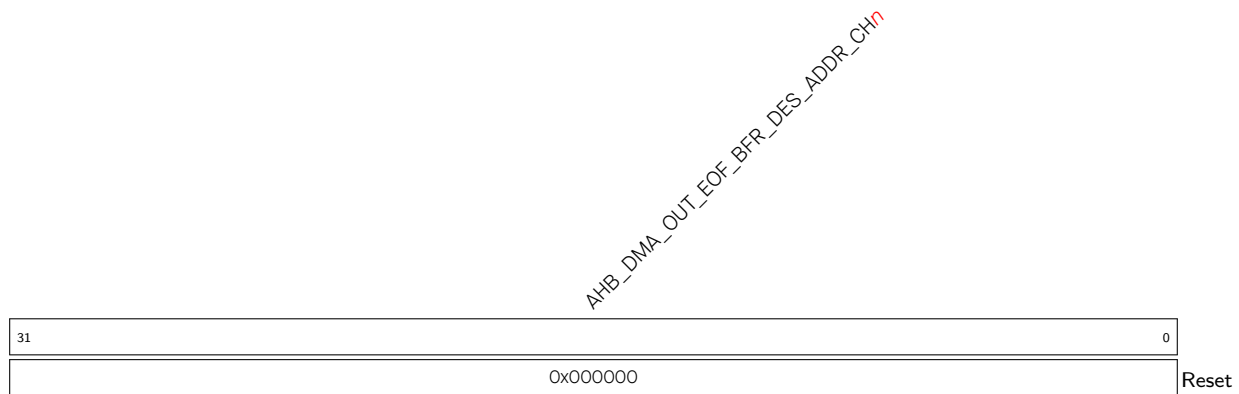
AHB_DMA_OUT_DSCR_STATE_CH_{*n*} Reserved. (RO)

AHB_DMA_OUT_STATE_CH_{*n*} Reserved. (RO)

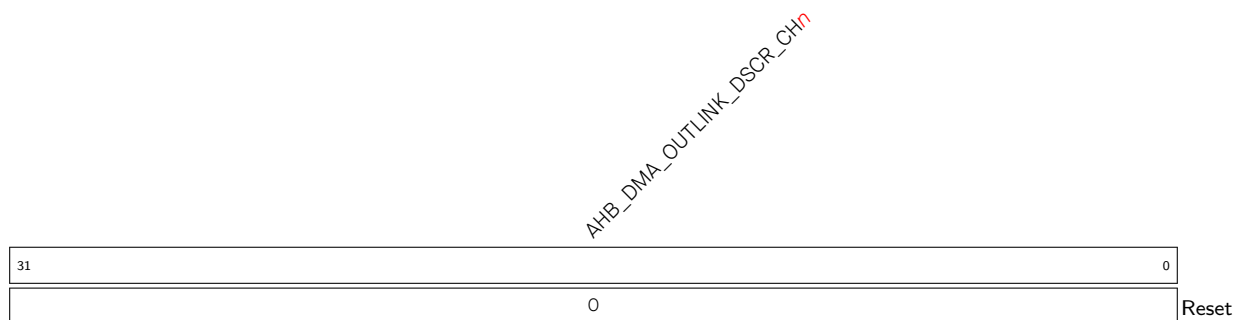
Register 3.39. AHB_DMA_OUT_EOF_DES_ADDR_CH_{*n*}_REG (*n*: 0-2) (0x00E8+0xC0n*)**

AHB_DMA_OUT_EOF_DES_ADDR_CH ⁿ																															
31																															0
0x000000																															
Reset																															

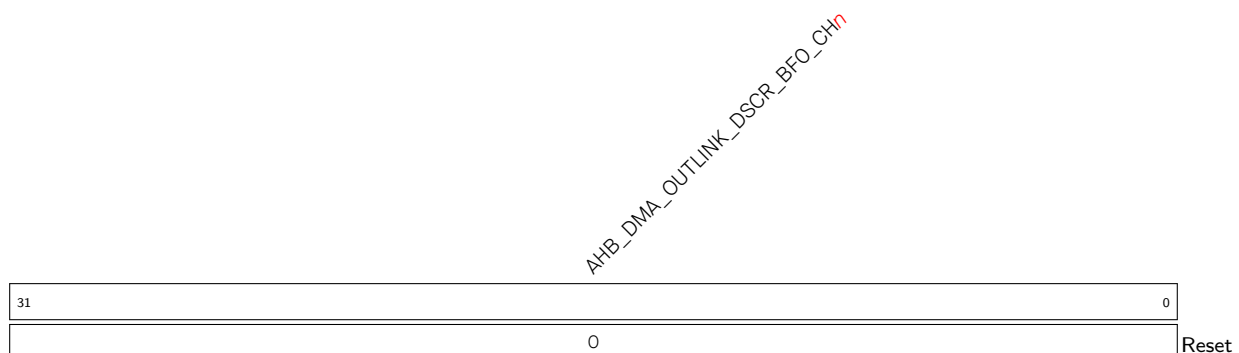
AHB_DMA_OUT_EOF_DES_ADDR_CH_{*n*} Represents the address of the transmit descriptor when the EOF bit in this descriptor is 1. (RO)

Register 3.40. AHB_DMA_OUT_EOF_BFR_DES_ADDR_CH n _REG (n : 0-2) (0x00EC+0xC0* n)

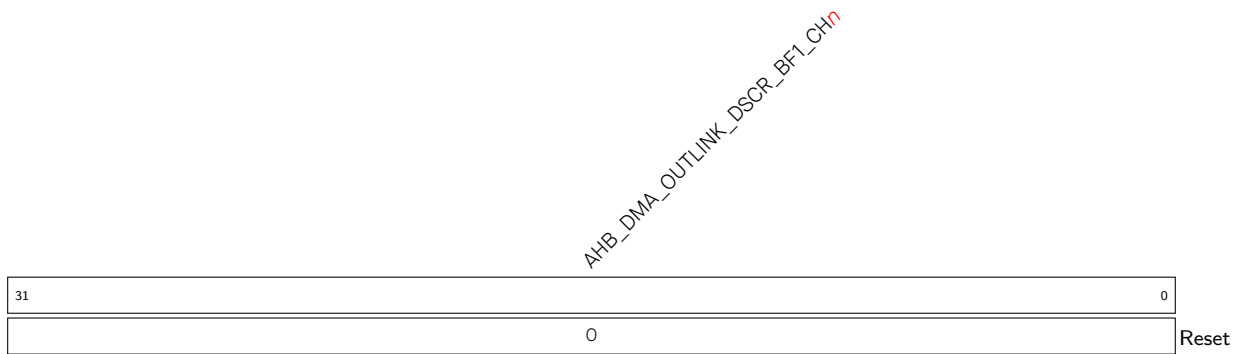
AHB_DMA_OUT_EOF_BFR_DES_ADDR_CH n Represents the address of the transmit descriptor before the last transmit descriptor. (RO)

Register 3.41. AHB_DMA_OUT_DSCR_CH n _REG (n : 0-2) (0x00F0+0xC0* n)

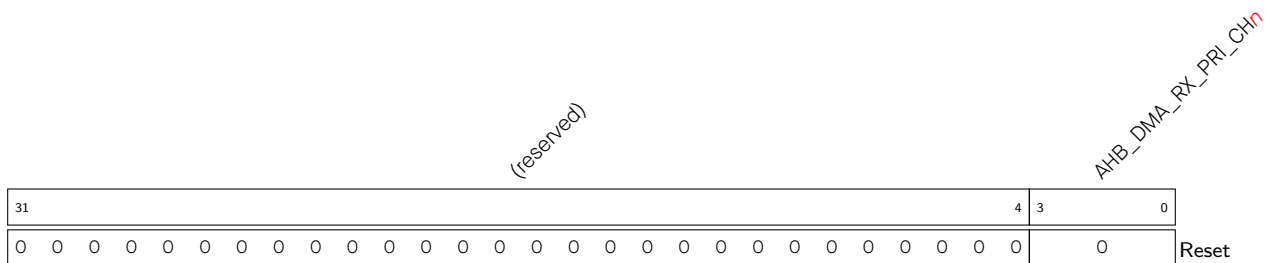
AHB_DMA_OUTLINK_DSCR_CH n Represents the address of the next transmit descriptor $y+1$ pointed by the current transmit descriptor that is pre-read. (RO)

Register 3.42. AHB_DMA_OUT_DSCR_BFO_CH n _REG (n : 0-2) (0x00F4+0xC0* n)

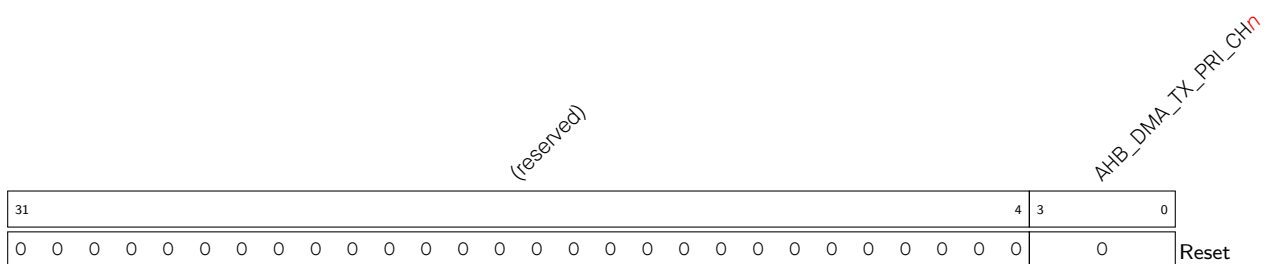
AHB_DMA_OUTLINK_DSCR_BFO_CH n Represents the address of the current transmit descriptor y that is pre-read. (RO)

Register 3.43. AHB_DMA_OUT_DSCR_BF1_CH_{*n*}_REG (*n*: 0-2) (0x00F8+0xC0n*)**

AHB_DMA_OUTLINK_DSCR_BF1_CH_{*n*} Represents the address of the previous transmit descriptor y-1 that is pre-read. (RO)

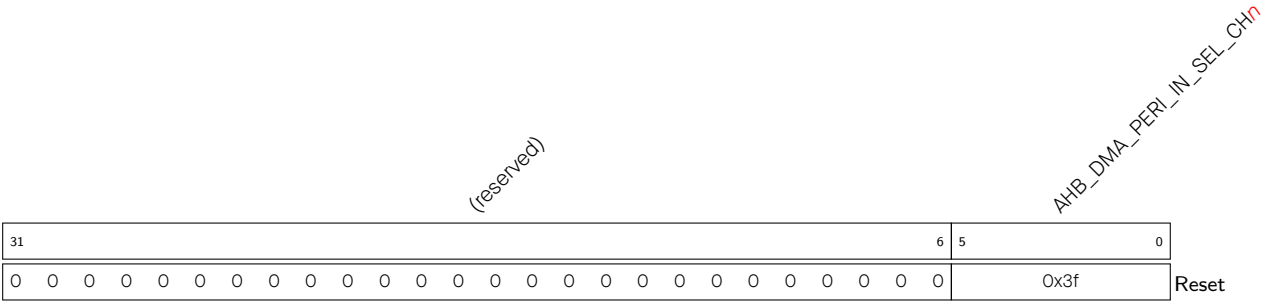
Register 3.44. AHB_DMA_IN_PRI_CH_{*n*}_REG (*n*: 0-2) (0x009C+0xC0n*)**

AHB_DMA_RX_PRI_CH_{*n*} Configures the priority of RX channel *n*. The larger the value, the higher the priority.
Value range: 0 ~ 5 (R/W)

Register 3.45. AHB_DMA_OUT_PRI_CH_{*n*}_REG (*n*: 0-2) (0x00FC+0xC0n*)**

AHB_DMA_TX_PRI_CH_{*n*} Configures the priority of TX channel *n*. The larger the value, the higher the priority.
Value range: 0 ~ 5 (R/W)

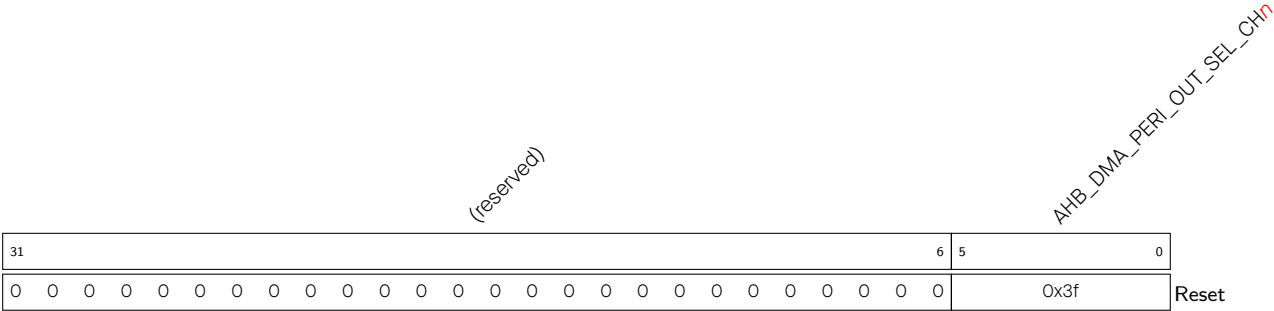
Register 3.46. AHB_DMA_IN_PERI_SEL_CHn_REG (n: 0-2) (0x00A0+0xC0*n)



AHB_DMA_PERI_IN_SEL_CHn Configures the peripheral connected to RX channel n.

- 0: Dummy
 - 1: GP-SPI
 - 2: UHCI
 - 3: I2S
 - 4: Dummy
 - 5: Dummy
 - 6: AES
 - 7: SHA
 - 8: ADC
 - 9: PARLIO
 - 10: Dummy
 - 11 ~ 15: Dummy
 - 16 ~ 63: Invalid
- (R/W)

Register 3.47. AHB_DMA_OUT_PERI_SEL_CHn_REG (n: 0-2) (0x0100+0xC0*n)



AHB_DMA_OUT_PERI_SEL_CHn Configures the peripheral connected to TX channel n.

- 0: Dummy
 - 1: GP-SPI
 - 2: UHCI
 - 3: I2S
 - 4: Dummy
 - 5: Dummy
 - 6: AES
 - 7: SHA
 - 8: ADC
 - 9: PARLIO
 - 10: Dummy
 - 11 ~ 15: Dummy
 - 16 ~ 63: Invalid
- (R/W)

Part II

Memory Organization

This part provides insights into the system's memory structure, discussing the organization and mapping of RAM, ROM, eFuse, and external memories, offering a framework for understanding memory-related subsystems.

Chapter 4

System and Memory

4.1 Overview

ESP32-C5 has an ultra-low power and highly-integrated system with two processors:

- a high-performance 32-bit RISC-V processor (HP CPU), five-stage pipeline, clock frequency up to 240 MHz.
- a low-power 32-bit RISC-V processor (LP CPU), two-stage pipeline, clock frequency up to 40 MHz or 48 MHz (See the note below).

Note:

ESP32-C5 supports selection of crystal frequencies, see Chapter [6 Reset and Clock](#).

All internal memory, external memory, and peripherals are located on the HP CPU and LP CPU buses.

4.2 Features

- **Address Space**
 - 720 KB of internal memory address space, accessible by the instruction bus or data bus
 - 832 KB of peripheral address space
 - 32 MB of virtual address space for external memory, accessible by the instruction bus or data bus
 - 384 KB of internal DMA address space
 - 32 MB of external DMA address space
- **Internal Memory**
 - 320 KB of ROM
 - 384 KB of HP SRAM
 - 16 KB of LP SRAM
- **External Memory**
 - Up to 32 MB of external flash
 - Up to 32 MB of external RAM
- **Peripheral Space**
 - 61 modules/peripherals in total

- **GDMA**
 - 7 GDMA-supported modules/peripherals

4.3 Functional Description

4.3.1 Address Mapping

Figure 4.3-1 illustrates the system structure and address mapping. All the non-reserved addresses are accessible via both the instruction bus and the data bus, meaning that the instruction bus and the data bus share the same address space.

Both the data bus and instruction bus of the HP CPU and LP CPU are little-endian. All buses have a 32-bit data width. HP CPU and LP CPU can access data via the data bus using single-byte, double-byte, and 4-byte alignment.

The HP CPU has the following access capabilities:

- Direct access to internal memory via both the data bus and instruction bus
 - 384 KB HP SRAM (0x4080_0000 ~ 0x4085_FFFF)
 - 16 KB LP SRAM (0x5000_0000 ~ 0x5000_3FFF)
- Access to internal memory via [ROM-Cache](#)
 - 320 KB ROM (0x4000_0000 ~ 0x4004_FFFF)

Note:

Software can access the ROM addresses in two ways: cache-able access or noncache-able access, depending on the control of CPU Physical Memory Attribution (PMA). Cache-able access is done via cache. Noncache-able access is done without cache.

- Access to external memory through the cache

The virtual address is mapped to the physical address space of the external memory through the MMU.

- Up to 32 MB external flash (0x4200_0000 ~ 0x43FF_FFFF)
 - Up to 32 MB external RAM (0x4200_0000 ~ 0x43FF_FFFF)
- Direct access to modules/peripherals via the data bus, including:
 - HP CPU Peripherals
 - HP Peripherals
 - LP Peripherals

The LP CPU has the following access capabilities:

- Direct access to HP SRAM and LP SRAM via both the data bus and instruction bus
 - 384 KB HP SRAM (0x4080_0000 ~ 0x4085_FFFF)
 - 16 KB LP SRAM (0x5000_0000 ~ 0x5000_3FFF)
- Direct access to modules/peripherals via the data bus, including:

- HP Peripherals
- LP Peripherals

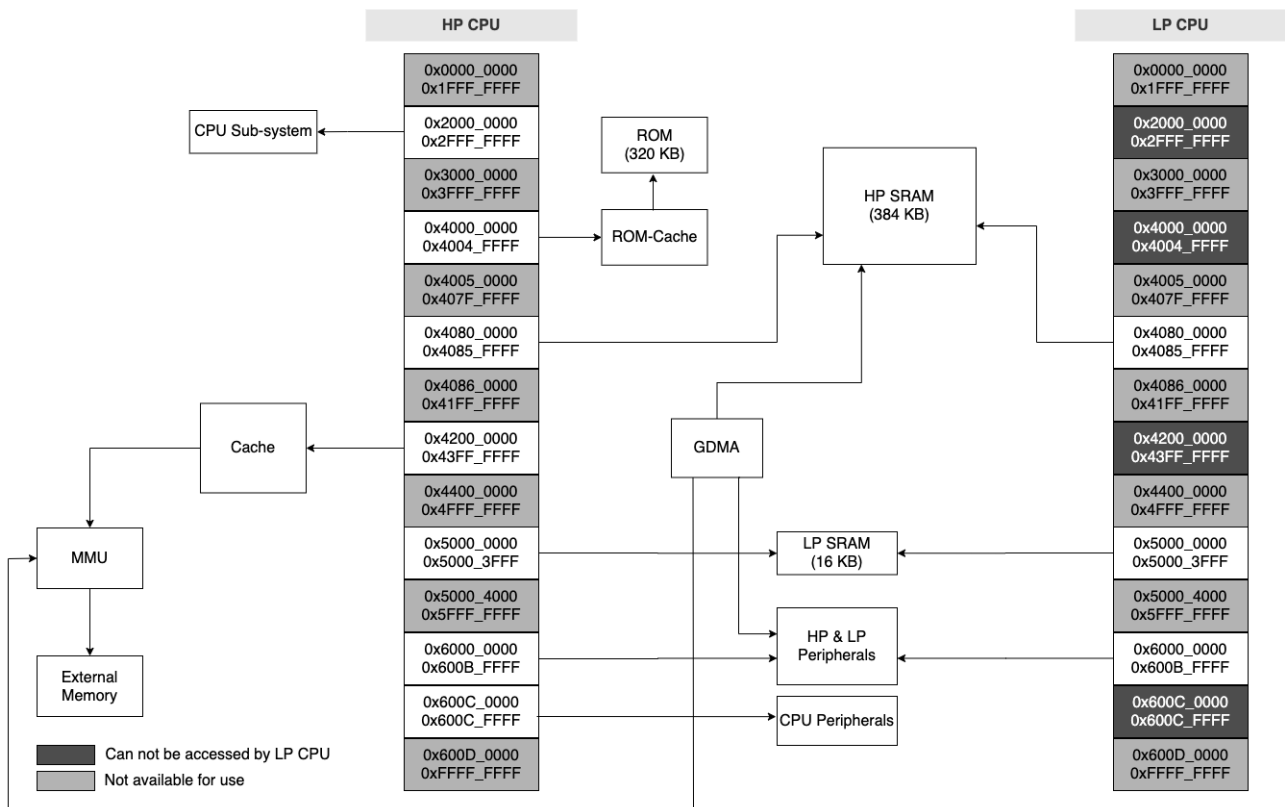


Figure 4.3-1. System Structure and Address Mapping

Note:

- The range of addresses available in the address space may be larger than the actual available memory of a particular type.
- For CPU Sub-system, please refer to Chapter 1 *High-Performance CPU [to be added later]*.
- Some of the address space can only be accessed by the HP CPU, not by the LP CPU, as indicated in Figure 4.3-1. This part of the address space will not be specifically distinguished in the following sections.

Table 4.3-1 lists the address ranges on the data bus and instruction bus and their corresponding target memories.

Table 4.3-1. Memory Address Mapping

Bus Type	Boundary Address		Size	Target
	Low Address	High Address		
	0x0000_0000	0x3FFF_FFFF		Reserved
Data/Instruction bus	0x4000_0000	0x4004_FFFF	320 KB	ROM*
	0x4005_0000	0x407F_FFFF		Reserved
Data/Instruction bus	0x4080_0000	0x4085_FFFF	384 KB	HP SRAM*

Cont'd on next page

Table 4.3-1 – cont'd from previous page

Bus Type	Boundary Address		Size	Target
	Low Address	High Address		
	0x4086_0000	0x41FF_FFFF		Reserved
Data/Instruction bus	0x4200_0000	0x43FF_FFFF	32 MB	External memory
	0x4400_0000	0x4FFF_FFFF		Reserved
Data/Instruction bus	0x5000_0000	0x5000_3FFF	16 KB	LP SRAM [*]
	0x5000_4000	0x5FFF_FFFF		Reserved
Data/Instruction bus	0x6000_0000	0x600C_FFFF	832 KB	Peripherals
	0x600D_0000	0xFFFF_FFFF		Reserved

^{*} All of the internal memories are managed by Permission Control module. An internal memory can only be accessed when it is allowed by Permission Control, then the internal memory can be available to the HP CPU and LP CPU. For more information about Permission Control, please refer to Chapter [13 Permission Control \(PMS\)](#).

4.3.2 Internal Memory

ESP32-C5 has various types of internal memory:

- ROM (320 KB): The ROM is a read-only memory and can not be programmable. It contains the ROM code of some low-level system software and read-only data. Note that this memory can only be accessed by HP CPU.
- HP SRAM (384 KB): The HP SRAM is a volatile memory that can be quickly accessed by the HP CPU or LP CPU (generally within a single HP CPU clock cycle for HP CPU).
- LP SRAM (16 KB): The LP SRAM is also a volatile memory, however, in Deep-sleep mode, data stored in the LP SRAM will not be lost. The LP SRAM can be accessed by the HP CPU or LP CPU and is usually used to store program instructions and data that need to be kept in sleep mode.

4.3.2.1 ROM

This 320 KB ROM is a read-only memory, accessed by the HP CPU directly through the instruction bus or through the data bus via 0x4000_0000 ~ 0x4004_FFFF or via ROM-Cache, see Table [4.3-1](#).

As shown in Figure [4.3-2](#), ESP32-C5 is able to access ROM via ROM-Cache. This 4 KB ROM-Cache is two-way set associative, with a block size of 64 bytes. Both the instruction bus and the data bus can access the ROM-Cache at the same time, and the ROM-Cache responds to one or the other through arbitration. If data missing happens, ROM-Cache controller initiates a request to the ROM.

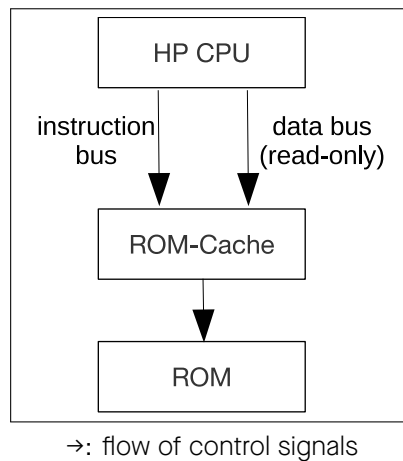


Figure 4.3-2. ROM-Cache Structure

4.3.2.2 HP SRAM

This 384 KB HP SRAM is a read-and-write memory, accessed by the HP CPU and LP CPU through the instruction bus or data bus via their shared addresses 0x4080_0000 ~ 0x4085_FFFF, see Table 4.3-1.

4.3.2.3 LP SRAM

This 16 KB LP SRAM is a read-and-write memory, accessed by the HP CPU or LP CPU through the instruction bus or data bus via their shared addresses 0x5000_0000 ~ 0x5000_3FFF, see Table 4.3-1.

LP SRAM can be accessed by the following modes:

- high-speed mode, i.e., the LP SRAM is accessed in HP CPU clock frequency. In this case, HP CPU can access the LP SRAM without any latency. But the latency of LP CPU accessing LP SRAM ranges from a few dozen to dozens of LP CPU cycles.
- low-speed mode, i.e., the LP SRAM is accessed in LP CPU clock frequency. In this case, LP CPU can access the LP SRAM without any latency. But the latency of HP CPU accessing LP SRAM ranges from a few dozen to dozens of HP CPU cycles.

Switch the modes based on application scenarios.

- If the LP CPU is not working, switch to high-speed mode to improve the access speed of the HP CPU.
- If the LP CPU is executing code in the LP SRAM, switch to the low-speed mode.
- If the HP CPU is in sleep mode, it is a must to switch to the low-speed mode.

Mode Switch Configuration

- Configure LP_AON_FAST_MEM_MUX_SEL to select the mode needed:
 - 0: low-speed mode
 - 1: high-speed mode
- Set LP_AON_FAST_MEM_MUX_SEL_UPDATE to start mode switch.
- Read LP_AON_FAST_MEM_MUX_SEL_STATUS to check if mode switch is done:
 - 0: mode is switched

- 1: mode is not switched

4.3.3 External Memory

ESP32-C5 supports SPI, Dual SPI, Quad SPI, and QPI interfaces that allow connection to external flash and RAM. ESP32-C5 also supports hardware manual encryption and automatic decryption based on XTS-AES algorithm to protect users' programs and data in the external flash and RAM.

4.3.3.1 External Memory Address Mapping

The external memory can be accessed by HP CPU via the cache or accessed by the GDMA. According to information inside the MMU (Memory Management Unit), the cache maps the HP CPU and GDMA's address (0x4200_0000 ~ 0x43FF_FFFF) into a physical address of the external memory. With this address mapping, ESP32-C5 can address up to 32 MB external flash and 32 MB external RAM.

Note that the instruction bus shares the same address space (32 MB) with the data bus to access the external memory.

4.3.3.2 Cache

As shown in Figure 4.3-3, ESP32-C5 has a shared four-way set-associative cache. Its size is 32 KB and its block size is 32 bytes. The instruction bus and data bus can access the cache simultaneously, but the cache can only respond to one of them at a time through arbitration. If a data missing happens, cache controller initiates a request to the external memory.

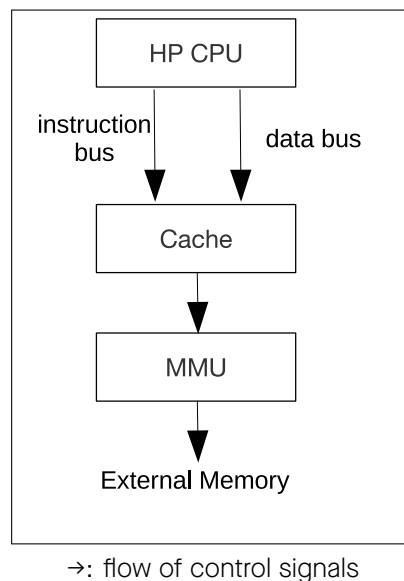


Figure 4.3-3. Cache Structure

4.3.3.3 Cache Operations

ESP32-C5 cache supports the following operations:

1. Write-back

- Clears dirty bits in the tag memory and updates new data to external memory.
- After the write-back operation, the new data is updated to external memory.

- Users can choose whether to invalidate the data in the cache.
- If the data is not invalidated, HP CPU will read/write data directly from/to the cache where data missing will not occur.

2. Clean

- Clears dirty bits in the tag memory without updating data to external memory.
- After the clean operation, old data remains in external memory, while the cache holds the new one (unaware of it).
- HP CPU can then read/write the data directly from/to the cache, where data missing will not occur.

3. Invalidate

- Clean the valid bits in tag memory. This is to remove valid data from the cache.
- Once this operation is done, the deleted data is stored only in external memory.
- HP CPU needs to access external memory if it wants to access the data again.
- Two types of invalidate operation:
 - Manual-Invalidate: is performed only on data in the specified cache area.
 - Invalidate-All: is performed on all data in the cache.

4. Preload

- Loads instructions and data into the cache in advance.
- Minimum unit of preload operation is one block.
- Two types of preload operation:
 - Manual-Preload: involves the hardware prefetching continuous data based on the virtual address specified by the software.
 - Auto-Preload: involves the hardware prefetching continuous data based on the current address where the cache hits or misses (configuration-dependent).

5. Lock/Unlock

- Lock operation prevents easily replacing data in the cache.
- Two types of lock:
 - Prelock: locks data in the specified area when filling missing data to cache memory, leaving the data outside the specified area unlocked.
 - Manual Lock: checks data in the cache memory and locks data only if it falls in the specified area, leaving data outside the specified area unlocked.
- When there is missing data, the cache replaces the data in the unlocked way first, ensuring that the data in the locked way remains in the cache and will not be replaced.
- When all ways within the cache are locked, the cache will replace data, as if it was not locked.
- Unlocking is the reverse of locking and can only be done manually.

- Manual-Invalidate operation only works on the unlocked data. If you plan to perform this operation on the locked data, please unlock them first.

4.3.4 GDMA Address Space

The General Direct Memory Access (GDMA) peripheral in ESP32-C5 consisting of three TX channels and three RX channels provides Direct Memory Access (DMA) service, including:

- data transfers between different locations of internal memory
- data transfers between modules/peripherals and internal memory
- data transfers between different locations of external memory
- data transfers between modules/peripherals and external memory
- data transfers between internal memory and external memory

GDMA uses the same addresses as the data bus to access HP SRAM and external RAM, i.e., GDMA uses address range 0x4080_0000 ~ 0x4085_FFFF to access HP SRAM, and 0x4200_0000 ~ 0x43FF_FFFF to external RAM. Seven modules/peripherals in ESP32-C5 work together with GDMA. As shown in Figure 4.3-4, seven vertical lines correspond to these seven modules/peripherals with GDMA function. The horizontal line represents a certain channel of GDMA (can be any channel), and the intersection of the vertical line and the horizontal line indicates that a module/peripheral has the ability to access the corresponding channel of GDMA.

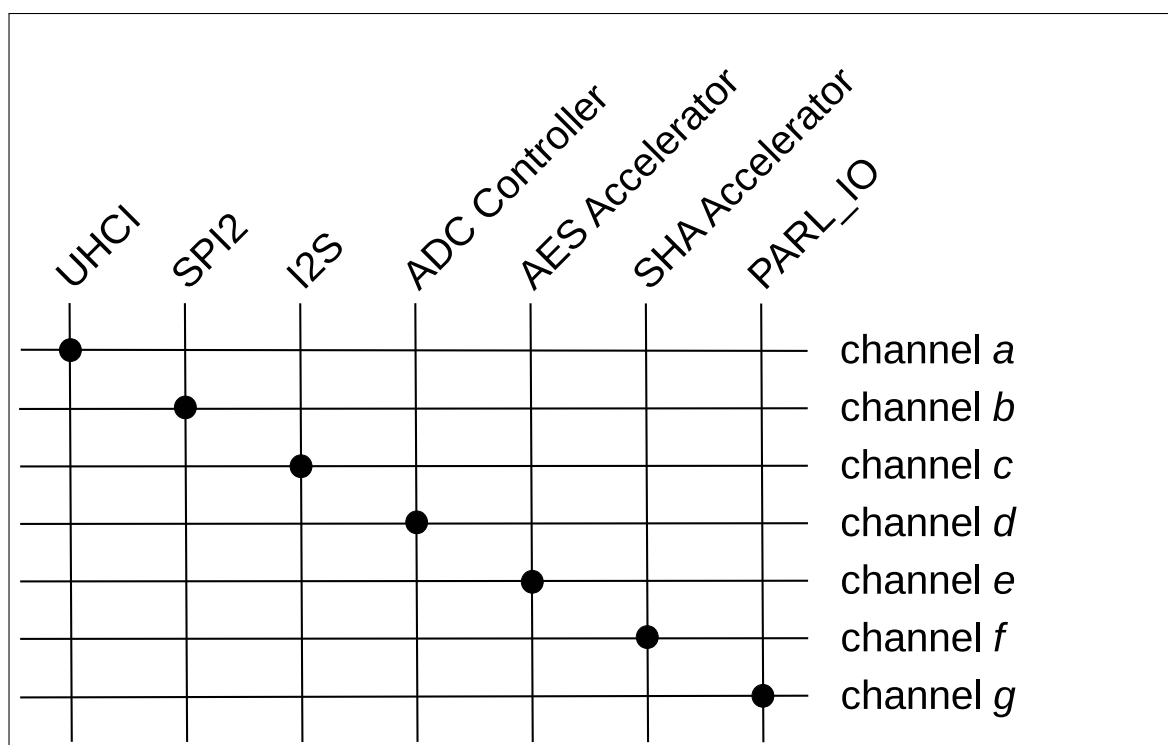


Figure 4.3-4. Modules/peripherals that can work with GDMA

These modules/peripherals can access any memory available to GDMA. For more information, please refer to Chapter 3 [GDMA Controller \(GDMA\)](#).

Note:

When accessing a memory via GDMA, a corresponding access permission is needed, otherwise this access may fail. For more information about permission control, please refer to Chapter 13 [Permission Control \(PMS\)](#).

4.3.5 Modules/Peripherals Address Mapping

Table 4.3-2 lists all the modules/peripherals and their respective address ranges. Note that the address space of specific modules/peripherals is defined by “Boundary Address” (including both Low Address and High Address).

Table 4.3-2. Module/Peripheral Address Mapping

Target	Boundary Address		Size (KB)
	Low Address	High Address	
UART Controller 0 (UART0)	0x6000_0000	0x6000_0FFF	4
UART Controller 1 (UART1)	0x6000_1000	0x6000_1FFF	4
External Memory Encryption and Decryption (XTS_AES) ¹	0x6000_2000	0x6000_2FFF	4
SPI Controller 0 (SPI0) ¹	0x6000_2000	0x6000_2FFF	4
SPI Controller 1 (SPI1)	0x6000_3000	0x6000_3FFF	4
I2C Controller (I2C)	0x6000_4000	0x6000_4FFF	4
UHCI Controller (UHCI)	0x6000_5000	0x6000_5FFF	4
Remote Control Peripheral (RMT)	0x6000_6000	0x6000_6FFF	4
LED PWM Controller (LEDC)	0x6000_7000	0x6000_7FFF	4
Timer Group 0 (TIMG0)	0x6000_8000	0x6000_8FFF	4
Timer Group 1 (TIMG1)	0x6000_9000	0x6000_9FFF	4
System Timer (SYSTIMER)	0x6000_A000	0x6000_AFFF	4
Two-wire Automotive Interface 0 (TWAIO)	0x6000_B000	0x6000_BFFF	4
I2S Controller (I2S)	0x6000_C000	0x6000_CFFF	4
Two-Wire Automotive Interface 1 (TWA1)	0x6000_D000	0x6000_DFFF	4
SAR ADC	0x6000_E000	0x6000_EFFF	4
Temperature Sensor	0x6000_E000	0x6000_EFFF	4
USB Serial/JTAG Controller (USB_SERIAL_JTAG)	0x6000_F000	0x6000_FFFF	4
Interrupt Matrix (INTMTX)	0x6001_0000	0x6001_0FFF	4
Reserved	0x6001_1000	0x6001_1FFF	
Pulse Count Controller (PCNT)	0x6001_2000	0x6001_2FFF	4
Event Task Matrix (SOC_ETM)	0x6001_3000	0x6001_3FFF	4
Motor Controller (MCPWM)	0x6001_4000	0x6001_4FFF	4
Parallel IO Controller (PARL_IO)	0x6001_5000	0x6001_5FFF	4
Reserved	0x6001_6000	0x6001_9FFF	
Memory Access Monitor (PSARM_MEM_MONITOR) ²	0x6001_A000	0x6001_AFFF	4
Reserved	0x6001_B000	0x6007_FFFF	

Cont'd on next page

Table 4.3-2 – cont'd from previous page

Target	Boundary Address		Size (KB)
	Low Address	High Address	
General DMA Controller (GDMA)	0x6008_0000	0x6008_0FFF	4
General Purpose SPI2 Controller (GP-SPI2)	0x6008_1000	0x6008_1FFF	4
Bit-scrambler (BITSCRAMBLER)	0x6008_2000	0x6008_2FFF	4
Reserved	0x6008_3000	0x6008_6FFF	
AES Accelerator (AES)	0x6008_8000	0x6008_8FFF	4
SHA Accelerator (SHA)	0x6008_9000	0x6008_9FFF	4
RSA Accelerator (RSA)	0x6008_A000	0x6008_AFFF	4
ECC Accelerator (ECC)	0x6008_B000	0x6008_BFFF	4
Digital Signature (DS)	0x6008_C000	0x6008_CFFF	4
HMAC Accelerator (HMAC)	0x6008_D000	0x6008_DFFF	4
ECDSA Accelerator (ECDSA)	0x6008_E000	0x6008_EFFF	4
Reserved	0x6008_F000	0x6008_FFFF	
IO MUX	0x6009_0000	0x6009_0FFF	4
GPIO Matrix	0x6009_1000	0x6009_1FFF	4
Memory Access Monitor 1 (TCM_MEM_MONITOR) ²	0x6009_2000	0x6009_2FFF	4
Reserved	0x6009_3000	0x6009_4FFF	
High-Performance System Registers (HP_SYSREG)	0x6009_5000	0x6009_5FFF	4
Power/Clock/Reset Registers (PCR)	0x6009_6000	0x6009_6FFF	4
Reserved	0x6009_7000	0x6009_7FFF	
Trusted Execution Environment (TEE) ²	0x6009_8000	0x6009_8FFF	4
High-Performance Access Permission Management (HP_APM)	0x6009_9000	0x6009_97FF	2
Low-Power Access Permission Management (LP_APMO)	0x6009_9800	0x6009_9FFF	2
CPU Access Permission Management Registers (CPU_APM_REG)	0x6009_A000	0x6009_AFFF	4
Power Management Unit (PMU)	0x600B_0000	0x600B_03FF	1
Low-Power Clock/Reset Registers (LP_CLKRST)	0x600B_0400	0x600B_07FF	1
Reserved	0x600B_0800	0x600B_0BFF	
Low-Power Timer (LP_TIMER)	0x600B_0C00	0x600B_0FFF	1
Low-Power Always-on Registers (LP_AON)	0x600B_1000	0x600B_13FF	1
Low-Power UART (LP_UART)	0x600B_1400	0x600B_17FF	1
Low-Power I2C (LP_I2C)	0x600B_1800	0x600B_1BFF	1
Low-Power Watchdog Timer (LP_WDT)	0x600B_1C00	0x600B_1FFF	1
Reserved	0x600B_2000	0x600B_23FF	
I2C Analog Master (I2C_ANA_MST)	0x600B_2400	0x600B_27FF	1
Low-Power Peripherals (LPPERI)	0x600B_2800	0x600B_2BFF	1

Cont'd on next page

Table 4.3-2 – cont'd from previous page

Target	Boundary Address		Size (KB)
	Low Address	High Address	
Low-Power Analog Peripherals (LP_ANA_PERI)	0x600B_2C00	0x600B_2FFF	1
Low-Power Trusted Execution Environment (LP_TEE) ²	0x600B_3400	0x600B_37FF	1
Low-Power Access Permission Management (LP_APM) ²	0x600B_3800	0x600B_3BFF	1
Reserved	0x600B_3C00	0x600B_3FFF	
Low-Power IO MUX (LP_IO_MUX)	0x600B_4000	0x600B_43FF	1
Low-Power GPIO Matrix (LP_GPIO)	0x600B_4400	0x600B_47FF	1
eFuse Controller	0x600B_4800	0x600B_4FFF	2
Reserved	0x600B_5000	0x600B_FFFF	
RISC-V Trace Encoder (TRACE)	0x600C_0000	0x600C_0FFF	4
Reserved	0x600C_1000	0x600C_1FFF	
Bus Access Monitor (BUS_MONITOR) ²	0x600C_2000	0x600C_2FFF	4
Reserved	0x600C_3000	0x600C_4FFF	
Interrupt Priority Registers (INTPRI)	0x600C_5000	0x600C_5FFF	4
Reserved	0x600C_6000	0x600C_7FFF	
Cache	0x600C_8000	0x600C_8FFF	4
Reserved	0x600C_9000	0x600C_FFFF	

¹ The registers of External Memory Encryption and Decryption (XTS_AES) module share the same address range of 0x6000_2000 ~ 0x6000_2FFF with SPI0 controller.

² The address space of this module/peripheral is not continuous.

Note:

As shown in Figure 4.3-1,

- HP CPU can access all peripherals listed in Table 4.3-2.
- LP CPU can access all peripherals listed in Table 4.3-2, except RISC-V Trace Encoder (TRACE), DEBUG ASSIST (ASSIST_DEBUG), Interrupt Priority Registers (INTPRI), and Cache.

Part III

System Component

Encompassing a range of system-level functionalities, this part describes components related to system boot, clocks, GPIO, timers, watchdogs, debug assistance, event and interrupt handling, low-power management, and various system registers.

Chapter 5

GPIO Matrix and IO MUX

5.1 Overview

GPIO matrix is a hardware structure that allows dynamic remapping between the GPIO pins and peripheral input/output signals to provide greater flexibility. IO MUX (Input/Output Multiplexing) is a technology that allows the same pin to perform different functions at different times, allowing dynamic switching of pin functions through configuration.

ESP32-C5 provides two groups of GPIO matrix and IO MUX:

- HP GPIO matrix and HP IO MUX
- LP GPIO matrix and LP IO MUX

The ESP32-C5 chip features 29 GPIO pins, numbered from GPIO0 to GPIO28, including

- 7 LP GPIO pins (GPIO0~GPIO6) for either HP or LP peripherals.
- 14 HP GPIO pins (GPIO7~GPIO14, GPIO23~GPIO28) for HP peripherals only.
- 8 dedicated GPIO pins (GPIO15~GPIO22) for external flash and PSRAM. **These pins can not be used for other purpose, and therefore are not described in subsequent sections.** For more information regarding these pins, please refer to [ESP32-C5 Datasheet](#) > Section *Pin Mapping Between Chip and Flash/PSRAM*.

Each pin can be used as a general-purpose I/O, or be connected to an internal peripheral signal. Together these modules provide highly configurable I/O.

- Through HP GPIO matrix and HP IO MUX, HP peripheral input signals can be from any GPIO pins, and HP peripheral output signals can be routed to any GPIO pins.
- Through LP GPIO matrix and LP IO MUX, LP peripheral input signals can be from any LP GPIO pins, and LP peripheral output signals can be routed to any LP GPIO pins.

Unless otherwise specified, GPIO matrix refers to both LP GPIO matrix and HP GPIO matrix, so as to IO MUX.

5.2 Features

5.2.1 HP GPIO Matrix and HP IO MUX

HP GPIO matrix has the following features:

- A full-switching matrix between HP peripheral input/output signals and the GPIO pins
- 76 HP peripheral input signals sourced from the input of any GPIO pins

- 77 HP peripheral output signals routed to the output of any GPIO pins
- Signal synchronization for HP peripheral inputs based on **HP IO MUX operating clock**
- GPIO Filter hardware for input signal filtering
- Glitch Filter hardware for second-time filtering on input signal
- Sigma delta modulated (SDM) output
- GPIO simple input and output
- HP GPIO Wakeup

HP IO MUX has the following features:

- Control of 21 GPIOs (GPIO0~GPIO14, GPIO23~GPIO28) for HP peripherals.
- A configuration register `IO_MUX_GPIO $n$ _REG` provided for each GPIO pin, to control the pin's input/output, pull-up/pull-down, drive strength, and function selection.
- Better high-frequency digital performance achieved by routing some digital signals (such as SPI) directly from HP IO MUX to peripherals.

5.2.2 LP GPIO Matrix and LP IO MUX

LP GPIO matrix has the following features:

- GPIO simple input and output
- LP GPIO Wakeup

LP IO MUX has the following feature:

- Control of 7 LP GPIO pins (GPIO0~GPIO6) for LP peripherals.
- A configuration register `LP_IO_MUX_GPIO $n$ _REG` provided for each LP GPIO pin, to control the pin's input/output, pull-up/pull-down, drive strength, and function selection.
- A field `LP_AON_GPIO_MUX_SEL` provided to select the source of the pin control signals, from HP IO MUX or LP IO MUX.

5.3 Architectural Overview

Figure 5.3-1 shows in details how HP GPIO matrix, HP IO MUX, LP GPIO matrix, and LP IO MUX route signals from pins to peripherals, and from peripherals to pins.

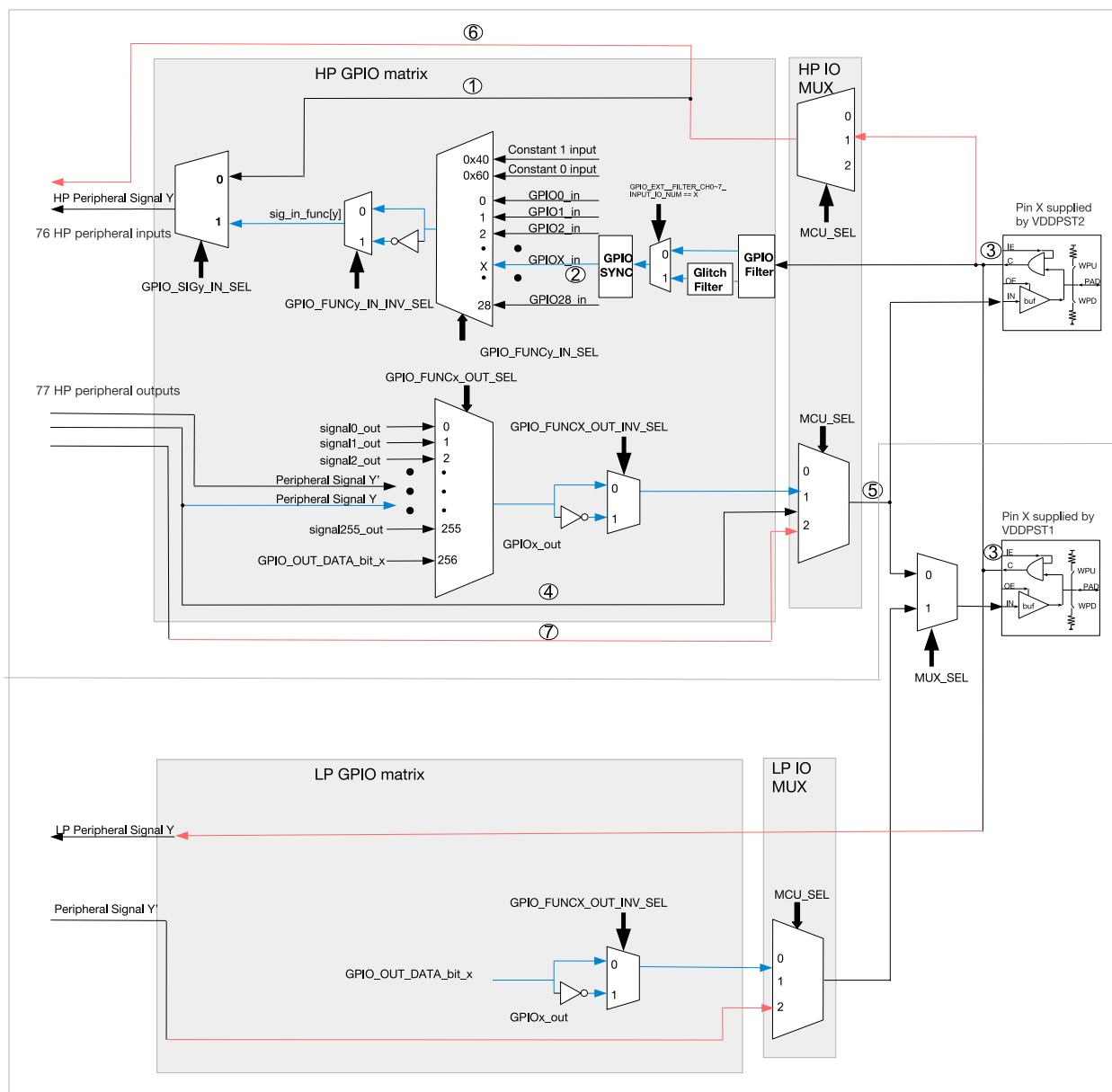


Figure 5.3-1. Architecture of HP GPIO Matrix, HP IO MUX, LP GPIO Matrix, and LP IO MUX

The following points explain the areas marked numerically in the figure above, taking the HP system as an example.

① Peripheral input signals marked “yes” in column “Direct input through HP IO MUX” in Table 5.12-1 can be routed to the peripherals via HP IO MUX, or via HP GPIO matrix, while the other input signals can only be routed to the peripherals via HP GPIO matrix.

② There are only 21 inputs from GPIO SYNC to HP GPIO matrix, since ESP32-C5 provides 21 GPIO pins available for users in total.

③ The pins are controlled by the signals: IE, OE, WPU, and WPD.

④ Peripheral outputs marked “yes” in column “Direct output through HP IO MUX” in Table 5.12-1 can be routed to pins via HP IO MUX or via HP GPIO matrix, while the other output signals can only be routed to pins via HP GPIO matrix.

⑤ There are 29 outputs (corresponding to GPIO pin X: 0~14, 23~28) from HP GPIO matrix to HP IO MUX.

- ⑥ There are peripheral inputs only through HP IO MUX.
- ⑦ There are peripheral outputs only through HP IO MUX.

Figure 5.3-2 shows the internal structure of a pad, which is an electrical interface between the chip logic and the GPIO pin. The structure is applicable to all GPIO pins and can be controlled using IE, OE, WPU, and WPD signals. For the configuration of these signals, see [IO_MUX_GPIOx_REG](#) or [LP_IO_MUX_GPIOx_REG](#).

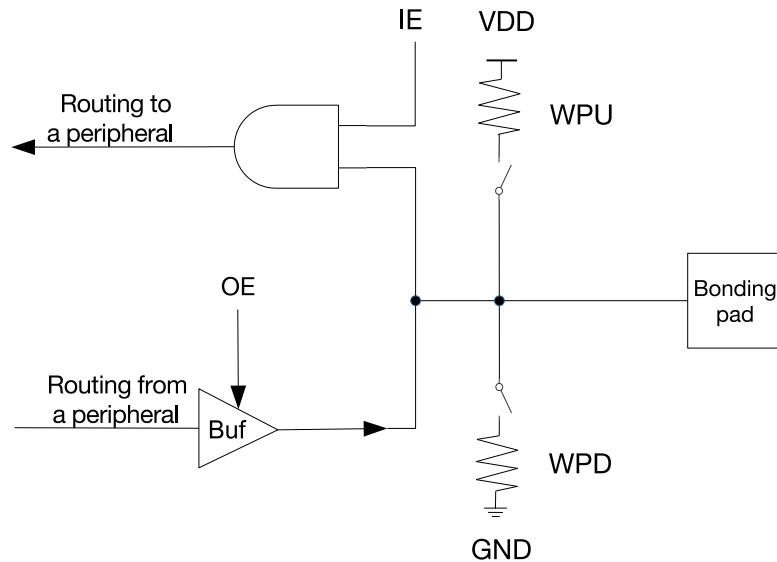


Figure 5.3-2. Internal Structure of a Pad

- IE: input enable
- OE: output enable
- WPU: internal weak pull-up resistor
- WPD: internal weak pull-down resistor
- Bonding pad: a terminal point of the chip logic used to make a physical connection from the chip die to GPIO pin in the chip package

5.4 Peripheral Input via GPIO Matrix

5.4.1 Overview

To receive a peripheral input signal via HP GPIO matrix,

- Configure the matrix to source the peripheral input signal from one of the 21 GPIOs (0~14, 23~28), see Table 5.12-1.
- Configure the peripheral signal to receive input signal via HP GPIO matrix.
- Configure the GPIO pin to be controlled by HP IO MUX.

For detailed configuration, see Figure 5.3-1 and Section 5.4.7.

As shown in Figure 5.3-1, when GPIO matrix is used to input a signal from the pin, all external input signals are sourced from the GPIO pins and then filtered by the GPIO Filter, as shown in Step 2 in Section 5.4.7.

The Glitch Filter is only available in HP GPIO matrix. The Glitch Filter hardware can filter eight of the output signals from the GPIO Filter, and the other unselected signals go directly to the GPIO SYNC hardware, as shown in Step 3 in Section 5.4.7.

All signals filtered by the GPIO Filter hardware or the Glitch Filter hardware are synchronized by the GPIO SYNC hardware to IO MUX operating clock and then enter the GPIO matrix, see Section 5.4.2. Such signal filtering and synchronization features apply to all GPIO matrix signals but do not apply when using the IO MUX.

5.4.2 Signal Synchronization

Only HP GPIO matrix supports this signal synchronization function. Figure 5.4-1 shows the functionality of GPIO SYNC. In the figure, negative sync and positive sync mean GPIO input is synchronized on falling edge and on rising edge of HP IO MUX operating clock respectively.

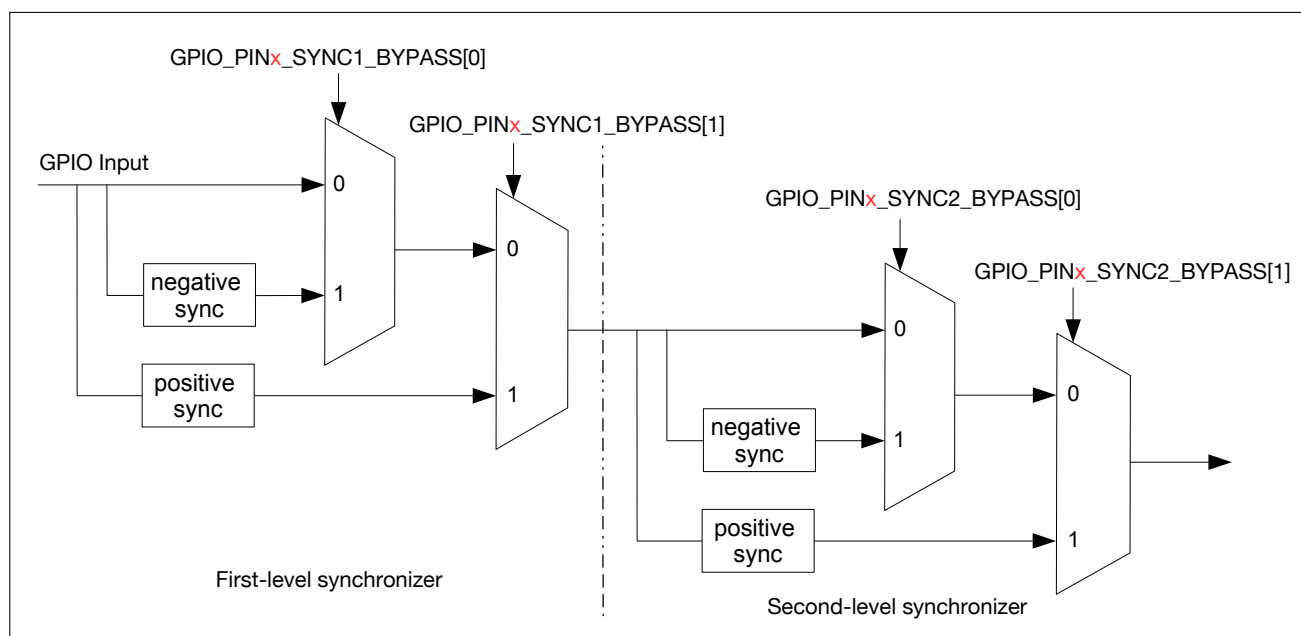


Figure 5.4-1. GPIO Input Synchronized on Rising Edge or on Falling Edge of HP IO MUX Operating Clock

The synchronization function is disabled by default by the synchronizer. But when an asynchronous peripheral signal is connected to the pin, the signal should be synchronized by the two-level synchronizer (i.e., the first-level synchronizer and the second-level synchronizer as shown in Figure 5.4-1) to lower the probability of causing metastability.

5.4.3 GPIO Filter

Only HP GPIO matrix supports this GPIO Filter function. When this function is enabled, only signals with a valid width of more than two clock cycles can be sampled, as illustrated in Figure 5.4-2.

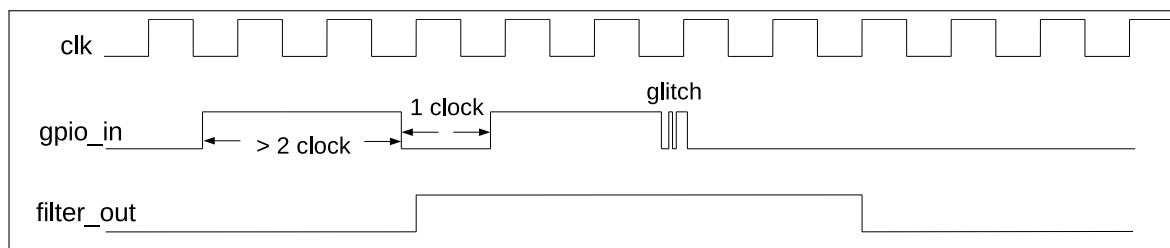


Figure 5.4-2. GPIO Filter Timing of GPIO Input Signals

5.4.4 Glitch Filter

The Glitch Filter function is exclusive to the HP GPIO matrix. The Glitch Filter hardware supports eight channels, each of which selects one signal from the 21 (0~14, 23~28) output signals generated by the GPIO Filter hardware. It then conducts a second round of filtering on the selected signal. This Glitch Filter hardware can be used to filter slow-speed signals. For more information, see Step 3 in Section 5.4.71.

5.4.5 Simple GPIO Input

Both the HP GPIO matrix and LP GPIO matrix support the Simple GPIO Input function. Enabling this function allows the direct reading of the input value of a GPIO pin at any time, without the need to route the GPIO input to any peripherals.

For HP GPIO matrix, to implement simple GPIO input, follow the steps below:

- Set `IO_MUX_GPIOx_MCU_IE` in `IO_MUX_GPIOx_REG`, to enable pin input.
- Read the GPIO input from `GPIO_IN_REG[x]`.

For LP GPIO matrix, to implement simple GPIO input, follow the steps below:

- Set `LP_IOMUX_PADx_FUN_IE` in `LP_IO_MUX_GPIOx_REG`, to enable pin input.
- Read the GPIO input from `LP_GPIO_IN_REG[x]`.

5.4.6 GPIO Wakeup

5.4.6.1 HP GPIO Wakeup

The HP GPIO wakeup feature is designed to awaken the system from Light-sleep. HP GPIO wakeup is not available when the HP system is powered down.

GPIO0~GPIO14 and GPIO23~GPIO28 can be used to generate HP GPIO wakeup by enabling `GPIO_PINx_WAKEUP_ENABLE` (x: 0~14, 23~28).

HP GPIO wakeup supports both high-voltage and low-voltage triggers, depending on the configuration of `GPIO_PINx_INT_TYPE`.

Table 5.4-1. HP GPIO Wakeup Signal Trigger and Clear Conditions

<code>GPIO_PINx_INT_TYPE</code> ^{1,2}	Wakeup is generated when	Wakeup is cleared when
5	GPIO input is high-level	GPIO input is low-level
4	GPIO input is low-level	GPIO input is high-level

¹ x ranges from 0 to 14, 23 to 28.

² If the field is configured to any other value, HP GPIO wakeup feature is disabled.

5.4.6.2 LP GPIO Wakeup

The LP GPIO wakeup feature is designed to awaken the system from Deep-sleep. LP GPIO wakeup is not available when the LP peripherals are powered down.

GPIO0~GPIO6 can be used to generate LP GPIO wakeup by enabling `LP_GPIO_PIN $x$ _WAKEUP_ENABLE` (x : 0~6).

LP GPIO wakeup supports the following types, depending on the configuration of `LP_GPIO_PIN $x$ _INT_TYPE`:

- posedge sensitive
- negedge sensitive
- both posedge and negedge sensitive
- high-voltage sensitive and low-voltage sensitive

Table 5.4-2. LP GPIO Wakeup Signal Trigger and Clear Conditions

<code>LP_GPIO_PINx_INT_TYPE</code> ^{1,2}	Wakeup is generated when	Wakeup is cleared when
1	GPIO input toggles from low-level to high-level	<code>LP_GPIO_PINx_EDGE_WAKEUP_CLR</code> is set
2	GPIO input toggles from high-level to low-level	<code>LP_GPIO_PINx_EDGE_WAKEUP_CLR</code> is set
3	GPIO input toggles from high-level to low-level, or vice versa	<code>LP_GPIO_PINx_EDGE_WAKEUP_CLR</code> is set
4	GPIO input is low-level	GPIO input is high-level
5	GPIO input is high-level.	GPIO input is low-level

¹ x ranges from 0 to 6.

² If the field is configured to any other value, LP GPIO wakeup feature is disabled.

5.4.7 Programming Procedure

5.4.7.1 HP GPIO Matrix

To read GPIO pin X ¹ into HP peripheral signal Y , follow the steps below:

1. Configure `GPIO_FUNC $y$ _IN_SEL_CFG_REG` corresponding to HP peripheral signal Y in HP GPIO matrix:
 - Set `GPIO_SIG $y$ _IN_SEL` to enable peripheral signal input via HP GPIO matrix.
 - Set `GPIO_FUNC $y$ _IN_SEL` to the desired GPIO pin, i.e., X here.

Note: some peripheral signals have no valid `GPIO_SIG $y$ _IN_SEL` bit, namely, these peripherals can only receive input signals via HP GPIO matrix.

2. Optionally enable the GPIO Filter for pin input signals by setting `IO_MUX_GPIO $x$ _FILTER_EN`.
3. Enable Glitch Filter feature as follows:

- Configure `GPIO_EXT_FILTER_CHn_INPUT_IO_NUM` to `m`. `n` (0~7) represents the channel number. `m` (0~14, 23~28) represents the GPIO pin number.
- Configure `GPIO_EXT_FILTER_CHn_WINDOW_WIDTH` to `VALUE1` and `GPIO_EXT_FILTER_CHn_WINDOW_THRES` to `VALUE2`. During `VALUE1 + 1` cycles, if there are `VALUE2 + 1` input signals that do not match the current output signal value, the Glitch Filter hardware inverts the output signal. `GPIO_EXT_FILTER_CHn_WINDOW_WIDTH` and `GPIO_EXT_FILTER_CHn_WINDOW_THRES` can be configured to the same value `VALUE3`, then only signals with a width greater than `VALUE3 + 1` clock cycles will be sampled.
- Set `GPIO_EXT_FILTER_CHn_EN` to enable channel `n`.

An example is shown in Figure 5.4-3, where `GPIO_EXT_FILTER_CHn_WINDOW_WIDTH` is configured to 3 and `GPIO_EXT_FILTER_CHn_WINDOW_THRES` to 2. The output signal value (`signal_out`) keeps as “0” in the four clock cycles before T1. The input signal value (`signal_in`) has been “1” for three clock cycles in the same period, then the output signal is inverted to “1” after T1.

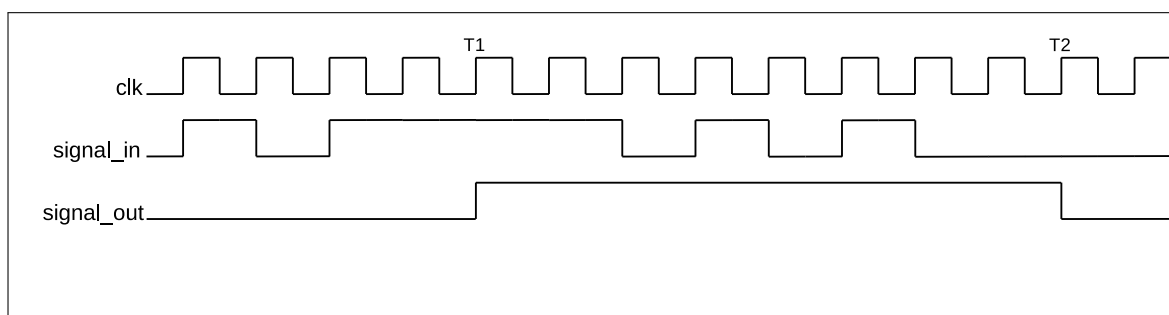


Figure 5.4-3. Glitch Filter Timing Example

4. Synchronize GPIO input signals. To do so, please set `GPIO_PINx_REG` corresponding to GPIO pin `X` as follows:
 - Set `GPIO_PINx_SYNC1_BYPASS` to enable input signal synchronized on rising edge or on falling edge in the first-level synchronization, see Figure 5.4-1.
 - Set `GPIO_PINx_SYNC2_BYPASS` to enable input signal synchronized on rising edge or on falling edge in the second-level synchronization, see Figure 5.4-1.
5. Configure HP IO MUX register to enable pin input. For this end, please set `IO_MUX_GPIOx_REG` corresponding to GPIO pin `X` as follows:
 - Set `IO_MUX_GPIOx_FUN_IE` to enable input^{2,3}.
 - Set or clear `IO_MUX_GPIOx_FUN_WPU` and `IO_MUX_GPIOx_FUN_WPD` as desired to enable or disable pull-up and pull-down resistors.

Note:

1. One input pin can be connected to multiple peripheral input signals.
2. The input signal can be inverted by configuring `GPIO_FUNCy_IN_INV_SEL`.
3. It is possible to have an HP peripheral read a constantly low or constantly high input value without connecting this input to a pin. This can be done by selecting a special `GPIO_FUNCy_IN_SEL` input, instead of a GPIO number:

- When `GPIO_FUNCy_IN_SEL` is set to 0x40, input signal is always 0.
- When `GPIO_FUNCy_IN_SEL` is set to 0x60, input signal is always 1.

Programming Example

To connect UART0 RXD input signal (UORXD_in, signal index 6) to GPIO7, please follow the steps below.

1. Set `GPIO_SIG6_IN_SEL` in `GPIO_FUNC6_IN_SEL_CFG_REG` to enable peripheral signal input via HP GPIO matrix.
2. Set `GPIO_FUNC6_IN_SEL` in `GPIO_FUNC6_IN_SEL_CFG_REG` to 7, i.e., select GPIO7.
3. Set `IO_MUX_GPIO7_FUN_IE` in `IO_MUX_GPIO7_REG` to enable pin input.

5.4.7.2 LP GPIO Matrix

For inputs, LP GPIO matrix only supports the GPIO wakeup function. For LP GPIO input wakeup programming, please refer to Section 5.4.6.2.

5.5 Peripheral Output via GPIO Matrix

5.5.1 Overview

To output a signal from a peripheral via GPIO matrix:

- Configure the HP GPIO matrix to route HP peripheral output signals (only signals with a name assigned in the column “Output signal” in Table 5.12-1) to one of the 21 GPIOs (0~14, 23~28).
- Then the output signal is routed from the peripheral into GPIO matrix and then into IO MUX. IO MUX must be configured to set the chosen pin to GPIO function. This enables the GPIO output signal to be connected to the pin.

For the detailed programming procedure, see Section 5.5.4.1.

Note:

There is a range of peripheral output signals (97~100 in Table 5.12-1) which are not connected to any peripheral, but to the input signals (97~100) directly.

5.5.2 Simple GPIO Output

GPIO matrix can also be used for simple GPIO output. For this case, one GPIO pin can be configured to directly output the desired value, without routing any peripheral output to this pin.

Follow the steps below to configure HP GPIO matrix for simple GPIO output:

- Set `GPIO_FUNCx_OUT_SEL` with a special peripheral index 256 (0x100).
- Configure the corresponding bit in `GPIO_OUT_REG` to the desired GPIO output value.

Note:

- The bits `GPIO_OUT_REG`[0~14, 23~28] correspond to GPIO0~GPIO14, GPIO23~GPIO28.
- Recommended operation: use `GPIO_OUT_W1TS` and `GPIO_OUT_W1TC` to set or clear `GPIO_OUT_REG`.

LP GPIO matrix supports only the Simple GPIO Output function. There is no need to configure function selection registers. Just configure the corresponding bit in [LP_GPIO_OUT_REG](#) to the desired GPIO output value.

Note:

- The bits [LP_GPIO_OUT_REG](#)[0~6] correspond to GPIO0~GPIO6.
- Recommended operation: use [LP_GPIO_OUT_DATA_W1TS](#) and [LP_GPIO_OUT_DATA_W1TC](#) to set or clear [LP_GPIO_OUT_REG](#).

5.5.3 Sigma Delta Modulated Output (SDM)

5.5.3.1 Functional Description

Four HP peripheral output signals (index: 76~79 in Table [5.12-1](#)) support 1-bit second-order sigma delta modulation. By default the output is enabled for these four channels. This Sigma Delta modulator can also output PDM (pulse density modulation) signal with configurable duty cycle. The function is:

$$H(z) = X(z)z^{-1} + E(z)(1-z^{-1})^2$$

$E(z)$ is quantization error and $X(z)$ is the input.

This modulator supports scaling down of IO MUX operating clock by divider 1~256:

- Set [GPIO_EXT_SIGMADELTA_CLK_EN](#) to enable the modulator clock.
- Configure [GPIO_EXT_SDn_PRESCALE](#) ($n = 0\sim3$ for the four channels).

After scaling, the clock cycle is equal to one pulse output cycle from the modulator.

[GPIO_EXT_SDn_IN](#) is a signed number with a range of [-128, 127] and is used to control the duty cycle¹ of PDM output signal.

- [GPIO_EXT_SDn_IN](#) = -128, the duty cycle of the output signal is 0%.
- [GPIO_EXT_SDn_IN](#) = 0, the duty cycle of the output signal is near 50%.
- [GPIO_EXT_SDn_IN](#) = 127, the duty cycle of the output signal is near 100%.

The formula for calculating PDM signal duty cycle is shown as below:

$$Duty\_Cycle = \frac{GPIO\_EXT\_SDn\_IN + 128}{256}$$

Note:

For PDM signals, duty cycle refers to the percentage of high level cycles to the whole statistical period (several pulse cycles, for example, 256 pulse cycles).

5.5.3.2 SDM Configuration

The configuration of SDM is shown below:

- Route one of SDM outputs to a pin via HP GPIO matrix, see Section 5.5.4.1.
- Enable the modulator clock by setting [GPIO_EXT_SIGMADELTA_CLK_EN](#).
- Configure the divider value by setting [GPIO_EXT_SD \$n\$ _PRESCALE](#).
- Configure the duty cycle of SDM output signal by setting [GPIO_EXT_SD \$n\$ _IN](#).

5.5.4 Programming Procedure

5.5.4.1 HP GPIO Matrix

The output signals with a name assigned in the column “Output signal” in Table 5.12-1 can be set to go through the HP GPIO matrix into HP IO MUX and then to a pin. Figure 5.3-1 illustrates the configuration.

To output peripheral signal Y to a particular GPIO pin X , follow the steps below:

1. Configure [GPIO_FUNC \$x\$ _OUT_SEL_CFG_REG](#) and [GPIO_ENABLE_REG\[\$x\$ \]](#) corresponding to GPIO pin X in HP GPIO matrix. Recommended operation: use corresponding [WITS](#) (write 1 to set) and [WITC](#) (write 1 to clear) registers to set or clear [GPIO_ENABLE_REG](#).
 - Set the [GPIO_FUNC \$x\$ _OUT_SEL](#) field in register [GPIO_FUNC \$x\$ _OUT_SEL_CFG_REG](#) to the index of the desired peripheral output signal Y .
 - If the signal should always be enabled as an output, set the bit [GPIO_FUNC \$x\$ _OE_SEL](#) and the corresponding bit in [GPIO_ENABLE_WITS_REG](#). To have the output enable signal decided by internal logic (for example, the SPIQ_oe in column “Output enable signal when [GPIO_FUNC \$n\$ _OE_SEL = 0” in Table 5.12-1\), clear the \[GPIO_FUNC \\$x\\$ _OE_SEL\]\(#\) bit instead.](#)
 - Set the corresponding bit in register [GPIO_ENABLE_WITC_REG](#) to disable the output from the GPIO pin.
2. For an open drain output, set the field [GPIO_PIN \$x\$ _PAD_DRIVER](#).
3. Configure HP IO MUX register to enable output via HP GPIO matrix. Set [IO_MUX_GPIO \$x\$ _REG](#) corresponding to GPIO pin X as follows:
 - Set the field [IO_MUX_GPIO \$x\$ _MCU_SEL](#) to desired HP IO MUX function corresponding to GPIO pin X . This is Function 1 (GPIO function), numeric value 1, for all pins.
 - Set the [IO_MUX_GPIO \$x\$ _FUN_DRV](#) field to the desired value for output strength (0~3). The higher the drive strength, the more current can be sourced/sunk from the pin.
 - If using open drain mode, set or clear the [IO_MUX_GPIO \$x\$ _FUN_WPU](#) and [IO_MUX_GPIO \$x\$ _FUN_WPD](#) bits to enable or disable the internal pull-up and pull-down resistors.
4. Enable hysteresis function:
 - See Section 5.10.

Note:

- The output signal from a single peripheral can be sent to multiple pins simultaneously.
- The output signal can be inverted by setting [GPIO_FUNC \$x\$ _OUT_INV_SEL](#).

5.5.4.2 LP GPIO Matrix

For outputs, LP GPIO matrix only supports the Simple GPIO Output function. For the detailed programming, please refer to Section [5.5.2](#).

5.6 Direct Input and Output via IO MUX

5.6.1 Overview

Some digital signals such as SPI and JTAG can bypass GPIO matrix for better high-frequency digital performance. In this case, IO MUX is used to connect these pins directly to peripherals. This option is less flexible than routing signals via GPIO matrix, as the IO MUX register for each GPIO pin can only select from a limited number of functions, but high-frequency digital performance can be improved.

ESP32-C5 provides 7 GPIO pins (GPIO0~GPIO6) with low power (LP) capabilities. These pins can be controlled by either HP IO MUX or LP IO MUX. If controlled by LP IO MUX, these pins will bypass HP IO MUX and HP GPIO matrix for the use by peripherals in LP system.

When configured as LP GPIOs, the pins can still be controlled by the peripherals in LP system during chip Deep-sleep, and wake up the chip from Deep-sleep.

5.6.2 Functional Description

The pins with LP functions (GPIO0~GPIO6) are controlled by bit[*n*] of LP_AON_GPIO_MUX_SEL in LP_AON_GPIO_MUX_REG (*n* = 0~6). By default, all these bits are set to 0, routing all input/output signals via HP IO MUX.

If bit[*n*] is set to 1 in LP_AON_GPIO_MUX_SEL, then input/output signals are controlled by LP IO MUX. In this mode, LP_IO_MUX_GPIO*n*_REG is used to control the LP GPIO pins. See [5.14-1](#) for the LP functions of each LP GPIO pin. Note that LP_IO_MUX_GPIO*n*_REG applies the LP GPIO pin numbering. In ESP32-C5, the numbering of LP GPIO pins is the same as that of the HP GPIO pins.

5.6.2.1 HP IO MUX

Two fields must be configured in order to bypass HP GPIO matrix for HP peripheral input signals:

1. IO_MUX_GPIO*n*_MCU_SEL for the GPIO pin must be set to the required pin function. For the list of pin functions, please refer to Table [5.13-1](#).
2. Clear GPIO_SIG*n*_IN_SEL to route the input directly to the peripheral.

To bypass HP GPIO matrix for HP peripheral output signals, IO_MUX_GPIO*n*_MCU_SEL for the GPIO pin must be set to the required pin function.

5.6.2.2 LP IO MUX

To bypass LP GPIO matrix for LP peripheral input/output signals, LP_IO_MUX_GPIO*n*_MCU_SEL for the GPIO pin must be set to the required pin function. For the list of pin functions, please refer to Table [5.14-1](#).

Note:

Not all signals can be directly connected to peripheral via IO MUX. Some input/output signals can only be connected to peripheral via GPIO matrix. See Table 5.12-1.

5.7 Analog Functions

5.7.1 Overview

ESP32-C5 provides 11 GPIO pins (GPIO0~GPIO6, GPIO8~GPIO9, GPIO13~GPIO14) with analog functions, including 6 LP GPIO pins and 5 HP GPIO pins. See Table 5.15-1.

5.7.2 Analog Functions

When the pin is used for analog purpose, make sure this pin is left floating by configuring registers. By such way, the external analog signal is directly connected to internal analog signal via GPIO pin. The configuration is as follows:

- Clear `IO_MUX_GPIO $n$ _FUN_IE`, `IO_MUX_GPIO $n$ _FUN_WPU`, and `IO_MUX_GPIO $n$ _FUN_WPD`, to set the pin floating.
- Write 1 to the corresponding bit in `GPIO_ENABLE_W1TC`, to clear output enable.

To use these functions listed in Table 5.15-1, please refer to the following related chapters:

- For XTAL_32K related functions, see Chapter 4 *Low-power Management [to be added later]*.
- For ADC related functions, see Chapter 36 *ADC Controller*.
- For voltage comparator related functions, see Chapter 37 *Analog Voltage Comparator*.

Note:

GPIO1 has two different analog functions, which can be used simultaneously.

5.8 Pin Functions in Light-sleep

Pins may provide different functions when ESP32-C5 is in Light-sleep mode. If `IO_MUX_GPIO $n$ _SLP_SEL` in register `IO_MUX_GPIO $n$ _REG` for a GPIO pin is set to 1, a different set of bits will be used to control the pin when the chip is in Light-sleep mode.

Table 5.8-1. Bit Used to Control IO MUX Functions in Light-sleep Mode

IO MUX Function	Normal Execution OR <code>IO_MUX_GPIOn_SLP_SEL</code> = 0	Light-sleep Mode AND <code>IO_MUX_GPIOn_SLP_SEL</code> = 1
Output Drive Strength	<code>IO_MUX_GPIOn_FUN_DRV</code>	<code>IO_MUX_GPIOn_MCU_DRV</code>
Pull-up Resistor	<code>IO_MUX_GPIOn_FUN_WPU</code>	<code>IO_MUX_GPIOn_MCU_WPU</code>
Pull-down Resistor	<code>IO_MUX_GPIOn_FUN_WPD</code>	<code>IO_MUX_GPIOn_MCU_WPD</code>

Cont'd on next page

Table 5.8-1 – cont'd from previous page

IO MUX Function	Normal Execution OR <code>IO_MUX_GPIOn_SLP_SEL = 0</code>	Light-sleep Mode AND <code>IO_MUX_GPIOn_SLP_SEL = 1</code>
Input Enable	<code>IO_MUX_GPIOn_FUN_IE</code>	<code>IO_MUX_GPIOn_MCU_IE</code>
Output Enable	<code>OE_SEL</code> from GPIO matrix*	<code>IO_MUX_GPIOn_MCU_OE</code>

* If `IO_MUX_GPIO $n$ _SLP_SEL` is set to 0, pin functions remain the same in both normal execution and in Light-sleep mode. Please refer to Section 5.5.4.1 for how to enable output in normal execution.

5.9 Pin Hold Feature

Each GPIO pin has an individual hold function controlled by Power Management Unit (PMU) or registers. When the pin is set to hold, the state is latched at that moment and will not change no matter how the internal signals change or how the IO MUX/GPIO configuration is modified. Users can use the Hold function for the pins to retain the pin state through a core reset triggered by watchdog time-out or Deep-sleep events.

The Hold state of each GPIO pin is controlled by the result of OR operation of the pin's Hold enable signal and the global Hold enable signal.

- Hold signal of each individual pin:
 - `LP_AON_GPIO_HOLD0_REG[ $n$ ]` ($n = 0\sim14, 23\sim28$) controls the Hold signal of each pin.
- Global Hold signal:
 - Global Hold signal of HP GPIO pins:
 - * `PMU_TIE_HIGH_HP_PAD_HOLD_ALL`: enables the global Hold signal of all HP GPIO pins.
 - * `PMU_TIE_LOW_HP_PAD_HOLD_ALL`: disables the global Hold signal of all HP GPIO pins.
 - Global Hold signal of LP GPIO pins:
 - * `PMU_TIE_HIGH_LP_PAD_HOLD_ALL`: enables the global Hold signal of all LP GPIO pins.
 - * `PMU_TIE_LOW_LP_PAD_HOLD_ALL`: disables the global Hold signal of all LP GPIO pins.

Enable or disable the Hold feature of the pins by one of the following ways:

- **Method 1: enable or disable the Hold feature of individual pins:**

To maintain the pin's input/output status in Deep-sleep, set `LP_AON_GPIO_HOLD0_REG[ $n$ ]` (where $n = 0\sim14, 23\sim28$ corresponds to GPIO0~GPIO14, GPIO23~GPIO28). To disable the hold function after waking up, clear the bit[n].
- **Method 2: enable or disable the Hold feature of all HP/LP pins:**
 - HP Pins

Set `PMU_TIE_HIGH_HP_PAD_HOLD_ALL` to maintain the input/output status of all HP pins and set `PMU_TIE_LOW_HP_PAD_HOLD_ALL` to disable the hold function for all HP pins.
 - LP Pins

Set `PMU_TIE_HIGH_LP_PAD_HOLD_ALL` to maintain the input/output status of all LP pins and set `PMU_TIE_LOW_LP_PAD_HOLD_ALL` to disable the hold function for all LP pins.
- **Method 3: Use PMU Task**

Configure the PMU task to hold the HP pins before Deep-sleep. In Deep-sleep, the HP pins would be hold automatically by the PMU.

5.10 Hysteresis Characteristics

Each GPIO pin has hysteresis functionality. When hysteresis is not enabled, as shown in Figure 5.10-1, the level flip of the signal (C) input to the chip from the PAD has only one threshold (V_t , about 1.7 V). When the voltage on the PAD is higher than V_t , the level on the C is high. Otherwise, it is low. However, noise on the PAD may affect the signal on C.

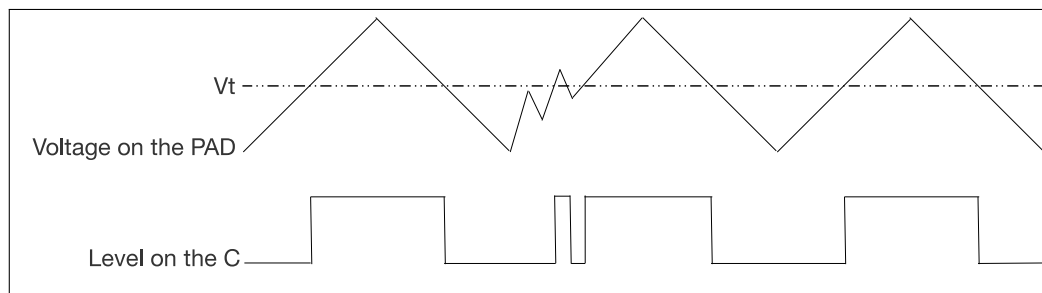


Figure 5.10-1. Example of Level Flip on the Chip Pad — Hysteresis Function Not Enabled

When hysteresis is enabled, as shown in Figure 5.10-2, the level flip of C has two thresholds, high-level threshold (V_{th} , about 1.7 V) and low-level threshold (V_{tl} , about 1.4 V). When the voltage of pad goes from low to high, if the voltage is higher than V_{th} , the level of C is high. When the voltage of PAD goes from high to low, if the voltage is lower than V_{tl} , the level of C is low. When the voltage of PAD is between V_{th} and V_{tl} , the level of C does not change. The hysteresis function plays an anti-interference role by mitigating the impact of noise, consequently reducing the level flip time of C.

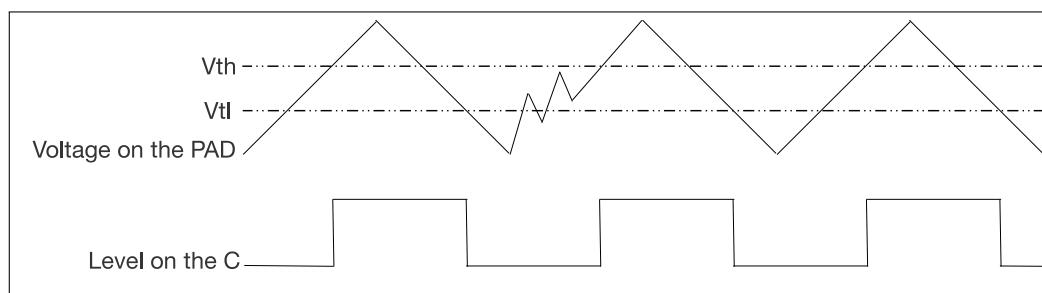


Figure 5.10-2. Example of Level Flip on the Chip Pad — Hysteresis Function Enabled

To enable the hysteresis function, follow the steps below:

- `IO_MUX_GPIO $n$ _HYS_SEL` = 0 (n = 0~14, 23~28, corresponding to GPIO0~GPIO14, GPIO23~GPIO28)
 - Set `EFUSE_HYS_EN_PAD` to enable the hysteresis function for GPIO0~GPIO14 and GPIO23~GPIO28.
 - Or clear `EFUSE_HYS_EN_PAD` to disable the hysteresis function for GPIO0~GPIO14 and GPIO23~GPIO28.
- `IO_MUX_GPIO $n$ _HYS_SEL` = 1 (n = 0~14, 23~28, corresponding to GPIO0~GPIO14, GPIO23~GPIO28)

- Set `IO_MUX_GPIO $n$ _HYS_EN` to enable the hysteresis function for GPIO n .
- Clear `IO_MUX_GPIO $n$ _HYS_EN` to disable the hysteresis function for GPIO n .

Recommended Operation:

- Set `IO_MUX_GPIO $n$ _HYS_SEL`.
- Then enable or disable the the hysteresis function for GPIO n using `IO_MUX_GPIO $n$ _HYS_EN`.

5.11 Power Supplies and Management of GPIO Pins

5.11.1 Power Supplies of GPIO Pins

For more information on the power supply for GPIO pins, please refer to Pin Definition in [ESP32-C5 Datasheet](#). All the pins can be used to wake up the chip from Light-sleep, but only the pins (GPIO0~GPIO6) in VDDPST1 domain can be used to wake up the chip from Deep-sleep.

5.11.2 Power Supply Management

Each ESP32-C5 pin is connected to one of the following power domains.

- VDDPST1: the input power supply for LP GPIO pins
- VDDPST2: the input power supply for HP GPIO pins

5.12 HP Peripheral Signal List

Table 5.12-1 shows the peripheral input/output signals via HP GPIO matrix.

Please pay attention to the configuration of the bit `GPIO_FUNC $n$ _OE_SEL`:

- `GPIO_FUNC $n$ _OE_SEL` = 1: the output enable is controlled by the corresponding bit n of `GPIO_ENABLE_REG`:
 - `GPIO_ENABLE_REG` = 0: output is disabled.
 - `GPIO_ENABLE_REG` = 1: output is enabled.
- `GPIO_FUNC $n$ _OE_SEL` = 0: use the output enable signal from peripheral, for example `SPIQ_oe` in the column “Output enable signal when `GPIO_FUNC $n$ _OE_SEL` = 0” of Table 5.12-1. Note that the signals such as `SPIQ_oe` can be 1 (1'd1) or 0 (1'd0), depending on the configuration of corresponding peripherals. If it's 1'd1 in column “Output enable signal when `GPIO_FUNC $n$ _OE_SEL` = 0”, it indicates that once `GPIO_FUNC $n$ _OE_SEL` is cleared, the output signal is always enabled by default.

Note:

Signals are numbered consecutively, but not all signals are valid.

- Only the signals with a name assigned in the column “Input signal” in Table 5.12-1 are valid input signals.
- Only the signals with a name assigned in the column “Output signal” in Table 5.12-1 are valid output signals.

Table 5.12-1. Peripheral Signals via HP GPIO Matrix

Signal No.	Input Signal	Default Value	Direct Input via HP IO MUX	Output Signal	Output Enable Signal when GPIO_FUNCn_OE_SEL = 0	Direct Output via HP IO MUX
0	—	-	-	ledc_ls_sig_out0	1'd1	no
1	—	—	—	ledc_ls_sig_out1	1'd1	no
2	—	—	—	ledc_ls_sig_out2	1'd1	no
3	—	—	—	ledc_ls_sig_out3	1'd1	no
4	—	—	—	ledc_ls_sig_out4	1'd1	no
5	—	—	—	ledc_ls_sig_out5	1'd1	no
6	UORXD_in	0	yes	UOTXD_out	1'd1	yes
7	UOCTS_in	0	no	UORTS_out	1'd1	no
8	UODSR_in	0	no	UODTR_out	1'd1	no
9	U1RXD_in	1	no	U1TXD_out	1'd1	no
10	U1CTS_in	0	no	U1RTS_out	1'd1	no
11	U1DSR_in	0	no	U1DTR_out	1'd1	no
12	I2S_MCLK_in	0	no	I2S_MCLK_out	1'd1	no
13	I2SO_BCK_in	0	no	I2SO_BCK_out	1'd1	no
14	I2SO_WS_in	0	no	I2SO_WS_out	1'd1	no
15	I2SI_SD_in	0	no	I2SO_SD_out	1'd1	no
16	I2SI_BCK_in	0	no	I2SI_BCK_out	1'd1	no
17	I2SI_WS_in	0	no	I2SI_WS_out	1'd1	no
18	—	—	—	I2SO_SD1_out	1'd1	no
19	—	—	—	—	—	—
20	—	—	—	—	—	—
21	—	—	—	—	—	—
22	—	—	—	—	—	—
23	—	—	—	—	—	—
24	—	—	—	—	—	—
25	—	—	—	—	—	—
26	—	—	—	—	—	—

Signal No.	Input Signal	Default Value	Direct Input via HP IO MUX	Output Signal	Output Enable Signal when GPIO_FUNCn_OE_SEL = 0	Direct Output via HP IO MUX
27	cpu_gpio_in0	0	no	cpu_gpio_out0	cpu_gpio_out_oen0	no
28	cpu_gpio_in1	0	no	cpu_gpio_out1	cpu_gpio_out_oen1	no
29	cpu_gpio_in2	0	no	cpu_gpio_out2	cpu_gpio_out_oen2	no
30	cpu_gpio_in3	0	no	cpu_gpio_out3	cpu_gpio_out_oen3	no
31	cpu_gpio_in4	0	no	cpu_gpio_out4	cpu_gpio_out_oen4	no
32	cpu_gpio_in5	0	no	cpu_gpio_out5	cpu_gpio_out_oen5	no
33	cpu_gpio_in6	0	no	cpu_gpio_out6	cpu_gpio_out_oen6	no
34	cpu_gpio_in7	0	no	cpu_gpio_out7	cpu_gpio_out_oen7	no
35	usb_jtag_tdo_bridge	0	no	—	—	—
36	—	—	—	—	—	—
37	—	—	—	—	—	—
38	—	—	—	—	—	—
39	—	—	—	—	—	—
40	—	—	—	—	—	—
41	—	—	—	—	—	—
42	—	—	—	—	—	—
43	—	—	—	—	—	—
44	—	—	—	—	—	—
45	—	—	—	—	—	—
46	I2CEXT0_SCL_in	1	no	I2CEXT0_SCL_out	I2CEXT0_SCL_oe	no
47	I2CEXT0_SDA_in	1	no	I2CEXT0_SDA_out	I2CEXT0_SDA_oe	no
48	parl_rx_data0	0	no	parl_tx_data0	1'd1	no
49	parl_rx_data1	0	no	parl_tx_data1	1'd1	no
50	parl_rx_data2	0	no	parl_tx_data2	1'd1	no
51	parl_rx_data3	0	no	parl_tx_data3	1'd1	no
52	parl_rx_data4	0	no	parl_tx_data4	1'd1	no
53	parl_rx_data5	0	no	parl_tx_data5	1'd1	no
54	parl_rx_data6	0	no	parl_tx_data6	1'd1	no

Signal No.	Input Signal	Default Value	Direct Input via HP IO MUX	Output Signal	Output Enable Signal when GPIO_FUNCn_OE_SEL = 0	Direct Output via HP IO MUX
55	parl_rx_data7	0	no	parl_tx_data7	1'd1	no
56	FSPICLK_in	0	yes	FSPICLK_out_mux	FSPICLK_oe	yes
57	FSPIQ_in	0	yes	FSPIQ_out	FSPIQ_oe	yes
58	FSPID_in	0	yes	FSPID_out	FSPID_oe	yes
59	FSPIHD_in	0	yes	FSPIHD_out	FSPIHD_oe	yes
60	FSPIWP_in	0	yes	FSPIWP_out	FSPIWP_oe	yes
61	FSPICSO_in	0	yes	FSPICSO_out	FSPICSO_oe	yes
62	parl_rx_clk_in	0	no	parl_rx_clk_out	1'd1	no
63	parl_tx_clk_in	0	no	parl_tx_clk_out	1'd1	no
64	rmt_sig_in0	0	no	rmt_sig_out0	1'd1	no
65	rmt_sig_in1	0	no	rmt_sig_out1	1'd1	no
66	twai0_rx	1	no	twai0_tx	1'd1	no
67	—	—	—	twai0_bus_off_on	1'd1	no
68	—	—	—	twai0_clkout	1'd1	no
69	—	—	—	twai0_standby	1'd1	no
70	twai1_rx	1	no	twai1_tx	1'd1	no
71	—	—	—	twai1_bus_off_on	1'd1	no
72	—	—	—	twai1_clkout	1'd1	no
73	—	—	—	twai1_standby	1'd1	no
74	—	—	—	—	—	—
75	—	—	—	—	—	—
76	pcnt_rst_in0	0	no	gpio_sd0_out	1'd1	no
77	pcnt_rst_in1	0	no	gpio_sd1_out	1'd1	no
78	pcnt_rst_in2	0	no	gpio_sd2_out	1'd1	no
79	pcnt_rst_in3	0	no	gpio_sd3_out	1'd1	no
80	pwm0_sync0_in	0	no	pwm0_out0a	1'd1	no
81	pwm0_sync1_in	0	no	pwm0_out0b	1'd1	no
82	pwm0_sync2_in	0	no	pwm0_out1a	1'd1	no

Signal No.	Input Signal	Default Value	Direct Input via HP IO MUX	Output Signal	Output Enable Signal when GPIO_FUNCn_OE_SEL = 0	Direct Output via HP IO MUX
83	pwm0_f0_in	0	no	pwm0_out1b	1'd1	no
84	pwm0_f1_in	0	no	pwm0_out2a	1'd1	no
85	pwm0_f2_in	0	no	pwm0_out2b	1'd1	no
86	pwm0_cap0_in	0	no	parl_tx_cs_o	1'd1	no
87	pwm0_cap1_in	0	no	—	—	—
88	pwm0_cap2_in	0	no	—	—	—
89	—	—	—	—	—	—
90	—	—	—	—	—	—
91	—	—	—	—	—	—
92	—	—	—	—	—	—
93	—	—	—	—	—	—
94	—	—	—	—	—	—
95	—	—	—	—	—	—
96	—	—	—	—	—	—
97	sig_in_func_97	0	no	sig_in_func97	1'd1	no
98	sig_in_func_98	0	no	sig_in_func98	1'd1	no
99	sig_in_func_99	0	no	sig_in_func99	1'd1	no
100	sig_in_func_100	0	no	sig_in_func100	1'd1	no
101	pcnt_sig_ch0_in0	0	no	FSPICS1_out	FSPICS1_oe	yes
102	pcnt_sig_ch1_in0	0	no	FSPICS2_out	FSPICS2_oe	yes
103	pcnt_ctrl_ch0_in0	0	no	FSPICS3_out	FSPICS3_oe	yes
104	pcnt_ctrl_ch1_in0	0	no	FSPICS4_out	FSPICS4_oe	yes
105	pcnt_sig_ch0_in1	0	no	FSPICS5_out	FSPICS5_oe	yes
106	pcnt_sig_ch1_in1	0	no	—	—	—
107	pcnt_ctrl_ch0_in1	0	no	—	—	—
108	pcnt_ctrl_ch1_in1	0	no	—	—	—
109	pcnt_sig_ch0_in2	0	no	—	—	—
110	pcnt_sig_ch1_in2	0	no	—	—	—

Signal No.	Input Signal	Default Value	Direct Input via HP IO MUX	Output Signal	Output Enable Signal when GPIO_FUNCn_OE_SEL = 0	Direct Output via HP IO MUX
111	pcnt_ctrl_ch0_in2	0	no	—	—	—
112	pcnt_ctrl_ch1_in2	0	no	—	—	—
113	pcnt_sig_ch0_in3	0	no	—	—	—
114	pcnt_sig_ch1_in3	0	no	—	—	—
115	pcnt_ctrl_ch0_in3	0	no	—	—	—
116	pcnt_ctrl_ch1_in3	0	no	—	—	—
117 ~ 255	—	—	—	—	—	—

5.13 HP IO MUX Function List

Table 5.13-1 shows the HP IO MUX functions and default states of each HP GPIO pin.

Table 5.13-1. HP IO MUX Pin Functions

GPIO No.	Name	Function 0	Function 1*	Function 2	Function 3	DRV	Reset	Notes
0	XTAL_32K_P	GPIO0	GPIO0	—	—	2	0	R
1	XTAL_32K_N	GPIO1	GPIO1	—	—	2	0	R
2	MTMS	MTMS	GPIO2	FSPIQ	—	2	1	R
3	MTDI	MTDI	GPIO3	—	—	2	1	R
4	MTCK	MTCK	GPIO4	FSPIHD	—	2	1*	R
5	MTDO	MTDO	GPIO5	FSPIWP	—	2	1	R
6	GPIO6	GPIO6	GPIO6	FSPICLK	—	2	0	R
7	GPIO7	SDIO_DATA1	GPIO7	FSPID	—	2	1	
8	GPIO8	SDIO_DATA0	GPIO8	—	—	2	0	—

Cont'd on next page

Table 5.13-1 – cont'd from previous page

GPIO No.	Name	Function 0	Function 1*	Function 2	Function 3	DRV	Reset	Notes
9	GPIO9	SDIO_CLK	GPIO9	—	—	2	0	—
10	GPIO10	SDIO_CMD	GPIO10	FSPICS0	—	2	0	—
11	U0TXD	U0TXD	GPIO11	—	—	2	4	—
12	U0RXD	U0RXD	GPIO12	—	—	2	1	—
13	GPIO13	SDIO_DATA3	GPIO13	—	—	3	1	USB
14	GPIO14	SDIO_DATA2	GPIO14	—	—	3	3*	USB
23	GPIO23	GPIO23	GPIO23	—	—	2	0	—
24	GPIO24	GPIO24	GPIO24	—	—	2	0	—
25	GPIO25	GPIO25	GPIO25	—	—	2	1	—
26	GPIO26	GPIO26	GPIO26	—	—	2	1	—
27	GPIO27	GPIO27	GPIO27	—	—	2	3	—
28	GPIO28	GPIO28	GPIO28	—	—	2	3	—

* In HP IO MUX, unused pins must be configured to GPIO function.

Drive Strength

“DRV” column shows the drive strength of each pin after reset:

- **0** - Drive current = ~5 mA
- **1** - Drive current = ~10 mA
- **2** - Drive current = ~20 mA
- **3** - Drive current = ~40 mA

Reset

The default configuration of each pin after reset:

- **0** - IE = 0 (input disabled)
- **1** - IE = 1 (input enabled)
- **3** - IE = 1, WPU = 1 (input enabled, pull-up resistor enabled)
- **4** - IE = 1, OE = 1 (input enabled, output enabled)
- **1*** - If EFUSE_DIS_PAD_JTAG = 1, the pin MTCK is left floating after reset, i.e., IE = 1. If EFUSE_DIS_PAD_JTAG = 0, the pin MTCK is connected to internal pull-up resistor, i.e., IE = 1, WPU = 1.
- **3*** - IE = 1, WPU = 0. The default value of GPIO14's USB pull-up is 1, which means the pull-up resistor is enabled. For details, please refer to the **Notes** below.

Notes

- **R** - LP pins. Some LP pins have analog functions. For details, see [5.15-1](#).
- **USB** - USB pull-up resistor enabled
 - By default, the USB function is enabled for USB pins (i.e., GPIO13 and GPIO14), and the pin pull-up is decided by the USB pull-up. The USB pull-up is controlled by USB_SERIAL_JTAG_DP/DM_PULLUP and the pull-up resistor value is controlled by USB_SERIAL_JTAG_PULLUP_VALUE. For details, see [ESP32-C5 Technical Reference Manual](#) > Chapter *USB Serial/JTAG Controller*.
 - When the USB function is disabled, USB pins are used as regular GPIOs and the pin's internal weak pull-up and pull-down resistors are disabled by default (configurable by IO_MUX_GPIO_n_MCU_WPU/WPD).

5.14 LP IO MUX Function List

Table [5.14-1](#) shows the LP IO MUX functions of each GPIO pin. GPIO pins are default controlled by HP IO MUX, so Table [5.14-1](#) only show functions of LP IO MUX.

Table 5.14-1. LP IO MUX Pin Functions

LP IO Name	Function 0	Function 1*	Function 2	Function 3
GPIO0	LP_UART_DTRN	LP_GPIO0	—	—
GPIO1	LP_UART_DSRN	LP_GPIO1	—	—
GPIO2	LP_UART_RTSN	LP_GPIO2	—	LP_I2C_SDA
GPIO3	LP_UART_CTSN	LP_GPIO3	—	LP_I2C_SCL

LP IO Name	Function 0	Function 1*	Function 2	Function 3
GPIO4	LP_UART_RXD_PAD	LP_GPIO4	—	—
GPIO5	LP_UART_TXD_PAD	LP_GPIO5	—	—
GPIO6	—	LP_GPIO6	—	—

Notice:

- Hardware flow control of LP_UART cannot be used with LP_I2C at the same time.
- In LP IO MUX, unused pins must be configured to LP_GPIO function.
- This column lists the LP GPIO names, since LP functions are configured with LP GPIO registers that use LP GPIO numbering.

5.15 GPIO Pin Analog Function List

Table 5.15-1 shows the GPIO pins and their corresponding analog functions.

Table 5.15-1. GPIO Pin Analog Functions

Name	Analog Function 0	Analog Function 1
XTAL_32K_P	XTAL_32K_P	
XTAL_32K_N	XTAL_32K_N	ADC1_CH0
MTMS	—	ADC1_CH1
MTDI	—	ADC1_CH2
MTCK	—	ADC1_CH3
MTDO	—	ADC1_CH4
GPIO6	—	ADC1_CH5
GPIO8	ZCD0*	—
GPIO9	ZCD1*	—
GPIO13	USB_D-	—
GPIO14	USB_D+	—

* ZCD0 and ZCD1 are analog PAD voltage comparator functions. See Chapter [37 Analog Voltage Comparator](#) for details.

5.16 Event Task Matrix Function

In ESP32-C5, GPIO supports ETM function, that is, the ETM task of GPIO can be triggered by the ETM event of any peripheral, or the ETM task of any peripheral can be triggered by the ETM event of GPIO. For more details about ETM, please refer to Chapter [8 Event Task Matrix \(ETM\)](#). Only ETM tasks and ETM events related to GPIO are introduced here.

The GPIO ETM provides eight task channels x (0~7). The ETM tasks that each task channel can receive are:

- GPIO_TASK_CH x _SET: GPIO goes high when triggered.

- GPIO_TASK_CH x _CLEAR: GPIO goes low when triggered.
- GPIO_TASK_CH x _TOGGLE: GPIO toggles level when triggered.

Below is an example to configure task channel x to control GPIO y :

- Configure IO_MUX_GPIO y _MCU_SEL to 1, to select Function 1 listed in Table 5.13-1.
- Configure GPIO_ENABLE_REG[y] to 1.
- Configure GPIO_EXT_ETM_TASK_GPIO y _SEL to x .
- Set GPIO_EXT_ETM_TASK_GPIO y _EN, to enable ETM task channel x to control GPIO y .

Note:

- One task channel can be selected by one or more GPIOs.
- When two or three of the signals GPIO_TASK_CH x _SET, GPIO_TASK_CH x _CLEAR, and GPIO_TASK_CH x _TOGGLE of the task channel x selected by GPIO y are valid at the same time, then GPIO_TASK_CH x _SET has the highest priority, GPIO_TASK_CH x _CLEAR takes the second higher priority, and GPIO_TASK_CH x _TOGGLE has the lowest priority.
- When GPIO y is controlled by ETM task channel, the values of GPIO_OUT_REG, GPIO_FUNC n _OUT_INV_SEL, and GPIO_FUNC n _OUT_SEL may be modified by the hardware. For such reason, it's recommended to reconfigure these registers when the GPIO is free from the control of ETM task channel.

GPIO has eight event channels, and the ETM events that each event channel can generate are:

- GPIO_EVT_CH x _RISE_EDGE: Indicates that the output signal of the corresponding GPIO Filter (see Figure 5.3-1) has a rising edge.
- GPIO_EVT_CH x _FALL_EDGE: Indicates that the output signal of the corresponding GPIO Filter (see Figure 5.3-1) has a falling edge.
- GPIO_EVT_CH x _ANY_EDGE: Indicates that the output signal of the corresponding GPIO Filter (see Figure 5.3-1) is reversed.

The specific configuration of the event channel is as follows:

- Set GPIO_EXT_ETM_CH x _EVENT_EN to enable event channel x (0~7).
- Configure GPIO_EXT_ETM_CH x _EVENT_SEL to y (0~14, 23~28), i.e., select one from the 21 GPIOs.

Note:

One GPIO can be selected by one or more event channels.

In some applications, GPIO ETM events can be used to trigger GPIO ETM tasks. For example, event channel 0 selects GPIO0, GPIO1 selects task channel 0, and the GPIO_EVT_CH0_RISE_EDGE event is used to trigger the GPIO_TASK_CH0_TOGGLE task. When a square wave signal is input to the chip through GPIO0, the chip outputs a square wave signal with a frequency divided by 2 through GPIO1.

5.17 Interrupts

ESP32-C5's HP IO MUX and HP GPIO matrix can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- GPIO_EXT_REG_INT
- GPIO_PROCPU_INT

Several internal interrupt sources from HP IO MUX and HP GPIO matrix can generate the above interrupt signals. The interrupt sources from HP IO MUX and HP GPIO matrix are listed with their trigger conditions and the resulted interrupt signals in Table 5.17-1.

Table 5.17-1. HP IO MUX and HP GPIO Matrix's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
GPIO_EXT_COMP_O_ALL_INT	See Chapter 37 Analog Voltage Comparator	GPIO_EXT_REG_INT
GPIO_EXT_COMP_O_NEG_INT	See Chapter 37 Analog Voltage Comparator	GPIO_EXT_REG_INT
GPIO_EXT_COMP_O_POS_INT	See Chapter 37 Analog Voltage Comparator	GPIO_EXT_REG_INT
GPIO_REG_INTR	GPIO_STATUS_INTERRUPT[n] & GPIO_PINn_INT_ENA[0] (n: 0~12, 23~28)	GPIO_PROCPU_INT

ESP32-C5's LP IO MUX and GPIO matrix can generate the following interrupt signal that will be sent to the LP_CORE.

- LP_GPIO_INTRO

The interrupt source from LP IO MUX and LP GPIO matrix is listed with its trigger condition and the resulted interrupt signal in Table 5.17-2.

Table 5.17-2. LP IO MUX and LP GPIO Matrix's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
LP_GPIO_REG_INTR	LP_GPIO_STATUS_INTERRUPT[n] (n: 0~6)	LP_GPIO_INTR

5.18 Register Summary

5.18.1 HP GPIO Matrix Register Summary

The addresses in this section are relative to HP GPIO matrix base address provided in Table 4.3-2 in Chapter [4 System and Memory](#).

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Configuration Registers			

Name	Description	Address	Access
GPIO_STRAP_REG	Strapping pin register	0x0000	RO
GPIO_OUT_REG	GPIO output register	0x0004	R/W/SC/WTC
GPIO_OUT_W1TS_REG	GPIO output set register	0x0008	WT
GPIO_OUT_W1TC_REG	GPIO output clear register	0x000C	WT
GPIO_ENABLE_REG	GPIO output enable register	0x0034	R/W/WTC
GPIO_ENABLE_W1TS_REG	GPIO output enable set register	0x0038	WT
GPIO_ENABLE_W1TC_REG	GPIO output enable clear register	0x003C	WT
GPIO_IN_REG	GPIO input register	0x0064	RO
GPIO_PROCPU_INT_REG	CPU interrupt status register for GPIO0~GPIO14, GPIO23~GPIO28	0x00A4	RO
Interrupt Status Registers			
GPIO_STATUS_REG	GPIO interrupt status register	0x0074	R/W/WTC
GPIO_STATUS_W1TS_REG	GPIO interrupt status set register	0x0078	WT
GPIO_STATUS_W1TC_REG	GPIO interrupt status clear register	0x007C	WT
GPIO_STATUS_NEXT_REG	GPIO interrupt source register	0x00B4	RO
Pin Configuration Registers			
GPIO_PIN0_REG	GPIO0 configuration register	0x00C4	R/W
GPIO_PIN1_REG	GPIO1 configuration register	0x00C8	R/W
GPIO_PIN2_REG	GPIO2 configuration register	0x00CC	R/W
GPIO_PIN3_REG	GPIO3 configuration register	0x00D0	R/W
GPIO_PIN4_REG	GPIO4 configuration register	0x00D4	R/W
GPIO_PIN5_REG	GPIO5 configuration register	0x00D8	R/W
GPIO_PIN6_REG	GPIO6 configuration register	0x00DC	R/W
GPIO_PIN7_REG	GPIO7 configuration register	0x00E0	R/W
GPIO_PIN8_REG	GPIO8 configuration register	0x00E4	R/W
GPIO_PIN9_REG	GPIO9 configuration register	0x00E8	R/W
GPIO_PIN10_REG	GPIO10 configuration register	0x00EC	R/W
GPIO_PIN11_REG	GPIO11 configuration register	0x00F0	R/W
GPIO_PIN12_REG	GPIO12 configuration register	0x00F4	R/W
GPIO_PIN13_REG	GPIO13 configuration register	0x00F8	R/W
GPIO_PIN14_REG	GPIO14 configuration register	0x00FC	R/W
GPIO_PIN23_REG	GPIO23 configuration register	0x0120	R/W
GPIO_PIN24_REG	GPIO24 configuration register	0x0124	R/W
GPIO_PIN25_REG	GPIO25 configuration register	0x0128	R/W
GPIO_PIN26_REG	GPIO26 configuration register	0x012C	R/W
GPIO_PIN27_REG	GPIO27 configuration register	0x0130	R/W
GPIO_PIN28_REG	GPIO28 configuration register	0x0134	R/W
Input Configuration Registers			
GPIO_FUNC6_IN_SEL_CFG_REG	Configuration register for input signal 6	0x02DC	R/W
GPIO_FUNC7_IN_SEL_CFG_REG	Configuration register for input signal 7	0x02E0	R/W
GPIO_FUNC8_IN_SEL_CFG_REG	Configuration register for input signal 8	0x02E4	R/W
GPIO_FUNC9_IN_SEL_CFG_REG	Configuration register for input signal 9	0x02E8	R/W
GPIO_FUNC10_IN_SEL_CFG_REG	Configuration register for input signal 10	0x02EC	R/W

Name	Description	Address	Access
GPIO_FUNC11_IN_SEL_CFG_REG	Configuration register for input signal 11	0x02F0	R/W
GPIO_FUNC12_IN_SEL_CFG_REG	Configuration register for input signal 12	0x02F4	R/W
GPIO_FUNC13_IN_SEL_CFG_REG	Configuration register for input signal 13	0x02F8	R/W
GPIO_FUNC14_IN_SEL_CFG_REG	Configuration register for input signal 14	0x02FC	R/W
GPIO_FUNC15_IN_SEL_CFG_REG	Configuration register for input signal 15	0x0300	R/W
GPIO_FUNC16_IN_SEL_CFG_REG	Configuration register for input signal 16	0x0304	R/W
GPIO_FUNC17_IN_SEL_CFG_REG	Configuration register for input signal 17	0x0308	R/W
GPIO_FUNC27_IN_SEL_CFG_REG	Configuration register for input signal 27	0x0330	R/W
GPIO_FUNC28_IN_SEL_CFG_REG	Configuration register for input signal 28	0x0334	R/W
GPIO_FUNC29_IN_SEL_CFG_REG	Configuration register for input signal 29	0x0338	R/W
GPIO_FUNC30_IN_SEL_CFG_REG	Configuration register for input signal 30	0x033C	R/W
GPIO_FUNC31_IN_SEL_CFG_REG	Configuration register for input signal 31	0x0340	R/W
GPIO_FUNC32_IN_SEL_CFG_REG	Configuration register for input signal 32	0x0344	R/W
GPIO_FUNC33_IN_SEL_CFG_REG	Configuration register for input signal 33	0x0348	R/W
GPIO_FUNC34_IN_SEL_CFG_REG	Configuration register for input signal 34	0x034C	R/W
GPIO_FUNC35_IN_SEL_CFG_REG	Configuration register for input signal 35	0x0350	R/W
GPIO_FUNC46_IN_SEL_CFG_REG	Configuration register for input signal 46	0x037C	R/W
GPIO_FUNC47_IN_SEL_CFG_REG	Configuration register for input signal 47	0x0380	R/W
GPIO_FUNC48_IN_SEL_CFG_REG	Configuration register for input signal 48	0x0384	R/W
GPIO_FUNC49_IN_SEL_CFG_REG	Configuration register for input signal 49	0x0388	R/W
GPIO_FUNC50_IN_SEL_CFG_REG	Configuration register for input signal 50	0x038C	R/W
GPIO_FUNC51_IN_SEL_CFG_REG	Configuration register for input signal 51	0x0390	R/W
GPIO_FUNC52_IN_SEL_CFG_REG	Configuration register for input signal 52	0x0394	R/W
GPIO_FUNC53_IN_SEL_CFG_REG	Configuration register for input signal 53	0x0398	R/W
GPIO_FUNC54_IN_SEL_CFG_REG	Configuration register for input signal 54	0x039C	R/W
GPIO_FUNC55_IN_SEL_CFG_REG	Configuration register for input signal 55	0x03A0	R/W
GPIO_FUNC56_IN_SEL_CFG_REG	Configuration register for input signal 56	0x03A4	R/W
GPIO_FUNC57_IN_SEL_CFG_REG	Configuration register for input signal 57	0x03A8	R/W
GPIO_FUNC58_IN_SEL_CFG_REG	Configuration register for input signal 58	0x03AC	R/W
GPIO_FUNC59_IN_SEL_CFG_REG	Configuration register for input signal 59	0x03B0	R/W
GPIO_FUNC60_IN_SEL_CFG_REG	Configuration register for input signal 60	0x03B4	R/W
GPIO_FUNC61_IN_SEL_CFG_REG	Configuration register for input signal 61	0x03B8	R/W
GPIO_FUNC62_IN_SEL_CFG_REG	Configuration register for input signal 62	0x03BC	R/W
GPIO_FUNC63_IN_SEL_CFG_REG	Configuration register for input signal 63	0x03C0	R/W
GPIO_FUNC64_IN_SEL_CFG_REG	Configuration register for input signal 64	0x03C4	R/W
GPIO_FUNC65_IN_SEL_CFG_REG	Configuration register for input signal 65	0x03C8	R/W
GPIO_FUNC66_IN_SEL_CFG_REG	Configuration register for input signal 66	0x03CC	R/W
GPIO_FUNC70_IN_SEL_CFG_REG	Configuration register for input signal 70	0x03DC	R/W
GPIO_FUNC75_IN_SEL_CFG_REG	Configuration register for input signal 75	0x03F0	R/W
GPIO_FUNC76_IN_SEL_CFG_REG	Configuration register for input signal 76	0x03F4	R/W
GPIO_FUNC77_IN_SEL_CFG_REG	Configuration register for input signal 77	0x03F8	R/W
GPIO_FUNC78_IN_SEL_CFG_REG	Configuration register for input signal 78	0x03FC	R/W
GPIO_FUNC79_IN_SEL_CFG_REG	Configuration register for input signal 79	0x0400	R/W

Name	Description	Address	Access
GPIO_FUNC80_IN_SEL_CFG_REG	Configuration register for input signal 80	0x0404	R/W
GPIO_FUNC81_IN_SEL_CFG_REG	Configuration register for input signal 81	0x0408	R/W
GPIO_FUNC82_IN_SEL_CFG_REG	Configuration register for input signal 82	0x040C	R/W
GPIO_FUNC83_IN_SEL_CFG_REG	Configuration register for input signal 83	0x0410	R/W
GPIO_FUNC84_IN_SEL_CFG_REG	Configuration register for input signal 84	0x0414	R/W
GPIO_FUNC85_IN_SEL_CFG_REG	Configuration register for input signal 85	0x0418	R/W
GPIO_FUNC86_IN_SEL_CFG_REG	Configuration register for input signal 86	0x041C	R/W
GPIO_FUNC87_IN_SEL_CFG_REG	Configuration register for input signal 87	0x0420	R/W
GPIO_FUNC88_IN_SEL_CFG_REG	Configuration register for input signal 88	0x0424	R/W
GPIO_FUNC97_IN_SEL_CFG_REG	Configuration register for input signal 97	0x0448	R/W
GPIO_FUNC98_IN_SEL_CFG_REG	Configuration register for input signal 98	0x044C	R/W
GPIO_FUNC99_IN_SEL_CFG_REG	Configuration register for input signal 99	0x0450	R/W
GPIO_FUNC100_IN_SEL_CFG_REG	Configuration register for input signal 100	0x0454	R/W
GPIO_FUNC101_IN_SEL_CFG_REG	Configuration register for input signal 101	0x0458	R/W
GPIO_FUNC102_IN_SEL_CFG_REG	Configuration register for input signal 102	0x045C	R/W
GPIO_FUNC103_IN_SEL_CFG_REG	Configuration register for input signal 103	0x0460	R/W
GPIO_FUNC104_IN_SEL_CFG_REG	Configuration register for input signal 104	0x0464	R/W
GPIO_FUNC105_IN_SEL_CFG_REG	Configuration register for input signal 105	0x0468	R/W
GPIO_FUNC106_IN_SEL_CFG_REG	Configuration register for input signal 106	0x046C	R/W
GPIO_FUNC107_IN_SEL_CFG_REG	Configuration register for input signal 107	0x0470	R/W
GPIO_FUNC108_IN_SEL_CFG_REG	Configuration register for input signal 108	0x0474	R/W
GPIO_FUNC109_IN_SEL_CFG_REG	Configuration register for input signal 109	0x0478	R/W
GPIO_FUNC110_IN_SEL_CFG_REG	Configuration register for input signal 110	0x047C	R/W
GPIO_FUNC111_IN_SEL_CFG_REG	Configuration register for input signal 111	0x0480	R/W
GPIO_FUNC112_IN_SEL_CFG_REG	Configuration register for input signal 112	0x0484	R/W
GPIO_FUNC113_IN_SEL_CFG_REG	Configuration register for input signal 113	0x0488	R/W
GPIO_FUNC114_IN_SEL_CFG_REG	Configuration register for input signal 114	0x048C	R/W
GPIO_FUNC115_IN_SEL_CFG_REG	Configuration register for input signal 115	0x0490	R/W
GPIO_FUNC116_IN_SEL_CFG_REG	Configuration register for input signal 116	0x0494	R/W
Output Configuration Registers			
GPIO_FUNC0_OUT_SEL_CFG_REG	Configuration register for GPIO0 output	0x0AC4	varies
GPIO_FUNC1_OUT_SEL_CFG_REG	Configuration register for GPIO1 output	0x0AC8	varies
GPIO_FUNC2_OUT_SEL_CFG_REG	Configuration register for GPIO2 output	0x0ACC	varies
GPIO_FUNC3_OUT_SEL_CFG_REG	Configuration register for GPIO3 output	0x0AD0	varies
GPIO_FUNC4_OUT_SEL_CFG_REG	Configuration register for GPIO4 output	0x0AD4	varies
GPIO_FUNC5_OUT_SEL_CFG_REG	Configuration register for GPIO5 output	0x0AD8	varies
GPIO_FUNC6_OUT_SEL_CFG_REG	Configuration register for GPIO6 output	0x0ADC	varies
GPIO_FUNC7_OUT_SEL_CFG_REG	Configuration register for GPIO7 output	0x0AE0	varies
GPIO_FUNC8_OUT_SEL_CFG_REG	Configuration register for GPIO8 output	0x0AE4	varies
GPIO_FUNC9_OUT_SEL_CFG_REG	Configuration register for GPIO9 output	0x0AE8	varies
GPIO_FUNC10_OUT_SEL_CFG_REG	Configuration register for GPIO10 output	0x0AEC	varies
GPIO_FUNC11_OUT_SEL_CFG_REG	Configuration register for GPIO11 output	0x0AF0	varies
GPIO_FUNC12_OUT_SEL_CFG_REG	Configuration register for GPIO12 output	0x0AF4	varies

Name	Description	Address	Access
GPIO_FUNC13_OUT_SEL_CFG_REG	Configuration register for GPIO13 output	0x0AF8	varies
GPIO_FUNC14_OUT_SEL_CFG_REG	Configuration register for GPIO14 output	0x0AFC	varies
GPIO_FUNC23_OUT_SEL_CFG_REG	Configuration register for GPIO23 output	0x0B20	varies
GPIO_FUNC24_OUT_SEL_CFG_REG	Configuration register for GPIO24 output	0x0B24	varies
GPIO_FUNC25_OUT_SEL_CFG_REG	Configuration register for GPIO25 output	0x0B28	varies
GPIO_FUNC26_OUT_SEL_CFG_REG	Configuration register for GPIO26 output	0x0B2C	varies
GPIO_FUNC27_OUT_SEL_CFG_REG	Configuration register for GPIO27 output	0x0B30	varies
GPIO_FUNC28_OUT_SEL_CFG_REG	Configuration register for GPIO28 output	0x0B34	varies
Clock Gate Register			
GPIO_CLOCK_GATE_REG	GPIO clock gate register	0x0DF8	R/W
Version Register			
GPIO_DATE_REG	GPIO version register	0x0DFC	R/W

5.18.2 HP IO MUX Register Summary

The addresses in this section are relative to the HP IO MUX base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
IO_MUX_GPIO0_REG	IO MUX configuration register for GPIO0	0x0000	R/W
IO_MUX_GPIO1_REG	IO MUX configuration register for GPIO1	0x0004	R/W
IO_MUX_GPIO2_REG	IO MUX configuration register for GPIO2	0x0008	R/W
IO_MUX_GPIO3_REG	IO MUX configuration register for GPIO3	0x000C	R/W
IO_MUX_GPIO4_REG	IO MUX configuration register for GPIO4	0x0010	R/W
IO_MUX_GPIO5_REG	IO MUX configuration register for GPIO5	0x0014	R/W
IO_MUX_GPIO6_REG	IO MUX configuration register for GPIO6	0x0018	R/W
IO_MUX_GPIO7_REG	IO MUX configuration register for GPIO7	0x001C	R/W
IO_MUX_GPIO8_REG	IO MUX configuration register for GPIO8	0x0020	R/W
IO_MUX_GPIO9_REG	IO MUX configuration register for GPIO9	0x0024	R/W
IO_MUX_GPIO10_REG	IO MUX configuration register for GPIO10	0x0028	R/W
IO_MUX_GPIO11_REG	IO MUX configuration register for GPIO11	0x002C	R/W
IO_MUX_GPIO12_REG	IO MUX configuration register for GPIO12	0x0030	R/W
IO_MUX_GPIO13_REG	IO MUX configuration register for GPIO13	0x0034	R/W
IO_MUX_GPIO14_REG	IO MUX configuration register for GPIO14	0x0038	R/W
IO_MUX_GPIO23_REG	IO MUX configuration register for GPIO23	0x005C	R/W
IO_MUX_GPIO24_REG	IO MUX configuration register for GPIO24	0x0060	R/W
IO_MUX_GPIO25_REG	IO MUX configuration register for GPIO25	0x0064	R/W
IO_MUX_GPIO26_REG	IO MUX configuration register for GPIO26	0x0068	R/W
IO_MUX_GPIO27_REG	IO MUX configuration register for GPIO27	0x006C	R/W
IO_MUX_GPIO28_REG	IO MUX configuration register for GPIO28	0x0070	R/W
Version Register			

Name	Description	Address	Access
IO_MUX_DATE_REG	Version control register	0x01FC	R/W

5.18.3 GPIO EXT Register Summary

GPIO EXT registers consist of SDM registers, Glitch Filter registers, and ETM registers.

The addresses in this section are relative to (HP GPIO matrix base address + 0x0F00). GPIO base address is provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
SDM Configuration Registers			
GPIO_EXT_SIGMADELTA_MISC_REG	MISC Register	0x0004	R/W
GPIO_EXT_SIGMADELTA0_REG	Duty cycle configuration register for SDM channel 0	0x0008	R/W
GPIO_EXT_SIGMADELTA1_REG	Duty cycle configuration register for SDM channel 1	0x000C	R/W
GPIO_EXT_SIGMADELTA2_REG	Duty cycle configuration register for SDM channel 2	0x0010	R/W
GPIO_EXT_SIGMADELTA3_REG	Duty cycle configuration register for SDM channel 3	0x0014	R/W
PAD Comparator Configuration Registers			
GPIO_EXT_PAD_COMP_CONFIG_0_REG	Configuration register for zero-crossing detection	0x0058	R/W
GPIO_EXT_PAD_COMP_FILTER_0_REG	Configuration register for interrupt source mask period of zero-crossing detection	0x005C	R/W
Glitch Filter Configuration Registers			
GPIO_EXT_GLITCH_FILTER_CH0_REG	Configuration register of channel 0	0x00D8	R/W
GPIO_EXT_GLITCH_FILTER_CH1_REG	Configuration register of channel 1	0x00DC	R/W
GPIO_EXT_GLITCH_FILTER_CH2_REG	Configuration register of channel 2	0x00E0	R/W
GPIO_EXT_GLITCH_FILTER_CH3_REG	Configuration register of channel 3	0x00E4	R/W
GPIO_EXT_GLITCH_FILTER_CH4_REG	Configuration register of channel 4	0x00E8	R/W
GPIO_EXT_GLITCH_FILTER_CH5_REG	Configuration register of channel 5	0x00EC	R/W
GPIO_EXT_GLITCH_FILTER_CH6_REG	Configuration register of channel 6	0x00F0	R/W
GPIO_EXT_GLITCH_FILTER_CH7_REG	Configuration register of channel 7	0x00F4	R/W
ETM Configuration Registers			
GPIO_EXT_ETM_EVENT_CH0_CFG_REG	Configuration register of channel 0	0x0118	R/W
GPIO_EXT_ETM_EVENT_CH1_CFG_REG	Configuration register of channel 1	0x011C	R/W
GPIO_EXT_ETM_EVENT_CH2_CFG_REG	Configuration register of channel 2	0x0120	R/W
GPIO_EXT_ETM_EVENT_CH3_CFG_REG	Configuration register of channel 3	0x0124	R/W
GPIO_EXT_ETM_EVENT_CH4_CFG_REG	Configuration register of channel 4	0x0128	R/W
GPIO_EXT_ETM_EVENT_CH5_CFG_REG	Configuration register of channel 5	0x012C	R/W
GPIO_EXT_ETM_EVENT_CH6_CFG_REG	Configuration register of channel 6	0x0130	R/W
GPIO_EXT_ETM_EVENT_CH7_CFG_REG	Configuration register of channel 7	0x0134	R/W

Name	Description	Address	Access
GPIO_EXT_ETM_TASK_P0_CFG_REG	GPIO selection register 0	0x0158	R/W
GPIO_EXT_ETM_TASK_P1_CFG_REG	GPIO selection register 1	0x015C	R/W
GPIO_EXT_ETM_TASK_P2_CFG_REG	GPIO selection register 2	0x0160	R/W
GPIO_EXT_ETM_TASK_P3_CFG_REG	GPIO selection register 3	0x0164	R/W
GPIO_EXT_ETM_TASK_P4_CFG_REG	GPIO selection register 4	0x0168	R/W
GPIO_EXT_ETM_TASK_P5_CFG_REG	GPIO selection register 5	0x016C	R/W
Interrupt Registers			
GPIO_EXT_INT_RAW_REG	GPIO_EXT interrupt raw register	0x01D0	RO/WTC/SS
GPIO_EXT_INT_ST_REG	GPIO_EXT interrupt masked register	0x01D4	RO
GPIO_EXT_INT_ENA_REG	GPIO_EXT interrupt enable register	0x01D8	R/W
GPIO_EXT_INT_CLR_REG	GPIO_EXT interrupt clear register	0x01DC	WT
Version Register			
GPIO_EXT_VERSION_REG	Version Control Register	0x01FC	R/W

5.18.4 LP GPIO Matrix Register Summary

The addresses in this section are relative to LP GPIO matrix base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
LP_GPIO_OUT_REG	LP GPIO output register	0x0004	R/W/WTC
LP_GPIO_OUT_WITS_REG	LP GPIO output set register	0x0008	WT
LP_GPIO_OUT_WITC_REG	LP GPIO output clear register	0x000C	WT
LP_GPIO_ENABLE_REG	LP GPIO output enable register	0x0010	R/W/WTC
LP_GPIO_ENABLE_WITS_REG	LP GPIO output enable set register	0x0014	WT
LP_GPIO_ENABLE_WITC_REG	LP GPIO output enable clear register	0x0018	WT
LP_GPIO_IN_REG	LP GPIO input register	0x001C	RO
LP_GPIO_STATUS_REG	LP GPIO interrupt status register	0x0020	R/W/WTC
LP_GPIO_STATUS_WITS_REG	LP GPIO interrupt status set register	0x0024	WT
LP_GPIO_STATUS_WITC_REG	LP GPIO interrupt status clear register	0x0028	WT
LP_GPIO_STATUS_NEXT_REG	LP GPIO interrupt source register	0x002C	RO
LP_GPIO_PIN0_REG	LP GPIO0 configuration register	0x0030	varies
LP_GPIO_PIN1_REG	LP GPIO1 configuration register	0x0034	varies
LP_GPIO_PIN2_REG	LP GPIO2 configuration register	0x0038	varies
LP_GPIO_PIN3_REG	LP GPIO3 configuration register	0x003C	varies
LP_GPIO_PIN4_REG	LP GPIO4 configuration register	0x0040	varies
LP_GPIO_PIN5_REG	LP GPIO5 configuration register	0x0044	varies
LP_GPIO_PIN6_REG	LP GPIO6 configuration register	0x0048	varies
LP_GPIO_PIN7_REG	LP GPIO7 configuration register	0x004C	varies
LP_GPIO_FUNC0_OUT_SEL_CFG_REG	Configuration register for GPIO0 output	0x02B0	R/W
LP_GPIO_FUNC1_OUT_SEL_CFG_REG	Configuration register for GPIO1 output	0x02B4	R/W

Name	Description	Address	Access
LP_GPIO_FUNC2_OUT_SEL_CFG_REG	Configuration register for GPIO2 output	0x02B8	R/W
LP_GPIO_FUNC3_OUT_SEL_CFG_REG	Configuration register for GPIO3 output	0x02BC	R/W
LP_GPIO_FUNC4_OUT_SEL_CFG_REG	Configuration register for GPIO4 output	0x02C0	R/W
LP_GPIO_FUNC5_OUT_SEL_CFG_REG	Configuration register for GPIO5 output	0x02C4	R/W
LP_GPIO_FUNC6_OUT_SEL_CFG_REG	Configuration register for GPIO6 output	0x02C8	R/W
LP_GPIO_CLOCK_GATE_REG	GPIO clock gate register	0x03F8	R/W
Version Register			
LP_GPIO_DATE_REG	Version control register	0x03FC	R/W

5.18.5 LP IO MUX Register Summary

The addresses in this section are relative to LP IO MUX base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configure Registers			
LP_IO_MUX_GPIO0_REG	LP IO MUX configuration register for pin GPIO0	0x0000	R/W
LP_IO_MUX_GPIO1_REG	LP IO MUX configuration register for pin GPIO1	0x0004	R/W
LP_IO_MUX_GPIO2_REG	LP IO MUX configuration register for pin GPIO2	0x0008	R/W
LP_IO_MUX_GPIO3_REG	LP IO MUX configuration register for pin GPIO3	0x000C	R/W
LP_IO_MUX_GPIO4_REG	LP IO MUX configuration register for pin GPIO4	0x0010	R/W
LP_IO_MUX_GPIO5_REG	LP IO MUX configuration register for pin GPIO5	0x0014	R/W
LP_IO_MUX_GPIO6_REG	LP IO MUX configuration register for pin GPIO6	0x0018	R/W
Version Register			
LP_IO_MUX_DATE_REG	Version control register	0x01FC	R/W

5.19 Registers

5.19.1 HP GPIO Matrix Registers

The addresses in this section are relative to HP GPIO matrix base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section VII .

Register 5.1. GPIO_STRAP_REG (0x0000)

(reserved)																GPIO_STRAPPING																
3116																150																
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x00																Reset

GPIO_STRAPPING Represents the values of GPIO strapping pins.

- Bit[0]: Invalid
- Bit[1]: Invalid
- Bit[2]: Invalid
- Bit[3]: GPIO27
- Bit[4]: GPIO28
- Bit[5]: GPIO7
- Bit[6]: GPIO25
- Bit[7]~bit[15]: Invalid

For more information about the functions controlled by strapping pins, see Chapter 7 [Chip Boot Control](#).
(RO)

Register 5.2. GPIO_OUT_REG (0x0004)

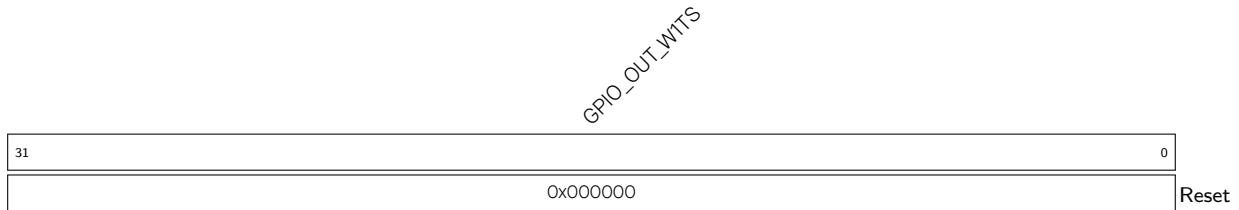
GPIO_OUT_DATA_ORIG																																
31																															0	
0x000000																																Reset

GPIO_OUT_DATA_ORIG Configures the output value of GPIO0~GPIO14 and GPIO23~GPIO28 output in simple GPIO output mode.

0: Low level
1: High level

The value of bit[0]~bit[14] and bit[23]~bit[28] corresponds to the output value of GPIO0~GPIO14 and GPIO23~GPIO28 respectively.

(R/W/SC/WTC)

Register 5.3. GPIO_OUT_W1TS_REG (0x0008)

GPIO_OUT_W1TS Configures whether or not to set the output register [GPIO_OUT_REG](#) of GPIO0~GPIO14 and GPIO23~GPIO28.

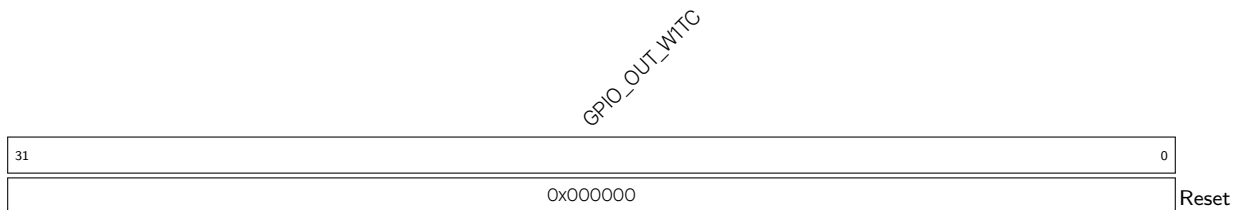
0: Not set

1: The corresponding bit in [GPIO_OUT_REG](#) will be set to 1

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Recommended operation: use this register to set [GPIO_OUT_REG](#).

(WT)

Register 5.4. GPIO_OUT_W1TC_REG (0x000C)

GPIO_OUT_W1TC Configures whether or not to clear the output register [GPIO_OUT_REG](#) of GPIO0~GPIO14 and GPIO23~GPIO28 output.

0: Not clear

1: The corresponding bit in [GPIO_OUT_REG](#) will be cleared.

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Recommended operation: use this register to clear [GPIO_OUT_REG](#).

(WT)

Register 5.5. GPIO_ENABLE_REG (0x0034)

<i>GPIO_ENABLE_DATA</i>	
31	0
0x000000	
Reset	

GPIO_ENABLE_DATA Configures whether or not to enable the output of GPIO0~GPIO14 and GPIO23~GPIO28.

0: Not enable

1: Enable

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

(R/W/WTC)

Register 5.6. GPIO_ENABLE_WITS_REG (0x0038)

<i>GPIO_ENABLE_WITS</i>	
31	0
0x000000	
Reset	

GPIO_ENABLE_WITS Configures whether or not to set the output enable register [GPIO_ENABLE_REG](#) of GPIO0~GPIO14 and GPIO23~GPIO28.

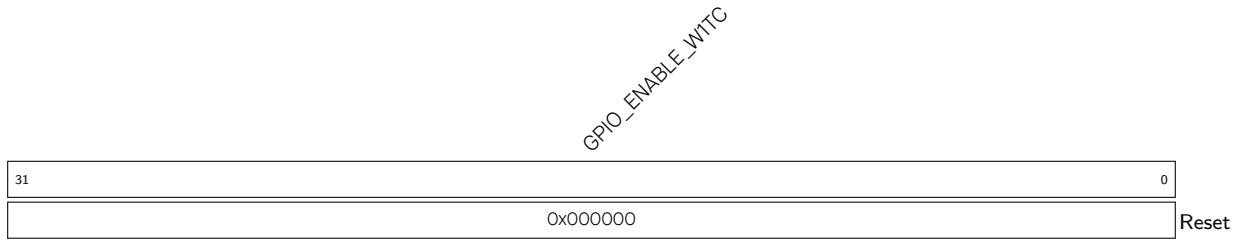
0: Not set

1: The corresponding bit in [GPIO_ENABLE_REG](#) will be set to 1

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Recommended operation: use this register to set [GPIO_ENABLE_REG](#).

(WT)

Register 5.7. GPIO_ENABLE_W1TC_REG (0x003C)

GPIO_ENABLE_W1TC Configures whether or not to clear the output enable register [GPIO_ENABLE_REG](#) of GPIO0~GPIO14 and GPIO23~GPIO28.

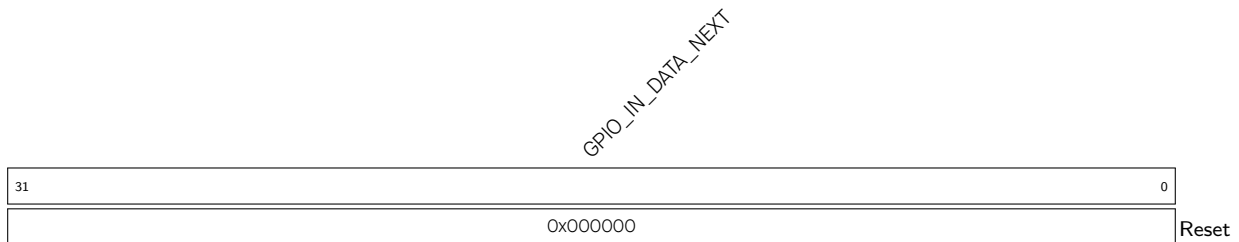
0: Not clear

1: The corresponding bit in [GPIO_ENABLE_REG](#) will be cleared

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Recommended operation: use this register to clear [GPIO_ENABLE_REG](#).

(WT)

Register 5.8. GPIO_IN_REG (0x0064)

GPIO_IN_DATA_NEXT Represents the input value of GPIO0~GPIO14 and GPIO23~GPIO28.

Each bit represents a pin input value:

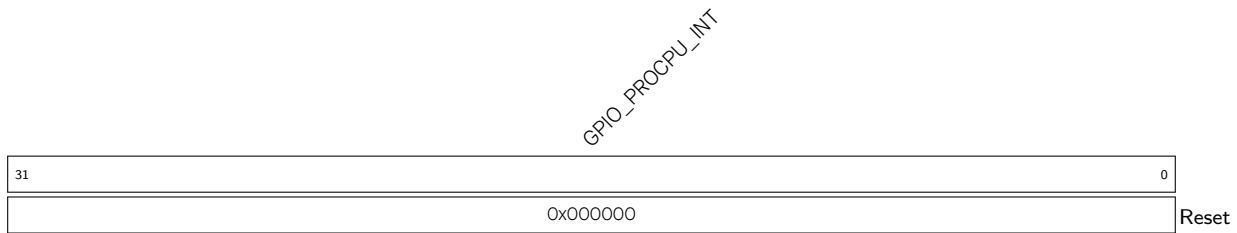
0: Low level

1: High level

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

(RO)

Register 5.9. GPIO_PROCPU_INT_REG (0x00A4)



GPIO_PROCPU_INT Represents the CPU interrupt status of GPIO0~GPIO14 and GPIO23~GPIO28.

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Each bit represents:

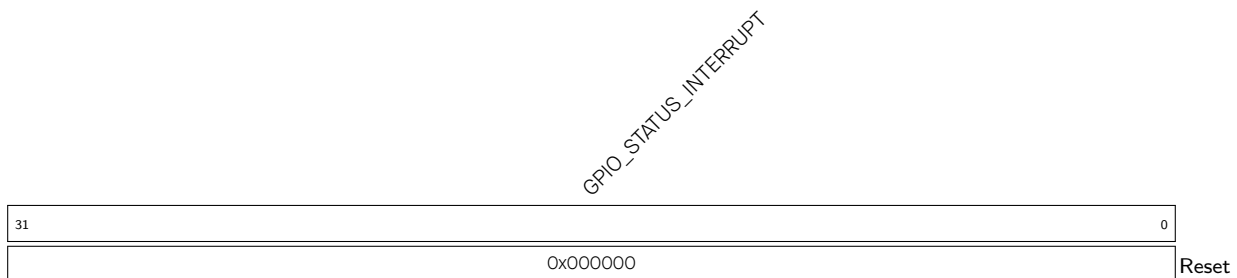
0: Represents CPU interrupt is not enabled, or the GPIO does not generate the interrupt configured by [GPIO_PIN \$n\$ _INT_TYPE](#).

1: Represents the GPIO generates an interrupt configured by [GPIO_PIN \$n\$ _INT_TYPE](#) after the CPU interrupt is enabled.

This interrupt status is corresponding to the bit in [GPIO_STATUS_REG](#) when assert (high) enable signal (bit13 of [GPIO_PIN \$n\$ _REG](#)).

(RO)

Register 5.10. GPIO_STATUS_REG (0x0074)



GPIO_STATUS_INTERRUPT The interrupt status of GPIO0~GPIO14 and GPIO23~GPIO28, can be configured by the software.

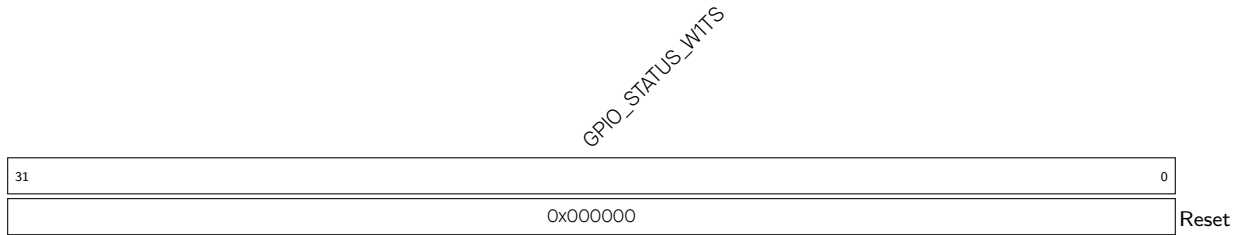
Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

Each bit represents the status of its corresponding GPIO:

- 0: Represents the GPIO does not generate the interrupt configured by [GPIO_PIN \$n\$ _INT_TYPE](#), or this bit is configured to 0 by the software.

- 1: Represents the GPIO generates the interrupt configured by [GPIO_PIN \$n\$ _INT_TYPE](#), or this bit is configured to 1 by the software.

(R/W/WTC)

Register 5.11. GPIO_STATUS_W1TS_REG (0x0078)

GPIO_STATUS_W1TS Configures whether or not to set the interrupt status register [GPIO_STATUS_INTERRUPT](#) of GPIO0~GPIO14 and GPIO23~GPIO28.

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

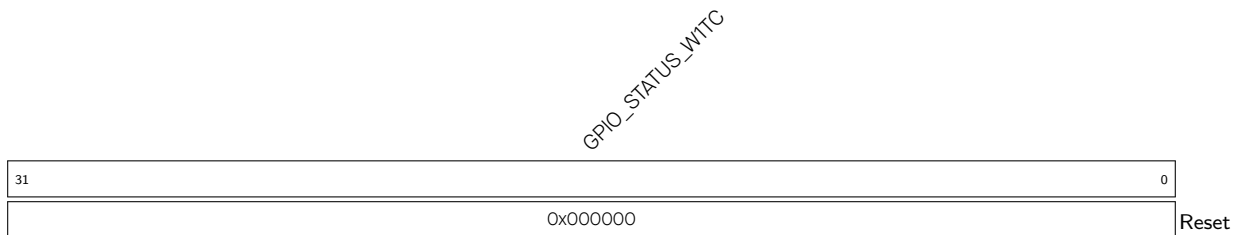
The value of each bit can be:

0: Not set

1: The corresponding bit in [GPIO_STATUS_INTERRUPT](#) will be set to 1.

Recommended operation: use this register to set [GPIO_STATUS_INTERRUPT](#).

(WT)

Register 5.12. GPIO_STATUS_W1TC_REG (0x007C)

GPIO_STATUS_W1TC Configures whether or not to clear the interrupt status register [GPIO_STATUS_INTERRUPT](#) of GPIO0~GPIO14 and GPIO23~GPIO28.

Bit[0]~bit[14] and bit[23]~bit[28] are corresponding to GPIO0~GPIO14 and GPIO23~GPIO28.

The value of each bit can be:

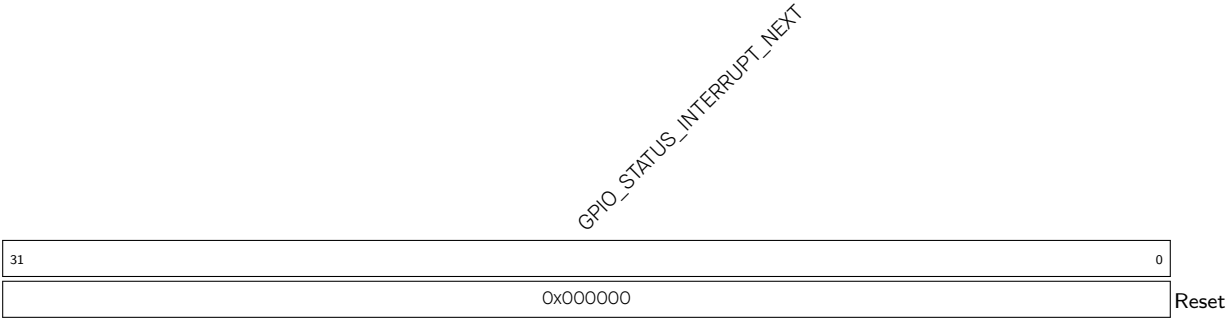
0: Not clear

1: The corresponding bit in [GPIO_STATUS_INTERRUPT](#) will be cleared.

Recommended operation: use this register to clear [GPIO_STATUS_INTERRUPT](#).

(WT)

Register 5.13. GPIO_STATUS_NEXT_REG (0x00B4)



GPIO_STATUS_INTERRUPT_NEXT Represents the interrupt source signal of GPIO0~GPIO14 and GPIO23~GPIO28.
Each bit represents:
0: The GPIO does not generate the interrupt configured by `GPIO_PINn_INT_TYPE`.
1: The GPIO generates an interrupt configured by `GPIO_PINn_INT_TYPE`.
The interrupt could be rising edge interrupt, falling edge interrupt, level sensitive interrupt and any edge interrupt.
(RO)

Register 5.14. GPIO_PIN*n*_REG (*n*: 0-28) (0x00C4+0x4**n*)

(reserved)																GPIO_PIN _n _INT_ENA				(reserved)				GPIO_PIN _n _WAKEUP_ENABLE				GPIO_PIN _n _INT_TYPE				(reserved)				GPIO_PIN _n _SYNC1_BYPASS				GPIO_PIN _n _PAD_DRIVER				GPIO_PIN _n _SYNC2_BYPASS			
31 18 17 13 12 11 10 9 7 6 5 4 3 2 1 0																																															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x0				0 0 0				0x0				0 0				0x0				0				0x0				Reset			

GPIO_PIN_n_SYNC2_BYPASS Configures whether or not to synchronize GPIO input data on either edge of IO MUX operating clock for the second-level synchronization.

0: Not synchronize

1: Synchronize on falling edge

2: Synchronize on rising edge

3: Synchronize on rising edge

(R/W)

GPIO_PIN n _PAD_DRIVER Configures to select pin drive mode.

0: Normal output

1: Open drain output

(R/W)

GPIO_PIN_n_SYNC1_BYPASS Configures whether or not to synchronize GPIO input data on either edge of IO MUX operating clock for the first-level synchronization.

0: Not synchronize

1: Synchronize on falling edge

2: Synchronize on rising edge

3: Synchronize on rising edge

(R/W)

GPIO_PIN_n_INT_TYPE Configures GPIO interrupt type.

0: GPIO interrupt disabled

1: Rising edge trigger

2: Falling edge trigger

3: Any edge trigger

4: Low level trigger

5: High level trigger

(R/W)

GPIO_PIN_n_WAKEUP_ENABLE Configures whether or not to enable GPIO wakeup function.

0: Disable

1: Enable

This function only wakes up the CPU from Light-sleep.

(R/W)

Continued on the next page...

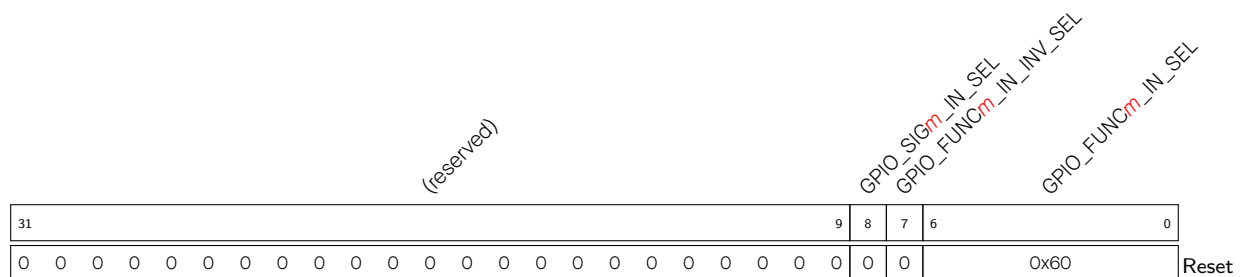
Register 5.14. GPIO_PIN n _REG (n : 0-28) (0x00C4+0x4* n)

Continued from the previous page...

GPIO_PIN n _INT_ENA Configures whether or not to enable CPU interrupt or CPU non-maskable interrupt.

- Bit[13]: Configures whether or not to enable CPU interrupt:
0: Disable
1: Enable
 - Bit[14]~bit[17]: Invalid
- (R/W)

Register 5.15. GPIO_FUNC*m*_IN_SEL_CFG_REG (*m*: 6-17) (0x02DC+0x4*(*m*-6))



GPIO_FUNC*m***_IN_SEL** Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal *m*.

- 0x0: Select GPIO0
0x1: Select GPIO1
.....
0x1B: Select GPIO27
0x1C: Select GPIO28
Or
0x40: A constantly high input
0x60: A constantly low input
(R/W)

GPIO_FUNC_mIN_INV_SEL Configures whether or not to invert the input value.

- 0: Not invert
1: Invert
(R/W)

GPIO_SIG_mIN_SEL Configures whether or not to route signals via GPIO matrix.

- 0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.
1: Route signals via GPIO matrix.
(R/W)

Register 5.16. GPIO_FUNC

IN_SEL_CFG_REG (*p*: 27-35) (0x0330+0x4*(*p*-27))

Diagram illustrating the GPIO configuration register structure (bits 31 to 0):

- Bits 31 to 9: (reserved)
- Bits 8 to 7: GPIO_SIG_IN_SEL
- Bit 6: GPIO_FUNC_IN_INV_SEL
- Bits 5 to 0: GPIO_FUNC_IN_SEL (Default value: 0x60)

Reset

GPIO_FUNC*p_IN_SEL* Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal *p*.

0x0: Select GPIO0

0x1: Select GPIO1

• • • • •

0x1B: Select GPIO27

0x1C: Select GPIO28

Or

0x40: A constantly high input

0x60: A constantly low input

(R/W)

GPIO_FUNC*p*_IN_INV_SEL Configures whether or not to invert the input value.

0: Not invert

1: Invert

(R/W)

GPIO_SIG_p_IN_SEL Configures whether or not to route signals via GPIO matrix.

0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.

1: Route signals via GPIO matrix.

(R/W)

Register 5.17. GPIO_FUNC*r*_IN_SEL_CFG_REG (*r*: 46-66) (0x037C+0x4*(*r*-46))

[illegible]

GPIO_FUNC*r***_IN_SEL** Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal *r*.

0x0: Select GPIO0

0x1: Select GPIO1

• • • • •

0x1B: Select GPIO27

0x1C: Select GPIO28

Or

0x40: A constantly high input

0x60: A constantly low input

(R/W)

GPIO_FUNC_r_IN_INV_SEL Configures whether or not to invert the input value.

0: Not invert

1: Invert

(R/W)

GPIO_SIG_r_IN_SEL Configures whether or not to route signals via GPIO matrix.

0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.

1: Route signals via GPIO matrix.

(R/W)

Register 5.18. GPIO_FUNC70_IN_SEL_CFG_REG (0x03DC)

[illegible]

GPIO_FUNC70_IN_SEL Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal 70.

0x0: Select GPIO0

0x1: Select GPIO1

• • • • •

0x1B: Select GPIO27

0x1C: Select GPIO28

Or

0x40: A constantly high input

0x60: A constantly low input

(R/W)

GPIO_FUNC70_IN_INV_SEL Configures whether or not to invert the input value.

0: Not invert

1: Invert

(R/W)

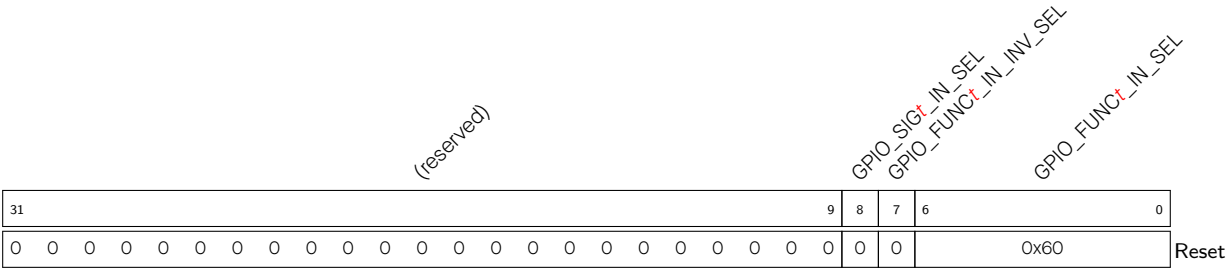
GPIO_SIG70_IN_SEL Configures whether or not to route signals via GPIO matrix.

0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.

1: Route signals via GPIO matrix.

(R/W)

Register 5.19. GPIO_FUNC*t*_IN_SEL_CFG_REG (*t*: 76-88) (0x03F4+0x4*(*t*-76))



GPIO_FUNC*t*_IN_SEL Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal *t*.

0x0: Select GPIO0
0x1: Select GPIO1
.....
0x1B: Select GPIO27
0x1C: Select GPIO28
Or
0x40: A constantly high input
0x60: A constantly low input
(R/W)

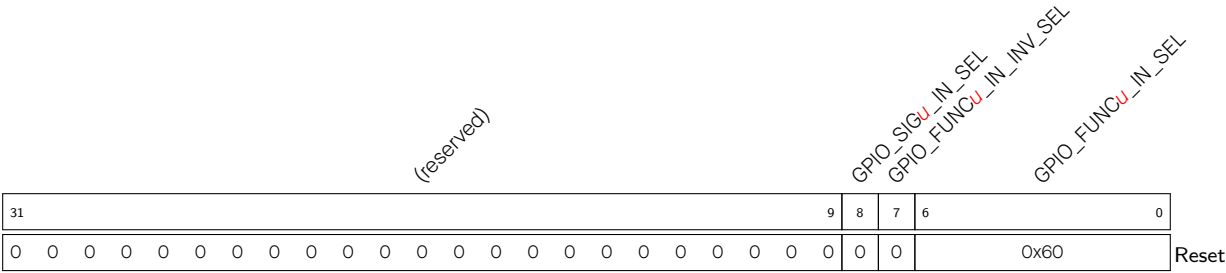
GPIO_FUNC*t*_IN_INV_SEL Configures whether or not to invert the input value.

0: Not invert
1: Invert
(R/W)

GPIO_SIG*t*_IN_SEL Configures whether or not to route signals via GPIO matrix.

0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.
1: Route signals via GPIO matrix.
(R/W)

Register 5.20. GPIO_FUNCU_IN_SEL_CFG_REG (*U*: 97-116) (0x0448+0x4*(*U*-97))



GPIO_FUNCU_IN_SEL Configures to select a pin from the 21 GPIO pins (GPIO0~GPIO14, GPIO23~GPIO28) to connect the input signal *U*.

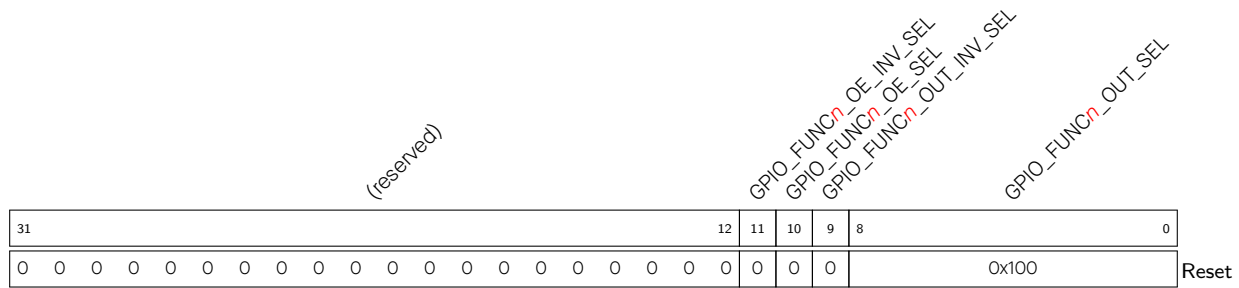
0x0: Select GPIO0
0x1: Select GPIO1
.....
0x1B: Select GPIO27
0x1C: Select GPIO28
Or
0x40: A constantly high input
0x60: A constantly low input
(R/W)

GPIO_FUNCU_IN_INV_SEL Configures whether or not to invert the input value.

0: Not invert
1: Invert
(R/W)

GPIO_SIGU_IN_SEL Configures whether or not to route signals via GPIO matrix.

0: Bypass GPIO matrix, i.e., connect signals directly to peripheral configured in IO MUX.
1: Route signals via GPIO matrix.
(R/W)

Register 5.21. GPIO_FUNC n _OUT_SEL_CFG_REG (n : 0-14, 23-28) (0x0AC4+0x4* n)

GPIO_FUNC n _OUT_SEL Configures to select a signal Y ($0 \leq Y < 256$) from peripheral signals to be output from GPIO n .

0: Select signal 0

1: Select signal 1

.....

254: Select signal 254

255: Select signal 255

Or

256: Bit[n] of [GPIO_OUT_REG](#) and [GPIO_ENABLE_REG](#) are selected as the output value and output enable.

For the detailed signal list, see Table 5.12-1.

(R/W/SC)

GPIO_FUNC n _OUT_INV_SEL Configures whether or not to invert the output value.

0: Not invert

1: Invert

(R/W/SC)

GPIO_FUNC n _OE_SEL Configures to select the source of output enable signal.

0: Use output enable signal from peripheral.

1: Force the output enable signal to be sourced from bit[n] of [GPIO_ENABLE_REG](#).

(R/W)

GPIO_FUNC n _OE_INV_SEL Configures whether or not to invert the output enable signal.

0: Not invert

1: Invert

(R/W)

Register 5.22. GPIO_CLOCK_GATE_REG (0x0DF8)

(reserved)																												GPIO_CLK_EN	
31																												1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

GPIO_CLK_EN Configures whether or not to enable clock gate.

0: Not enable

1: Enable, the clock is free running.

(R/W)

Register 5.23. GPIO_DATE_REG (0x0DFC)

(reserved)				GPIO_DATE																								
31			28	27																								0
0	0	0	0		0x2312140																							Reset

GPIO_DATE Version control register.

(R/W)

5.19.2 HP IO MUX Registers

The addresses in this section are relative to the HP IO MUX base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section VII .

Register 5.24. IO_MUX_GPIO n _REG (n : 0-14, 23-28) (0x0000+0x4* n)

(reserved)																IO_MUX_GPIO _n _HYS_SEL IO_MUX_GPIO _n _HYS_EN IO_MUX_GPIO _n _FILTER_EN IO_MUX_GPIO _n _MCU_SEL IO_MUX_GPIO _n _FUN_DRV IO_MUX_GPIO _n _FUN_IE IO_MUX_GPIO _n _FUN_WPU IO_MUX_GPIO _n _FUN_WPD IO_MUX_GPIO _n _MCU_DRV IO_MUX_GPIO _n _MCU_IE IO_MUX_GPIO _n _MCU_WPU IO_MUX_GPIO _n _MCU_WPD IO_MUX_GPIO _n _SLP_SEL IO_MUX_GPIO _n _MCU_OE															
31											18	17	16	15	14	12		11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x1	2		0	0	0	0	0	0	0	0	0	0	0	Reset			

IO_MUX_GPIO n _MCU_OE Configures whether or not to enable the output of GPIO n in sleep mode.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _SLP_SEL Configures whether or not to enter sleep mode for GPIO n .

0: Not enter

1: Enter

(R/W)

IO_MUX_GPIO n _MCU_WPD Configure whether or not to enable pull-down resistor of GPIO n in sleep mode.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _MCU_WPU Configures whether or not to enable pull-up resistor of GPIO n during sleep mode.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _MCU_IE Configures whether or not to enable the input of GPIO n during sleep mode.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _MCU_DRV Configures the drive strength of GPIO n during sleep mode.

0: ~5 mA

1: ~10 mA

2: ~20 mA

3: ~40 mA

(R/W)

IO_MUX_GPIO n _FUN_WPD Configures whether or not to enable pull-down resistor of GPIO n .

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 5.24. IO_MUX_GPIO n _REG (n : 0-14, 23-28) (0x0000+0x4* n)

Continued from the previous page...

IO_MUX_GPIO n _FUN_WPU Configures whether or not enable pull-up resistor of GPIO n .

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _FUN_IE Configures whether or not to enable input of GPIO n .

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _FUN_DRV Configures the drive strength of GPIO n .

0: ~5 mA

1: ~10 mA

2: ~20 mA

3: ~40 mA

(R/W)

IO_MUX_GPIO n _MCU_SEL Configures to select IO MUX function for this signal.

0: Select Function 0

1: Select Function 1

.....

(R/W)

IO_MUX_GPIO n _FILTER_EN Configures whether or not to enable filter for pin input signals.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _HYS_EN Configures whether or not to enable the hysteresis function of the pin when IO_MUX_GPIO n _HYS_SEL is set to 1.

0: Disable

1: Enable

(R/W)

IO_MUX_GPIO n _HYS_SEL Configures to choose the signal for enabling the hysteresis function for GPIO n .

0: Choose the output enable signal of eFuse

1: Choose the output enable signal of IO_MUX_GPIO n _HYS_EN

(R/W)

Register 5.25. IO_MUX_DATE_REG (0x01FC)

(reserved)				IO_MUX_REG_DATE																								
31	28	27																										0
0	0	0	0	0x2311270																								Reset

IO_MUX_REG_DATE Version control register (R/W)

5.19.3 GPIO EXT Registers

The addresses in this section are relative to (HP GPIO matrix base address + 0x0F00). GPIO base address is provided in Table 4.3-2 in Chapter 4 *System and Memory*.
For how to program reserved fields, please refer to Section VII .

Register 5.26. GPIO_EXT_SIGMADELTA_MISC_REG (0x0004)

(reserved)																															GPIO_EXT_SIGMADELTA_CLK_EN		
31																															1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

GPIO_EXT_SIGMADELTA_CLK_EN Configures whether or not to enable the clock for sigma delta modulation.
0: Not enable
1: Enable
(R/W)

Register 5.27. GPIO_EXT_SIGMADELTA n _REG (n : 0-3) (0x0008+0x4* n)

Diagram of the 32-bit GPIO_EXT_CR1 register structure:

- Bits 31-16: (reserved)
- Bits 15-8: GPIO_EXT_SD_n_PRESCALE
- Bits 7-0: GPIO_EXT_SD_n_IN

Register Value: 0x00000000

GPIO_EXT_SD*n*_IN Configures the duty cycle of sigma delta modulation output.
(R/W)

GPIO_EXT_SD*n*_PRESCALE Configures the divider value to divide IO MUX operating clock.

0: Do not divide

1~255: The clock is divided by 2~256

(R/W)

Register 5.28. GPIO_EXT_PAD_COMP_CONFIG_0_REG (0x0058)

(reserved)																GPIO_EXT_DREF_COMP_0		GPIO_EXT_MODE_COMP_0		GPIO_EXT_XPD_COMP_0	
31															5	4	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x0		0	0	Reset	

GPIO_EXT_XPD_COMP_0 Configures whether to enable the function of analog PAD voltage comparator.

0: Disable

1: Enable

(R/W)

GPIO_EXT_MODE_COMP_0 Configures the reference voltage for analog PAD voltage comparator.

0: Reference voltage is the internal reference voltage, meanwhile GPIO8 PAD can be used as a regular GPIO

1: Reference voltage is the voltage on the GPIO8 PAD

(R/W)

GPIO_EXT_DREF_COMP_0 Configures the internal reference voltage for analog PAD voltage comparator.

0: Internal reference voltage is 0 * VDDPST1

1: Internal reference voltage is 0.1 * VDDPST1

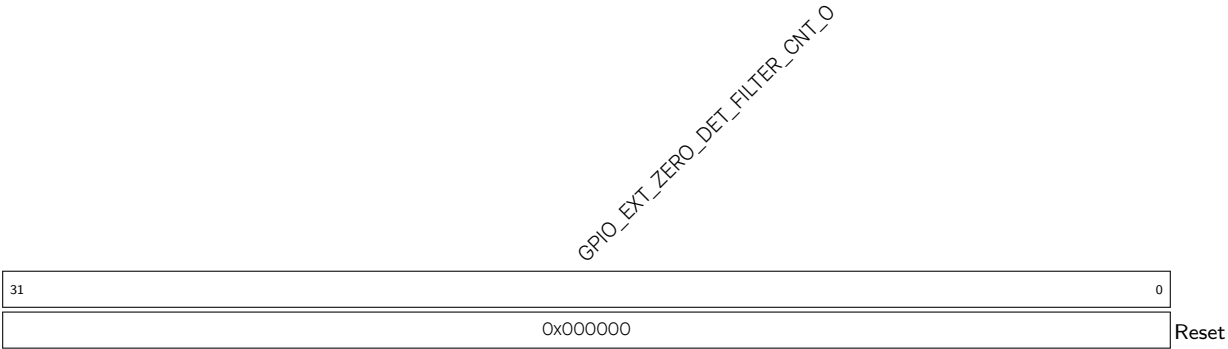
.....

6: Internal reference voltage is 0.6 * VDDPST1

7: Internal reference voltage is 0.7 * VDDPST1

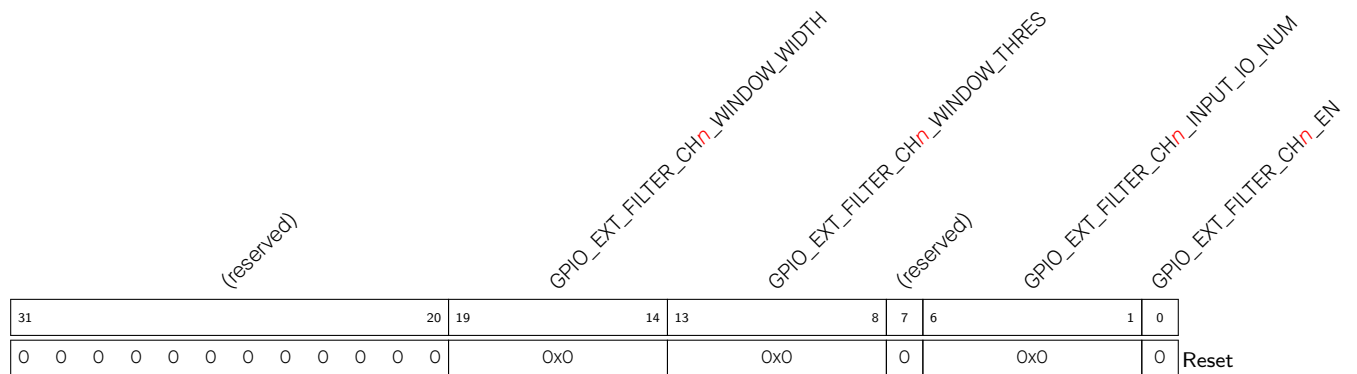
(R/W)

Register 5.29. GPIO_EXT_PAD_COMP_FILTER_O_REG (0x005C)



GPIO_EXT_ZERO_DET_FILTER_CNT_O Configures the period of masking new interrupt source for analog PAD voltage comparator.
Measurement unit: IO MUX operating clock cycle
(R/W)

Register 5.30. GPIO_EXT_GLITCH_FILTER_CH*n*_REG (*n*: 0-7) (0x00D8+0x4**n*)



GPIO_EXT_FILTER_CH n _EN Configures whether or not to enable channel n of Glitch Filter.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_FILTER_CH*n***_INPUT_IO_NUM** Configures to select the input GPIO for Glitch Filter.

0: Select GPIO0

1: Select GPIO1

• • • • •

27: Select GPIO27

28: Select GPIO28

29~63: Reserved

(R/W)

GPIO_EXT_FILTER_CH n _WINDOW_THRES Configures the window threshold for Glitch Filter. The window threshold should be less than or equal to GPIO_EXT_FILTER_CH n _WINDOW_WIDTH.

Measurement unit: IO MUX operating clock cycle

(R/W)

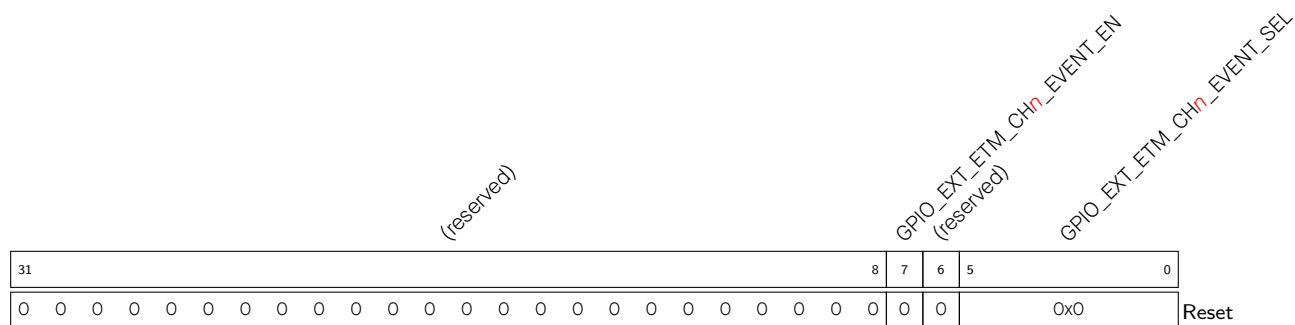
GPIO_EXT_FILTER_CHn_WINDOW_WIDTH Configures the window width for Glitch Filter.

The effective value of window width is 0~63.

Measurement unit: IO MUX operating clock cycle

(R/W)

Register 5.31. GPIO_EXT_ETM_EVENT_CH n _CFG_REG (n : 0-7) (0x0118+0x4* n)



GPIO_EXT_ETM_CH n _EVENT_SEL Configures to select GPIO for ETM event channel.

0: Select GPIO0

1: Select GPIO1

• • • • •

27: Select GPIO27

28: Select GPIO28

29~63: Reserved

(R/W)

GPIO_EXT_ETM_CHn_EVENT_EN Configures whether or not to enable ETM event send.

0: Not enable

1: Enable

(R/W)

Register 5.32. GPIO_EXT_ETM_TASK_PO_CFG_REG (0x0158)

(reserved)		GPIO_EXT_ETM_TASK_GPIO4_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO4_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO3_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO3_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO2_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO2_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO1_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO1_SEL		(reserved)		GPIO_EXT_ETM_TA	
31	30	29	28	27	26	24	23	22	21	20	18	17	16	15	14	12	11	10	9	8	6	5	4	3	2	0									
0	0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	Reset									

GPIO_EXT_ETM_TASK_GPIO0_SEL Configures to select an ETM task channel for GPIO0.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO0_EN Configures whether or not to enable GPIO0 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO1_SEL Configures to select an ETM task channel for GPIO1.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO1_EN Configures whether or not to enable GPIO1 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO2_SEL Configures to select an ETM task channel for GPIO2.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO2_EN Configures whether or not to enable GPIO2 to response ETM task.

0: Not enable

1: Enable

(R/W)

Continued on the next page...

Register 5.32. GPIO_EXT_ETM_TASK_PO_CFG_REG (0x0158)

Continued from the previous page...

GPIO_EXT_ETM_TASK_GPIO3_SEL Configures to select an ETM task channel for GPIO3.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO3_EN Configures whether or not to enable GPIO3 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO4_SEL Configures to select an ETM task channel for GPIO4.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO4_EN Configures whether or not to enable GPIO4 to response ETM task.

0: Not enable

1: Enable

(R/W)

Register 5.33. GPIO_EXT_ETM_TASK_P1_CFG_REG (0x015C)

(reserved)		GPIO_EXT_ETM_TASK_GPIO9_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO9_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO8_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO8_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO7_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO7_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO6_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO6_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO5_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO5_SEL	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset							
0	0	0	0	0	0x0	0	0	0	0	0	0x0	0	0	0	0	0	0x0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0x0								

GPIO_EXT_ETM_TASK_GPIO5_SEL Configures to select an ETM task channel for GPIO5.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO5_EN Configures whether or not to enable GPIO5 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO6_SEL Configures to select an ETM task channel for GPIO6.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO6_EN Configures whether or not to enable GPIO6 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO7_SEL Configures to select an ETM task channel for GPIO7.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO7_EN Configures whether or not to enable GPIO7 to response ETM task.

0: Not enable

1: Enable

(R/W)

Continued on the next page...

Register 5.33. GPIO_EXT_ETM_TASK_P1_CFG_REG (0x015C)

Continued from the previous page...

GPIO_EXT_ETM_TASK_GPIO8_SEL Configures to select an ETM task channel for GPIO8.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO8_EN Configures whether or not to enable GPIO8 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO9_SEL Configures to select an ETM task channel for GPIO9.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO9_EN Configures whether or not to enable GPIO9 to response ETM task.

0: Not enable

1: Enable

(R/W)

Register 5.34. GPIO_EXT_ETM_TASK_P2_CFG_REG (0x0160)

(reserved)		GPIO_EXT_ETM_TASK_GPIO14_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO14_SEL		GPIO_EXT_ETM_TASK_GPIO13_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO13_SEL		GPIO_EXT_ETM_TASK_GPIO12_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO12_SEL		GPIO_EXT_ETM_TASK_GPIO11_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO11_SEL		GPIO_EXT_ETM_TASK_GPIO10_EN		(reserved)		GPIO_EXT_ETM_T	
31	30	29	28	27	26	24	23	22	21	20	18	17	16	15	14	12	11	10	9	8	6	5	4	3	2	0					
0	0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	Reset					

GPIO_EXT_ETM_TASK_GPIO10_SEL Configures to select an ETM task channel for GPIO10.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO10_EN Configures whether or not to enable GPIO10 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO11_SEL Configures to select an ETM task channel for GPIO11.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO11_EN Configures whether or not to enable GPIO11 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO12_SEL Configures to select an ETM task channel for GPIO12.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO12_EN Configures whether or not to enable GPIO12 to response ETM task.

0: Not enable

1: Enable

(R/W)

Continued on the next page...

Register 5.34. GPIO_EXT_ETM_TASK_P2_CFG_REG (0x0160)

Continued from the previous page...

GPIO_EXT_ETM_TASK_GPIO13_SEL Configures to select an ETM task channel for GPIO13.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO13_EN Configures whether or not to enable GPIO13 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO14_SEL Configures to select an ETM task channel for GPIO14.

0: Select channel 0

1: Select channel 1

.....

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO14_EN Configures whether or not to enable GPIO14 to response ETM task.

0: Not enable

1: Enable

(R/W)

Register 5.35. GPIO_EXT_ETM_TASK_P4_CFG_REG (0x0168)

(reserved)		GPIO_EXT_ETM_TASK_GPIO24_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO24_SEL		(reserved)		GPIO_EXT_ETM_TASK_GPIO23_EN		(reserved)		GPIO_EXT_ETM_TASK_GPIO23_SEL		(reserved)		(reserved)		(reserved)		(reserved)		(reserved)		(reserved)		(reserved)		(reserved)	
31	30	29	28	27	26	24	23	22	21	20	18	17	16	15	14	12	11	10	9	8	6	5	4	3	2	0					
0	0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	0	0	0	0x0	Reset					

GPIO_EXT_ETM_TASK_GPIO23_SEL Configures to select an ETM task channel for GPIO23.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO23_EN Configures whether or not to enable GPIO23 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO24_SEL Configures to select an ETM task channel for GPIO24.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO24_EN Configures whether or not to enable GPIO24 to response ETM task.

0: Not enable

1: Enable

(R/W)

Register 5.36. GPIO_EXT_ETM_TASK_P5_CFG_REG (0x016C)

[illegible]

GPIO_EXT_ETM_TASK_GPIO25_SEL Configures to select an ETM task channel for GPIO25.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO25_EN Configures whether or not to enable GPIO25 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO26_SEL Configures to select an ETM task channel for GPIO26.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO26_EN Configures whether or not to enable GPIO26 to response ETM task.

0: Not enable

1: Enable

(R/W)

GPIO_EXT_ETM_TASK_GPIO27_SEL Configures to select an ETM task channel for GPIO27.

0: Select channel 0

1: Select channel 1

• • • • •

7: Select channel 7

GPIO_EXT_ETM_TASK_GPIO27_EN Configures whether or not to enable GPIO27 to response ETM task.

0: Not enable

1: Enable

(R/W)

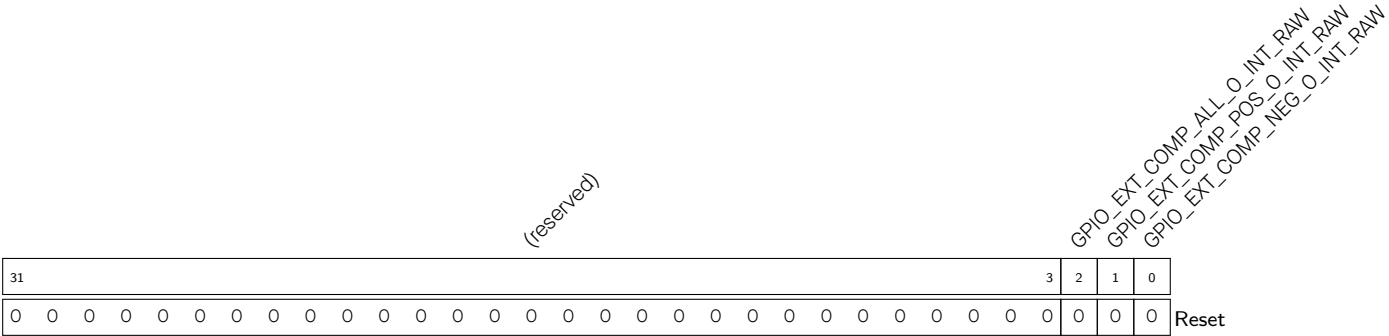
Continued on the next page...

Register 5.36. GPIO_EXT_ETM_TASK_P5_CFG_REG (0x016C)

Continued from the previous page...

- GPIO_EXT_ETM_TASK_GPIO28_SEL** Configures to select an ETM task channel for GPIO28.
0: Select channel 0
1: Select channel 1
.....
7: Select channel 7
- GPIO_EXT_ETM_TASK_GPIO28_EN** Configures whether or not to enable GPIO28 to response ETM task.
0: Not enable
1: Enable
(R/W)

Register 5.37. GPIO_EXT_INT_RAW_REG (0x01D0)



- GPIO_EXT_COMP_NEG_O_INT_RAW** The raw interrupt status of GPIO_EXT_COMP_NEG_O_INT.
(RO/WTC/SS)
- GPIO_EXT_COMP_POS_O_INT_RAW** The raw interrupt status of GPIO_EXT_COMP_POS_O_INT.
(RO/WTC/SS)
- GPIO_EXT_COMP_ALL_O_INT_RAW** The raw interrupt status of GPIO_EXT_COMP_ALL_O_INT.
(RO/WTC/SS)

Register 5.38. GPIO_EXT_INT_ST_REG (0x01D4)

(reserved)																																GPIO_EXT_COMP_ALL_O_INT_ST GPIO_EXT_COMP_POS_O_INT_ST GPIO_EXT_COMP_NEG_O_INT_ST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPIO_EXT_COMP_NEG_O_INT_ST The interrupt status of GPIO_EXT_COMP_NEG_O_INT.
(RO)

GPIO_EXT_COMP_POS_O_INT_ST The interrupt status of GPIO_EXT_COMP_POS_O_INT.
(RO)

GPIO_EXT_COMP_ALL_O_INT_ST The interrupt status of GPIO_EXT_COMP_ALL_O_INT.
(RO)

Register 5.39. GPIO_EXT_INT_ENA_REG (0x01D8)

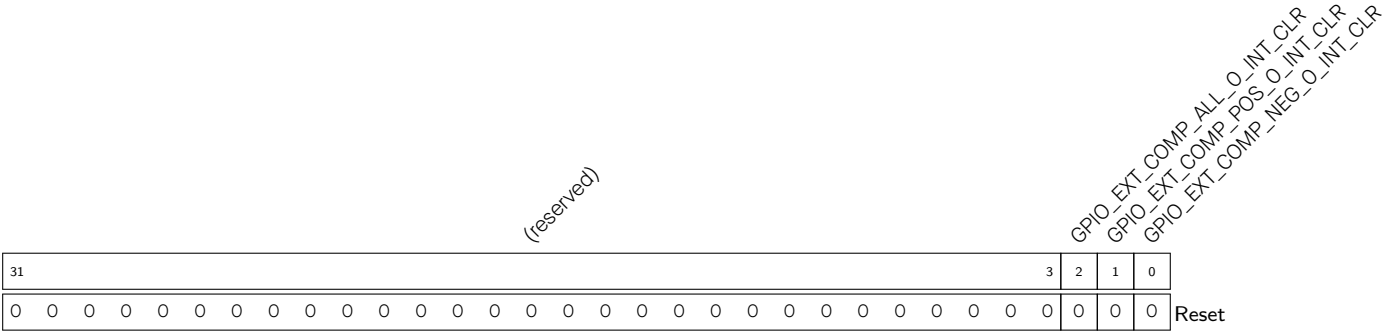
(reserved)																												GPIO_EXT_COMP_ALL_O GPIO_EXT_COMP_POS_O GPIO_EXT_COMP_NEG_O			
31																												3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	Reset

GPIO_EXT_COMP_NEG_O_INT_ENA Write 1 to enable GPIO_EXT_COMP_NEG_O_INT.
(R/W)

GPIO_EXT_COMP_POS_O_INT_ENA Write 1 to enable GPIO_EXT_COMP_POS_O_INT.
(R/W)

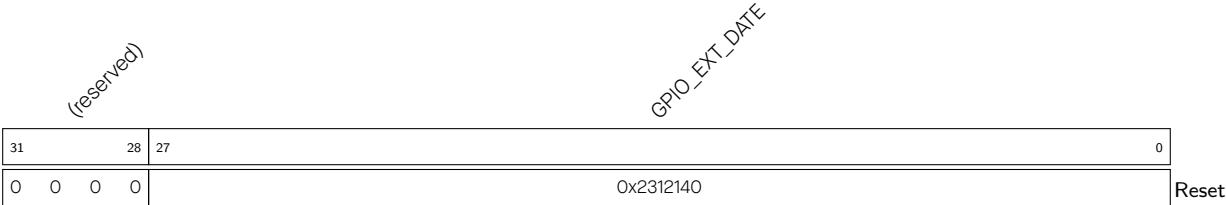
GPIO_EXT_COMP_ALL_O_INT_ENA Write 1 to enable GPIO_EXT_COMP_ALL_O_INT.
(R/W)

Register 5.40. GPIO_EXT_INT_CLR_REG (0x01DC)



- GPIO_EXT_COMP_NEG_O_INT_CLR Write 1 to clear GPIO_EXT_COMP_NEG_O_INT.
(R/W)
- GPIO_EXT_COMP_POS_O_INT_CLR Write 1 to clear GPIO_EXT_COMP_POS_O_INT.
(R/W)
- GPIO_EXT_COMP_ALL_O_INT_CLR Write 1 to clear GPIO_EXT_COMP_ALL_O_INT.
(R/W)

Register 5.41. GPIO_EXT_VERSION_REG (0x01FC)



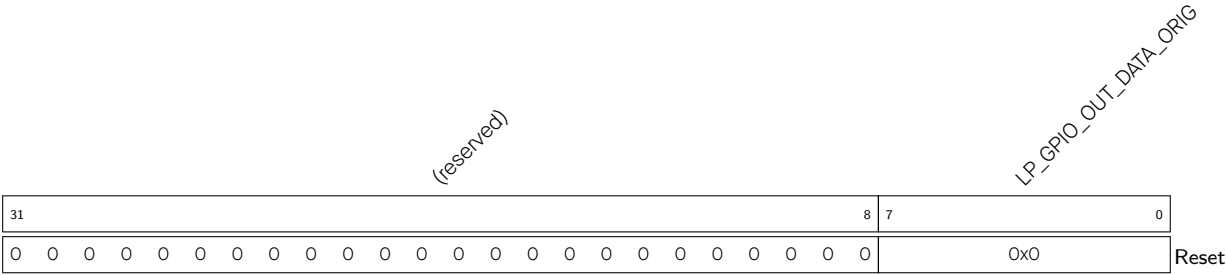
GPIO_EXT_DATE Version control register. (R/W)

5.19.4 LP GPIO Matrix Registers

The addresses in this section are relative to LP GPIO matrix base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

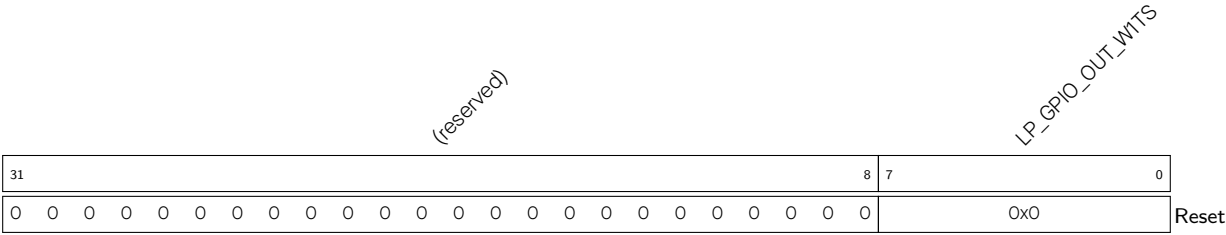
For how to program reserved fields, please refer to Section VII .

Register 5.42. LP_GPIO_OUT_REG (0x0004)



LP_GPIO_OUT_DATA_ORIG Configures the output of GPIO0~GPIO6.
The value of each bit can be:
0: Low level
1: High level
Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.
(R/W/WTC)

Register 5.43. LP_GPIO_OUT_W1TS_REG (0x0008)



LP_GPIO_OUT_W1TS Configures whether or not to enable the output register LP_IO_OUT_REG of GPIO0~GPIO6.
The value of each bit can be:
0: Not set
1: The corresponding bit in LP_GPIO_OUT_REG will be set to 1
Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.
Recommended operation: use this register to set LP_GPIO_OUT_REG.
(WT)

Register 5.44. LP_GPIO_OUT_W1TC_REG (0x000C)

(reserved)																				LP_GPIO_OUT_W1TC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
31																				8	7	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

LP_GPIO_OUT_W1TC Configures whether or not to clear the output register [LP_GPIO_OUT_REG](#) of GPIO0~GPIO6.

The value of each bit can be:

0: Not clear

1: The corresponding bit in [LP_GPIO_OUT_REG](#) will be cleared.

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

Recommended operation: use this register to clear [LP_GPIO_OUT_REG](#).

(WT)

Register 5.45. LP_GPIO_ENABLE_REG (0x0010)

(reserved)																								LP_GPIO_ENABLE_DATA									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0x0								Reset	

LP_GPIO_ENABLE_DATA Configures whether or not to enable the output of GPIO0~GPIO6.

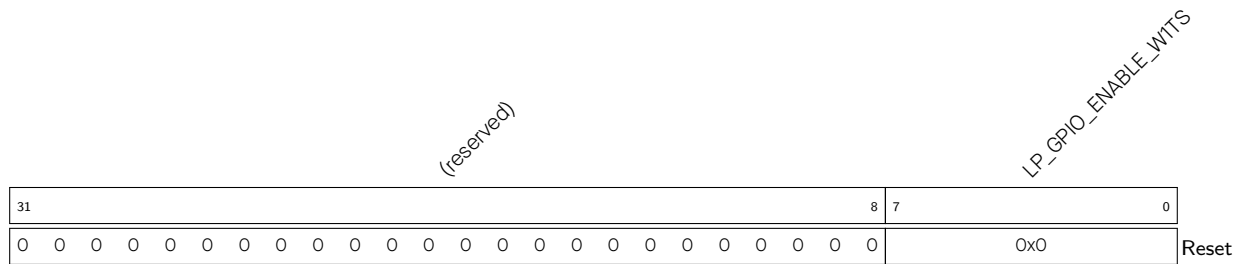
The value of each bit can be:

0: Not enable

1: Enable

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

(R/W/WTC)

Register 5.46. LP_GPIO_ENABLE_W1TS_REG (0x0014)

LP_GPIO_ENABLE_W1TS Configures whether or not to set the output enable register [LP_GPIO_ENABLE_REG](#) of GPIO0~GPIO6.

The value of each bit can be:

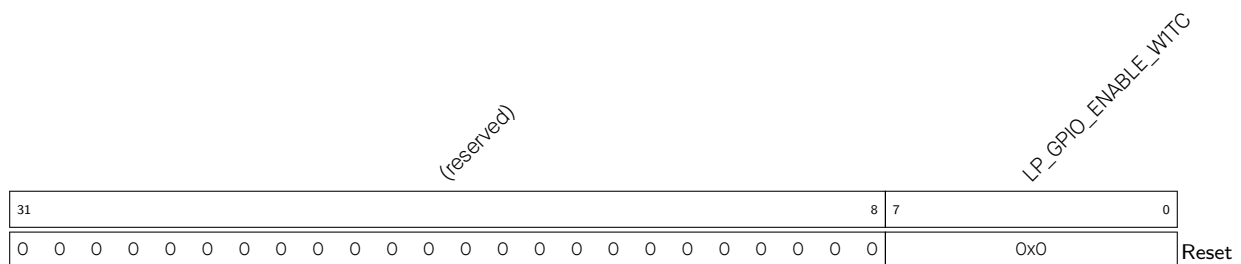
0: Not set

1: The corresponding bit in [LP_GPIO_ENABLE_REG](#) will be set to 1

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

Recommended operation: use this register to set [LP_GPIO_ENABLE_REG](#).

(WT)

Register 5.47. LP_GPIO_ENABLE_W1TC_REG (0x0018)

LP_GPIO_ENABLE_W1TC Configures whether or not to clear the output enable register [LP_GPIO_ENABLE_REG](#) of GPIO0~GPIO6.

The value of each bit can be:

0: Not clear

1: The corresponding bit in [LP_GPIO_ENABLE_REG](#) will be cleared

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

Recommended operation: use this register to clear [LP_GPIO_ENABLE_REG](#).

(WT)

Register 5.48. LP_GPIO_IN_REG (0x001C)

(reserved)																LP_GPIO_IN_DATA_NEXT																	
31																8	7	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x0																Reset	

LP_GPIO_IN_DATA_NEXT Represents the input value of GPIO0~GPIO6.

Each bit represents a pin input value:

0: Low level input

1: High level input

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

(RO)

Register 5.49. LP_GPIO_STATUS_REG (0x0020)

(reserved)																								LP_GPIO_STATUS_INTERRUPT									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0x0								Reset	

LP_GPIO_STATUS_INTERRUPT Configures the interrupt status of GPIO0~GPIO6.

The value of each bit can be:

0: No interrupt

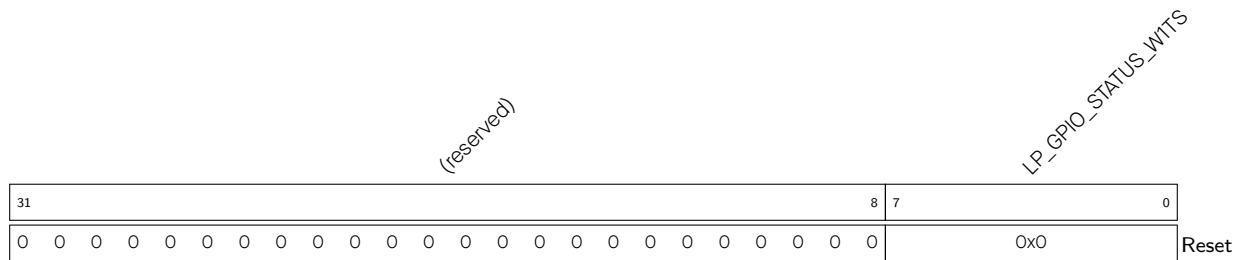
1: Interrupt is triggered

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

This field is used together with LP_GPIO_PIN n _INT_TYPE in register LP_GPIO_PIN n _REG.

(R/W/WTC)

Register 5.50. LP_GPIO_STATUS_W1TS_REG (0x0024)



LP_GPIO_STATUS_W1TS Configures whether or not to set the interrupt status register [LP_GPIO_STATUS_INT](#) of GPIO0~GPIO6.

The value of each bit can be:

0: Not set

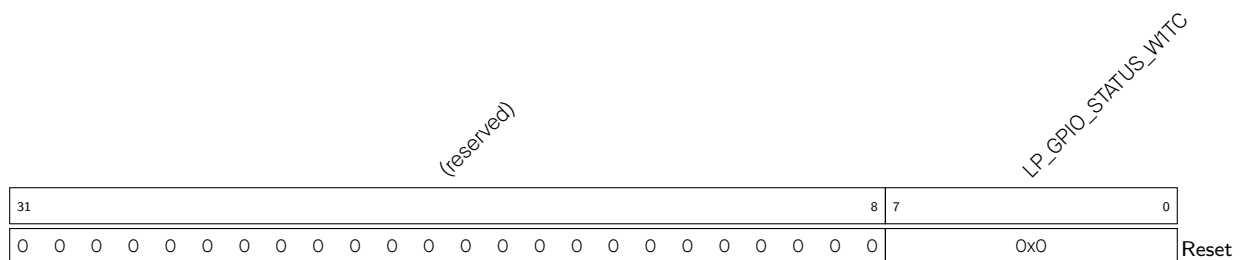
1: The corresponding bit in [LP_GPIO_STATUS_INT](#) will be set

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

Recommended operation: use this register to set [LP_GPIO_STATUS_INT](#).

(WT)

Register 5.51. LP_GPIO_STATUS_W1TC_REG (0x0028)



LP_GPIO_STATUS_W1TC Configures whether or not to clear the interrupt status register [LP_GPIO_STATUS_INT](#) of GPIO0~GPIO6.

The value of each bit can be:

0: Not clear

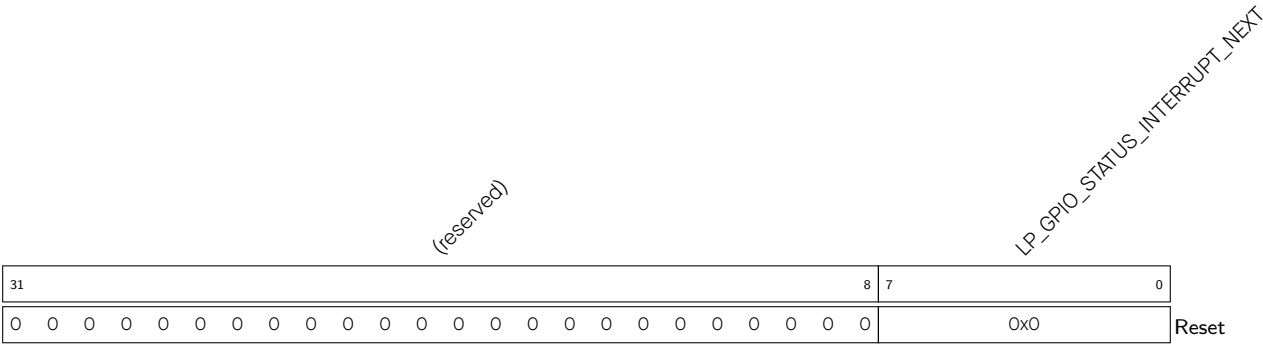
1: The corresponding bit in [LP_GPIO_STATUS_INT](#) will be cleared

Bit[0]~bit[6] are corresponding to GPIO0~GPIO6.

Recommended operation: use this register to clear [LP_GPIO_STATUS_INT](#).

(WT)

Register 5.52. LP_GPIO_STATUS_NEXT_REG (0x002C)



LP_GPIO_STATUS_INTERRUPT_NEXT Represents the interrupt source status of GPIO0~GPIO6.
Bit[0]~bit[6] are corresponding to GPIO0~7.
Each bit represents:
0: Interrupt source status is invalid.
1: Interrupt source status is valid.
The interrupt here can be rising-edge triggered, falling-edge triggered, any edge triggered, or level triggered.
(RO)

Register 5.53. LP_GPIO_PIN n _REG (n : 0-6) (0x0030+0x4* n)

(reserved)																						LP_GPIO_PIN _n _WAKEUP_ENABLE				LP_GPIO_PIN _n _INT_TYPE				(reserved)				LP_GPIO_PIN _n _EDGE_WAKEUP_CLR				LP_GPIO_PIN _n _SYNC1_BYPASS				LP_GPIO_PIN _n _PAD_DRIVER				LP_GPIO_PIN _n _SYNC2_BYPASS			
31																						11		10	9		7		6	5	4	3	2	1	0														
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																						0		0x0		0		0	0x0		0		0x0		Reset														

Reset

LP_GPIO_PIN n _SYNC2_BYPASS Configures whether or not to synchronize GPIO input data on either edge of LP IO MUX operating clock for the second-level synchronization.

0: Not synchronize

1: Synchronize on falling edge

2: Synchronize on rising edge

3: Synchronize on rising edge

(R/W)

LP_GPIO_PIN n _PAD_DRIVER Configures to select the pin dirve mode of GPIO n .

0: Normal output

1: Open drain output

(R/W)

LP_GPIO_PIN n _SYNC1_BYPASS Configures whether or not to synchronize GPIO input data on either edge of LP IO MUX operating clock for the first-level synchronization.

0: Not synchronize

1: Synchronize on falling edge

2: Synchronize on rising edge

3: Synchronize on rising edge

(R/W)

LP_GPIO_PIN n _EDGE_WAKEUP_CLR Configures whether or not to clear the edge wake-up status of GPIO0~GPIO6.

0: No effect

1: Clear

(WT)

LP_GPIO_PIN n _INT_TYPE Configures GPIO n interrupt type.

0: GPIO interrupt disabled

1: Rising edge trigger

2: Falling edge trigger

3: Any edge trigger

4: Low level trigger

5: High level trigger

(R/W)

Continued on the next page...

Register 5.53. LP_GPIO_PIN n _REG (n : 0-6) (0x0030+0x4* n)

Continued from the previous page...

LP_GPIO_PIN n _WAKEUP_ENABLE Configures whether or not to enable GPIO n wake-up function.

0: Not enable

1: Enable

This function is disabled when PD_LP_PERI is powered off.

(R/W)

Register 5.54. LP_GPIO_FUNC n _OUT_SEL_CFG_REG (n : 0-6) (0x02B0+0x4* n)

(reserved)																												LP_GPIO_FUNC _n _OE_INV_SEL (reserved) LP_GPIO_FUNC _n _OUT_INV_SEL																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31																												3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_GPIO_FUNC n _OUT_INV_SEL Configures whether or not to invert the output value.

0: Not invert

1: Invert

(R/W)

LP_GPIO_FUNC n _OE_INV_SEL Configures whether or not to invert the output enable signal.

0: Not invert

1: Invert

(R/W)

Register 5.55. LP_GPIO_CLOCK_GATE_REG (0x03F8)

(reserved)																															LP_GPIO_CLK_EN		
31																															1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

LP_GPIO_CLK_EN Configure to enable the GPIO clock gate or not.

0: Not enable

1: Enable

(R/W)

Register 5.56. LP_GPIO_DATE_REG (0x03FC)

(reserved)				LP_GPIO_DATE																									
31	28	27	0																										
0	0	0	0	0x2312010																									Reset

LP_GPIO_DATE Version control register.
(R/W)

5.19.5 LP IO MUX Registers

The addresses in this section are relative to LP IO MUX base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.
For how to program reserved fields, please refer to Section VII .

Register 5.57. LP_IO_MUX_GPIO n _REG (n : 0-6) (0x0000+0x4* n)

(reserved)																LP_IO_MUX_GPIO _n _HYS_SEL LP_IO_MUX_GPIO _n _HYS_EN LP_IO_MUX_GPIO _n _FILTER_EN LP_IO_MUX_GPIO _n _MCU_SEL LP_IO_MUX_GPIO _n _FUN_DRV LP_IO_MUX_GPIO _n _FUN_IE LP_IO_MUX_GPIO _n _FUN_WPU LP_IO_MUX_GPIO _n _FUN_WPD LP_IO_MUX_GPIO _n _MCU_DRV LP_IO_MUX_GPIO _n _MCU_IE LP_IO_MUX_GPIO _n _MCU_WPU LP_IO_MUX_GPIO _n _MCU_WPD LP_IO_MUX_GPIO _n _SLP_SEL LP_IO_MUX_GPIO _n _MCU_OE															
31													18	17	16	15	14	12		11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x1	2		0	0	0	0	0	0	0	0	0	0	Reset				

LP_IO_MUX_GPIO n _MCU_OE Configures whether or not to enable the output of GPIO n during sleep mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _SLP_SEL Configures whether or not to enable the sleep mode for GPIO n .

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _MCU_WPD Configures whether or not to enable the pull-down resistor of GPIO n during sleep mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _MCU_WPU Configures whether or not to enable the pull-up resistor of GPIO n during sleep mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _MCU_IE Configures whether or not to enable the input of GPIO n during sleep mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _MCU_DRV Configures the drive strength of GPIO n during sleep mode.

0: ~5 mA

1: ~10 mA

2: ~20 mA

3: ~40 mA

(R/W)

Continued on the next page...

Register 5.57. LP_IO_MUX_GPIO n _REG (n : 0-7) (0x0000+0x4* n)

Continued from the previous page...

LP_IO_MUX_GPIO n _FUN_WPD Configures whether or not to enable the pull-down resistor of GPIO n in normal execution mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _FUN_WPU Configures whether or not to enable the pull-up resistor of GPIO n in normal execution mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _FUN_IE Configures whether or not to enable the input of GPIO n in normal execution mode.

0: Not enable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _FUN_DRV Configures the drive strength of GPIO n in normal execution mode.

0: ~5 mA

1: ~10 mA

2: ~20 mA

3: ~40 mA

(R/W)

LP_IO_MUX_GPIO n _MCU_SEL Configures to select the LP IO MUX function for GPIO n in normal execution mode.

0: Select Function 0

1: Select Function 1

.....

(R/W)

LP_IO_MUX_GPIO n _FILTER_EN Configures whether or not to enable filter for pin input signals.

0: Disable

1: Enable

(R/W)

LP_IO_MUX_GPIO n _HYS_EN Configures whether or not to enable the hysteresis function of the pin when LP_IO_MUX_GPIO n _HYS_SEL is set to 1.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 5.57. LP_IO_MUX_GPIO n _REG (n : 0-7) (0x0000+0x4* n)

Continued from the previous page...

LP_IO_MUX_GPIO n _HYS_SEL Configures to choose the signal for enabling the hysteresis function for GPIO n .
0: Choose the output enable signal of eFuse
1: Choose the output enable signal of LP_IO_MUX_GPIO n _HYS_EN
(R/W)

Register 5.58. LP_IO_MUX_DATE_REG (0x01FC)

(reserved)				LP_IO_MUX_REG_DATE																									
31	28	27																											0
0	0	0	0	0x2211270																									Reset

LP_IO_MUX_REG_DATE Version control register.
(R/W)

Chapter 6

Reset and Clock

6.1 Reset

6.1.1 Overview

ESP32-C5 provides four types of reset that occur at different levels, namely CPU Reset, Core Reset, System Reset, and Chip Reset. All reset types mentioned above (except Chip Reset) preserve the data stored in internal memory. Figure 6.1-1 shows the scopes of affected subsystems by each type of reset.

6.1.2 Architectural Overview

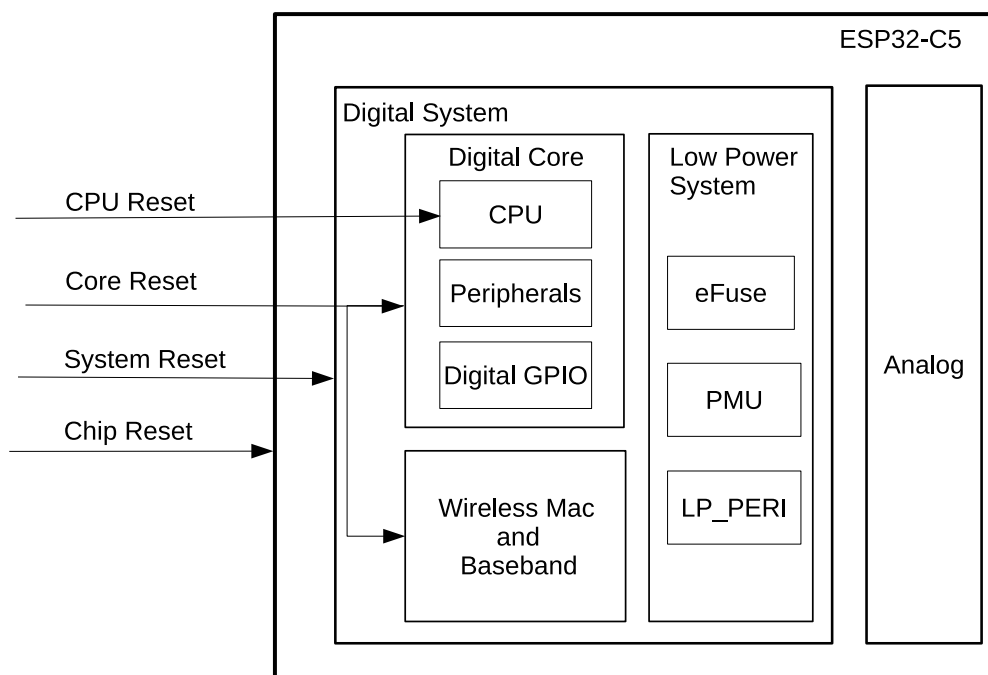


Figure 6.1-1. Reset Types

ESP32-C5's Digital System can be divided into two parts: [High Performance System \(HP system\)](#) that includes Digital Core, Wireless Mac and Baseband, and HP SRAM, and [Low Power System \(LP system\)](#) that only includes some low-power peripherals and LP SRAM. See Figure 6.1-1 for details (note that HP SRAM and LP SRAM would not be reset, so they are not shown in the figure).

6.1.3 Features

- Four reset types:

- CPU Reset: resets CPU core. Once such a reset is released, the instructions from the CPU reset vector (0x40000000) will be executed.
- Core Reset: resets the whole digital system except LP system, including CPU, peripherals, digital GPIOs, Wireless MAC and Baseband.
- System Reset: resets the whole digital system, including LP system.
- Chip Reset: resets the whole chip, including the digital system and analog system.
- Software reset and hardware reset:
 - Software Reset: triggered via software by configuring the corresponding registers of CPU, see Chapter 4 [Low-power Management \[to be added later\]](#).
 - Hardware Reset: triggered directly by the hardware.

6.1.4 Functional Description

CPU will be reset immediately when any type of reset above occurs. Users can retrieve reset source codes by reading `LP_CLKRST_RESET_CAUSE` after the reset is released. The current reset source recorded in the `LP_CLKRST_RESET_CAUSE` register can be cleared by configuring `LP_CLKRST_COREO_RESET_CAUSE_CLR`.

Table 6.1-1 lists possible reset sources and the types of reset they trigger. When multiple reset sources are active simultaneously, the reset source recorded in `LP_CLKRST_RESET_CAUSE` is the most top one among the active reset sources listed in Table 6.1-1. When multiple reset sources are active but not at the same time, `LP_CLKRST_RESET_CAUSE` records the most recent reset source.

Table 6.1-1. Reset Source

Code	Source	Reset Type	Note
0x1A	CPU lockup reset	CPU Reset	—
0x01	Chip reset ¹	Chip Reset	—
0x12	Super watchdog reset	System Re-set	See Chapter 11 Watchdog Timers (WDT)
0x19	Power glitch reset	System Re-set	See Chapter 15 Power Supply Detector
0x0F	Brown-out system re-set	Chip Reset or System Reset	Triggered by brown-out detector ²
0x10	RWDT system reset	System Re-set	See Chapter 11 Watchdog Timers (WDT)
0x09	RWDT core reset	Core Reset	See Chapter 11 Watchdog Timers (WDT)
0x14	eFuse reset	Core Reset	Triggered by eFuse CRC error check
0x07	MWDT0 core reset	Core Reset	See Chapter 11 Watchdog Timers (WDT)
0x08	MWDT1 core reset	Core Reset	See Chapter 11 Watchdog Timers (WDT)
0x15	USB (UART) reset	Core Reset	Triggered when external USB host sends a specific command to the serial interface of USB Serial/JTAG Controller. See Chapter 28 USB Serial/JTAG Controller

Cont'd on next page

Table 6.1-1 – cont'd from previous page

Code	Source	Reset Type	Note
0x16	USB (JTAG) reset	Core Reset	Triggered when external USB host sends a specific command to the JTAG interface of USB Serial/JTAG Controller. See Chapter 28 USB Serial/JTAG Controller
0x03	Software system reset	Core Reset	Triggered by configuring LP_AON_HPSYS_SW_RESET
0x0D	RWDT CPU reset	CPU Reset	See Chapter 11 Watchdog Timers (WDT)
0x0C	Software CPU reset	CPU Reset	Triggered by configuring LP_AON_CPU_CORE0_SW_RESET
0x0B	MWDT0 CPU reset	CPU Reset	See Chapter 11 Watchdog Timers (WDT)
0x11	MWDT1 CPU reset	CPU Reset	See Chapter 11 Watchdog Timers (WDT)
0x18	JTAG CPU reset	CPU Reset	Triggered when a "JDB Resetting CPU" instruction is received
0x05	Deep-sleep reset	Core Reset	See Chapter 4 Low-power Management [to be added later]

¹ Chip Reset can be triggered by the following sources:

- Chip power-up.
- Brown-out detector.

² Once brown-out status is detected, the detector will trigger System Reset or Chip Reset, depending on the register configuration. See Chapter [4 Low-power Management \[to be added later\]](#).

6.1.5 Peripheral Reset

Peripherals can be reset individually by configuring corresponding registers, or globally by core reset, system reset, or chip reset. After releasing peripherals from reset, the reset release status registers can be read to determine the reset status. These registers turning to 1 indicates that peripherals have been released from reset and can operate properly.

The reset registers of ESP32-C5's HP system peripherals are controlled by the Power/Clock/Reset (PCR) module. Please see [6.4.1 PCR Register Summary](#) for the reset registers of HP system peripherals and [6.4.2 LP System Clock Register Summary](#) for the clock registers of low-power system peripherals.

6.2 Clock

6.2.1 Overview

ESP32-C5 clocks are mainly sourced from oscillator (OSC), RC, and PLL circuits, and then processed by the dividers or selectors. This allows functional modules to select their working clock according to their power consumption and performance requirements. Figure [6.2-1](#) shows the system clock structure.

6.2.2 Architectural Overview

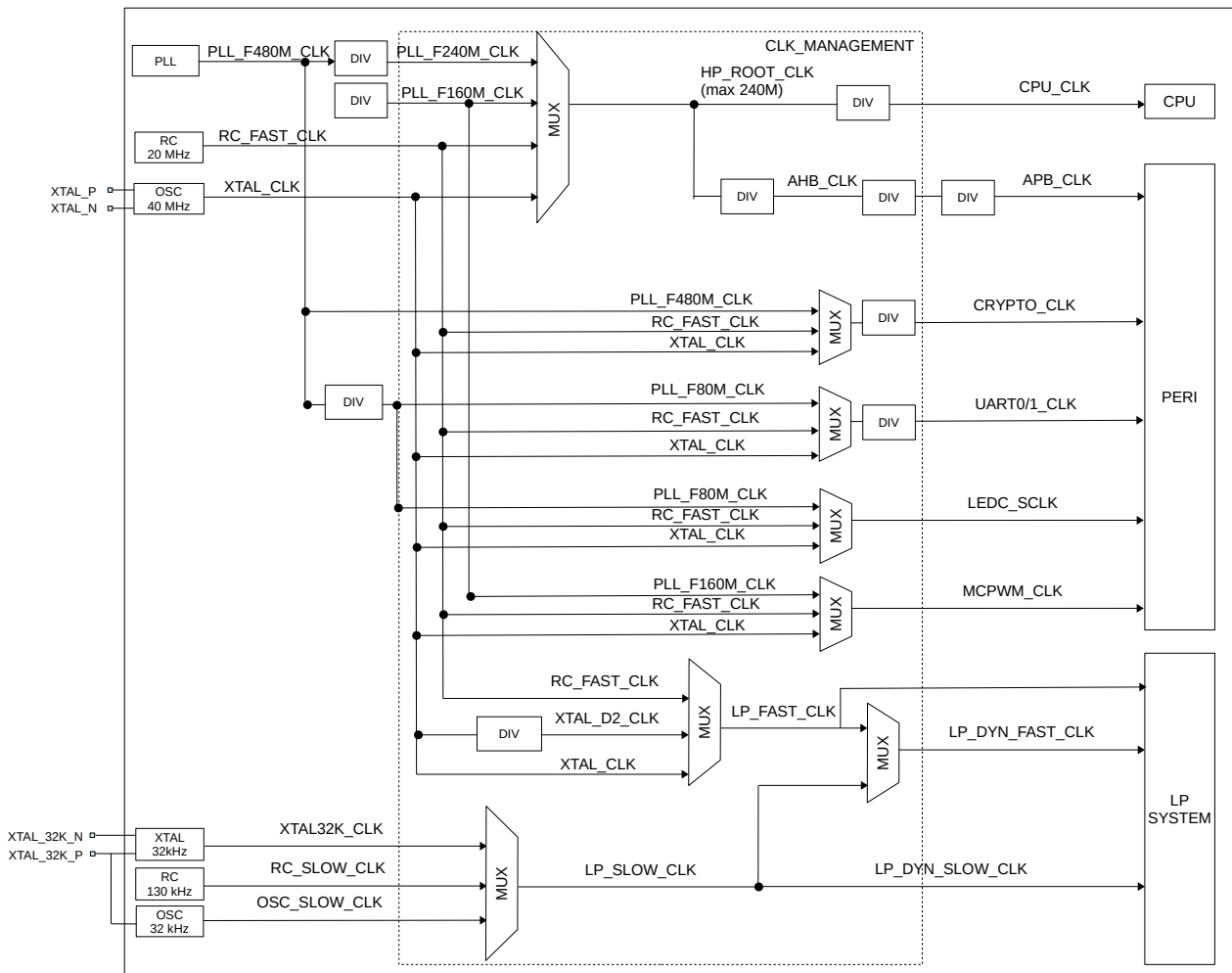


Figure 6.2-1. System Clock

6.2.3 Features

ESP32-C5 clocks can be classified into two types depending on their frequencies:

- High-performance (HP) clocks for devices working at a higher frequency, such as CPU and digital peripherals
 - PLL_F480M_CLK (480 MHz): internal PLL clock. Its reference clock is XTAL_CLK
 - XTAL_CLK (48 MHz/40 MHz): external crystal clock
- Low-power (LP) clocks for low-power system and some peripherals working in low-power mode
 - XTAL32K_CLK (32 kHz): external crystal clock
 - RC_FAST_CLK (20 MHz by default): internal fast RC oscillator with adjustable frequency
 - RC_SLOW_CLK (130 kHz by default): internal slow RC oscillator
 - OSC_SLOW_CLK (32 kHz by default): external slow clock input through `XTAL_32K_P`. After configuring these two GPIO, also configure the Hold function (see Chapter 5 *GPIO Matrix and IO*)

[MUX > 5.9 Pin Hold Feature\)](#)

The crystal frequency of ESP32-C5 can be either 40 MHz or 48 MHz. The boot mode, along with MTMS, EFUSE_XTAL_48M_SEL and EFUSE_XTAL_48M_SEL_MODE, determines the crystal frequency.

Table 6.2-1. Crystal Frequency

Crystal Frequency	Boot Mode	eFuse 1	eFuse 2 ^b	MTMS
40 MHz	SPI Boot	x	even number of 1	x ^d
48 MHz			odd number of 1	x
40 MHz	Joint Download Boot	1	even number of 1	x
48 MHz			odd number of 1	x
40 MHz		0	x	0 ^c
48 MHz				1 ^c

^a **eFuse 1:** EFUSE_XTAL_48M_SEL_MODE

^b **eFuse 2:** EFUSE_XTAL_48M_SEL[2:0]

^c In the **Joint Download Boot** mode, the user needs to specify the level of the strapping pin MTMS.

^d x: values that have no effect on the result and can therefore be ignored.

6.2.4 Functional Description

6.2.4.1 HP System Clock

As Figure 6.2-1 shows, CPU_CLK is the master clock for CPU and its frequency can be as high as 240 MHz. Alternatively, CPU can run at lower frequencies, such as at 2 MHz, to achieve lower power consumption. CPU_CLK shares the same clock sources with AHB_CLK and APB_CLK. Users can select from XTAL_CLK, PLL_240M_CLK, PLL_160M_CLK, or RC_FAST_CLK as the clock source of CPU_CLK by configuring PCR_SOC_CLK_SEL. For details, see Table 6.2-2 and Table 6.2-3. By default, the CPU clock is sourced from XTAL_CLK with a division factor of 1.

Table 6.2-2. CPU_CLK Clock Source

PCR_SOC_CLK_SEL	CPU Clock Source
0	XTAL_CLK
1	RC_FAST_CLK
2	PLL_F160M_CLK
3	PLL_F240M_CLK

Table 6.2-3. Frequency of CPU_CLK, AHB_CLK and HP_ROOT_CLK

Clock	Source	Frequency
HP_ROOT_CLK	PLL_F240M_CLK	240 MHz
	PLL_F160M_CLK	160 MHz
	XTAL_CLK	48 MHz/40 MHz
	RC_FAST_CLK	20 MHz
CPU_CLK ¹	HP_ROOT_CLK	$f_{\text{HP\_ROOT\_CLK}} / (\text{PCR\_CPU\_DIV\_NUM} + 1)$
AHB_CLK ²	HP_ROOT_CLK	$f_{\text{HP\_ROOT\_CLK}} / (\text{PCR\_AHB\_DIV\_NUM} + 1)$

¹ CPU_CLK frequency must be larger than or equal to AHB_CLK frequency, and must be an integer multiple of AHB_CLK frequency.

² AHB_CLK frequency can not exceed XTAL_CLK frequency.

Note:

When selecting the clock source of HP_ROOT_CLK, or configuring the clock divisor for CPU_CLK and AHB_CLK, please also set [PCR_BUS_CLOCK_UPDATE](#) to apply the new configuration, and read [PCR_BUS_CLOCK_UPDATE](#) to see if new configuration takes effect.

As shown in [6.2-1](#), to generate APB_CLK, AHB_CLK might be divided twice. The first division is compulsory. That is, AHB_CLK is always divided by the divisor ([PCR_APB_DIV_NUM](#) + 1). The second division (also called automatic frequency reduction) is optional. When there is no request from the host in the chip to access peripheral registers, AHB_CLK will be further divided by ([APB_DECREASE_DIV_NUM](#) + 1) to lower power consumption. If the host initiates a request to access peripheral registers, APB_CLK will be restored to the frequency after the first division.

Note that the function of automatic frequency reduction can be disabled (already disabled by default) by configuring [APB_DECREASE_DIV_NUM](#) to 0.

6.2.4.2 LP System Clock

The LP system can operate when most other clocks are disabled. LP system clocks include LP_SLOW_CLK and LP_FAST_CLK.

The clock sources for LP_SLOW_CLK and LP_FAST_CLK are low-frequency clocks:

- LP_SLOW_CLK can be derived from:
 - RC_SLOW_CLK
 - XTAL32K_CLK
 - OSC_SLOW_CLK
- LP_FAST_CLK can be derived from:
 - 20 MHz/24 MHz XTAL_D2_CLK, which is XTAL_CLK divided by 2
 - RC_FAST_CLK
 - XTAL_CLK

The clock source of LP_DYN_SLOW_CLK is LP_SLOW_CLK. There is no frequency change from the clock source.

The clock source of LP_DYN_FAST_CLK depends on the chip's power mode (see Chapter 4 [Low-power Management \[to be added later\]](#)).

- Select LP_FAST_CLK as its clock source in Active and Modem-sleep mode
- Select LP_SLOW_CLK as its clock source in Light-sleep and Deep-sleep mode

6.2.4.3 Peripheral Clocks

Table 6.2-4, Table 6.2-5, Table 6.2-6, and Table 6.2-7 list the derived HP clock sources, HP clocks for each peripheral, derived LP clock sources and LP clocks for each peripheral.

Table 6.2-4. Derived HP Clock Source

	Source Clock		Derived Clock		Source Clock				Derived Clock									Source Clock
Derived Clock	XTAL_CLK 48/40 MHz	PLL_F480M_CLK 480 MHz	PLL_F240M_CLK 240 MHz	PLL_F160M_CLK 160 MHz	RC_FAST_CLK 20 MHz	RC_SLOW_CLK 130 kHz	OSC_SLOW_CLK 32 kHz	XTAL32K_CLK 32 kHz	HP_ROOT_CLK 240 MHz/160 MHz/48(/40) MHz/20 MHz	CRYPTO_CLK	APB_CLK	AHB_CLK	CPU_CLK	LP_DYN_FAST_CLK	LP_DYN_SLOW_CLK	XTAL_D2_CLK 24/20 MHz	LP_FAST_CLK	Clock from IO
PLL_F240M_CLK		Y																
PLL_F160M_CLK		Y																
PLL_F120M_CLK		Y																
PLL_F80M_CLK		Y																
PLL_F48M_CLK		Y																
PLL_F40M_CLK		Y																
HP_ROOT_CLK	Y		Y	Y	Y													
CRYPTO_CLK	Y	Y			Y													
APB_CLK									Y									
AHB_CLK									Y									
CPU_CLK									Y									

Table 6.2-5. HP Clocks Used by Each Peripheral

Peripheral	Source Clock	Derived Clock					Source Clock	Derived Clock										Source Clock
	XTAL_CLK 48/40 MHz	PLL_F240M_CLK 240 MHz	PLL_F160M_CLK 160 MHz	PLL_F120M_CLK 120 MHz	PLL_F80M_CLK 80 MHz	PLL_F48M_CLK 48 MHz	RC_FAST_CLK 20 MHz	HP_ROOT_CLK 240 MHz/160 MHz/48(/40) MHz/20 MHz	CRYPTO_CLK	APB_CLK	AHB_CLK	CPU_CLK	LP_DYN_ FAST_CLK	LP_DYN_ SLOW_CLK	XTAL_D2_CLK 24/20 MHz	LP_FAST_CLK	Clock from IO	
Timer Group (TIMG)	Y				Y		Y											
Main System Watchdog Timers (MWD1)	Y				Y		Y											
I2S Controller (I2S)	Y	Y	Y														I2S_MCLK_in	
UART Controller (UART)	Y				Y		Y											
Remote Control Peripheral (RMT)	Y					Y												
Motor Control PWM (MCPWM)	Y		Y				Y											
I2C Controller (I2C)	Y						Y											
SPI2	Y		Y	Y			Y											
SAR ADC	Y				Y		Y											
USB Serial/JTAG Controller						Y												
Controller Area Network Flexible Data- Rate [to be added later]	Y						Y											
LED PWM Controller (LEDC)	Y	Y				Y												
System Timer	Y						Y											
Parallel IO Controller (PARLIO)	Y	Y					Y										parl_rx_clk_in parl_tx_clk_in	
IO MUX	Y				Y		Y											
AES Accelerator (AES) SHA Accelerator (SHA) RSA Accelerator (RSA) ECC Accelerator (ECC) Digital Signature Algorithm (DSA) HMAC Accelerator (HMAC) Elliptic Curve Digital Signature Algorithm (ECDSA)									Y									
Interrupt Matrix [to be added later]											Y							
SDIO Slave Controller (SDIO)											Y							
Pulse Count Controller (PCNT) [to be added later]										Y								
Event Task Matrix (ETM)											Y							
GDMA Controller (GDMA)											Y							
UHCI											Y							

Continued on the next page...

Table 6.2-5 – Continued from the previous page...

Peripheral	Source Clock	Derived Clock						Source Clock	Derived Clock								Source Clock
	XTAL_CLK 48/40 MHz	PLL_F240M_CLK 240 MHz	PLL_F160M_CLK 160 MHz	PLL_F120M_CLK 120 MHz	PLL_F80M_CLK 80 MHz	PLL_F48M_CLK 48 MHz	RC_FAST_CLK 20 MHz	HP_ROOT_CLK 240 MHz/160 MHz/48(/40) MHz/20 MHz	CRYPTO_CLK	APB_CLK	AHB_CLK	CPU_CLK	LP_DYN_ FAST_CLK	LP_DYN_ SLOW_CLK	XTAL_D2_CLK 24/20 MHz	LP_FAST_CLK	Clock from IO
Debug Assistant												Y					
RISC-V Trace Encoder (TRACE)											Y	Y					
Interrupt priority registers (INTPRI)												Y					

Table 6.2-6. Derived LP Clock Source

Derived Clock	Source Clock		Derived Clock		Source Clock				Derived Clock								Source Clock	
	XTAL_CLK 48/40 MHz	PLL_F480M_CLK 480 MHz	PLL_F240M_CLK 240 MHz	PLL_F160M_CLK 160 MHz	RC_FAST_CLK 20 MHz	RC_SLOW_CLK 130 kHz	OSC_SLOW_CLK 32 kHz	XTAL32K_CLK 32 kHz	HP_ROOT_CLK 240 MHz/160 MHz/48(/40) MHz/20 MHz	CRYPTO_CLK	APB_CLK	AHB_CLK	CPU_CLK	LP_DYN_FAST_CLK	LP_DYN_SLOW_CLK	XTAL_D2_CLK 24/20 MHz	LP_FAST_CLK	Clock from IO
LP_DYN_FAST_CLK						Y	Y	Y									Y	
LP_DYN_SLOW_CLK						Y	Y	Y										
XTAL_D2_CLK	Y																	
LP_FAST_CLK	Y				Y											Y		

Table 6.2-7. LP Clocks Used by Each Peripheral

Derived Clock	Source Clock		Derived Clock		Source Clock				Derived Clock								Source Clock	
	XTAL_CLK 48/40 MHz	PLL_F480M_CLK 480 MHz	PLL_CLK PLL_F120M_CLK 240 MHz	PLL_F160M_CLK 160 MHz	RC_FAST_CLK 20 MHz	RC_SLOW_CLK 130 kHz	OSC_SLOW_CLK 32 kHz	XTAL32K_CLK 32 kHz	HP_ROOT_CLK 240 MHz/160 MHz/48(/40) MHz/20 MHz	CRYPTO_CLK	APB_CLK	AHB_CLK	CPU_CLK	LP_DYN_ FAST_CLK	LP_DYN_ SLOW_CLK	XTAL_D2_CLK 24/20 MHz	LP_FAST_CLK	Clock from IO
eFuse Controller (eFuse)																Y		
RTC Watchdog Timer (RWDt)														Y	Y			
RTC Timer (RTC Timer)														Y	Y			
Brownout Detector														Y				
Power Management Unit (PMU)														Y	Y		Y	

PLL_F480M_CLK

PLL_F480M_CLK is a 480 MHz clock. PLL_F240M_CLK, PLL_F160M_CLK, PLL_F80M_CLK, PLL_F48M_CLK and PLL_F40M_CLK are divided from PLL_F480M_CLK.

CRYPTO_CLK

As shown in Figure 6.2-4, CRYPTO_CLK can be derived from XTAL_CLK, PLL_F480M_CLK, or RC_FAST_CLK, and its frequency is up to 160 MHz.

To protect encryption and decryption peripherals from DPA (Differential Power Analysis) attacks, a random divider strategy is implemented for the functional clock of these peripherals. Four security levels are available, depending on the range of random divider. Users can select the security level by configuring HP_SYSTEM_SEC_DPA_CONF_REG. If HP_SYSTEM_SEC_DPA_CFG_SEL is set to 1, the security level is determined by the configuration of EFUSE_SEC_DPA_LEVEL, otherwise, by the value of HP_SYSTEM_SEC_DPA_LEVEL.

LED_PWM Clock

LEDC module uses PLL_F80M_CLK, RC_FAST_CLK or XTAL_CLK as its clock source. When the system is in low-power mode (APB_CLK is disabled), most peripherals are halted, but LEDC can still work via RC_FAST_CLK.

Slow Clock for Timer Group 0

Using XTAL_CLK as a reference, TIMGO can calculate the frequency of slow clock sources provided by ESP32-C5. The slow clock source can be selected by configuring [PCR_32K_SEL](#):

- 0: Select XTAL32K_CLK clock
- 1: Select OSC_SLOW_CLK clock
- 2: Select RC_SLOW_CLK clock
- 3: Select RC_FAST_CLK clock

Low-Power System Clock Registers

See [6.4.2 LP System Clock Register Summary](#) for the clock registers of low-power system peripherals.

6.2.4.4 PMU Control of High-Performance System Clock Gating

In various operating modes of ESP32-C5, the following register fields can be pre-configured to enable the [Power Management Unit \(PMU\)](#) to control the clock gating of high-performance system peripherals:

- PMU_HP_x_DIG_ICG_APB_EN (x = ACTIVE/MODEM/SLEEP): Controls the clock gating for reading and writing registers of high-performance system peripherals.
- PMU_HP_x_DIG_ICG_FUNC_EN (x = ACTIVE/MODEM/SLEEP): Controls the functional clock gating of high-performance system peripherals.

For the specific configuration process, please refer to the Chapter [4 Low-power Management \[to be added later\]](#).

Tables [6.2-8](#) and [6.2-9](#) list the correspondence between PMU pre-configured register bits and high-performance system clock gating.

Table 6.2-8. Mapping between controlling the clock gating of read/write registers and related PMU registers

PMU_HP_X_DIG_ICG_APB_EN bit	Control the clock gating of read/write registers
0	AES Accelerator (AES) SHA Accelerator (SHA) RSA Accelerator (RSA) ECC Accelerator (ECC) Digital Signature Algorithm (DSA) HMAC Accelerator (HMAC) Elliptic Curve Digital Signature Algorithm (ECDSA)
1	GDMA Controller (GDMA)
2	SPI2
3	Interrupt Matrix [to be added later]
4	I2S Controller (I2S)
6	UART0
7	UART1
8	UHCI
11	Timer Group 0
12	Timer Group 1
13	I2C Controller (I2C)
14	LED PWM Controller (LEDC)
15	Remote Control Peripheral (RMT)
16	System Timer
17	USB Serial/JTAG Controller
18	Two-wire Automotive Interface 0
19	Two-wire Automotive Interface 1
20	Pulse Count Controller (PCNT) [to be added later]
21	Motor Control PWM (MCPWM)
22	Event Task Matrix (ETM)
23	Parallel IO Controller (PARLIO)
25	Debug Assistant
26	GPIO Matrix and IO MUX
28	BitScrambler

Table 6.2-9. Mapping between controlling the functional clock gating of high-performance system peripherals and related PMU registers

PMU_HP_x_DIG_ICG_FUNC_EN bit	control the functional clock gating of high-performance system peripherals
0	GDMA Controller (GDMA)
1	SPI2
2	I2S Receive Side
3	UART0
4	UART1
5	UHCI
6	USB Serial/JTAG Controller
7	I2S Transmit Side
10	MEM_MONITOR
11	SDIO Slave Controller (SDIO)
12	Temperature Sensor
13	Timer Group 0
14	Timer Group 1
16	Event Task Matrix (ETM)
17	High-Performance CPU [to be added later] ASSIST_DEBUG
18	System Timer
19	AES Accelerator (AES) SHA Accelerator (SHA) RSA Accelerator (RSA) ECC Accelerator (ECC) Digital Signature Algorithm (DSA) HMAC Accelerator (HMAC) Elliptic Curve Digital Signature Algorithm (ECDSA)
21	Remote Control Peripheral (RMT)
22	Motor Control PWM (MCPWM)
24	BitScrambler
25	Parallel IO Controller (PARLIO)
27	LED PWM Controller (LEDC)
28	GPIO Matrix and IO MUX
29	I2C Controller (I2C)
30	Two-wire Automotive Interface 0
31	Two-wire Automotive Interface 1

6.3 Programming Procedures

6.3.1 HP System Clock Configuration

When configuring `PCR_SOC_CLK_SEL` to select the clock source of `HP_ROOT_CLK`, or configuring the clock divisor for `CPU_CLK` via `PCR_CPU_DIV_NUM` and `AHB_CLK` via `PCR_AHB_DIV_NUM`, please also set the enable register to apply the new configuration. To check whether the new configuration takes effect, read

[PCR_BUS_CLOCK_UPDATE](#) and see if it is 0.

6.3.2 LP System Clock Configuration

The clock source of LP_SLOW_CLK can be configured via [LP_CLKRST_SLOW_CLK_SEL](#).

The clock source of LP_FAST_CLK can be configured via [LP_CLKRST_FAST_CLK_SEL](#).

6.3.3 Peripheral Clock Reset and Configuration

Notice:

ESP32-C5 features low power consumption. This is why some peripheral clocks are gated (disabled) by default. Before using any of these peripherals, it is mandatory to enable the clock for the given peripheral by setting the corresponding CLK_EN bit to 1, and release the peripheral from reset state by setting the RST_EN bit to 0.

The clocks of most peripherals can be classified into two types:

- Bus clock: used to configure peripheral registers.
- Functional clock: such as UART's reference clock, used by peripherals to operate.

The functional clock of most peripherals can be selected from multiple clock sources. For clock gating registers, whether they are used to gate the bus clock (AHB_CLK, APB_CLK) or the functional clock will be stated in the corresponding register descriptions.

Bus clock switches, functional clock switches, and configuration registers for clock source selection and clock frequency division are grouped into the PCR module. For more information, see Section [6.4 Register Summary](#).

When a peripheral is not working, users can turn off its functional clock by configuring related PCR registers. Turning off the peripheral's functional clock does not affect the rest of the system.

Take the I2C clock configuration as an example.

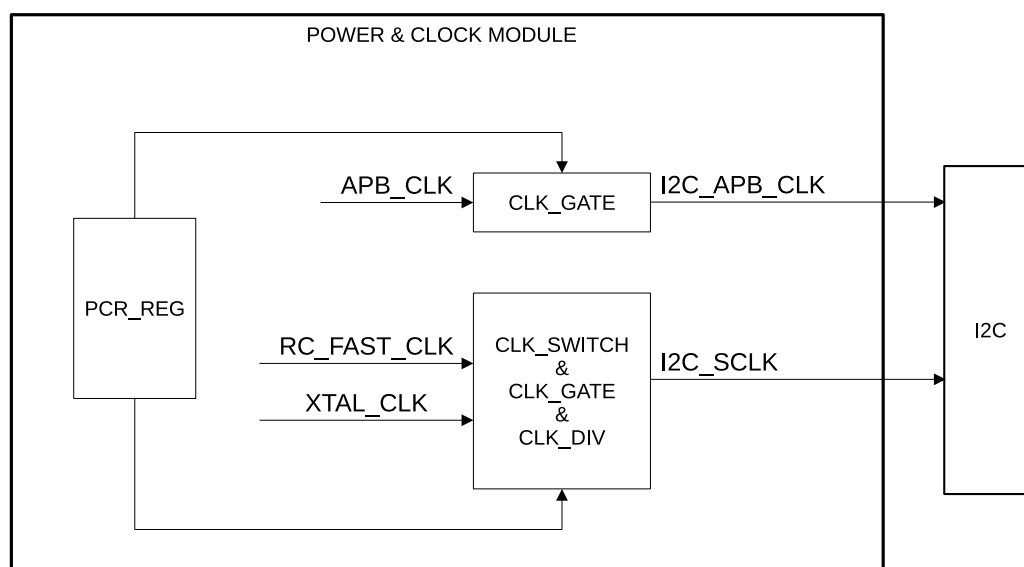


Figure 6.3-1. Clock Configuration Example

Figure 6.3-1 shows the clock structure of I2C. The clock structure of other peripherals is similar to this one. CLK_SWITCH is used to select a clock output and CLK_GATE to turn on/off the clock.

In scenarios that require low power consumption, when the peripheral is not in use, in addition to turning off the functional clock, the bus clock of the peripheral can also be turned off to further lower the power consumption.

Note that if you turn off the bus clock first, the functional clock may continue working. Therefore, when turning off clocks, it is recommended to turn off the functional clock first and then the bus clock; when turning on clocks, it is recommended to turn on the bus clock first and then the functional clock.

Note:

In this chapter, all divisor configuration registers are configured with the actual divisor minus 1.

6.4 Register Summary

6.4.1 PCR Register Summary

The addresses in this section are relative to Power/Clock/Reset (PCR) Register base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Configuration Register			
PCR_UART0_CONF_REG	UART0 configuration register	0x0000	varies
PCR_UART0_SCLK_CONF_REG	UART0_SCLK configuration register	0x0004	R/W
PCR_UART0_PD_CTRL_REG	UART0 power control register	0x0008	R/W
PCR_UART1_CONF_REG	UART1 configuration register	0x000C	varies
PCR_UART1_SCLK_CONF_REG	UART1_SCLK configuration register	0x0010	R/W
PCR_UART1_PD_CTRL_REG	UART1 power control register	0x0014	R/W
PCR_I2C_CONF_REG	I2C configuration register	0x0020	R/W
PCR_I2C_SCLK_CONF_REG	I2C_SCLK configuration register	0x0024	R/W
PCR_TWAIO_CONF_REG	TWAIO configuration register	0x0028	varies
PCR_TWAIO_FUNC_CLK_CONF_REG	TWAIO_FUNC_CLK configuration register	0x002C	R/W
PCR_TWAI1_CONF_REG	TWAI1 configuration register	0x0030	varies
PCR_TWAI1_FUNC_CLK_CONF_REG	TWAI1_FUNC_CLK configuration register	0x0034	R/W
PCR_UHCI_CONF_REG	UHCI configuration register	0x0038	varies
PCR_RMT_CONF_REG	RMT configuration register	0x003C	R/W
PCR_RMT_SCLK_CONF_REG	RMT_SCLK configuration register	0x0040	R/W
PCR_RMT_PD_CTRL_REG	RMT power control register	0x0044	R/W
PCR_LEDC_CONF_REG	LEDC configuration register	0x0048	varies
PCR_LEDC_SCLK_CONF_REG	LEDC_SCLK configuration register	0x004C	R/W
PCR_LEDC_PD_CTRL_REG	LEDC power control register	0x0050	R/W
PCR_TIMERGROUPO_CONF_REG	TIMERGROUPO configuration register	0x0054	varies
PCR_TIMERGROUPO_TIMER_CLK_CONF_REG	TIMERGROUPO_TIMER_CLK configuration register	0x0058	R/W
PCR_TIMERGROUPO_WDT_CLK_CONF_REG	TIMERGROUPO_WDT_CLK configuration register	0x005C	R/W
PCR_TIMERGROUP1_CONF_REG	TIMERGROUP1 configuration register	0x0060	varies
PCR_TIMERGROUP1_TIMER_CLK_CONF_REG	TIMERGROUP1_TIMER_CLK configuration register	0x0064	R/W
PCR_TIMERGROUP1_WDT_CLK_CONF_REG	TIMERGROUP1_WDT_CLK configuration register	0x0068	R/W
PCR_SYSTIMER_CONF_REG	SYSTIMER configuration register	0x006C	varies
PCR_SYSTIMER_FUNC_CLK_CONF_REG	SYSTIMER_FUNC_CLK configuration register	0x0070	R/W
PCR_I2S_CONF_REG	I2S configuration register	0x0074	varies
PCR_I2S_TX_CLKM_CONF_REG	I2S_TX_CLKM configuration register	0x0078	R/W

Name	Description	Address	Access
PCR_I2S_TX_CLKM_DIV_CONF_REG	I2S_TX_CLKM_DIV configuration register	0x007C	R/W
PCR_I2S_RX_CLKM_CONF_REG	I2S_RX_CLKM configuration register	0x0080	R/W
PCR_I2S_RX_CLKM_DIV_CONF_REG	I2S_RX_CLKM_DIV configuration register	0x0084	R/W
PCR_SARADC_CONF_REG	SARADC configuration register	0x0088	R/W
PCR_SARADC_CLKM_CONF_REG	SARADC_CLKM configuration register	0x008C	R/W
PCR_TSENS_CLK_CONF_REG	TSENS_CLK configuration register	0x0090	R/W
PCR_USB_DEVICE_CONF_REG	USB_DEVICE configuration register	0x0094	varies
PCR_INTMTX_CONF_REG	INTMTX configuration register	0x0098	varies
PCR_PCNT_CONF_REG	PCNT configuration register	0x009C	varies
PCR_ETM_CONF_REG	ETM configuration register	0x00A0	varies
PCR_PWM_CONF_REG	PWM configuration register	0x00A4	varies
PCR_PWM_CLK_CONF_REG	PWM_CLK configuration register	0x00A8	R/W
PCR_PARL_IO_CONF_REG	PARL_IO configuration register	0x00AC	varies
PCR_PARL_CLK_RX_CONF_REG	PARL_CLK_RX configuration register	0x00B0	R/W
PCR_PARL_CLK_TX_CONF_REG	PARL_CLK_TX configuration register	0x00B4	R/W
PCR_GDMA_CONF_REG	GDMA configuration register	0x00C0	R/W
PCR_SPI2_CONF_REG	SPI2 configuration register	0x00C4	varies
PCR_SPI2_CLKM_CONF_REG	SPI2_CLKM configuration register	0x00C8	R/W
PCR_AES_CONF_REG	AES configuration register	0x00CC	varies
PCR_SHA_CONF_REG	SHA configuration register	0x00D0	varies
PCR_RSA_CONF_REG	RSA configuration register	0x00D4	varies
PCR_RSA_PD_CTRL_REG	RSA power control register	0x00D8	R/W
PCR_ECC_CONF_REG	ECC configuration register	0x00DC	varies
PCR_ECC_PD_CTRL_REG	ECC power control register	0x00E0	R/W
PCR_DS_CONF_REG	DS configuration register	0x00E4	varies
PCR_HMAC_CONF_REG	HMAC configuration register	0x00E8	varies
PCR_ECDSA_CONF_REG	ECDSA configuration register	0x00EC	varies
PCR_IOMUX_CONF_REG	IOMUX configuration register	0x00F0	R/W
PCR_IOMUX_CLK_CONF_REG	IOMUX_CLK configuration register	0x00F4	R/W
PCR_TRACE_CONF_REG	TRACE configuration register	0x00FC	R/W
PCR_ASSIST_CONF_REG	ASSIST configuration register	0x0100	R/W
PCR_CACHE_CONF_REG	CACHE configuration register	0x0104	R/W
PCR_TIMEOUT_CONF_REG	TIMEOUT configuration register	0x010C	R/W
PCR_SYSCLK_CONF_REG	SYSCLK configuration register	0x0110	varies
PCR_CPU_WAITI_CONF_REG	CPU_WAITI configuration register	0x0114	varies
PCR_CPU_FREQ_CONF_REG	CPU_FREQ configuration register	0x0118	R/W
PCR_AHB_FREQ_CONF_REG	AHB_FREQ configuration register	0x011C	R/W
PCR_APB_FREQ_CONF_REG	APB_FREQ configuration register	0x0120	R/W
PCR_PLL_DIV_CLK_EN_REG	SPLL DIV clock-gating configuration register	0x0128	R/W
PCR_CTRL_32K_CONF_REG	32KHz clock configuration register	0x0130	R/W

Name	Description	Address	Access
PCR_SEC_CONF_REG	Clock source configuration register for External Memory Encryption and De-cryption	0x013C	R/W
PCR_BUS_CLK_UPDATE_REG	Configuration register for applying up-dated high-performance system clock sources	0x0144	R/W/WTC
PCR_SAR_CLK_DIV_REG	SAR ADC clock divisor configuration register	0x0148	R/W
PCR_BS_CONF_REG	BS configuration register	0x0150	R/W
PCR_BS_FUNC_CONF_REG	BS_FUNC_CLK configuration register	0x0154	R/W
PCR_BS_PD_CTRL_REG	BS power control register	0x0158	R/W
PCR_TIMERGROUP_WDT_CONF_REG	TIMERGROUP_WDT configuration register	0x015C	R/W
PCR_TIMERGROUP_XTAL_CONF_REG	TIMERGROUP1 configuration register	0x0160	R/W
PCR_TCM_MEM_MONITOR_CONF_REG	TCM_MEM_MONITOR configuration register	0x016C	varies
PCR_PSRAM_MEM_MONITOR_CONF_REG	PSRAM_MEM_MONITOR configuration register	0x0170	varies
PCR_HPCORE_O_PD_CTRL_REG	HP CORE0 power control register	0x0178	R/W
PCR_SDIO_SLAVE_CONF_REG	SDIO_SLAVE configuration register	0x017C	R/W
Frequency Statistics Register			
PCR_SYSCLOCK_FREQ_QUERY_O_REG	SYSCLOCK frequency query 0 register	0x0124	HRO
Version Register			
PCR_DATE_REG	Date register.	0x0FFC	R/W

6.4.2 LP System Clock Register Summary

The addresses of the last two registers with the LPPER1 prefix in this section are relative to the Low-power Peripheral Register (LPPER1) base address. The addresses of the third and fourth to last registers with the LP_AON prefix in this section are relative to the Low-power Always-on Register (LP_AON) base address. The other addresses in this section are relative to the Low-power Clock/Reset Register (LP_CLKRST) base address. For base address, please refer to Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Configuration Registers			
LP_CLKRST_LP_CLK_CONF_REG	LP system clock source configuration register	0x0000	R/W
LP_CLKRST_LP_CLK_PO_EN_REG	Configuration register for gating clock signals to pins	0x0004	R/W
LP_CLKRST_LP_CLK_EN_REG	Configuration register for gating LP clock source	0x0008	R/W
LP_CLKRST_LP_RST_EN_REG	Configuration register for LP peripheral software reset	0x000C	R/W

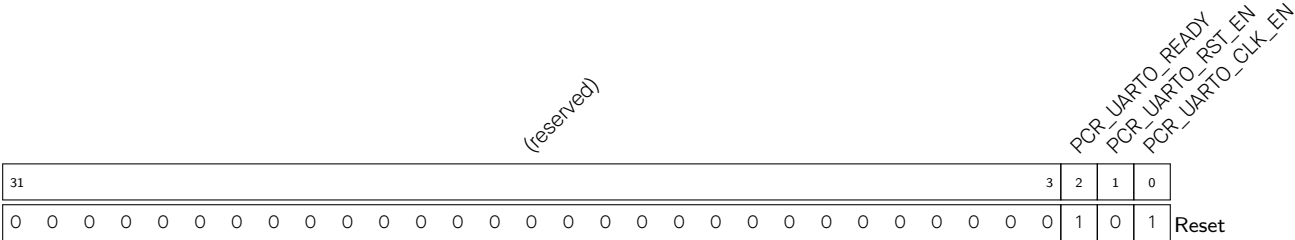
Name	Description	Address	Access
LP_CLKRST_RESET_CAUSE_REG	Reset cause status register	0x0010	varies
LP_CLKRST_CPU_RESET_REG	CPU reset configuration register	0x0014	R/W
LP_CLKRST_FOSC_CNTL_REG	RC_FAST_CLK frequency configuration register	0x0018	R/W
LP_CLKRST_CLK_TO_HP_REG	Configuration register for gating clock signals to HP system	0x0020	R/W
LP_CLKRST_LPMEM_FORCE_REG	Configuration register for forcing the gate of LP memory clock	0x0024	R/W
LP_CLKRST_LPPERI_REG	LP peripheral clock configuration register	0x0028	R/W
LP_CLKRST_XTAL32K_REG	XTAL32K_CLK configuration register	0x002C	R/W
LP_CLKRST_DATE_REG	Version control register	0x03FC	R/W
LP_AON_SYS_CFG_REG	Software system reset register	0x0034	WT
LP_AON_CPUCORE0_CFG_REG	Software CPU reset register	0x0038	WT
LPPERI_CLK_EN_REG	eFuse clock gating configuration register	0x0000	R/W
LPPERI_RESET_EN_REG	eFuse reset configuration register	0x0004	R/W

6.5 Registers

6.5.1 PCR Registers

The addresses in this section are relative to the Power/Clock/Reset (PCR) Register base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 6.1. PCR_UARTO_CONF_REG (0x0000)



PCR_UARTO_CLK_EN Configures whether or not to enable APB_CLK for UARTO.
0: Not enable
1: Enable
(R/W)

PCR_UARTO_RST_EN Configures whether or not to reset UARTO.
0: Not reset
1: Reset
(R/W)

PCR_UARTO_READY Represents whether or not UARTO is released from reset.
0: Not released
1: Released
(RO)

Register 6.2. PCR_UARTO_SCLK_CONF_REG (0x0004)

(reserved)										PCR_UARTO_SCLK_EN				PCR_UARTO_SCLK_SEL				PCR_UARTO_SCLK_DIV_NUM				PCR_UARTO_SCLK_DIV_B				PCR_UARTO_SCLK_DIV_A			
31					23	22	21	20	19					12	11					6	5					0			
0	0	0	0	0	0	0	0	0	0	1	3				0				0				0				Reset		

PCR_UARTO_SCLK_DIV_A Configures the denominator of the divisor's fractional part's fractional part for UARTO functional clock. (R/W)

PCR_UARTO_SCLK_DIV_B Configures the numerator of the divisor's fractional part for UARTO functional clock. (R/W)

PCR_UARTO_SCLK_DIV_NUM Configures the integral part of the divisor for UARTO functional clock. (R/W)

PCR_UARTO_SCLK_SEL Configures the clock source of UARTO.

- 0: No clock source
 - 1: PLL_F80M_CLK
 - 2: RC_FAST_CLK
 - 3 (default): XTAL_CLK
- (R/W)

PCR_UARTO_SCLK_EN Configures whether or not to enable UARTO functional clock.

- 0: Not enable
 - 1: Enable
- (R/W)

Register 6.3. PCR_UART0_PD_CTRL_REG (0x0008)

(reserved)																																PCR_UART0_MEM_FORCE_PD PCR_UART0_MEM_FORCE_PU (reserved)				
31																															3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset

Reset

PCR_UART0_MEM_FORCE_PU Configures whether or not to force power up UART0 memory.

0: Not force power up UART0 memory

1: Force power up UART0 memory

(R/W)

PCR_UART0_MEM_FORCE_PD Configures whether or not to force power down UART0 memory.

0: Not force power down UART0 memory

1: Force power down UART0 memory

(R/W)

Register 6.4. PCR_UART1_CONF_REG (0x000C)

(reserved)																												PCR_UART1_READY PCR_UART1_RST_EN PCR_UART1_CLK_EN			
31																												3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Reset

PCR_UART1_CLK_EN Configures whether or not to enable APB_CLK for UART1.

0: Not enable

1: Enable

(R/W)

PCR_UART1_RST_EN Configures whether or not to reset UART1.

0: Not reset

1: Reset

(R/W)

PCR_UART1_READY Represents whether or not UART1 is released from reset.

0: Not released

1: Released

(RO)

Register 6.5. PCR_UART1_SCLK_CONF_REG (0x0010)

(reserved)										PCR_UART1_SCLK_EN PCR_UART1_SCLK_SEL				PCR_UART1_SCLK_DIV_NUM				PCR_UART1_SCLK_DIV_B				PCR_UART1_SCLK_DIV_A			
31								23	22	21	20	19				12	11			6	5			0	
0	0	0	0	0	0	0	0	0	0	1	3	0			0			0						Reset	

Reset

PCR_UART1_SCLK_DIV_A Configures the denominator of the divisor's fractional part for UART1 functional clock. (R/W)

PCR_UART1_SCLK_DIV_B Configures the numerator of the divisor's fractional part for UART1 functional clock. (R/W)

PCR_UART1_SCLK_DIV_NUM Configures the integral part of the divisor for UART1 functional clock. (R/W)

PCR_UART1_SCLK_SEL Configures the clock source of UART1.

- 0: No clock source
 - 1: PLL_F80M_CLK
 - 2: RC_FAST_CLK
 - 3 (default): XTAL_CLK
- (R/W)

PCR_UART1_SCLK_EN Configures whether or not to enable UART1 functional clock.

- 0: Not enable
 - 1: Enable
- (R/W)

Register 6.6. PCR_UART1_PD_CTRL_REG (0x0014)

(reserved)																															PCR_UART1_MEM_FORCE_PD PCR_UART1_MEM_FORCE_PU (reserved)						
31																															3	2	1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset

PCR_UART1_MEM_FORCE_PU Configures whether or not to force power up UART1 memory.

0: Not force power up UART1 memory

1: Force power up UART1 memory

(R/W)

PCR_UART1_MEM_FORCE_PD Configures whether or not to force power down UART1 memory.

0: Not force power down UART1 memory

1: Force power down UART1 memory

(R/W)

Register 6.7. PCR_I2C_CONF_REG (0x0020)

(reserved)																																PCR_I2C_READY PCR_I2C_RST_EN PCR_I2C_CLK_EN		
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_I2C_CLK_EN Configures whether or not to enable APB_CLK for I2C.

0: Not enable

1: Enable

(R/W)

PCR_I2C_RST_EN Configures whether or not to reset I2C.

0: Not reset

1: Reset

(R/W)

PCR_I2C_READY Represents whether or not I2C is released from reset.

0: Not released

1: Released

(RO)

Register 6.8. PCR_I2C_SCLK_CONF_REG (0x0024)

(reserved)										PCR_I2C_SCLK_EN (reserved)		PCR_I2C_SCLK_SEL				PCR_I2C_SCLK_DIV_NUM						PCR_I2C_SCLK_DIV_B				PCR_I2C_SCLK_DIV_A				
31										23		22	21	20		19						12		11		6		5	0	
0 0 0 0 0 0 0 0 0 0										1		0	0	0						0		0		0		Reset				

PCR_I2C_SCLK_DIV_A Configures the denominator of the divisor’s fractional part for I2C functional clock. (R/W)

PCR_I2C_SCLK_DIV_B Configures the numerator of the divisor’s fractional part for I2C functional clock. (R/W)

PCR_I2C_SCLK_DIV_NUM Configures the integral part of the divisor for I2C functional clock. (R/W)

PCR_I2C_SCLK_SEL Configures the clock source of I2C.

- 0 (default): XTAL_CLK
- 1: RC_FAST_CLK

(R/W)

PCR_I2C_SCLK_EN Configures whether or not to enable I2C functional clock.

- 0: Not enable
- 1: Enable

(R/W)

Register 6.9. PCR_TWAI0_CONF_REG (0x0028)

(reserved)																												3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_TWAI0_CLK_EN Configures whether or not to enable TWAI0 APB_CLK.

0: Not enable

1: Enable

(R/W)

PCR_TWAIO_RST_EN Configures whether or not to reset TWAIO.

0: Not reset

1: Reset

(R/W)

PCR_TWAIO_READY Represents whether or not TWAIO is released from reset.

0: Not released

1: Released

(RO)

Register 6.10. PCR_TWAI0_FUNC_CLK_CONF_REG (0x002C)

[illegible]

PCR_TWAIO_FUNC_CLK_SEL Configures the clock source of TWAIO.

0 (default): XTAL_CLK

1: RC_FAST_CLK

(R/W)

PCR_TWAI0_FUNC_CLK_EN Configures whether or not to enable TWAI0 functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.11. PCR_TWAI1_CONF_REG (0x0030)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(reserved)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Reset

PCR_TWI1_CLK_EN Configures whether or not to enable TWI1 APB_CLK.

0: Not enable

1: Enable

(R/W)

PCR_TWA11_RST_EN Configures whether or not to reset TWA11.

0: Not reset

1: Reset

(R/W)

PCR_TWA1_READY Represents whether or not TWA1 is released from reset.

0: Not released

1: Released

(RO)

Register 6.12. PCR_TWA11_FUNC_CLK_CONF_REG (0x0034)

[illegible]

PCR_TWA11_FUNC_CLK_SEL Configures the clock source of TWA11.

0 (default): XTAL_CLK

1: RC FAST CLK

(R/W)

PCR_TWA11_FUNC_CLK_EN Configures whether or not to enable TWA11 functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.13. PCR_UHCI_CONF_REG (0x0038)

(reserved)																															PCR_UHCI_READY PCR_UHCI_RST_EN PCR_UHCI_CLK_EN			
31																															3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															1	0	1	Reset

PCR_UHCI_CLK_EN Configures whether or not to enable UHCI clock.

0: Not enable

1: Enable

(R/W)

PCR_UHCI_RST_EN Configures whether or not to reset UHCI.

0: Not reset

1: Reset

(R/W)

PCR_UHCI_READY Represents whether or not UCHI is released from reset.

0: Not released

1: Released

(RO)

Register 6.14. PCR_RMT_CONF_REG (0x003C)

(reserved)																												PCR_RMT_READY PCR_RMT_RST_EN PCR_RMT_CLK_EN			
31																												3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_RMT_CLK_EN Configures whether or not to enable APB_CLK for RMT.

0: Enable

1: Not enable

(R/W)

PCR_RMT_RST_EN Configures whether or not to reset RMT.

0: Not reset

1: Reset

(R/W)

PCR_RMT_READY Represents whether or not RMT is released from reset.

0: Not released

1: Released

(RO)

Register 6.15. PCR_RMT_SCLK_CONF_REG (0x0040)

(reserved)										PCR_RMT_SCLK_EN PCR_RMT_SCLK_SEL		PCR_RMT_SCLK_DIV_NUM					PCR_RMT_SCLK_DIV_B		PCR_RMT_SCLK_DIV_A		
31	23	22	21	20	19	12	11	6	5	0											
0	0	0	0	0	0	0	0	0	0	0	0	1	1			0		0			Reset

Reset

PCR_RMT_SCLK_DIV_A Configures the denominator of the divisor's fractional part for RMT functional clock. (R/W)

PCR_RMT_SCLK_DIV_B Configures the numerator of the divisor's fractional part for RMT functional clock. (R/W)

PCR_RMT_SCLK_DIV_NUM Configures the integral part of the divisor for RMT functional clock. (R/W)

PCR_RMT_SCLK_SEL Configures the clock source of RMT.

0: XTAL_CLK

1 (default): RC_FAST_CLK

2: PLL_F80M_CLK

(R/W)

PCR_RMT_SCLK_EN Configures whether or not to enable RMT functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.16. PCR_RMT_PD_CTRL_REG (0x0044)

(reserved)																															PCR_RMT_MEM_FORCE_PD PCR_RMT_MEM_FORCE_PU (reserved)			
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	Reset	

PCR_RMT_MEM_FORCE_PU Configures whether or not to force power up RMT memory.

0: Not force power up

1: Force power up

(R/W)

PCR_RMT_MEM_FORCE_PD Configures whether or not to force power down RMT memory.

0: Not force power down

1: Force power down

(R/W)

Register 6.17. PCR_LEDC_CONF_REG (0x0048)

(reserved)																															PCR_LEDC_READY PCR_LEDC_RST_EN PCR_LEDC_CLK_EN			
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_LEDC_CLK_EN Configures whether or not to enable APB_CLK for LEDC.

0: Not enable

1: Enable

(R/W)

PCR_LEDC_RST_EN Configures whether or not to reset LEDC.

0: Not reset

1: Reset

(R/W)

PCR_LEDC_READY Represents whether or not LEDC is released from reset.

0: Not released

1: Released

(RO)

Register 6.18. PCR_LEDC_SCLK_CONF_REG (0x004C)

[illegible]

PCR_LEDC_SCLK_SEL Configures the clock source of LEDC.

0 (default): XTAL_CLK

1: RC_FAST_CLK

2: PLL F80M CLK

3: No clock source

(R/W)

PCR_LEDC_SCLK_EN Configures whether or not to enable LEDC functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.19. PCR_LEDC_PD_CTRL_REG (0x0050)

Register 0x00000000: PCR_LED_CTRL

31	3	2	1	0
(reserved)				
Reset				

PCR_LED_CTRL_MEM_FORCE_PD
PCR_LED_CTRL_MEM_FORCE_PU
(reserved)

PCR_LEDC_MEM_FORCE_PU Configures whether or not to force power up LEDC memory.

0: Not force power up

1: Force power up

(R/W)

PCR_LEDC_MEM_FORCE_PD Configures whether or not to force power down LEDC memory.

0: Not force power down

1: Force power down

(R/W)

Register 6.20. PCR_TIMERGROUP0_CONF_REG (0x0054)

(reserved)																										PCR_TGO_TIMER1_READY					
																										PCR_TGO_TIMER0_READY					
																										PCR_TGO_WDT_READY					
																										PCR_TGO_RST_EN					
																										PCR_TGO_CLK_EN					
31																										5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	Reset

PCR_TGO_CLK_EN Configures whether or not to enable APB_CLK for Timer Group 0.

0: Not enable

1: Enable

(R/W)

PCR_TGO_RST_EN Configures whether or not to reset Timer Group 0.

0: Not reset

1: Reset

(R/W)

PCR_TGO_WDT_READY Represents whether or not the WDT in Timer Group 0 is released from reset.

0: Not released

1: Released

(RO)

PCR_TGO_TIMER0_READY Represents whether or not Timer 0 in Timer Group 0 is released from reset.

0: Not released

1: Released

(RO)

PCR_TGO_TIMER1_READY Represents whether or not Timer 1 in Timer Group 0 is released from reset.

0: Not released

1: Released

(RO)

Register 6.21. PCR_TIMERGROUPO_TIMER_CLK_CONF_REG (0x0058)

[illegible]

PCR_TGO_TIMER_CLK_SEL Configures the clock source of general-purpose timers in Timer Group 0.

0 (default): XTAL CLK

1: RC_FAST_CLK

2: PLL F48M CLK

3: No clock source

(R/W)

PCR_TGO_TIMER_CLK_EN Configures whether or not to enable the clock of general-purpose timers in Timer Group 0.

0: Not enable

1: Enable

(R/W)

Register 6.22. PCR_TIMERGROUP0_WDT_CLK_CONF_REG (0x005C)

[illegible]

PCR_TGO_WDT_CLK_SEL Configures the clock source of WDT in Timer Group 0.

0 (default): XTAL CLK

1: RC_FAST_CLK

2: PLL F80M CLK

3: No clock source

(R/W)

PCR_TGO_WDT_CLK_EN Configures whether or not to enable the clock of WDT in Timer Group 0.

0: Not enable

1: Enable

(R/W)

Register 6.23. PCR_TIMERGROUP1_CONF_REG (0x0060)

(reserved)																												PCR_TG1_TIMER1_READY PCR_TG1_TIMER0_READY PCR_TG1_WDT_READY PCR_TG1_RST_EN PCR_TG1_CLK_EN					
31																												5	4	3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												1	1	1	0	1	Reset

- PCR_TG1_CLK_EN** Configures whether or not to enable APB_CLK for Timer Group 1.
0: Not enable
1: Enable
(R/W)
- PCR_TG1_RST_EN** Configures whether or not to reset Timer Group 1.
0: Not reset
1: Reset
(R/W)
- PCR_TG1_WDT_READY** Represents whether or not the WDT in Timer Group 1 is released from reset.
0: Not released
1: Released
(RO)
- PCR_TG1_TIMER0_READY** Represents whether or not Timer 0 in Timer Group 1 is released from reset.
0: Not released
1: Released
(RO)timer_group0 Timer 0module (RO)
- PCR_TG1_TIMER1_READY** Represents whether or not Timer 1 in Timer Group 1 is released from reset.
0: Not released
1: Released
(RO)

Register 6.24. PCR_TIMERGROUP1_TIMER_CLK_CONF_REG (0x0064)

[illegible]

PCR_TG1_TIMER_CLK_SEL Configures the clock source of general-purpose timers in Timer Group

1.

0 (default): XTAL_CLK

1: RC_FAST_CLK

2: PLL_F48M_CLK

3: No clock source

(R/W)

PCR_TG1_TIMER_CLK_EN Configures whether or not to enable the clock of general-purpose timers in Timer Group 1.

0: Not enable

1: Enable

(R/W)

Register 6.25. PCR_TIMERGROUP1_WDT_CLK_CONF_REG (0x0068)

[illegible]

PCR_TG1_WDT_CLK_SEL Configures the clock source of WDT in Timer Group 1.

0 (default): XTAL_CLK

1: RC_FAST_CLK

2: PLL_F80M_CLK

3: No clock source

(R/W)

PCR_TG1_WDT_CLK_EN Configures whether or not to enable the clock for WDT in Timer Group 1.

0: Not enable

1: Enable

(R/W)

Register 6.26. PCR_SYSTIMER_CONF_REG (0x006C)

(reserved)																															PCR_SYSTIMER_READY PCR_SYSTIMER_RST_EN PCR_SYSTIMER_CLK_EN			
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset		

PCR_SYSTIMER_CLK_EN Configures whether or not to enable APB_CLK for System Timer.

0: Not enable

1: Enable

(R/W)

PCR_SYSTIMER_RST_EN Configures whether or not to reset System Timer.

0: Not reset

1: Reset

(R/W)

PCR_SYSTIMER_READY Represents whether or not the System Timer is released from reset.

0: Not released

1: Released

(RO)

Register 6.27. PCR_SYSTIMER_FUNC_CLK_CONF_REG (0x0070)

(reserved)										PCR_SYSTIMER_FUNC_CLK_EN (reserved)				PCR_SYSTIMER_FUNC_CLK_SEL (reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
31																				23	22	21	20	19											0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

PCR_SYSTIMER_FUNC_CLK_SEL Configures the clock source of System Timer.

0 (default): XTAL_CLK

1: RC_FAST_CLK

(R/W)

PCR_SYSTIMER_FUNC_CLK_EN Configures whether or not to enable System Timer functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.28. PCR_I2S_CONF_REG (0x0074)

(reserved)																												PCR_I2S_TX_READY PCR_I2S_RX_READY PCR_I2S_RST_EN PCR_I2S_CLK_EN				
31																												4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	Reset	

PCR_I2S_CLK_EN Configures whether or not to enable APB_CLK for I2S.
0: Not enable
1: Enable
(R/W)

PCR_I2S_RST_EN Configures whether or not to reset I2S.
0: Not reset
1: Reset
(R/W)

PCR_I2S_RX_READY Represents whether or not I2S RX is released from reset.
0: Not released
1: Released
(RO)

PCR_I2S_TX_READY Represents whether or not I2S TX is released from reset.
0: Not released
1: Released
(RO)

Register 6.29. PCR_I2S_TX_CLKM_CONF_REG (0x0078)

(reserved)																PCR_I2S_TX_CLKM_EN				PCR_I2S_TX_CLKM_SEL				PCR_I2S_TX_CLKM_DIV_NUM								(reserved)																			
31								23								22	21	20	19								12								11	0															
0	0	0	0	0	0	0	0	0	0	1	0		2						0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset																		

PCR_I2S_TX_CLKM_DIV_NUM Configures the integral part of I2S TX clock divisor.
(R/W)

PCR_I2S_TX_CLKM_SEL Configures the clock source of I2S TX.
0: XTAL_CLK
1: PLL_F240M_CLK
2: PLL_F160M_CLK
3: I2S_MCLK_in
(R/W)

PCR_I2S_TX_CLKM_EN Configures whether or not to enable I2S TX functional clock.
0: Not enable
1: Enable
(R/W)

Register 6.30. PCR_I2S_TX_CLKM_DIV_CONF_REG (0x007C)

(reserved)				PCR_I2S_TX_CLKM_DIV_YN1				PCR_I2S_TX_CLKM_DIV_X				PCR_I2S_TX_CLKM_DIV_Y				PCR_I2S_TX_CLKM_DIV_Z			
31	28	27	26	18				17	9				8	0					
0	0	0	0	0	0				1	0				0	Reset				

Reset

PCR_I2S_TX_CLKM_DIV_Z For $b \leq a/2$, the value of I2S_TX_CLKM_DIV_Z is b. For $b > a/2$, the value of I2S_TX_CLKM_DIV_Z is $(a - b)$. (R/W)

PCR_I2S_TX_CLKM_DIV_Y For $b \leq a/2$, the value of I2S_TX_CLKM_DIV_Y is $(a \% b)$. For $b > a/2$, the value of I2S_TX_CLKM_DIV_Y is $(a \% (a - b))$. (R/W)

PCR_I2S_TX_CLKM_DIV_X For $b \leq a/2$, the value of I2S_TX_CLKM_DIV_X is $\text{floor}(a/b) - 1$. For $b > a/2$, the value of I2S_TX_CLKM_DIV_X is $\text{floor}(a/(a - b)) - 1$. (R/W)

PCR_I2S_TX_CLKM_DIV_YN1 For $b \leq a/2$, the value of I2S_TX_CLKM_DIV_YN1 is 0. For $b > a/2$, the value of I2S_TX_CLKM_DIV_YN1 is 1. (R/W)

Note:

“a” and “b” represent the denominator and the numerator of the fractional divisor, respectively. For more information, see Section 27.6 in Chapter *I2S Controller (I2S)*.

Register 6.31. PCR_I2S_RX_CLKM_CONF_REG (0x0080)

(reserved)								PCR_I2S_MCLK_SEL				PCR_I2S_RX_CLKM_EN				PCR_I2S_RX_CLKM_SEL				PCR_I2S_RX_CLKM_DIV_NUM								(reserved)									
31								24	23	22	21	20	19								12	11															0
0	0	0	0	0	0	0	0	0	0	1	0	2								0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset			

PCR_I2S_RX_CLKM_DIV_NUM Configures the integral divisor for I2S clock. (R/W)

PCR_I2S_RX_CLKM_SEL Configures the clock source of I2S RX.

- 0: XTAL_CLK
- 1: PLL_F240M_CLK
- 2: PLL_F160M_CLK
- 3: I2S_MCLK_in
- (R/W)

PCR_I2S_RX_CLKM_EN Configures whether or not to enable I2S RX functional clock.

- 0: Not enable
- 1: Enable
- (R/W)

PCR_I2S_MCLK_SEL Configures to select master clock.

- 0 (default): I2S_RX_CLK
- 1: I2S_TX_CLK
- (R/W)

Register 6.32. PCR_I2S_RX_CLKM_DIV_CONF_REG (0x0084)

(reserved)				PCR_I2S_RX_CLKM_DIV_YN1				PCR_I2S_RX_CLKM_DIV_X				PCR_I2S_RX_CLKM_DIV_Y				PCR_I2S_RX_CLKM_DIV_Z				
31	28	27	26					18	17					9	8					0
0	0	0	0	0	0				1				0				Reset			

Reset

PCR_I2S_RX_CLKM_DIV_Z For $b \leq a/2$, the value of I2S_RX_CLKM_DIV_Z is b. For $b > a/2$, the value of I2S_RX_CLKM_DIV_Z is $(a - b)$. (R/W)

PCR_I2S_RX_CLKM_DIV_Y For $b \leq a/2$, the value of I2S_RX_CLKM_DIV_Y is $(a \% b)$. For $b > a/2$, the value of I2S_RX_CLKM_DIV_Y is $(a \% (a - b))$. (R/W)

PCR_I2S_RX_CLKM_DIV_X For $b \leq a/2$, the value of I2S_RX_CLKM_DIV_X is $\text{floor}(a/b) - 1$. For $b > a/2$, the value of I2S_RX_CLKM_DIV_X is $\text{floor}(a/(a-b)) - 1$. (R/W)

PCR_I2S_RX_CLKM_DIV_YN1 For $b \leq a/2$, the value of I2S_RX_CLKM_DIV_YN1 is 0. For $b > a/2$, the value of I2S_RX_CLKM_DIV_YN1 is 1. (R/W)

Note:

“a” and “b” represent the denominator and the numerator of the fractional divisor, respectively. For more information, see Section [27.6](#).

Register 6.33. PCR_SARADC_CONF_REG (0x0088)

(reserved)																												4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	Reset

PCR_SARADC_RST_EN Configures whether or not to reset functional registers of SAR ADC.

0: Not reset

1: Reset

(R/W)

PCR_SARADC_REG_CLK_EN Configures whether or not to enable APB_CLK for SAR ADC.

0: Not enable

1: Enable

(R/W)

PCR_SARADC_REG_RST_EN Configures whether or not to reset APB registers of SAR ADC.

0: Not reset

1: Reset

(R/W)

Register 6.34. PCR_SARADC_CLKM_CONF_REG (0x008C)

(reserved)										PCR_SARADC_CLKM_EN PCR_SARADC_CLKM_SEL				PCR_SARADC_CLKM_DIV_NUM				PCR_SARADC_CLKM_DIV_B				PCR_SARADC_CLKM_DIV_A								
31									23	22	21	20	19					12	11					6	5					0
0	0	0	0	0	0	0	0	0	0	1	0	4				0				0								Reset		

PCR_SARADC_CLKM_DIV_A Configures the denominator of the divisor's fractional part for SAR ADC functional clock. (R/W)

PCR_SARADC_CLKM_DIV_B Configures the numerator of the divisor's fractional part for SAR ADC functional clock. (R/W)

PCR_SARADC_CLKM_DIV_NUM Configures the integral part of the divisor for SAR ADC functional clock. (R/W)

PCR_SARADC_CLKM_SEL Configures the clock source of SAR ADC.

0 (default): XTAL_CLK

1: RC_FAST_CLK

2: PLL_F80M_CLK

3: No clock source

(R/W)

PCR_SARADC_CLKM_EN Configures whether or not to enable SAR ADC functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.35. PCR_TSENS_CLK_CONF_REG (0x0090)

[illegible]

PCR_TSENS_CLK_SEL Configures the clock source of the temperature sensor.

0 (default): XTAL_CLK

1: RC_FAST_CLK

(R/W)

PCR_TSENS_CLK_EN Configures whether or not to enable the clock of the temperature sensor.

0: Not enable

1: Enable

(R/W)

PCR_TSENS_RST_EN Configures whether or not to reset the temperature sensor.

0: Not reset

1: Reset

(R/W)

Register 6.36. PCR_USB_DEVICE_CONF_REG (0x0094)

(reserved)																																PCR_USB_DEVICE_READY PCR_USB_DEVICE_RST_EN PCR_USB_DEVICE_CLK_EN			
31																															3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1			

PCR_USB_DEVICE_JTAG_CLK_EN Configures whether or not to enable USB Serial/JTAG clock.
0: Not enable
1: Enable
(R/W)

PCR_USB_DEVICE_JTAG_RST_EN Configures whether or not to reset USB Serial/JTAG.
0: Not reset
1: Reset
(R/W)

PCR_USB_DEVICE_JTAG_READY Represents whether or not USB Serial/JTAG is released from re-set.
0: Not released
1: Released
(RO)

Register 6.37. PCR_INTMTX_CONF_REG (0x0098)

(reserved)																															PCR_INTMTX_READY PCR_INTMTX_RST_EN PCR_INTMTX_CLK_EN			
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_INTMTX_CLK_EN Configures whether or not to enable Interrupt Matrix clock.

0: Not enable

1: Enable

(R/W)

PCR_INTMTX_RST_EN Configures whether or not to reset Interrupt Matrix.

0: Not reset

1: Reset

(R/W)

PCR_INTMTX_READY Represents whether or not Interrupt Matrix is released from reset.

0: Not released

1: Released

(RO)

Register 6.38. PCR_PCNT_CONF_REG (0x009C)

(reserved)																																PCR_PONT_READY PCR_PONT_RST_EN PCR_PONT_CLK_EN			
31																															3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset		

PCR_PCNT_CLK_EN Configures whether or not to enable PCNT clock.

0: Not enable

1: Enable

(R/W)

PCR_PCNT_RST_EN Configures whether or not to reset PCNT.

0: Not reset

1: Reset

(R/W)

PCR_PCNT_READY Represents whether or not PCNT is released from reset.

0: Not released

1: Released

(RO)

Register 6.39. PCR_ETM_CONF_REG (0x00A0)

(reserved)																												PCR_ETM_READY PCR_ETM_RST_EN PCR_ETM_CLK_EN			
31																												3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_ETM_CLK_EN Configures whether or not to enable ETM clock.

0: Not enable

1: Enable

(R/W)

PCR_ETM_RST_EN Configures whether or not to reset ETM.

0: Reset

1: Not reset

(R/W)

PCR_ETM_READY Represents whether or not ETM is released from reset.

0: Not released

1: Released

(RO)

Register 6.40. PCR_PWM_CONF_REG (0x00A4)

(reserved)																															PCR_PWM_READY PCR_PWM_RST_EN PCR_PWM_CLK_EN			
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_PWM_CLK_EN Configures whether or not to enable MCPWM bus clock.

0: Not enable

1: Enable

(R/W)

PCR_PWM_RST_EN Configures whether or not to reset MCPWM.

0: Not reset

1: Reset

(R/W)

PCR_PWM_READY Represents whether or not MCPWM is released from reset.

0: Not released

1: Released

(RO)

Register 6.41. PCR_PWM_CLK_CONF_REG (0x00A8)

(reserved)										PCR_PWM_CLKM_EN				PCR_PWM_CLKM_SEL				PCR_PWM_DIV_NUM				(reserved)											
31									23	22	21	20	19					12	11									0					
0	0	0	0	0	0	0	0	0	0	1	0	4								0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PCR_PWM_DIV_NUM Configures the integral part of the divisor for MCPWM functional clock. (R/W)

PCR_PWM_CLKM_SEL Configures the clock source of MCPWM.

0 (default): XTAL_CLK

1: RC_FAST_CLK

2: PLL_F160M_CLK

3: No clock source

(R/W)

PCR_PWM_CLKM_EN Configures whether or not to enable MCPWM functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.42. PCR_PARL_IO_CONF_REG (0x00AC)

(reserved)																												PCR_PARL_READY				PCR_PARL_RST_EN				PCR_PARL_CLK_EN			
31																												3	2	1	0								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset			

PCR_PARL_CLK_EN Configures whether or not to enable APB_CLK for Parallel IO.

0: Not enable

1: Enable

(R/W)

PCR_PARL_RST_EN Configures whether or not to reset Parallel IO APB registers.

0: Not reset

1: Reset

(R/W)

PCR_PARL_READY Represents whether or not Parallel IO is released from reset.

0: Not released

1: Released

(RO)

Register 6.43. PCR_PARL_CLK_RX_CONF_REG (0x00B0)

(reserved)																PCR_PARL_RX_RST_EN			PCR_PARL_CLK_RX_EN			PCR_PARL_CLK_RX_SEL			PCR_PARL_CLK_RX_DIV_NUM															
31																20	19	18	17	16	15															0				
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0	1	0	0																Reset					

PCR_PARL_CLK_RX_DIV_NUM Configures the integral divisor for Parallel IO RX clock. (R/W)

PCR_PARL_CLK_RX_SEL Configures the clock source of Parallel IO RX.

- 0 (default): XTAL
- 1: RC_FAST_CLK
- 2: PLL_F240M_CLK
- 3: Use the clock from chip pin

(R/W)

PCR_PARL_CLK_RX_EN Configures whether or not to enable Parallel IO RX clock.

- 0: Not enable
- 1: Enable

(R/W)

PCR_PARL_RX_RST_EN Configures whether or not to reset Parallel IO RX.

- 0: Not reset
- 1: Reset

(R/W)

Register 6.44. PCR_PARL_CLK_TX_CONF_REG (0x00B4)

(reserved)												PCR_PARL_TX_RST_EN			PCR_PARL_CLK_TX_EN			PCR_PARL_CLK_TX_SEL			PCR_PARL_CLK_TX_DIV_NUM										
31													20	19	18	17	16	15													0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0												Reset		

Reset

PCR_PARL_CLK_TX_DIV_NUM Configures the integral divisor for Parallel IO TX clock. (R/W)**PCR_PARL_CLK_TX_SEL** Configures the clock source of Parallel IO TX.

0 (default): XTAL

1: RC_FAST_CLK

2: PLL_F240M_CLK

3: Use the clock from chip pin

(R/W)

PCR_PARL_CLK_TX_EN Configures whether or not to enable Parallel IO TX clock.

0: Not enable

1: Enable

(R/W)

PCR_PARL_TX_RST_EN Configures whether or not to reset Parallel IO TX.

0: Not reset

1: Reset

(R/W)

Register 6.45. PCR_GDMA_CONF_REG (0x00C0)

(reserved)																															PCR_GDMA_RST_EN			PCR_GDMA_CLK_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
31																															2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

PCR_GDMA_CLK_EN Configures whether or not to enable GDMA clock.

0: Not enable

1: Enable

(R/W)

PCR_GDMA_RST_EN Configures whether or not to reset GDMA.

0: Not reset

1: Reset

(R/W)

Register 6.46. PCR_SPI2_CONF_REG (0x00C4)

(reserved)																																PCR_SPI2_READY PCR_SPI2_RST_EN PCR_SPI2_CLK_EN			
31																															3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset		

PCR_SPI2_CLK_EN Configures whether or not to enable APB_CLK for SPI2.

0: Not enable

1: Enable

(R/W)

PCR_SPI2_RST_EN Configures whether or not to reset SPI2.

0: Not reset

1: Reset

(R/W)

PCR_SPI2_READY Represents whether or not SPI2 is released from reset.

0: Not released

1: Released

(RO)

Register 6.47. PCR_SPI2_CLKM_CONF_REG (0x00C8)

(reserved)										PCR_SPI2_CLKM_EN PCR_SPI2_CLKM_SEL										(reserved)											
31											23	22	21	20	19															0	
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PCR_SPI2_CLKM_SEL Configures the clock source of SPI2.

0 (default): XTAL_CLK

1: PLL_F160M_CLK

2: RC_FAST_CLK

3: PLL_F120M_CLK

(R/W)

PCR_SPI2_CLKM_EN Configures whether or not to enable SPI2 functional clock.

0: Not enable

1: Enable

(R/W)

Register 6.48. PCR_AES_CONF_REG (0x00CC)

(reserved)																																PCR_AES_READY PCR_AES_RST_EN PCR_AES_CLK_EN		
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_AES_CLK_EN Configures whether or not to enable AES clock.

0: Not enable

1: Enable

(R/W)

PCR_AES_RST_EN Configures whether or not to reset AES.

0: Not reset

1: Reset

(R/W)

PCR_AES_READY Represents whether or not AES is released from reset.

0: Not released

1: Released

(RO)

Register 6.49. PCR_SHA_CONF_REG (0x00D0)

(reserved)																																PCR_SHA_READY PCR_SHA_RST_EN PCR_SHA_CLK_EN			
31																															3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_SHA_CLK_EN Configures whether or not to enable SHA clock.

0: Not enable

1: Enable

(R/W)

PCR_SHA_RST_EN Configures whether or not to reset SHA.

0: Not reset

1: Reset

(R/W)

PCR_SHA_READY Represents whether or not SHA is released from reset.

0: Not released

1: Released

(RO)

Register 6.50. PCR_RSA_CONF_REG (0x00D4)

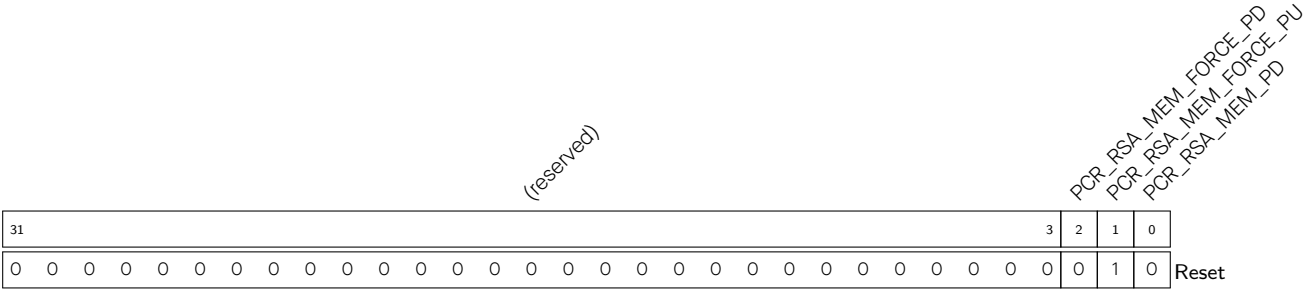
(reserved)																																PCR_RSA_READY PCR_RSA_RST_EN PCR_RSA_CLK_EN		
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_RSA_CLK_EN Configures whether or not to enable RSA clock.
0: Not enable
1: Enable
(R/W)

PCR_RSA_RST_EN Configures whether or not to reset RSA.
0: Not reset
1: Reset
(R/W)

PCR_RSA_READY Represents whether or not RSA is released from reset.
0: Not released
1: Released
(RO)

Register 6.51. PCR_RSA_PD_CTRL_REG (0x00D8)



PCR_RSA_MEM_PD Configures whether or not to power down RSA internal memory.
0: Not power down
1: Power down
(R/W)

PCR_RSA_MEM_FORCE_PU Configures whether or not to force power up RSA internal memory.
0: Not force power up
1: Force power up
(R/W)

PCR_RSA_MEM_FORCE_PD Configures whether or not to force power down RSA internal memory.
0: Not force power down
1: Force power down
(R/W)

Register 6.52. PCR_ECC_CONF_REG (0x00DC)

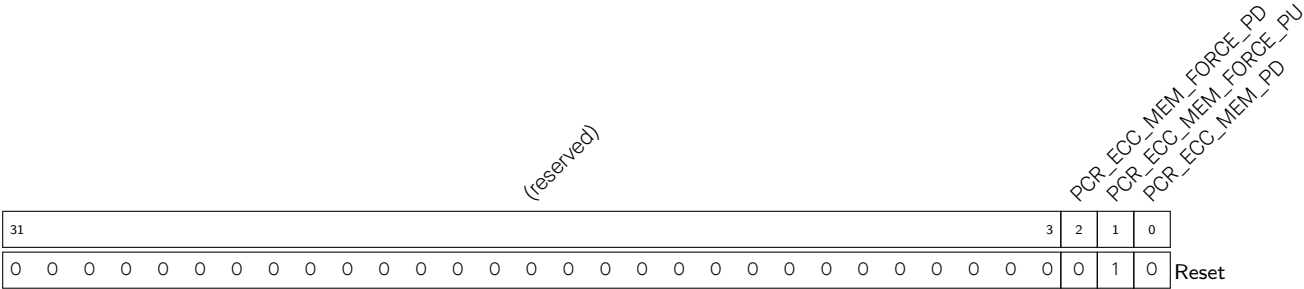
(reserved)																																PCR_ECC_READY PCR_ECC_RST_EN PCR_ECC_CLK_EN			
31																															3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1			

PCR_ECC_CLK_EN Configures whether or not to enable ECC clock.
0: Not enable
1: Enable
(R/W)

PCR_ECC_RST_EN Configures whether or not to reset ECC.
0: Not reset
1: Reset
(R/W)

PCR_ECC_READY Represents whether or not ECC is released from reset.
0: Not released
1: Released
(RO)

Register 6.53. PCR_ECC_PD_CTRL_REG (0x00E0)



PCR_ECC_MEM_PD Configures whether or not to power down ECC internal memory.

0: Not power down

1: Power down

(R/W)

PCR_ECC_MEM_FORCE_PU Configures whether or not to force power up ECC internal memory.

0: Not force power up

1: Force power up

(R/W)

PCR_ECC_MEM_FORCE_PD Configures whether or not to force power down ECC internal memory.

0: Not force power down

1: Force power down

(R/W)

Register 6.54. PCR_DS_CONF_REG (0x00E4)

(reserved)																																PCR_DS_READY PCR_DS_RST_EN PCR_DS_CLK_EN			
31																															3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_DS_CLK_EN Configures whether or not to enable DS clock.

0: Not enable

1: Enable

(R/W)

PCR_DS_RST_EN Configures whether or not to reset DS.

0: Not reset

1: Reset

(R/W)

PCR_DS_READY Represents whether or not DS is released from reset.

0: Not released

1: Released

(RO)

Register 6.55. PCR_HMAC_CONF_REG (0x00E8)

(reserved)																																PCR_HMAC_READY PCR_HMAC_RST_EN PCR_HMAC_OLK_EN		
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset	

PCR_HMAC_CLK_EN Configures whether or not to enable HMAC clock.

0: Not enable

1: Enable

(R/W)

PCR_HMAC_RST_EN Configures whether or not to reset HMAC.

0: Not reset

1: Reset

(R/W)

PCR_HMAC_READY Represents whether or not HMAC is released from reset.

0: Not released

1: Released

(RO)

Register 6.56. PCR_ECDSA_CONF_REG (0x00EC)

(reserved)																																PCR_ECDSA_READY PCR_ECDSA_RST_EN PCR_ECDSA_CLK_EN		
31																															3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_ECDSA_CLK_EN Configures whether or not to enable ECDSA clock.

0: Not enable

1: Enable

(R/W)

PCR_ECDSA_RST_EN Configures whether or not to reset ECDSA.

0: Not reset

1: Reset

(R/W)

PCR_ECDSA_READY Represents whether or not ECDSA is released from reset.

0: Not released

1: Released

(RO)

Register 6.57. PCR_IOMUX_CONF_REG (0x00F0)

(reserved)																															PCR_IOMUX_RST_EN PCR_IOMUX_CLK_EN			
31																															2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

PCR_IOMUX_CLK_EN Configures whether or not to enable APB_CLK for IO MUX.

0: Not enable

1: Enable

(R/W)

PCR_IOMUX_RST_EN Configures whether or not to reset IO MUX.

0: Not reset

1: Reset

(R/W)

Register 6.58. PCR_IOMUX_CLK_CONF_REG (0x00F4)

(reserved)										PCR_IOMUX_FUNC_CLK_EN PCR_IOMUX_FUNC_CLK_SEL										(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31									23	22	21	20	19															0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PCR_IOMUX_FUNC_CLK_SEL Configures the clock source of IO MUX.

- 0: XTAL_CLK
 - 1: RC_FAST_CLK
 - 2: PLL_F48M_CLK
 - 3: No clock source
- (R/W)

PCR_IOMUX_FUNC_CLK_EN Configures whether or not to enable IO MUX functional clock.

- 0: Not enable
 - 1: Enable
- (R/W)

Register 6.59. PCR_TRACE_CONF_REG (0x00FC)

(reserved)																															PCR_TRACE_RST_EN		PCR_TRACE_CLK_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31																															2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

PCR_TRACE_CLK_EN Configures whether or not to enable the clock of RISC-V Trace Encoder.

- 0: Not enable
 - 1: Enable
- (R/W)

PCR_TRACE_RST_EN Configures whether or not to reset RISC-V Trace Encoder.

- 0: Not reset
 - 1: Reset
- (R/W)

Register 6.60. PCR_ASSIST_CONF_REG (0x0100)

(reserved)																												PCR_ASSIST_RST_EN		PCR_ASSIST_CLK_EN		
31																											2	1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

Reset

PCR_ASSIST_CLK_EN Configures whether or not to enable Debug Assistant clock.

0: Not enable

1: Enable

(R/W)

PCR_ASSIST_RST_EN Configures whether or not to reset Debug Assistant.

0: Not reset

1: Reset

(R/W)

Register 6.61. PCR_CACHE_CONF_REG (0x0104)

(reserved)																															PCR_CACHE_RST_EN		PCR_CACHE_CLK_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
31																													2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

PCR_CACHE_CLK_EN Configures whether or not to enable Cache clock.

0: Not enable

1: Enable

(R/W)

PCR_CACHE_RST_EN Configures whether or not to reset Cache.

0: Not reset

1: Reset

(R/W)

Register 6.62. PCR_TIMEOUT_CONF_REG (0x010C)

(reserved)																															PCR_HP_TIMEOUT_RST_EN PCR_CPU_TIMEOUT_RST_EN (reserved)			
31																															3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															0	0	0	Reset

PCR_CPU_TIMEOUT_RST_EN Configures whether or not to reset CPU Peripheral Timeout Protection.

0: Not reset

1: Reset

(R/W)

PCR_HP_TIMEOUT_RST_EN Configures whether or not to reset HP Peripheral Timeout Protection.

0: Not reset

1: Reset

(R/W)

Register 6.63. PCR_SYSCCLK_CONF_REG (0x0110)

(reserved)		PCR_CLK_XTAL_FREQ												(reserved)		PCR_SOC_CLK_SEL												(reserved)											
31	30	24		23	18			17	16	15															0														
0	32			0	0	0	0	0	0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0															Reset												

PCR_SOC_CLK_SEL Configures to select clock source of HP_ROOT_CLK.

0: XTAL_CLK

1: RC_FAST_CLK

2: PLL_F160M_CLK

3: PLL_F240M_CLK □□

(R/W)

PCR_CLK_XTAL_FREQ Represents the frequency of XTAL.

Measurement unit: MHz.

(RO)

Register 6.64. PCR_CPU_WAITI_CONF_REG (0x0114)

(reserved)																								PCR_CPU_WAITI_DELAY_NUM				PCR_CPU_WAIT_MODE_FORCE_ON				(reserved)			
31																								8	7	4		3	2	0		Reset			
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0		1	0 0 0								

PCR_CPU_WAIT_MODE_FORCE_ON Configures whether or not to force on the gated CPU clock when CPU is in WFI (wait for interrupt) mode.

0: Not force enable

1: Force enable

Usually, after executing the WFI instruction, CPU enters the WFI mode, during which the gated CPU clock is turned off until any interrupts occur. In this way, power consumption is saved. If this bit is set, the gated CPU clock is always on and will not be turned off by the WFI instruction.

(R/W)

PCR_CPU_WAITI_DELAY_NUM Configures the number of delay cycles to turn off the CPU clock after the CPU enters the WFI mode because of WFI instruction.

Measurement unit: CPU_CLK cycles.

(R/W)

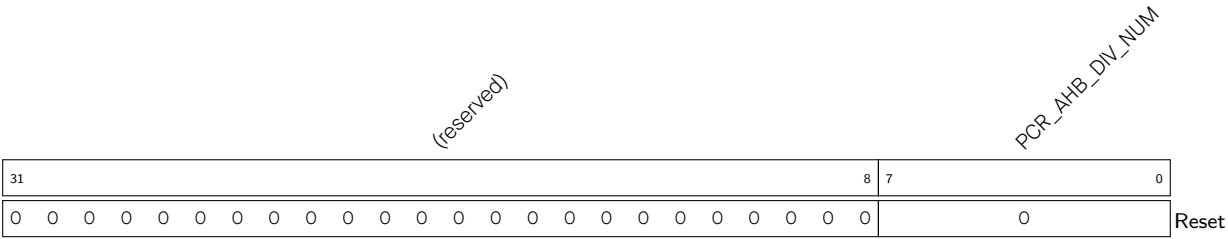
Register 6.65. PCR_CPU_FREQ_CONF_REG (0x0118)

(reserved)																								PCR_CPU_DIV_NUM									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0								Reset	

PCR_CPU_DIV_NUM Configures the divisor of HP_ROOT_CLK to generate CPU_CLK.

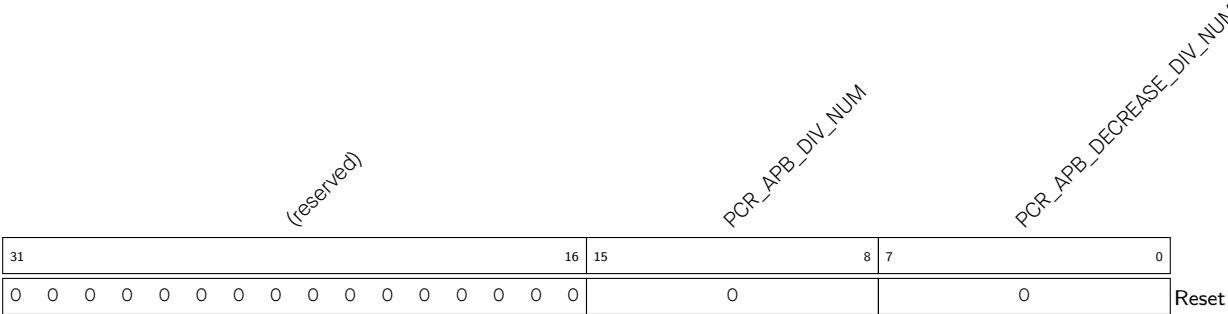
This field should be used together with [PCR_AHB_DIV_NUM](#).

Register 6.66. PCR_AHB_FREQ_CONF_REG (0x011C)



PCR_AHB_DIV_NUM Configures the divisor of HP_ROOT_CLK to generate AHB_CLK.
This field should be used together with [PCR_CPU_DIV_NUM](#).
(R/W)

Register 6.67. PCR_APB_FREQ_CONF_REG (0x0120)



PCR_APB_DECREASE_DIV_NUM Configures the divisor AHB_CLK to generate APB_CLK during the first division.
(R/W)

PCR_APB_DIV_NUM Configures the divisor of AHB_CLK to generate APB_CLK during the second division.
(R/W)

Register 6.68. PCR_PLL_DIV_CLK_EN_REG (0x0128)

(reserved)																															PCR_PLL_48M_CLK_EN						
																															reserved						
																															PCR_PLL_80M_CLK_EN						
																															PCR_PLL_120M_CLK_EN						
																															PCR_PLL_160M_CLK_EN						
																															PCR_PLL_240M_CLK_EN						
31																								6	5	4	3	2	1	0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	Reset				

PCR_PLL_240M_CLK_EN Configures whether or not to enable PLL_F240M_CLK.

0: Not enable

1 (default): Enable

(R/W)

PCR_PLL_160M_CLK_EN Configures whether or not to enable PLL_F160M_CLK.

0: Not enable

1 (default): Enable

(R/W)

PCR_PLL_120M_CLK_EN Configures whether or not to enable PCR_PLL_120M_CLK.

0: Not enable

1 (default): Enable

(R/W)

PCR_PLL_80M_CLK_EN Configures whether or not to enable PCR_PLL_80M_CLK.

0: Not enable

1 (default): Enable

(R/W)

PCR_PLL_48M_CLK_EN Configures whether or not to enable PCR_PLL_48M_CLK.

0: Not enable

1 (default): Enable

(R/W)

Register 6.69. PCR_CTRL_32K_CONF_REG (0x0130)

(reserved)																PCR_FOSC_TICK_NUM								(reserved)				PCR_32K_SEL																																	
31																16																15								8								7				3				2		0			
0																0																0								0								0				0				0		0		Reset	

Reset

PCR_32K_SEL Configures the 32 kHz clock for TIMER_GROUP.

- 0: XTAL32K_CLK
- 1: OSC_SLOW_CLK
- 2: RC_SLOW_CLK
- 2: RC_FAST_CLK
- (R/W)

PCR_FOSC_TICK_NUM Configure the division factor for the RC_FAST_CLK to enter the calibration module. (Effective when PCR_32K_SEL is set to 4).

(R/W)

Register 6.70. PCR_SEC_CONF_REG (0x013C)

(reserved)																												PCR_SEC_RST_EN				PCR_SEC_CLK_SEL																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31																												3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

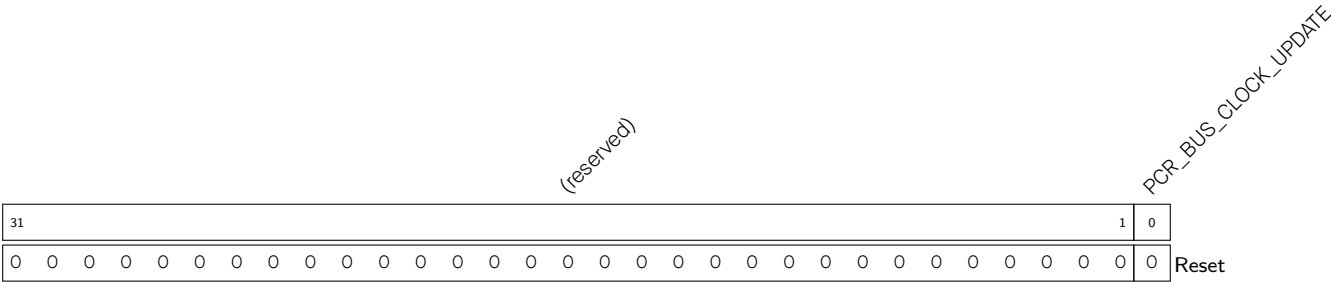
PCR_SEC_CLK_SEL Configures the clock source for the External Memory Encryption and Decryption module.

- 0 (default): XTAL_CLK
- 1: RC_FAST_CLK
- 2: PLL_F480M_CLK
- 3: No clock source
- (R/W)

PCR_SEC_RST_EN Configures whether or not to reset the SEC.

- 0: Not reset
- 1: Reset
- (R/W)

Register 6.71. PCR_BUS_CLK_UPDATE_REG (0x0144)



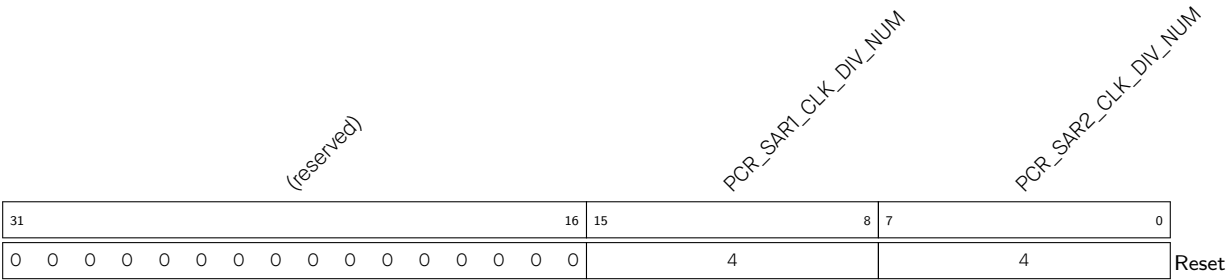
PCR_BUS_CLOCK_UPDATE Configures whether or not to update configurations for CPU_CLK division, AHB_CLK division and HP_ROOT_CLK clock source selection.

0: Not update configurations

1: Update configurations

This bit is automatically cleared when configurations have been updated. (R/W/WTC)

Register 6.72. PCR_SAR_CLK_DIV_REG (0x0148)



PCR_SAR2_CLK_DIV_NUM Configures the divisor for SAR ADC2 clock to generate ADC analog control signals. (R/W)

PCR_SAR1_CLK_DIV_NUM Configures the divisor for SAR ADC clock to generate ADC analog control signals. (R/W)

Register 6.73. PCR_BS_CONF_REG (0x0150)

(reserved)																												PCR_BS_RST_EN PCR_BS_CLK_EN	
31																											2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

PCR_BS_CLK_EN Configures whether or not to enable the clock of the Bit Scrambler.

0: Not enable

1: Enable

(R/W)

PCR_BS_RST_EN Configures whether or not to reset the Bit Scrambler. 0: Not reset

1: Reset

(R/W)

Register 6.74. PCR_BS_FUNC_CONF_REG (0x0154)

(reserved)										PCR_BS_RX_RST_EN PCR_BS_TX_RST_EN										(reserved)									
31											25	24	23	22															0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

PCR_BS_TX_RST_EN Configures whether or not to reset the transmit side of Bit Scrambler.

0: Not reset

1: Reset

(R/W)

PCR_BS_RX_RST_EN Configures whether or not to reset the receive side of Bit Scrambler.

0: Not reset

1: Reset

(R/W)

Register 6.75. PCR_BS_PD_CTRL_REG (0x0158)

(reserved)																												PCR_BS_MEM_FORCE_PD PCR_BS_MEM_FORCE_PU (reserved)			
31																												3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												1	0	0	Reset

PCR_BS_MEM_FORCE_PU Configures whether or not to force power up Bit Scrambler memory.

0: Not force power up

1: Force power up

(R/W)

PCR_BS_MEM_FORCE_PD Configures whether or not to force power down Bit Scrambler memory.

0: Not force power down

1: Force power down

(R/W)

Register 6.76. PCR_TIMERGROUP_WDT_CONF_REG (0x015C)

(reserved)																															PCR_TG1_WDT_RST_EN PCR_TGO_WDT_RST_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

PCR_TGO_WDT_RST_EN Configures whether or not to reset the watchdog of Timer Group 0.

0: Not reset

1: Reset

(R/W)

PCR_TG1_WDT_RST_EN Configures whether or not to reset the watchdog of Timer Group 1.

0: Not reset

1: Reset

(R/W)

Register 6.77. PCR_TIMERGROUP_XTAL_CONF_REG (0x0160)

(reserved)																															3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	Reset

PCR_TGO_XTAL_RST_EN Configures whether to reset the clock calibration function of the Timer Group 0.

0: Not reset

1: Reset

(R/W)

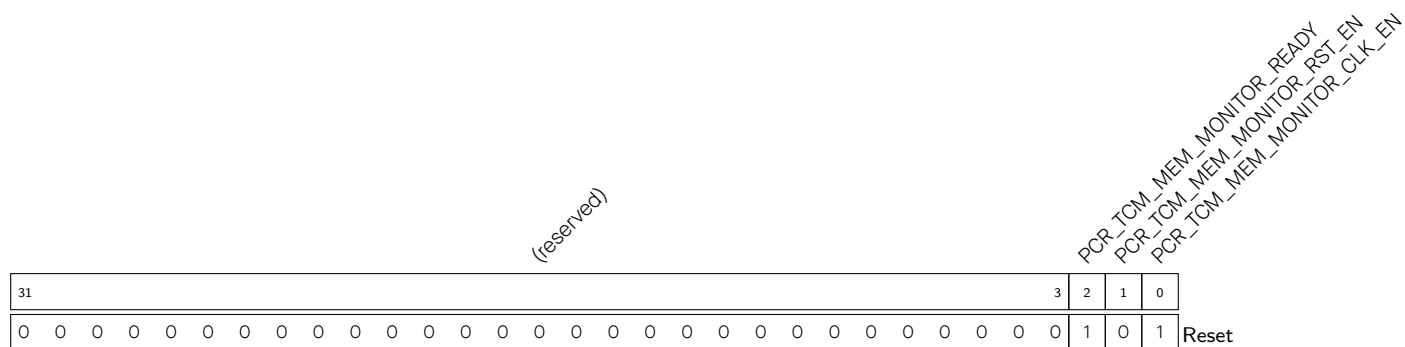
PCR_TGO_XTAL_CLK_EN Configures whether to enable the clock calibration function of the Timer Group 0.

0: Not enable

1: Enable

(R/W)

Register 6.78. PCR_TCM_MEM_MONITOR_CONF_REG (0x016C)



PCR_TCM_MEM_MONITOR_CLK_EN Configures whether to enable TCM_MEM_MONITOR_CLK.

0: Not enable

1: Enable

(R/W)

PCR_TCM_MEM_MONITOR_RST_EN Configures whether to reset the memory monitor.

0: Not reset

1: Reset

(R/W)

PCR_TCM_MEM_MONITOR_READY Represents whether or not memory monitor is released from reset.

0: Not released

1: Released

(RO)

Register 6.79. PCR_PSRAM_MEM_MONITOR_CONF_REG (0x0170)

(reserved)																												PCR_PSRAM_MEM_MONITOR_READY PCR_PSRAM_MEM_MONITOR_RST_EN PCR_PSRAM_MEM_MONITOR_CLK_EN			
31																									3	2	1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	Reset

PCR_PSRAM_MEM_MONITOR_CLK_EN Configures whether to enable the PSRAM_MEM_MONITOR_CLK.
0: Not enable
1: Enable
(R/W)

PCR_PSRAM_MEM_MONITOR_RST_EN Configures whether to reset the memory monitor of PSRAM.
0: Not reset
1: Reset
(R/W)

PCR_PSRAM_MEM_MONITOR_READY Represents whether or not memory monitor of PSRAM is released from reset.
0: Not released
1: Released
(RO)

Register 6.80. PCR_HPCORE_O_PD_CTRL_REG (0x0178)

(reserved)																												PCR_HPCORE_O_MEM_FORCE_PD PCR_HPCORE_O_MEM_FORCE_PU (reserved)				
31																											3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset

Reset

PCR_HPCORE_O_MEM_FORCE_PU Configures whether or not to force power up HP CORE0 memory.

0: Not force power up

1: Force power up

(R/W)

PCR_HPCORE_O_MEM_FORCE_PD Configures whether or not to force power down Key Manager memory.

0: Not force power down

1: Force power down

(R/W)

Register 6.81. PCR_SDIO_SLAVE_CONF_REG (0x017C)

(reserved)																															PCR_SDIO_SLAVE_RST_EN PCR_SDIO_SLAVE_CLK_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																											2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

PCR_SDIO_SLAVE_RST_EN Configures whether to reset the SDIO slave module.

0: Not reset

1: Reset

(R/W)

PCR_SDIO_SLAVE_CLK_EN Configures whether to enable the SDIO_SLAVE_CLK.

0: Not enable

1: Enable

(R/W)

Register 6.82. PCR_SYSCLK_FREQ_QUERY_0_REG (0x0120)

(reserved)																PCR_PLL_FREQ																PCR_FOSC_FREQ																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																18																17																8																7																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0															

PCR_FOSC_FREQ Represents the frequency of RC_FAST_CLK.
Measurement unit: MHz.
(HRO)

PCR_PLL_FREQ Represents the frequency of PLL_F480M_CLK .
Measurement unit: MHz.
(HRO)

Register 6.83. PCR_DATE_REG (0x0FFC)

(reserved)				PCR_DATE																							
31				28	27																					0	
0	0	0	0	0x2210080																							Reset

PCR_DATE Version control register. (R/W)

6.5.2 LP System Clock Registers

The addresses of the last two registers with the LPPERI prefix in this section are relative to the Low-power Peripheral Register (LPPERI) base address. The addresses of the third and fourth to last registers with the LP_AON prefix in this section are relative to the Low-power Always-on Register (LP_AON) base address.The other addresses in this section are relative to the Low-power Clock/Reset Register (LP_CLKRST) base address. For base address, please refer to Table 4.3-2 in Chapter 4 *System and Memory*.

Register 6.84. LP_CLKRST_LP_CLK_CONF_REG (0x0000)

(reserved)																				LP_CLKRST_FAST_CLK_SEL				LP_CLKRST_SLOW_CLK_SEL			
31																				4	3	2	1	0	Reset		
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																				1	0						

LP_CLKRST_SLOW_CLK_SEL Configures the source of LP_SLOW_CLK.

- 0: RC_SLOW_CLK
- 1: XTAL32K_CLK
- 2: OSC_SLOW_CLK
- 3: Invalid. No effect

(R/W)

LP_CLKRST_FAST_CLK_SEL Configures the source of LP_FAST_CLK.

- 0: RC_FAST_CLK
- 1: XTAL_D2_CLK

(R/W)

Register 6.85. LP_CLKRST_LP_CLK_PO_EN_REG (0x0004)

(reserved)																						LP_CLKRST_LPBUS_OEN LP_CLKRST_RING_OEN LP_CLKRST_FAST_OEN LP_CLKRST_SLOW_OEN LP_CLKRST_CORE_OEN (reserved) LP_CLKRST_FOSC_OEN LP_CLKRST_SOSC_OEN LP_CLKRST_AON_FAST_OEN LP_CLKRST_AON_SLOW_OEN													
31																						11		10	9	8	7	6	5	4	3	2	1	0	Reset
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																						1	1	1	1	1	1	1	1	1	1	1	1		

LP_CLKRST_AON_SLOW_OEN Configures whether to gate the LP_DYN_SLOW_CLK signals to pad.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_AON_FAST_OEN Configures whether to gate the LP_DYN_FAST_CLK signals to pad.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_SOSC_OEN Configures whether to gate the OSC_SLOW_CLK signals to pad.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_FOSC_OEN Configures whether to gate the RC_FAST_CLK signals to pad.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_XTAL32K_OEN Configures whether to gate the XTAL32K_CLK signals to pad.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

Continued on the next page...

Register 6.85. LP_CLKRST_LP_CLK_PO_EN_REG (0x0004)

Continued from the previous page...

LP_CLKRST_CORE_EFUSE_OEN Configures whether to gate the EFUSE_CTRL clock.

- 0: Disable the clock gate
 - 1: Enable the clock gate
- (R/W)

LP_CLKRST_SLOW_OEN Configures whether to gate the LP_SLOW_CLK signals to pad.

- 0: Disable the clock gate
 - 1: Enable the clock gate
- (R/W)

LP_CLKRST_FAST_OEN Configures whether to gate the LP_FAST_CLK signals to pad.

- 0: Disable the clock gate
 - 1: Enable the clock gate
- (R/W)

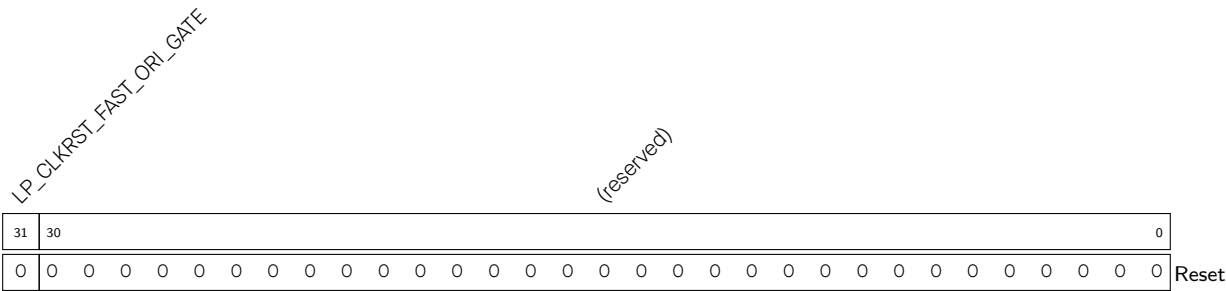
LP_CLKRST_RNG_OEN Configures whether to gate the RNG clock signals to pad.

- 0: Disable the clock gate
 - 1: Enable the clock gate
- (R/W)

LP_CLKRST_LPBUS_OEN Configures whether to gate the LP bus clock signals to pad.

- 0: Disable the clock gate
 - 1: Enable the clock gate
- (R/W)

Register 6.86. LP_CLKRST_LP_CLK_EN_REG (0x0008)



LP_CLKRST_FAST_ORI_GATE Configures the clock gate to LP_FAST_CLK.

- 0: Invalid. The clock gate controlled by hardware FSM
 - 1: Force the clock to pass the clock gate
- (R/W)

Register 6.87. LP_CLKRST_LP_RST_EN_REG (0x000C)

LP_CLKRST_ANA_PERI_RESET_EN				(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31	30	29	28	27	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_CLKRST_AON_EFUSE_CORE_RESET_EN Configures whether to reset the always-on part of EFUSE_CTRL.
0: Invalid. No effect
1: Reset
(R/W)

LP_CLKRST_LP_TIMER_RESET_EN Configures whether to reset RTC timer.
0: Invalid. No effect
1: Reset
(R/W)

LP_CLKRST_WDT_RESET_EN Configures whether to reset RWDT and Super Watchdog Timer.
0: Invalid. No effect
1: Reset
(R/W)

LP_CLKRST_ANA_PERI_RESET_EN Configures whether to reset analog peripherals, including the Brownout Detector.
0: Invalid. No effect
1: Reset
(R/W)

Register 6.88. LP_CLKRST_RESET_CAUSE_REG (0x0010)

LP_CLKRST_COREO_RESET_FLAG_CLR				(reserved)																	LP_CLKRST_COREO_RESET_FLAG				LP_CLKRST_RESET_CAUSE			
31	30	29	28																			6	5	4				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0				Reset

- LP_CLKRST_RESET_CAUSE Represents the reset cause. (RO)
- LP_CLKRST_COREO_RESET_FLAG Represents whether the reset occurred is recorded in LP_CLKRST_RESET_CAUSE.
0: Illegal reset
1: Recorded reset
(RO)
- LP_CLKRST_COREO_RESET_CAUSE_CLR Write 1 to clear LP_CLKRST_RESET_CAUSE. (WT)
- LP_CLKRST_COREO_RESET_FLAG_SET Write 1 to set LP_CLKRST_COREO_RESET_FLAG. (WT)
- LP_CLKRST_COREO_RESET_FLAG_CLR Write 1 to clear LP_CLKRST_COREO_RESET_FLAG. (WT)

Register 6.89. LP_CLKRST_CPU_RESET_REG (0x0014)

[illegible]

LP_CLKRST_RTC_WDT_CPU_RESET_LENGTH Configures the length of RWDT CPU reset.

Measurement unit: LP_DYN_FAST_CLK clock cycles.

(R/W)

LP_CLKRST_RTC_WDT_CPU_RESET_EN Configures whether or not to enable RWDT CPU reset.

0: Enable RWDT CPU reset.

1: Disable RWDT CPU reset.

(R/W)

LP_CLKRST_CPU_STALL_WAIT Configure the time interval between CPU stall and reset.

Measurement unit: LP_DYN_FAST_CLK clock cycles.

(R/W)

LP_CLKRST_CPU_STALL_EN Configures whether or not CPU will stall before RWDT and software reset CPU.

0: CPU will not stall.

1: CPU will stall.

(R/W)

Register 6.90. LP_CLKRST_FOSC_CNTL_REG (0x0018)

Register diagram for LP_CLKRST_FOSC_DFREQ:

- Bit 31**: Reset
- Bits 22-21**: Reserved
- Bits 0-20**: OXAC (Oscillator Amplitude Control)

31	22	21	0
			Reset
			(reserved)
			Oxac

LP_CLKRST_FOSC_DFREQ Configures the frequency of RC_FAST_CLK frequency. (R/W)

Register 6.91. LP_CLKRST_CLK_TO_HP_REG (0x0020)

LP_CLKRST_ICG_HP_FOSC (reserved))																															LP_CLKRST_ICG_HP_SOSC (reserved)															LP_CLKRST_ICG_HP_XTAL32K (reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
31	30	29	28	27	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_CLKRST_ICG_HP_XTAL32K Configures whether to gate the XTAL32K_CLK signals to HP system.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_ICG_HP_SOSC Configures whether to gate the RC_SLOW_CLK signals to HP system.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LP_CLKRST_ICG_HP_FOSC Configures whether to gate the RC_FAST_CLK signals to HP system.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

Register 6.92. LP_CLKRST_LPMEM_FORCE_REG (0x0024)

LP_CLKRST_LPMEM_CLK_FORCE_ON																																	(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31	30																															0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

LP_CLKRST_LPMEM_CLK_FORCE_ON Configures whether to force enable the gate of LP Memory.

0: Invalid. The clock gate controlled by hardware FSM

1: Force open clock gate of LP Memory

(R/W)

Register 6.93. LP_CLKRST_LPPERI_REG (0x0028)

(reserved)																LP_CLKRST_LP_SEL_XTAL32K																LP_CLKRST_LP_SEL_XTAL																LP_CLKRST_LP_SEL_OSC_FAST																LP_CLKRST_LP_SEL_OSC_SLOW																(reserved))																LP_CLKRST_LP_BLETIMER_DIV_NUM																(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31	30	29	28	27	26	25	24	23															12	11															0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	1	0	0	0	0	0	0	0	0														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_CLKRST_LP_BLETIMER_DIV_NUM Configures the divisor of the clock divider for RTC BLE Timer.

The working clock frequency of RTC BLE Timer is equal to the source clock frequency divided by $((LP\_CLKRST\_LP\_BLETIMER\_DIV\_NUM - 1) / 2)$. (R/W)

LP_CLKRST_LP_SEL_OSC_SLOW Configures the clock source for RTC BLE Timer.

1: Selects RC_SLOW_CLK as the source clock for RTC BLE Timer.

0: Disables RC_SLOW_CLK as the source clock for RTC BLE Timer. (R/W)

LP_CLKRST_LP_SEL_OSC_FAST Configures the clock source for RTC BLE Timer.

1: Selects RC_FAST_CLK as the source clock for RTC BLE Timer.

0: Disables RC_FAST_CLK as the source clock for RTC BLE Timer. (R/W)

LP_CLKRST_LP_SEL_XTAL Configures the clock source for RTC BLE Timer.

1: Selects XTAL_CLK as the source clock for RTC BLE Timer.

0: Disables XTAL_CLK as the source clock for RTC BLE Timer. (R/W)

LP_CLKRST_LP_SEL_XTAL32K Configures the clock source for RTC BLE Timer.

1: Selects XTAL32K_CLK as the source clock for RTC BLE Timer.

0: Disables XTAL32K_CLK as the source clock for RTC BLE Timer. (R/W)

Register 6.94. LP_CLKRST_XTAL32K_REG (0x002C)

LP_CLKRST_DAC_XTAL32K																															LP_CLKRST_DBUF_XTAL32K																															LP_CLKRST_DGM_XTAL32K																															LP_CLKRST_DRES_XTAL32K																															(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31		29		28		27		25		24		22		21																																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
3				0				3				3				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0				0			

LP_CLKRST_DRES_XTAL32K Configures DRES (R/W)

LP_CLKRST_DGM_XTAL32K Configures DGM (R/W)

LP_CLKRST_DBUF_XTAL32K Configures DBUF (R/W)

LP_CLKRST_DAC_XTAL32K Configures DAC (R/W)

Register 6.95. LP_CLKRST_DATE_REG (0x03FC)

31	30	0
0	0x2207280	

Reset

LP_CLKRST_CLKRST_DATE Version control register. (R/W)

LP_CLKRST_CLK_EN Configures whether to force enable the clock gate for header registers.

0: Invalid

1: Force enable

(R/W)

Register 6.96. LP_AON_SYS_CFG_REG (0x0034)

Diagram of the LP_AON_HPSYS_SW_RESET register structure:

- Bit 31: LP_AON_HPSYS_SW_RESET
- Bits 30-0: (reserved)

LP_AON_HPSYS_SW_RESET Configures whether to do software reset of the system.

0: Not reset

1: Reset

(WT)

Register 6.97. LP_AON_CPUCORE0_CFG_REG (0x0038)

(reserved)																													LP_AON_CPU_CORE0_SW_RESET																															(reserved)			0
31	29	28	27																																			0																									
0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset																													

LP_AON_CPU_CORE0_SW_RESET Configures whether to do software reset of the CPU.

0: Not reset

1: Reset

(WT)

Register 6.98. LPPERI_CLK_EN_REG (0x0000)

(reserved)																															LPPERI_EFUSE_CK_EN												(reserved)																						0
31	30	29																							25	24	23																																						0
0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset																													

LPPERI_RNG_CK_EN Configures whether to gate the clock signals of RNG module.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

LPPERI_EFUSE_CK_EN Configures whether to gate the clock signals of eFuse Controller.

0: Disable the clock gate

1: Enable the clock gate

(R/W)

Register 6.99. LPPERI_RESET_EN_REG (0x0004)

[illegible]

LPPEI_RNG_RESET_EN Configures whether to do software reset of the RNG Controller.

0: Not reset

1: Reset

(R/W)

LPPERI_EFUSE_RESET_EN Configures whether to do software reset of the eFuse Controller.

0: Not reset

1: Reset

(R/W)

Chapter 7

Chip Boot Control

7.1 Overview

The chip allows for configuring the following boot parameters through strapping pins and eFuse parameters at power-up or a hardware reset, without microcontroller interaction.

- **Chip boot mode**
 - Strapping pins: GPIO26, GPIO27, and GPIO28
- **SDIO sampling and driving clock edge**
 - Strapping pins: GPIO25 and MTDI
- **ROM messages printing**
 - Strapping pin: GPIO27
 - eFuse parameters: EFUSE_UART_PRINT_CONTROL and EFUSE_DIS_USB_SERIAL_JTAG_ROM_PRINT
- **JTAG signal source**
 - Strapping pin: GPIO7
 - eFuse parameters: EFUSE_DIS_PAD_JTAG, EFUSE_DIS_USB_JTAG, and EFUSE_JTAG_SEL_ENABLE
- **Crystal frequency selection**
 - Strapping pin: MTMS (effective only in Joint Download Boot mode)
 - eFuse parameters: EFUSE_XTAL_48M_SEL and EFUSE_XTAL_48M_SEL_MODE

The default values of all the above eFuse parameters are 0, which means that they are not burnt. Given that eFuse is one-time programmable, once an eFuse bit is programmed to 1, it can never be reverted to 0. For how to program eFuse bits, please refer to Chapter [2 eFuse Controller \(eFuse\) \[to be added later\]](#).

During Chip Reset (see Chapter [6 Reset and Clock](#)), hardware captures samples and stores the voltage level of strapping pins as strapping bit of “0” or “1” in latches, and holds these bits until the chip is powered down or next chip reset. Software can read the latch status (strapping value) from [GPIO_STRAPPING](#).

7.2 Functional Description

This section introduces chip reset functions and the patterns of the strapping pins and eFuse values to invoke each function.

Notice: Only documented patterns should be used. If an undocumented pattern is used, it may trigger unexpected behaviors.

7.2.1 Default Configuration

The default values of the strapping pins, namely the logic levels, are determined by pins' internal weak pull-up/pull-down resistors at reset if the pins are not connected to any circuit, or connected to an external high-impedance circuit.

Table 7.2-1. Default Configuration of Strapping Pins

Strapping Pin	Default Configuration	Bit Value
GPIO25	Floating	–
GPIO26	Floating	–
GPIO27	Pull-up	1
GPIO28	Pull-up	1
GPIO7	Floating	–
MTMS	Floating	–
MTDI	Floating	–

To change the strapping bit values, users can apply external pull-down/pull-up resistors, or use host MCU GPIOs to control the voltage level of these pins when powering up ESP32-C5. After the reset is released, the strapping pins work as normal-function pins.

7.2.2 Chip Boot Mode Control

GPIO26, GPIO27 and GPIO28 control the boot mode after the reset is released. See Table 7.2-2 [Boot Mode Control](#).

Table 7.2-2. Boot Mode Control

Boot Mode	GPIO26	GPIO27	GPIO28
SPI Boot ¹	Any value	Any value	1 ¹
Joint Download Boot 0 ²	Any value	1	0
Joint Download Boot 1 ³	0	0	0

¹ **Bold** marks the default value and configuration.

² Joint Download Boot 0 mode supports the following download methods:

- USB-Serial-JTAG Download Boot
- UART Download Boot

³ Joint Download Boot 1 mode supports the following download methods:

- UART Download Boot
- SDIO Download Boot

In SPI Boot mode, the ROM bootloader loads and executes the program from SPI flash to boot the

system.

In Joint Download Boot 0 mode, users can download binary files into flash using UART0, USB, or SPI Slave interfaces. It is also possible to download binary files into SRAM and execute it from SRAM.

In Joint Download Boot 1 mode, users can download binary files into flash using UART0 or SDIO interfaces. It is also possible to download binary files into SRAM and execute it from SRAM.

Figure 7.2-1 *Chip Boot Flow* shows the detailed boot flow of the chip.

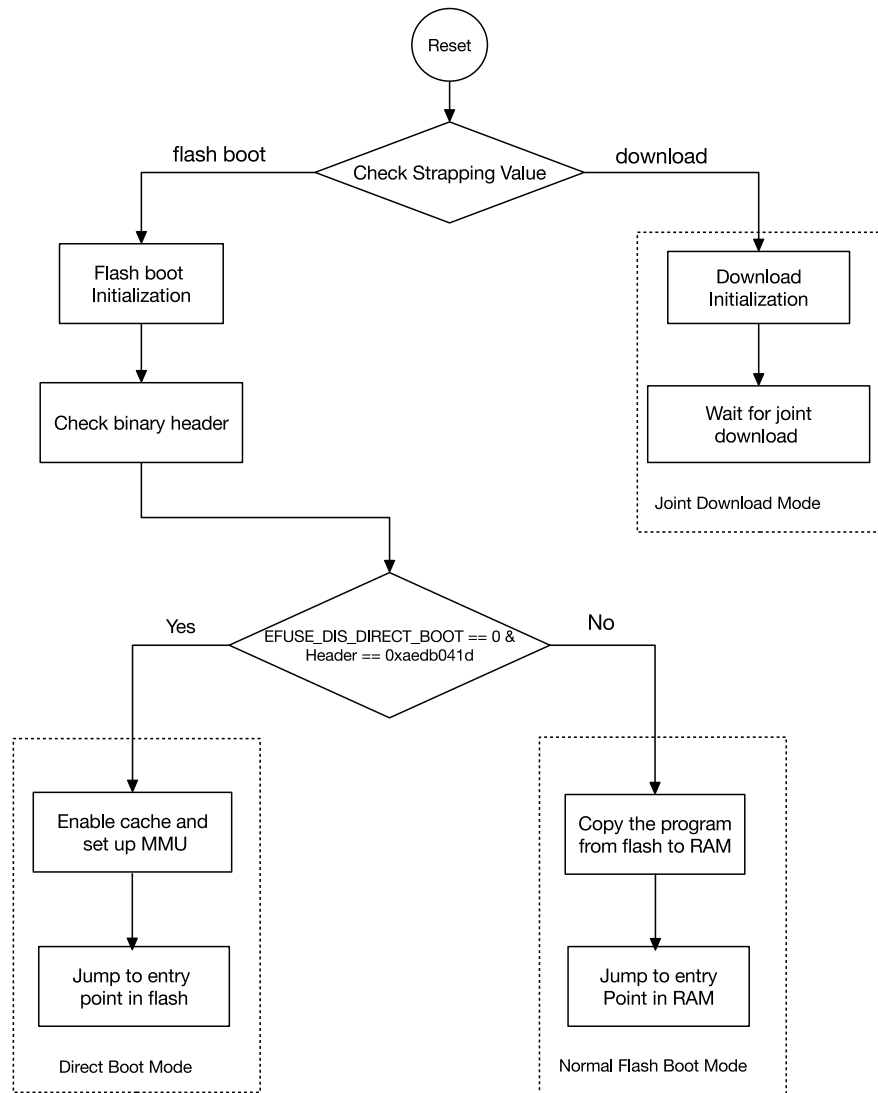


Figure 7.2-1. Chip Boot Flow

The following eFuse parameters control boot mode behaviors:

- **EFUSE_DIS_FORCE_DOWNLOAD**

- If this eFuse is 0 (default), software can force switch the chip from SPI Boot mode to Joint Download Boot mode by setting register LP_AON_FORCE_DOWNLOAD_BOOT and triggering a CPU reset. In this case, hardware overwrites [GPIO_STRAPPING\[4:3\]](#) from "1x" to "01".

- If this eFuse is 1, LP_AON_FORCE_DOWNLOAD_BOOT is disabled, and [GPIO_STRAPPING](#) can not be overwritten.

- **EFUSE_DIS_DOWNLOAD_MODE**

If this eFuse is 1, Joint Download Boot mode is disabled, and [GPIO_STRAPPING](#) will not be overwritten by LP_AON_FORCE_DOWNLOAD_BOOT.

- **EFUSE_ENABLE_SECURITY_DOWNLOAD**

If this eFuse is 1, Joint Download Boot mode only allows reading, writing, and erasing plaintext flash and does not support any SRAM or register operations. Ignore this eFuse if Joint Download Boot mode is disabled.

- **EFUSE_DIS_DIRECT_BOOT**

If this eFuse is 1, Direct Boot mode is disabled.

USB Serial/JTAG Controller can also force switch the chip to Joint Download Boot mode from SPI Boot mode, and vice versa. For detailed information, please refer to Chapter [28 USB Serial/JTAG Controller](#).

7.2.3 SDIO Sampling and Driving Clock Edge Control

The strapping pins GPIO25 and MTDI can be used to decide on which clock edge to sample signals and drive output lines. See Table [7.2-3 SDIO Input Sampling Edge/Output Driving Edge Control](#).

Table 7.2-3. SDIO Input Sampling Edge/Output Driving Edge Control

Edge behavior	GPIO25	MTDI
Falling edge sampling, falling edge output	0	0
Falling edge sampling, rising edge output	0	1
Rising edge sampling, falling edge output	1	0
Rising edge sampling, rising edge output	1	1

¹ GPIO25 and MTDI are floating by default, so above are not default configurations.

7.2.4 ROM Messages Printing Control

During the boot process, the messages by the ROM code can be printed to:

- (Default) **UART0** and **USB Serial/JTAG controller**
- **UART0**
- **USB Serial/JTAG controller**

To print ROM messages to **UART0** or **USB Serial/JTAG controller**, see the description below.

EFUSE_UART_PRINT_CONTROL and GPIO27 control printing ROM messages to **UART0** as shown in Table [7.2-4 UART0 ROM Message Printing Control](#).

Table 7.2-4. UART0 ROM Message Printing Control

UART0 ROM Message Printing	Register ²	eFuse ³	GPIO27
ROM messages are always printed to UART0 during boot	0	0 (0b00)	x ⁴
Print is enabled during boot		1 (0b01)	0
Print is disabled during boot			1
Print is disabled during boot		2 (0b10)	0
Print is enabled during boot			1
Print is disabled during boot		3 (0b11)	x
Print is disabled during boot	1	x	x

¹ **Bold** marks the default value and configuration.

² Register: LP_AON_STORE4_REG[0]

³ eFuse: EFUSE_UART_PRINT_CONTROL

⁴ x: x indicates that the value has no effect on the result and can be ignored.

EFUSE_DIS_USB_SERIAL_JTAG_ROM_PRINT controls the printing to **USB Serial/JTAG controller** as shown in Table 7.2-5 *USB Serial/JTAG ROM Message Printing Control*.

Table 7.2-5. USB Serial/JTAG ROM Message Printing Control

USB Serial/JTAG ROM Message Printing	EFUSE_DIS_USB_SERIAL_JTAG_ROM_PRINT
Enabled	0
Disabled	1
	Ignored

¹ **Bold** marks the default value and configuration.

7.2.5 JTAG Signal Source Control

The strapping pin GPIO7 can be used to control the JTAG signal source during the early boot process. This pin does not have internal pull-up or pull-down resistors, so its strapping value must be controlled by an external circuit that must not be in a high-impedance state.

GPIO7 controls the source of JTAG signals together with EFUSE_DIS_PAD_JTAG, EFUSE_DIS_USB_JTAG and EFUSE_JTAG_SEL_ENABLE. See Table 7.2-6 *JTAG Signal Source Control*.

Table 7.2-6. JTAG Signal Source Control

JTAG Signal Source	eFuse 1 ²	eFuse 2 ³	eFuse 3 ⁴	GPIO7
USB Serial/JTAG Controller ⁶	0	0	0	x ⁵
			1	1
JTAG pins MTDI, MTCK, MTMS, and MTDO		x	x	0
	1	1		x
USB Serial/JTAG Controller ⁶		0		
JTAG is disabled		x		
		1		

¹ **Bold** marks the default value and configuration.

² eFuse 1: EFUSE_DIS_PAD_JTAG

³ eFuse 2: EFUSE_DIS_USB_JTAG

⁴ eFuse 3: EFUSE_JTAG_SEL_ENABLE

⁵ x: x indicates that the value has no effect on the result and can be ignored.

⁶ In Joint Download Boot 1 mode, the USB Serial/JTAG controller is forcibly disabled, and the JTAG signal only comes from JTAG pins. If PAD_JTAG is also disabled, then JTAG is disabled.

7.2.6 Crystal Frequency Selection

ESP32-C5 supports crystal frequencies of 40 MHz and 48 MHz.

Chip Boot mode together with MTMS, EFUSE_XTAL_48M_SEL, and EFUSE_XTAL_48M_SEL_MODE collectively control the crystal frequency.

Table 7.2-7. Crystal Frequency Selection

Crystal Frequency	Boot Mode	eFuse 1 ²	eFuse 2 ³	MTMS
40 MHz	SPI Boot	x ⁴	Even number of 1s	x
48 MHz			Odd number of 1s	
40 MHz	Joint Download Boot 0/1	1	Even number of 1s	
48 MHz			Odd number of 1s	
40 MHz		0	x	0 ⁵
48 MHz				1 ⁵

¹ **Bold** marks the default value and configuration.

² eFuse 1: EFUSE_XTAL_48M_SEL_MODE

³ eFuse 2: EFUSE_XTAL_48M_SEL

⁴ x: x indicates that the value has no effect on the result and can be ignored.

⁵ In Joint Download Boot 0/1 mode, users need to set the signal level on MTMS strapping pin.

Chapter 8

Event Task Matrix (ETM)

8.1 Overview

The Event Task Matrix (ETM) peripheral contains 50 configurable channels. Each channel can map an event of any specified peripheral to a task of any specified peripheral. In this way, peripherals can be triggered to execute specified tasks without CPU intervention.

8.2 Features

The Event Task Matrix has the following features:

- Receive various events from multiple peripherals
- Generate various tasks for multiple peripherals
- 50 independently configurable ETM channels
- An ETM channel can be set up to receive any event, and map it to any task
- Each ETM channel can be enabled or disabled independently. If disabled, the channel will not respond to the configured event and generate the task mapped to that event
- Support for checking event and task status
- Peripherals supporting ETM include GPIO, LED PWM, general-purpose timers, RTC Timer, system timer, MCPWM, temperature sensor, ADC, I2S, LP CPU, GDMA, and PMU

Note that the 50 ETM channels are identical regarding their features and operations. Thus, in the following sections, ETM channels are collectively referred to as channel *n* (where *n* ranges from 0 to 49).

8.3 Functional Description

8.3.1 Architecture

The Event Task Matrix has 50 independent channels. Each channel can select any event as input, and map the event to any task as output (see Section 8.3.2 and Section 8.3.3 respectively). Each channel has an individual enable bit (see Section 8.3.6).

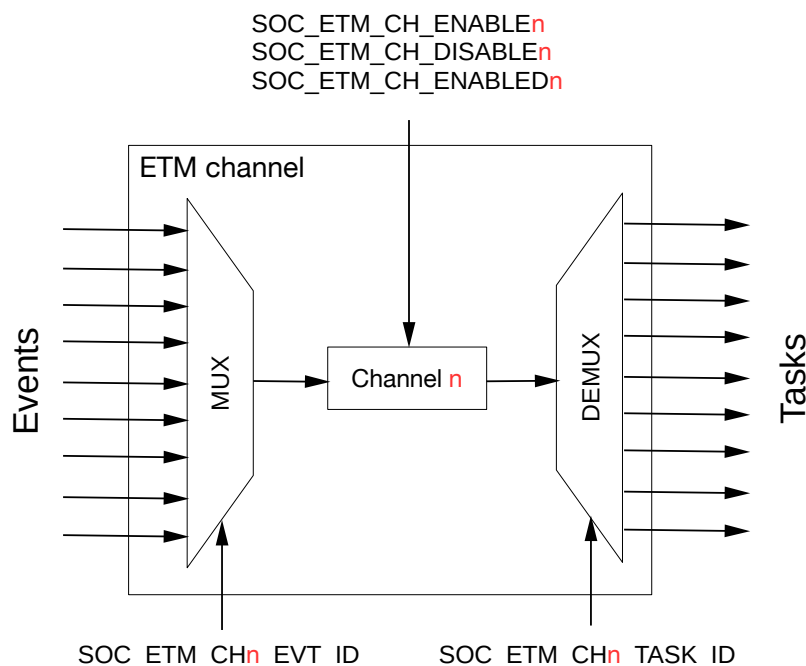
Figure 8.3-1. ETM Channel n Architecture

Figure 8.3-1 illustrates the structure of an ETM channel. The `SOC_ETM_CH $n$ _EVT_ID` field configures the MUX (multiplexer) to select one of the events as the input of channel n . The `SOC_ETM_CH $n$ _TASK_ID` field configures the DEMUX (demultiplexer) to map the event selected by channel n to one of the tasks. `SOC_ETM_CH_ENABLE $n$` and `SOC_ETM_CH_DISABLE $n$` are used to enable or disable channel n . `SOC_ETM_CH_ENABLED $n$` is used to indicate the status of the channel n .

8.3.2 Events

An ETM channel can be set up to select one event to receive by configuring the `SOC_ETM_CH $n$ _EVT_ID` field. Table 8.3-1 shows the configuration values of `SOC_ETM_CH $n$ _EVT_ID` and their corresponding events.

Table 8.3-1. Selectable Events for ETM Channel n

<code>SOC_ETM_CHn_EVT_ID</code>	Selected Event	Peripheral Generating This Event
1	<code>GPIO_EVT_CHO_RISE_EDGE</code>	GPIO
2	<code>GPIO_EVT_CH1_RISE_EDGE</code>	
3	<code>GPIO_EVT_CH2_RISE_EDGE</code>	
4	<code>GPIO_EVT_CH3_RISE_EDGE</code>	
5	<code>GPIO_EVT_CH4_RISE_EDGE</code>	
6	<code>GPIO_EVT_CH5_RISE_EDGE</code>	
7	<code>GPIO_EVT_CH6_RISE_EDGE</code>	
8	<code>GPIO_EVT_CH7_RISE_EDGE</code>	
9	<code>GPIO_EVT_CHO_FALL_EDGE</code>	
10	<code>GPIO_EVT_CH1_FALL_EDGE</code>	
11	<code>GPIO_EVT_CH2_FALL_EDGE</code>	

SOC_ETM_CH _n _EVT_ID	Selected Event	Peripheral Generating This Event
12	GPIO_EVT_CH3_FALL_EDGE	
13	GPIO_EVT_CH4_FALL_EDGE	
14	GPIO_EVT_CH5_FALL_EDGE	
15	GPIO_EVT_CH6_FALL_EDGE	
16	GPIO_EVT_CH7_FALL_EDGE	
17	GPIO_EVT_CH0_ANY_EDGE	
18	GPIO_EVT_CH1_ANY_EDGE	
19	GPIO_EVT_CH2_ANY_EDGE	
20	GPIO_EVT_CH3_ANY_EDGE	
21	GPIO_EVT_CH4_ANY_EDGE	
22	GPIO_EVT_CH5_ANY_EDGE	
23	GPIO_EVT_CH6_ANY_EDGE	
24	GPIO_EVT_CH7_ANY_EDGE	
25	GPIO_EVT_ZERO_DET_POS0	
26	GPIO_EVT_ZERO_DET_NEG0	
27	LEDC_EVT_DUTY_CHNG_END_CH0	LED PWM Controller (LEDC)
28	LEDC_EVT_DUTY_CHNG_END_CH1	
29	LEDC_EVT_DUTY_CHNG_END_CH2	
30	LEDC_EVT_DUTY_CHNG_END_CH3	
31	LEDC_EVT_DUTY_CHNG_END_CH4	
32	LEDC_EVT_DUTY_CHNG_END_CH5	
33	LEDC_EVT_OVF_CNT_PLS_CH0	
34	LEDC_EVT_OVF_CNT_PLS_CH1	
35	LEDC_EVT_OVF_CNT_PLS_CH2	
36	LEDC_EVT_OVF_CNT_PLS_CH3	
37	LEDC_EVT_OVF_CNT_PLS_CH4	
38	LEDC_EVT_OVF_CNT_PLS_CH5	
39	LEDC_EVT_TIME_OVF_TIMER0	
40	LEDC_EVT_TIME_OVF_TIMER1	
41	LEDC_EVT_TIME_OVF_TIMER2	
42	LEDC_EVT_TIME_OVF_TIMER3	
43	LEDC_EVT_TIMER0_CMP	
44	LEDC_EVT_TIMER1_CMP	
45	LEDC_EVT_TIMER2_CMP	
46	LEDC_EVT_TIMER3_CMP	
47	TGO_EVT_CNT_CMP_TIMER0	General-purpose timer group 0
49	TG1_EVT_CNT_CMP_TIMER0	General-purpose timer group 1
51	SYSTIMER_EVT_CNT_CMP0	System Timer
52	SYSTIMER_EVT_CNT_CMP1	
53	SYSTIMER_EVT_CNT_CMP2	
54	MCPWMO_EVT_TIMER0_STOP	MCPWMO
55	MCPWMO_EVT_TIMER1_STOP	
56	MCPWMO_EVT_TIMER2_STOP	
57	MCPWMO_EVT_TIMER0_TEZ	
58	MCPWMO_EVT_TIMER1_TEZ	

SOC_ETM_CH_n_EVT_ID	Selected Event	Peripheral Generating This Event
59	MCPWMO_EVT_TIMER2_TEZ	
60	MCPWMO_EVT_TIMER0_TEP	
61	MCPWMO_EVT_TIMER1_TEP	
62	MCPWMO_EVT_TIMER2_TEP	
63	MCPWMO_EVT_OPO_TEA	
64	MCPWMO_EVT_OP1_TEA	
65	MCPWMO_EVT_OP2_TEA	
66	MCPWMO_EVT_OPO_TEB	
67	MCPWMO_EVT_OP1_TEB	
68	MCPWMO_EVT_OP2_TEB	
69	MCPWMO_EVT_F0	
70	MCPWMO_EVT_F1	
71	MCPWMO_EVT_F2	
72	MCPWMO_EVT_F0_CLR	
73	MCPWMO_EVT_F1_CLR	
74	MCPWMO_EVT_F2_CLR	
75	MCPWMO_EVT_TZ0_CBC	
76	MCPWMO_EVT_TZ1_CBC	
77	MCPWMO_EVT_TZ2_CBC	
78	MCPWMO_EVT_TZ0_OST	
79	MCPWMO_EVT_TZ1_OST	
80	MCPWMO_EVT_TZ2_OST	
81	MCPWMO_EVT_CAP0	
82	MCPWMO_EVT_CAP1	
83	MCPWMO_EVT_CAP2	
84	MCPWMO_EVT_OPO_TEE1	
85	MCPWMO_EVT_OP1_TEE1	
86	MCPWMO_EVT_OP2_TEE1	
87	MCPWMO_EVT_OPO_TEE2	
88	MCPWMO_EVT_OP1_TEE2	
89	MCPWMO_EVT_OP2_TEE2	
90	ADC_EVT_CONV_CMPLT0	ADC Controller
91	ADC_EVT_EQ_ABOVE_THRESH0	
92	ADC_EVT_EQ_ABOVE_THRESH1	
93	ADC_EVT_EQ_BELOW_THRESH0	
94	ADC_EVT_EQ_BELOW_THRESH1	
96	ADC_EVT_STOPPED0	
97	ADC_EVT_STARTED0	
106	TMPSNSR_EVT_OVER_LIMIT	Temperature Sensor
107	I2SO_EVT_RX_DONE	I2S
108	I2SO_EVT_TX_DONE	
109	I2SO_EVT_X_WORDS_RECEIVED	
110	I2SO_EVT_X_WORDS_SENT	
111	ULP_EVT_ERR_INTR	Low-Power CPU

SOC_ETM_CHn_EVT_ID	Selected Event	Peripheral Generating This Event
113	ULP_EVT_START_INTR	
114	RTC_EVT_TICK	RTC Timer
115	RTC_EVT_OVF	
116	RTC_EVT_CMP	
117	GDMA_EVT_IN_DONE_CHO	GDMA Controller (GDMA)
118	GDMA_EVT_IN_DONE_CH1	
119	GDMA_EVT_IN_DONE_CH2	
120	GDMA_EVT_IN_SUC_EOF_CHO	
121	GDMA_EVT_IN_SUC_EOF_CH1	
122	GDMA_EVT_IN_SUC_EOF_CH2	
123	GDMA_EVT_IN_FIFO_EMPTY_CHO	
124	GDMA_EVT_IN_FIFO_EMPTY_CH1	
125	GDMA_EVT_IN_FIFO_EMPTY_CH2	
126	GDMA_EVT_IN_FIFO_FULL_CHO	
127	GDMA_EVT_IN_FIFO_FULL_CH1	
128	GDMA_EVT_IN_FIFO_FULL_CH2	
129	GDMA_EVT_OUT_DONE_CHO	
130	GDMA_EVT_OUT_DONE_CH1	
131	GDMA_EVT_OUT_DONE_CH2	
132	GDMA_EVT_OUT_EOF_CHO	
133	GDMA_EVT_OUT_EOF_CH1	
134	GDMA_EVT_OUT_EOF_CH2	
135	GDMA_EVT_OUT_TOTAL_EOF_CHO	
136	GDMA_EVT_OUT_TOTAL_EOF_CH1	
137	GDMA_EVT_OUT_TOTAL_EOF_CH2	
138	GDMA_EVT_OUT_FIFO_EMPTY_CHO	
139	GDMA_EVT_OUT_FIFO_EMPTY_CH1	
140	GDMA_EVT_OUT_FIFO_EMPTY_CH2	
141	GDMA_EVT_OUT_FIFO_FULL_CHO	
142	GDMA_EVT_OUT_FIFO_FULL_CH1	
143	GDMA_EVT_OUT_FIFO_FULL_CH2	
144	PMU_EVT_SLEEP_WEEKUP	PMU

Whenever any of these events occurs, the corresponding peripheral generates a pulse signal. As soon as the pulse signal is high, the event is considered as being received.

For more detailed descriptions of an event, please refer to the chapter for the peripheral generating this event.

8.3.3 Tasks

An ETM channel can be set up to map its event to one of the tasks by configuring the `SOC_ETM_CH $n$ _TASK_ID` field. Table 8.3-2 shows the configuration values of `SOC_ETM_CH $n$ _TASK_ID` and their corresponding tasks.

Table 8.3-2. Mappable Tasks for ETM Channel *n*

SOC_ETM_CH <i>n</i> _TASK_ID	Mapped Task	Peripheral Receiving This Task
1	GPIO_TASK_CH0_SET	GPIO
2	GPIO_TASK_CH1_SET	
3	GPIO_TASK_CH2_SET	
4	GPIO_TASK_CH3_SET	
5	GPIO_TASK_CH4_SET	
6	GPIO_TASK_CH5_SET	
7	GPIO_TASK_CH6_SET	
8	GPIO_TASK_CH7_SET	
9	GPIO_TASK_CH0_CLEAR	
10	GPIO_TASK_CH1_CLEAR	
11	GPIO_TASK_CH2_CLEAR	
12	GPIO_TASK_CH3_CLEAR	
13	GPIO_TASK_CH4_CLEAR	
14	GPIO_TASK_CH5_CLEAR	
15	GPIO_TASK_CH6_CLEAR	
16	GPIO_TASK_CH7_CLEAR	
17	GPIO_TASK_CH0_TOGGLE	
18	GPIO_TASK_CH1_TOGGLE	
19	GPIO_TASK_CH2_TOGGLE	
20	GPIO_TASK_CH3_TOGGLE	
21	GPIO_TASK_CH4_TOGGLE	
22	GPIO_TASK_CH5_TOGGLE	
23	GPIO_TASK_CH6_TOGGLE	
24	GPIO_TASK_CH7_TOGGLE	
25	LEDC_TASK_TIMER0_RES_UPDATE	LED PWM Controller (LEDC)
26	LEDC_TASK_TIMER1_RES_UPDATE	
27	LEDC_TASK_TIMER2_RES_UPDATE	
28	LEDC_TASK_TIMER3_RES_UPDATE	
29	LEDC_TASK_DUTY_SCALE_UPDATE_CH0	
30	LEDC_TASK_DUTY_SCALE_UPDATE_CH1	
31	LEDC_TASK_DUTY_SCALE_UPDATE_CH2	
32	LEDC_TASK_DUTY_SCALE_UPDATE_CH3	
33	LEDC_TASK_DUTY_SCALE_UPDATE_CH4	
34	LEDC_TASK_DUTY_SCALE_UPDATE_CH5	
35	LEDC_TASK_TIMER0_CAP	
36	LEDC_TASK_TIMER1_CAP	
37	LEDC_TASK_TIMER2_CAP	
38	LEDC_TASK_TIMER3_CAP	
39	LEDC_TASK_SIG_OUT_DIS_CH0	
40	LEDC_TASK_SIG_OUT_DIS_CH1	
41	LEDC_TASK_SIG_OUT_DIS_CH2	
42	LEDC_TASK_SIG_OUT_DIS_CH3	
43	LEDC_TASK_SIG_OUT_DIS_CH4	

SOC_ETM_CHn_TASK_ID	Mapped Task	Peripheral Receiving This Task
44	LEDC_TASK_SIG_OUT_DIS_CH5	
45	LEDC_TASK_OVF_CNT_RST_CHO	
46	LEDC_TASK_OVF_CNT_RST_CH1	
47	LEDC_TASK_OVF_CNT_RST_CH2	
48	LEDC_TASK_OVF_CNT_RST_CH3	
49	LEDC_TASK_OVF_CNT_RST_CH4	
50	LEDC_TASK_OVF_CNT_RST_CH5	
51	LEDC_TASK_TIMER0_RST	
52	LEDC_TASK_TIMER1_RST	
53	LEDC_TASK_TIMER2_RST	
54	LEDC_TASK_TIMER3_RST	
55	LEDC_TASK_TIMER0_RESUME	
56	LEDC_TASK_TIMER1_RESUME	
57	LEDC_TASK_TIMER2_RESUME	
58	LEDC_TASK_TIMER3_RESUME	
59	LEDC_TASK_TIMER0_PAUSE	
60	LEDC_TASK_TIMER1_PAUSE	
61	LEDC_TASK_TIMER2_PAUSE	
62	LEDC_TASK_TIMER3_PAUSE	
63	LEDC_TASK_GAMMA_RESTART_CHO	
64	LEDC_TASK_GAMMA_RESTART_CH1	
65	LEDC_TASK_GAMMA_RESTART_CH2	
66	LEDC_TASK_GAMMA_RESTART_CH3	
67	LEDC_TASK_GAMMA_RESTART_CH4	
68	LEDC_TASK_GAMMA_RESTART_CH5	
69	LEDC_TASK_GAMMA_PAUSE_CHO	
70	LEDC_TASK_GAMMA_PAUSE_CH1	
71	LEDC_TASK_GAMMA_PAUSE_CH2	
72	LEDC_TASK_GAMMA_PAUSE_CH3	
73	LEDC_TASK_GAMMA_PAUSE_CH4	
74	LEDC_TASK_GAMMA_PAUSE_CH5	
75	LEDC_TASK_GAMMA_RESUME_CHO	
76	LEDC_TASK_GAMMA_RESUME_CH1	
77	LEDC_TASK_GAMMA_RESUME_CH2	
78	LEDC_TASK_GAMMA_RESUME_CH3	
79	LEDC_TASK_GAMMA_RESUME_CH4	
80	LEDC_TASK_GAMMA_RESUME_CH5	
81	TGO_TASK_CNT_START_TIMER0	General-purpose timer group 0
82	TGO_TASK_ALARM_START_TIMER0	
83	TGO_TASK_CNT_STOP_TIMER0	
84	TGO_TASK_CNT_RELOAD_TIMER0	
85	TGO_TASK_CNT_CAP_TIMER0	
91	TG1_TASK_CNT_START_TIMER0	General-purpose timer group 1
92	TG1_TASK_ALARM_START_TIMER0	
93	TG1_TASK_CNT_STOP_TIMER0	

SOC_ETM_CHn_TASK_ID	Mapped Task	Peripheral Receiving This Task
94	TG1_TASK_CNT_RELOAD_TIMER0	MCPWMO
95	TG1_TASK_CNT_CAP_TIMER0	
101	MCPWMO_TASK_CMPRO_A_UP	
102	MCPWMO_TASK_CMPR1_A_UP	
103	MCPWMO_TASK_CMPR2_A_UP	
104	MCPWMO_TASK_CMPRO_B_UP	
105	MCPWMO_TASK_CMPR1_B_UP	
106	MCPWMO_TASK_CMPR2_B_UP	
107	MCPWMO_TASK_GEN_STOP	
108	MCPWMO_TASK_TIMER0_SYN	
109	MCPWMO_TASK_TIMER1_SYN	
110	MCPWMO_TASK_TIMER2_SYN	
111	MCPWMO_TASK_TIMER0_PERIOD_UP	
112	MCPWMO_TASK_TIMER1_PERIOD_UP	
113	MCPWMO_TASK_TIMER2_PERIOD_UP	
114	MCPWMO_TASK_TZO_OST	
115	MCPWMO_TASK_TZ1_OST	
116	MCPWMO_TASK_TZ2_OST	
117	MCPWMO_TASK_CLRO_OST	
118	MCPWMO_TASK_CLR1_OST	
119	MCPWMO_TASK_CLR2_OST	
120	MCPWMO_TASK_CAPO	ADC Controller
121	MCPWMO_TASK_CAP1	
122	MCPWMO_TASK_CAP2	
123	ADC_TASK_SAMPLE0	ADC Controller
125	ADC_TASK_START0	
126	ADC_TASK_STOP0	
131	TMPSNSR_TASK_START_SAMPLE	Temperature Sensor
132	TMPSNSR_TASK_STOP_SAMPLE	
133	I2SO_TASK_START_RX	I2S
134	I2SO_TASK_START_TX	
135	I2SO_TASK_STOP_RX	
136	I2SO_TASK_STOP_TX	
137	ULP_TASK_WEAKUP_CPU	Low-Power CPU
143	GDMA_TASK_IN_START_CH0	GDMA Controller (GDMA)
144	GDMA_TASK_IN_START_CH1	
145	GDMA_TASK_IN_START_CH2	
146	GDMA_TASK_OUT_START_CH0	
147	GDMA_TASK_OUT_START_CH1	
148	GDMA_TASK_OUT_START_CH2	
149	PMU_TASK_SLEEP_REQ	PMU

When a channel receives a valid event pulse signal, it generates the mapped task pulse signal.

For more detailed descriptions of a task, please refer to the chapter for the peripheral receiving this

task.

Events from different channels can be optionally mapped to the same task. For example, field `SOC_ETM_CH $n$ _TASK_ID` of multiple channels can be configured with the same value, and field `SOC_ETM_CH $n$ _EVT_ID` can be configured with the same or different values. In this case, when the event received by any of the channels is valid, the task will be generated. If events received by multiple channels are valid at the same time, the task will be generated only once.

8.3.4 Event and Task Status

The ETM module supports checking the status of events and tasks by reading registers as follows:

- Event status can be checked by reading the `SOC_ETM_EVT_ST $n$ _REG` register. If the field corresponding to an event is 1, it indicates that the event has been received by ETM. Otherwise, it indicates that the event has not been received by ETM. The fields in the `SOC_ETM_EVT_ST $n$ _REG` register can be cleared by writing 1 to the corresponding fields in the `SOC_ETM_EVT_ST $n$ _CLR_REG` register.
- Task status can be checked by reading the `SOC_ETM_TASK_ST $n$ _REG` register. If the field corresponding to a task is 1, it indicates that the task has been generated by ETM. Otherwise, it indicates that the task has not been generated by ETM. The fields in the `SOC_ETM_TASK_ST $n$ _REG` register can be cleared by writing 1 to the corresponding fields in the `SOC_ETM_TASK_ST $n$ _CLR_REG` register.

8.3.5 Timing Considerations

Figure 8.3-2 shows the structure of clocks that drive received events, sent tasks, and ETM channels.

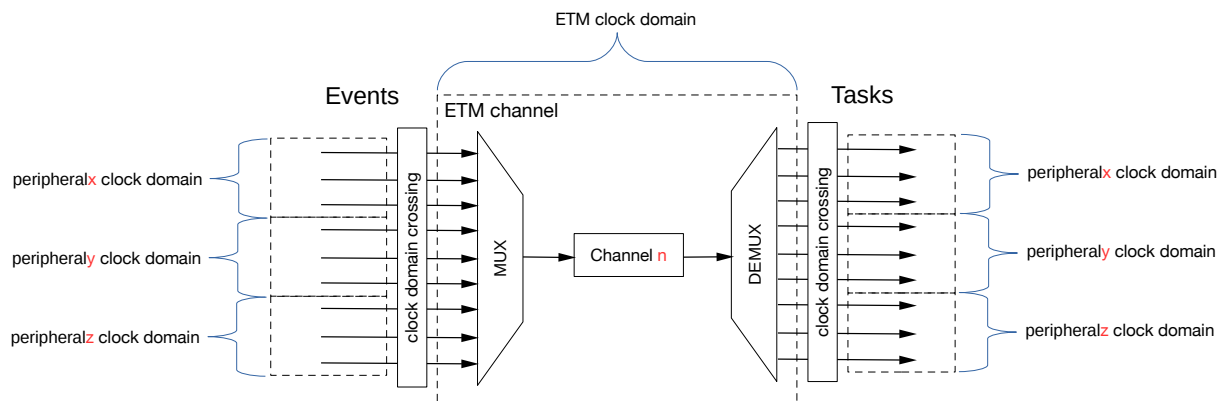


Figure 8.3-2. Event Task Matrix Clock Architecture

ETM is running at the `SYS_CLK` domain (see Chapter 6 *Reset and Clock*). Each event corresponds to a pulse signal generated by the corresponding peripheral in its clock domain, while each task is mapped by the ETM to a pulse signal under its corresponding peripheral clock domain. The peripherals generating events, the Event Task Matrix, and peripherals receiving tasks are not necessarily running off the same clock and as such need to be synchronized. Therefore, to avoid event loss, there should be a delay between two consecutive event pulses which is dependent on the clock of the peripheral generating the event, on the ETM's clock, and on the clock of the peripheral executing the task.

To make sure the Event Task Matrix receives every event successfully, for peripherals generating event pulses, the interval between two consecutive pulses must be greater than one ETM clock cycle, namely $\text{ceil}(\frac{\text{peripheral\_clock\_frequency}}{\text{ETM\_clock\_frequency}})$ in the unit of peripheral clock cycles.

For example, assuming that peripheral A is in the 80 MHz clock domain (PLL_F80M_CLK), and the ETM runs in the 40 MHz clock domain (SYS_CLK). Peripheral A generates event 1, which should be received by ETM. To receive each event 1 successfully, the interval between two consecutive event 1 must be greater than two peripheral A clock cycles (i.e., one ETM clock cycle).

Likewise, to make sure the Event Task Matrix maps the received event (i.e., event synchronized to the ETM's clock domain) successfully to a task, the interval between two consecutive event pulses in the ETM clock domain must be greater than one peripheral clock cycle, namely $\text{ceil}(\frac{\text{ETM\_clock\_frequency}}{\text{peripheral\_clock\_frequency}})$ in the unit of ETM clock cycles.

For example, assuming that peripheral B is in the 20 MHz clock domain (RC_FAST_CLK), and the ETM runs in the 40 MHz clock domain (SYS_CLK). ETM maps an event to task 1, which should be received by peripheral B. To map each received event successfully to task 1, the interval between two consecutive events must be greater than two ETM clock cycles (i.e., one peripheral B clock cycle).

As a result, to map two consecutive events generated by peripheral A to peripheral B, the interval between these two events must be $\text{ceil}(\frac{\text{peripheral\_A\_clock\_frequency}}{\text{ETM\_clock\_frequency}}) * \text{ceil}(\frac{\text{ETM\_clock\_frequency}}{\text{peripheral\_B\_clock\_frequency}})$ in the unit of peripheral A clock cycles.

For example, assuming that peripheral A is in the 80 MHz clock domain (PLL_F80M_CLK), peripheral B is in the 20 MHz clock domain (RC_FAST_CLK), and the ETM runs in the 40 MHz clock domain (SYS_CLK). ETM maps event 1 generated by peripheral A to task 1 received by peripheral B. To successfully map each event 1 to task 1, the interval between two consecutive event 1 must be greater than $2 * 2 = 4$ peripheral A clock cycles.

8.3.6 Channel Control

Each ETM channel can be independently configured to be enabled or disabled. When channel n is enabled and receives the event configured via `SOC_ETM_CH $n$ _EVT_ID`, it maps the event to the task configured via `SOC_ETM_CH $n$ _TASK_ID`. When channel n is disabled, even if it receives the event configured via `SOC_ETM_CH $n$ _EVT_ID`, no task will be generated.

To enable ETM channel n , write 1 to `SOC_ETM_CH_ENABLE $n$` . To disable ETM channel n , write 1 to `SOC_ETM_CH_DISABLE $n$` .

The status of ETM channel n can be obtained by reading `SOC_ETM_CH_ENABLED $n$` . 1 indicates that channel n has been enabled, and 0 indicates disabled.

If `SOC_ETM_CH $n$ _EVT_ID` or `SOC_ETM_CH $n$ _TASK_ID` is configured to 0, ETM channel n will also be disabled.

The complete procedure to configure ETM channel n is as follows:

1. Enable the ETM's clock by writing 1 to `PCR_ETM_CLK_EN`
2. Select the event to be received by channel n via `SOC_ETM_CH $n$ _EVT_ID`
3. Select the task mapped to the received event via `SOC_ETM_CH $n$ _TASK_ID`
4. Enable channel n by setting `SOC_ETM_CH_ENABLE $n$`

5. When channel *n* no longer needs to map the selected event to the selected task, disable channel *n* by setting `SOC_ETM_CH_DISABLEn`. To configure a new event and task mapping, repeat Steps 2 to 4. If no further configurations, channel *n* will remain disabled
6. The whole ETM module (i.e., all ETM channels) can be reset by writing 1 and then 0 to the `PCR_ETM_RST_EN` field

8.4 Register Summary

The addresses in this section are relative to Event Task Matrix base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Status register			
SOC_ETM_CH_ENA_ADO_REG	Channel enable status register	0x0000	R/WTC/WTS
SOC_ETM_CH_ENA_AD1_REG	Channel enable status register	0x000C	R/WTC/WTS
SOC_ETM_EVT_STO_REG	Event trigger status register	0x01A8	R/WTC/SS
SOC_ETM_EVT_ST1_REG	Event trigger status register	0x01B0	R/WTC/SS
SOC_ETM_EVT_ST2_REG	Event trigger status register	0x01B8	R/WTC/SS
SOC_ETM_EVT_ST3_REG	Event trigger status register	0x01C0	R/WTC/SS
SOC_ETM_EVT_ST4_REG	Event trigger status register	0x01C8	R/WTC/SS
SOC_ETM_TASK_STO_REG	Task trigger status register	0x01D0	R/WTC/SS
SOC_ETM_TASK_ST1_REG	Task trigger status register	0x01D8	R/WTC/SS
SOC_ETM_TASK_ST2_REG	Task trigger status register	0x01E0	R/WTC/SS
SOC_ETM_TASK_ST3_REG	Task trigger status register	0x01E8	R/WTC/SS
SOC_ETM_TASK_ST4_REG	Task trigger status register	0x01F0	R/WTC/SS
Configuration Register			
SOC_ETM_CH_ENA_ADO_SET_REG	Channel enable register	0x0004	WT
SOC_ETM_CH_ENA_ADO_CLR_REG	Channel disable register	0x0008	WT
SOC_ETM_CH_ENA_AD1_SET_REG	Channel enable register	0x0010	WT
SOC_ETM_CH_ENA_AD1_CLR_REG	Channel disable register	0x0014	WT
SOC_ETM_CH0_EVT_ID_REG	Channel 0 event ID register	0x0018	R/W
SOC_ETM_CH0_TASK_ID_REG	Channel 0 task ID register	0x001C	R/W
SOC_ETM_CH1_EVT_ID_REG	Channel 1 event ID register	0x0020	R/W
SOC_ETM_CH1_TASK_ID_REG	Channel 1 task ID register	0x0024	R/W
SOC_ETM_CH2_EVT_ID_REG	Channel 2 event ID register	0x0028	R/W
SOC_ETM_CH2_TASK_ID_REG	Channel 2 task ID register	0x002C	R/W
SOC_ETM_CH3_EVT_ID_REG	Channel 3 event ID register	0x0030	R/W
SOC_ETM_CH3_TASK_ID_REG	Channel 3 task ID register	0x0034	R/W
SOC_ETM_CH4_EVT_ID_REG	Channel 4 event ID register	0x0038	R/W
SOC_ETM_CH4_TASK_ID_REG	Channel 4 task ID register	0x003C	R/W
SOC_ETM_CH5_EVT_ID_REG	Channel 5 event ID register	0x0040	R/W
SOC_ETM_CH5_TASK_ID_REG	Channel 5 task ID register	0x0044	R/W
SOC_ETM_CH6_EVT_ID_REG	Channel 6 event ID register	0x0048	R/W
SOC_ETM_CH6_TASK_ID_REG	Channel 6 task ID register	0x004C	R/W
SOC_ETM_CH7_EVT_ID_REG	Channel 7 event ID register	0x0050	R/W
SOC_ETM_CH7_TASK_ID_REG	Channel 7 task ID register	0x0054	R/W
SOC_ETM_CH8_EVT_ID_REG	Channel 8 event ID register	0x0058	R/W
SOC_ETM_CH8_TASK_ID_REG	Channel 8 task ID register	0x005C	R/W
SOC_ETM_CH9_EVT_ID_REG	Channel 9 event ID register	0x0060	R/W

Name	Description	Address	Access
SOC_ETM_CH9_TASK_ID_REG	Channel 9 task ID register	0x0064	R/W
SOC_ETM_CH10_EVT_ID_REG	Channel 10 event ID register	0x0068	R/W
SOC_ETM_CH10_TASK_ID_REG	Channel 10 task ID register	0x006C	R/W
SOC_ETM_CH11_EVT_ID_REG	Channel 11 event ID register	0x0070	R/W
SOC_ETM_CH11_TASK_ID_REG	Channel 11 task ID register	0x0074	R/W
SOC_ETM_CH12_EVT_ID_REG	Channel 12 event ID register	0x0078	R/W
SOC_ETM_CH12_TASK_ID_REG	Channel 12 task ID register	0x007C	R/W
SOC_ETM_CH13_EVT_ID_REG	Channel 13 event ID register	0x0080	R/W
SOC_ETM_CH13_TASK_ID_REG	Channel 13 task ID register	0x0084	R/W
SOC_ETM_CH14_EVT_ID_REG	Channel 14 event ID register	0x0088	R/W
SOC_ETM_CH14_TASK_ID_REG	Channel 14 task ID register	0x008C	R/W
SOC_ETM_CH15_EVT_ID_REG	Channel 15 event ID register	0x0090	R/W
SOC_ETM_CH15_TASK_ID_REG	Channel 15 task ID register	0x0094	R/W
SOC_ETM_CH16_EVT_ID_REG	Channel 16 event ID register	0x0098	R/W
SOC_ETM_CH16_TASK_ID_REG	Channel 16 task ID register	0x009C	R/W
SOC_ETM_CH17_EVT_ID_REG	Channel 17 event ID register	0x00A0	R/W
SOC_ETM_CH17_TASK_ID_REG	Channel 17 task ID register	0x00A4	R/W
SOC_ETM_CH18_EVT_ID_REG	Channel 18 event ID register	0x00A8	R/W
SOC_ETM_CH18_TASK_ID_REG	Channel 18 task ID register	0x00AC	R/W
SOC_ETM_CH19_EVT_ID_REG	Channel 19 event ID register	0x00B0	R/W
SOC_ETM_CH19_TASK_ID_REG	Channel 19 task ID register	0x00B4	R/W
SOC_ETM_CH20_EVT_ID_REG	Channel 20 event ID register	0x00B8	R/W
SOC_ETM_CH20_TASK_ID_REG	Channel 20 task ID register	0x00BC	R/W
SOC_ETM_CH21_EVT_ID_REG	Channel 21 event ID register	0x00C0	R/W
SOC_ETM_CH21_TASK_ID_REG	Channel 21 task ID register	0x00C4	R/W
SOC_ETM_CH22_EVT_ID_REG	Channel 22 event ID register	0x00C8	R/W
SOC_ETM_CH22_TASK_ID_REG	Channel 22 task ID register	0x00CC	R/W
SOC_ETM_CH23_EVT_ID_REG	Channel 23 event ID register	0x00D0	R/W
SOC_ETM_CH23_TASK_ID_REG	Channel 23 task ID register	0x00D4	R/W
SOC_ETM_CH24_EVT_ID_REG	Channel 24 event ID register	0x00D8	R/W
SOC_ETM_CH24_TASK_ID_REG	Channel 24 task ID register	0x00DC	R/W
SOC_ETM_CH25_EVT_ID_REG	Channel 25 event ID register	0x00E0	R/W
SOC_ETM_CH25_TASK_ID_REG	Channel 25 task ID register	0x00E4	R/W
SOC_ETM_CH26_EVT_ID_REG	Channel 26 event ID register	0x00E8	R/W
SOC_ETM_CH26_TASK_ID_REG	Channel 26 task ID register	0x00EC	R/W
SOC_ETM_CH27_EVT_ID_REG	Channel 27 event ID register	0x00F0	R/W
SOC_ETM_CH27_TASK_ID_REG	Channel 27 task ID register	0x00F4	R/W
SOC_ETM_CH28_EVT_ID_REG	Channel 28 event ID register	0x00F8	R/W
SOC_ETM_CH28_TASK_ID_REG	Channel 28 task ID register	0x00FC	R/W
SOC_ETM_CH29_EVT_ID_REG	Channel 29 event ID register	0x0100	R/W
SOC_ETM_CH29_TASK_ID_REG	Channel 29 task ID register	0x0104	R/W
SOC_ETM_CH30_EVT_ID_REG	Channel 30 event ID register	0x0108	R/W
SOC_ETM_CH30_TASK_ID_REG	Channel 30 task ID register	0x010C	R/W

Name	Description	Address	Access
SOC_ETM_CH31_EVT_ID_REG	Channel 31 event ID register	0x0110	R/W
SOC_ETM_CH31_TASK_ID_REG	Channel 31 task ID register	0x0114	R/W
SOC_ETM_CH32_EVT_ID_REG	Channel 32 event ID register	0x0118	R/W
SOC_ETM_CH32_TASK_ID_REG	Channel 32 task ID register	0x011C	R/W
SOC_ETM_CH33_EVT_ID_REG	Channel 33 event ID register	0x0120	R/W
SOC_ETM_CH33_TASK_ID_REG	Channel 33 task ID register	0x0124	R/W
SOC_ETM_CH34_EVT_ID_REG	Channel 34 event ID register	0x0128	R/W
SOC_ETM_CH34_TASK_ID_REG	Channel 34 task ID register	0x012C	R/W
SOC_ETM_CH35_EVT_ID_REG	Channel 35 event ID register	0x0130	R/W
SOC_ETM_CH35_TASK_ID_REG	Channel 35 task ID register	0x0134	R/W
SOC_ETM_CH36_EVT_ID_REG	Channel 36 event ID register	0x0138	R/W
SOC_ETM_CH36_TASK_ID_REG	Channel 36 task ID register	0x013C	R/W
SOC_ETM_CH37_EVT_ID_REG	Channel 37 event ID register	0x0140	R/W
SOC_ETM_CH37_TASK_ID_REG	Channel 37 task ID register	0x0144	R/W
SOC_ETM_CH38_EVT_ID_REG	Channel 38 event ID register	0x0148	R/W
SOC_ETM_CH38_TASK_ID_REG	Channel 38 task ID register	0x014C	R/W
SOC_ETM_CH39_EVT_ID_REG	Channel 39 event ID register	0x0150	R/W
SOC_ETM_CH39_TASK_ID_REG	Channel 39 task ID register	0x0154	R/W
SOC_ETM_CH40_EVT_ID_REG	Channel 40 event ID register	0x0158	R/W
SOC_ETM_CH40_TASK_ID_REG	Channel 40 task ID register	0x015C	R/W
SOC_ETM_CH41_EVT_ID_REG	Channel 41 event ID register	0x0160	R/W
SOC_ETM_CH41_TASK_ID_REG	Channel 41 task ID register	0x0164	R/W
SOC_ETM_CH42_EVT_ID_REG	Channel 42 event ID register	0x0168	R/W
SOC_ETM_CH42_TASK_ID_REG	Channel 42 task ID register	0x016C	R/W
SOC_ETM_CH43_EVT_ID_REG	Channel 43 event ID register	0x0170	R/W
SOC_ETM_CH43_TASK_ID_REG	Channel 43 task ID register	0x0174	R/W
SOC_ETM_CH44_EVT_ID_REG	Channel 44 event ID register	0x0178	R/W
SOC_ETM_CH44_TASK_ID_REG	Channel 44 task ID register	0x017C	R/W
SOC_ETM_CH45_EVT_ID_REG	Channel 45 event ID register	0x0180	R/W
SOC_ETM_CH45_TASK_ID_REG	Channel 45 task ID register	0x0184	R/W
SOC_ETM_CH46_EVT_ID_REG	Channel 46 event ID register	0x0188	R/W
SOC_ETM_CH46_TASK_ID_REG	Channel 46 task ID register	0x018C	R/W
SOC_ETM_CH47_EVT_ID_REG	Channel 47 event ID register	0x0190	R/W
SOC_ETM_CH47_TASK_ID_REG	Channel 47 task ID register	0x0194	R/W
SOC_ETM_CH48_EVT_ID_REG	Channel 48 event ID register	0x0198	R/W
SOC_ETM_CH48_TASK_ID_REG	Channel 48 task ID register	0x019C	R/W
SOC_ETM_CH49_EVT_ID_REG	Channel 49 event ID register	0x01A0	R/W
SOC_ETM_CH49_TASK_ID_REG	Channel 49 task ID register	0x01A4	R/W
SOC_ETM_EVT_STO_CLR_REG	Event trigger status clear register	0x01AC	WT
SOC_ETM_EVT_ST1_CLR_REG	Event trigger status clear register	0x01B4	WT
SOC_ETM_EVT_ST2_CLR_REG	Event trigger status clear register	0x01BC	WT
SOC_ETM_EVT_ST3_CLR_REG	Event trigger status clear register	0x01C4	WT
SOC_ETM_EVT_ST4_CLR_REG	Event trigger status clear register	0x01CC	WT

Name	Description	Address	Access
SOC_ETM_TASK_ST0_CLR_REG	Task trigger status clear register	0x01D4	WT
SOC_ETM_TASK_ST1_CLR_REG	Task trigger status clear register	0x01DC	WT
SOC_ETM_TASK_ST2_CLR_REG	Task trigger status clear register	0x01E4	WT
SOC_ETM_TASK_ST3_CLR_REG	Task trigger status clear register	0x01EC	WT
SOC_ETM_TASK_ST4_CLR_REG	Task trigger status clear register	0x01F4	WT
SOC_ETM_CLK_EN_REG	ETM clock enable register	0x01F8	R/W
Version Register			
SOC_ETM_DATE_REG	Version control register	0x01FC	R/W

8.5 Registers

The addresses in this section are relative to Event Task Matrix base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 8.1. SOC_ETM_CH_ENA_ADO_REG (0x0000)

SOC_ETM_CH_ENABLED31	SOC_ETM_CH_ENABLED30	SOC_ETM_CH_ENABLED29	SOC_ETM_CH_ENABLED28	SOC_ETM_CH_ENABLED27	SOC_ETM_CH_ENABLED26	SOC_ETM_CH_ENABLED25	SOC_ETM_CH_ENABLED24	SOC_ETM_CH_ENABLED23	SOC_ETM_CH_ENABLED22	SOC_ETM_CH_ENABLED21	SOC_ETM_CH_ENABLED20	SOC_ETM_CH_ENABLED19	SOC_ETM_CH_ENABLED18	SOC_ETM_CH_ENABLED17	SOC_ETM_CH_ENABLED16	SOC_ETM_CH_ENABLED15	SOC_ETM_CH_ENABLED14	SOC_ETM_CH_ENABLED13	SOC_ETM_CH_ENABLED12	SOC_ETM_CH_ENABLED11	SOC_ETM_CH_ENABLED10	SOC_ETM_CH_ENABLED9	SOC_ETM_CH_ENABLED8	SOC_ETM_CH_ENABLED7	SOC_ETM_CH_ENABLED6	SOC_ETM_CH_ENABLED5	SOC_ETM_CH_ENABLED4	SOC_ETM_CH_ENABLED3	SOC_ETM_CH_ENABLED2	SOC_ETM_CH_ENABLED1	SOC_ETM_CH_ENABLED0
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

SOC_ETM_CH_ENABLED n (n : 0-31) Represents channel n enable status.

0: Disable

1: Enable

(R/WTC/WTs)

Register 8.2. SOC_ETM_CH_ENA_AD1_REG (0x000C)

(reserved)																		SOC_ETM_CH_ENABLED49																		SOC_ETM_CH_ENABLED48																		SOC_ETM_CH_ENABLED47																		SOC_ETM_CH_ENABLED46																		SOC_ETM_CH_ENABLED45																		SOC_ETM_CH_ENABLED44																		SOC_ETM_CH_ENABLED43																		SOC_ETM_CH_ENABLED42																		SOC_ETM_CH_ENABLED41																		SOC_ETM_CH_ENABLED40																		SOC_ETM_CH_ENABLED39																		SOC_ETM_CH_ENABLED38																		SOC_ETM_CH_ENABLED37																		SOC_ETM_CH_ENABLED36																		SOC_ETM_CH_ENABLED35																		SOC_ETM_CH_ENABLED34																		SOC_ETM_CH_ENABLED33																		SOC_ETM_CH_ENABLED32																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																		18																		17																		16																		15																		14																		13																		12																		11																		10																		9																		8																		7																		6																		5																		4																		3																		2																		1																		0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																		0																	

Reset

SOC_ETM_CH_ENABLED n (n : 32-49) Represents channel n enable status.

0: Disable

1: Enable

(R/WTC/WTs)

Register 8.3. SOC_ETM_EVT_STO_REG (0x01A8)

Soc																																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Soc_ETM_LEDC_EVT_DUTY_CHNG_END_CH5_ST																																
Soc_ETM_LEDC_EVT_DUTY_CHNG_END_CH4_ST																																
Soc_ETM_LEDC_EVT_DUTY_CHNG_END_CH3_ST																																
Soc_ETM_LEDC_EVT_DUTY_CHNG_END_CH2_ST																																
Soc_ETM_LEDC_EVT_DUTY_CHNG_END_CH1_ST																																
Soc_ETM_GPIO_EVT_ZERO_DET_NEGO_ST																																
Soc_ETM_GPIO_EVT_CH7_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH6_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH5_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH4_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH3_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH2_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH1_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH0_ANY_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH7_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH6_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH5_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH4_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH3_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH2_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH1_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH0_FALL_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH7_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH6_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH5_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH4_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH3_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH2_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH1_RISE_EDGE_ST																																
Soc_ETM_GPIO_EVT_CH0_RISE_EDGE_ST																																

SOC_ETM_GPIO_EVT_CH0_RISE_EDGE_ST Represents GPIO_EVT_CH0_RISE_EDGE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH1_RISE_EDGE_ST Represents GPIO_EVT_CH1_RISE_EDGE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH2_RISE_EDGE_ST Represents GPIO_EVT_CH2_RISE_EDGE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH3_RISE_EDGE_ST Represents GPIO_EVT_CH3_RISE_EDGE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH4_RISE_EDGE_ST Represents GPIO_EVT_CH4_RISE_EDGE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.3. SOC_ETM_EVT_STO_REG (0x01A8)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH5_RISE_EDGE_ST Represents GPIO_EVT_CH5_RISE_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH6_RISE_EDGE_ST Represents GPIO_EVT_CH6_RISE_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH7_RISE_EDGE_ST Represents GPIO_EVT_CH7_RISE_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH0_FALL_EDGE_ST Represents GPIO_EVT_CH0_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH1_FALL_EDGE_ST Represents GPIO_EVT_CH1_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH2_FALL_EDGE_ST Represents GPIO_EVT_CH2_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH3_FALL_EDGE_ST Represents GPIO_EVT_CH3_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.3. SOC_ETM_EVT_STO_REG (0x01A8)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH4_FALL_EDGE_ST Represents GPIO_EVT_CH4_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH5_FALL_EDGE_ST Represents GPIO_EVT_CH5_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH6_FALL_EDGE_ST Represents GPIO_EVT_CH6_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH7_FALL_EDGE_ST Represents GPIO_EVT_CH7_FALL_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CHO_ANY_EDGE_ST Represents GPIO_EVT_CHO_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH1_ANY_EDGE_ST Represents GPIO_EVT_CH1_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH2_ANY_EDGE_ST Represents GPIO_EVT_CH2_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH3_ANY_EDGE_ST Represents GPIO_EVT_CH3_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.3. SOC_ETM_EVT_STO_REG (0x01A8)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH4_ANY_EDGE_ST Represents GPIO_EVT_CH4_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH5_ANY_EDGE_ST Represents GPIO_EVT_CH5_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH6_ANY_EDGE_ST Represents GPIO_EVT_CH6_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_CH7_ANY_EDGE_ST Represents GPIO_EVT_CH7_ANY_EDGE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_ZERO_DET_POS0_ST Represents GPIO_EVT_ZERO_DET_POS0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_EVT_ZERO_DET_NEG0_ST Represents GPIO_EVT_ZERO_DET_NEG0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CHO_ST Represents LEDC_EVT_DUTY_CHNG_END_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.3. SOC_ETM_EVT_STO_REG (0x01A8)

Continued from the previous page...

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH1_ST Represents LEDC_EVT_DUTY_CHNG_END_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH2_ST Represents LEDC_EVT_DUTY_CHNG_END_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH3_ST Represents LEDC_EVT_DUTY_CHNG_END_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH4_ST Represents LEDC_EVT_DUTY_CHNG_END_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH5_ST Represents LEDC_EVT_DUTY_CHNG_END_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.4. SOC_ETM_EVT_ST1_REG (0x01B0)

SOC_ETM_MCPWM0_EVT_OPI_TEA_ST																														
SOC_ETM_MCPWM0_EVT_OPO_TEA_ST																														
SOC_ETM_MCPWM0_EVT_TIMER2_TEP_ST																														
SOC_ETM_MCPWM0_EVT_TIMER1_TEP_ST																														
SOC_ETM_MCPWM0_EVT_TIMER0_TEP_ST																														
SOC_ETM_MCPWM0_EVT_TIMER2_TEZ_ST																														
SOC_ETM_MCPWM0_EVT_TIMER1_TEZ_ST																														
SOC_ETM_MCPWM0_EVT_TIMER0_TEZ_ST																														
SOC_ETM_MCPWM0_EVT_TIMER2_STOP_ST																														
SOC_ETM_MCPWM0_EVT_TIMER1_STOP_ST																														
SOC_ETM_MCPWM0_EVT_TIMER0_STOP_ST																														
SOC_ETM_TG1_EVT_CNT_CMP2_ST																														
SOC_ETM_TG1_EVT_CNT_CMP1_ST																														
SOC_ETM_TG0_EVT_CNT_CMP0_ST																														
SOC_ETM_TG0_EVT_CNT_TIMER1_ST																														
SOC_ETM_LEDC_EVT_CNT_TIMER0_ST																														
SOC_ETM_LEDC_EVT_TIMER3_CMP_ST																														
SOC_ETM_LEDC_EVT_TIMER2_CMP_ST																														
SOC_ETM_LEDC_EVT_TIMER1_CMP_ST																														
SOC_ETM_LEDC_EVT_TIMER0_CMP_ST																														
SOC_ETM_LEDC_EVT_TIME_OVF_TIMER3_ST																														
SOC_ETM_LEDC_EVT_TIME_OVF_TIMER2_ST																														
SOC_ETM_LEDC_EVT_TIME_OVF_TIMER1_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH5_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH4_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH3_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH2_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH1_ST																														
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CHO_ST																														
Reset																														

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CHO_ST Represents LEDC_EVT_OVF_CNT_PLS_CHO trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH1_ST Represents LEDC_EVT_OVF_CNT_PLS_CH1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH2_ST Represents LEDC_EVT_OVF_CNT_PLS_CH2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH3_ST Represents LEDC_EVT_OVF_CNT_PLS_CH3 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH4_ST Represents LEDC_EVT_OVF_CNT_PLS_CH4 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH5_ST Represents LEDC_EVT_OVF_CNT_PLS_CH5 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Register 8.4. SOC_ETM_EVT_ST1_REG (0x01B0)

Continued from the previous page...

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER0_ST Represents LEDC_EVT_TIME_OVF_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER1_ST Represents LEDC_EVT_TIME_OVF_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER2_ST Represents LEDC_EVT_TIME_OVF_TIMER2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER3_ST Represents LEDC_EVT_TIME_OVF_TIMER3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIMER0_CMP_ST Represents LEDC_EVT_TIMER0_CMP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIMER1_CMP_ST Represents LEDC_EVT_TIMER1_CMP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIMER2_CMP_ST Represents LEDC_EVT_TIMER2_CMP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_EVT_TIMER3_CMP_ST Represents LEDC_EVT_TIMER3_CMP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.4. SOC_ETM_EVT_ST1_REG (0x01B0)

Continued from the previous page...

SOC_ETM_TGO_EVT_CNT_CMP_TIMER0_ST Represents TGO_EVT_CNT_CMP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_EVT_CNT_CMP_TIMER1_ST Represents TGO_EVT_CNT_CMP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_EVT_CNT_CMP_TIMER0_ST Represents TG1_EVT_CNT_CMP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_EVT_CNT_CMP_TIMER1_ST Represents TG1_EVT_CNT_CMP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_SYSTIMER_EVT_CNT_CMPO_ST Represents SYSTIMER_EVT_CNT_CMPO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_SYSTIMER_EVT_CNT_CMP1_ST Represents SYSTIMER_EVT_CNT_CMP1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_SYSTIMER_EVT_CNT_CMP2_ST Represents SYSTIMER_EVT_CNT_CMP2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER0_STOP_ST Represents MCPWMO_EVT_TIMER0_STOP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.4. SOC_ETM_EVT_ST1_REG (0x01B0)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_TIMER1_STOP_ST Represents MCPWMO_EVT_TIMER1_STOP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER2_STOP_ST Represents MCPWMO_EVT_TIMER2_STOP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER0_TEZ_ST Represents MCPWMO_EVT_TIMER0_TEZ trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER1_TEZ_ST Represents MCPWMO_EVT_TIMER1_TEZ trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER2_TEZ_ST Represents MCPWMO_EVT_TIMER2_TEZ trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER0_TEP_ST Represents MCPWMO_EVT_TIMER0_TEP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TIMER1_TEP_ST Represents MCPWMO_EVT_TIMER1_TEP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.4. SOC_ETM_EVT_ST1_REG (0x01B0)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_TIMER2_TEP_ST Represents MCPWMO_EVT_TIMER2_TEP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OPO_TEA_ST Represents MCPWMO_EVT_OPO_TEA trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP1_TEA_ST Represents MCPWMO_EVT_OP1_TEA trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.5. SOC_ETM_EVT_ST2_REG (0x01B8)

SOC_ETM_ADC_EVT_STOPPED0_ST	SOC_ETM_ADC_EVT_RESULT_DONE0_ST	SOC_ETM_ADC_EVT_EQ_BELOW_THRESH0_ST	SOC_ETM_ADC_EVT_EQ_BELOW_THRESH1_ST	SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH0_ST	SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH1_ST	SOC_ETM_MCPWMO_EVT_CONV_CMPLT0_ST	SOC_ETM_MCPWMO_EVT_OP2_TEE2_ST	SOC_ETM_MCPWMO_EVT_OP1_TEE2_ST	SOC_ETM_MCPWMO_EVT_OP0_TEE2_ST	SOC_ETM_MCPWMO_EVT_OP2_TEE1_ST	SOC_ETM_MCPWMO_EVT_OP1_TEE1_ST	SOC_ETM_MCPWMO_EVT_OP0_TEE1_ST	SOC_ETM_MCPWMO_EVT_CAP2_ST	SOC_ETM_MCPWMO_EVT_CAP1_ST	SOC_ETM_MCPWMO_EVT_CAPO_ST	SOC_ETM_MCPWMO_EVT_TZ2_OST_ST	SOC_ETM_MCPWMO_EVT_TZ1_OST_ST	SOC_ETM_MCPWMO_EVT_TZ0_OST_ST	SOC_ETM_MCPWMO_EVT_TZ2_OBC_ST	SOC_ETM_MCPWMO_EVT_TZ1_OBC_ST	SOC_ETM_MCPWMO_EVT_TZ0_OBC_ST	SOC_ETM_MCPWMO_EVT_F2_CLR_ST	SOC_ETM_MCPWMO_EVT_F1_CLR_ST	SOC_ETM_MCPWMO_EVT_F0_CLR_ST	SOC_ETM_MCPWMO_EVT_F2_ST	SOC_ETM_MCPWMO_EVT_F1_ST	SOC_ETM_MCPWMO_EVT_F0_ST	SOC_ETM_MCPWMO_EVT_OP2_TEB_ST	SOC_ETM_MCPWMO_EVT_OP1_TEB_ST	SOC_ETM_MCPWMO_EVT_OP0_TEB_ST	SOC_ETM_MCPWMO_EVT_OP2_TEA_ST
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

SOC_ETM_MCPWMO_EVT_OP2_TEA_ST Represents MCPWMO_EVT_OP2_TEA trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP0_TEB_ST Represents MCPWMO_EVT_OP0_TEB trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP1_TEB_ST Represents MCPWMO_EVT_OP1_TEB trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP2_TEB_ST Represents MCPWMO_EVT_OP2_TEB trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_F0_ST Represents MCPWMO_EVT_F0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_F1_ST Represents MCPWMO_EVT_F1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.5. SOC_ETM_EVT_ST2_REG (0x01B8)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_F2_ST Represents MCPWMO_EVT_F2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_F0_CLR_ST Represents MCPWMO_EVT_F0_CLR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_F1_CLR_ST Represents MCPWMO_EVT_F1_CLR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_F2_CLR_ST Represents MCPWMO_EVT_F2_CLR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TZO_CBC_ST Represents MCPWMO_EVT_TZO_CBC trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TZ1_CBC_ST Represents MCPWMO_EVT_TZ1_CBC trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TZ2_CBC_ST Represents MCPWMO_EVT_TZ2_CBC trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TZO_OST_ST Represents MCPWMO_EVT_TZO_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.5. SOC_ETM_EVT_ST2_REG (0x01B8)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_TZ1_OST_ST Represents MCPWMO_EVT_TZ1_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_TZ2_OST_ST Represents MCPWMO_EVT_TZ2_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_CAPO_ST Represents MCPWMO_EVT_CAPO trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_CAP1_ST Represents MCPWMO_EVT_CAP1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_CAP2_ST Represents MCPWMO_EVT_CAP2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OPO_TEE1_ST Represents MCPWMO_EVT_OPO_TEE1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP1_TEE1_ST Represents MCPWMO_EVT_OP1_TEE1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP2_TEE1_ST Represents MCPWMO_EVT_OP2_TEE1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.5. SOC_ETM_EVT_ST2_REG (0x01B8)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_OPO_TEE2_ST Represents MCPWMO_EVT_OPO_TEE2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP1_TEE2_ST Represents MCPWMO_EVT_OP1_TEE2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_EVT_OP2_TEE2_ST Represents MCPWMO_EVT_OP2_TEE2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ADC_EVT_CONV_CMPLT0_ST Represents ADC_EVT_CONV_CMPLT0 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH0_ST Represents ADC_EVT_EQ_ABOVE_THRESH0 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH1_ST Represents ADC_EVT_EQ_ABOVE_THRESH1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ADC_EVT_EQ_BELOW_THRESH0_ST Represents ADC_EVT_EQ_BELOW_THRESH0 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.5. SOC_ETM_EVT_ST2_REG (0x01B8)

Continued from the previous page...

SOC_ETM_ADC_EVT_EQ_BELOW_THRESH1_ST Represents ADC_EVT_EQ_BELOW_THRESH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_ADC_EVT_RESULT_DONE0_ST Represents ADC_EVT_RESULT_DONE0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_ADC_EVT_STOPPED0_ST Represents ADC_EVT_STOPPED0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.6. SOC_ETM_EVT_ST3_REG (0x01C0)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH2_ST																															SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH1_ST																															SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH0_ST																															SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH2_ST																															SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH1_ST																															SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH0_ST																															SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH2_ST																															SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH1_ST																															SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH0_ST																															SOC_ETM_RTC_EVT_IN_DONE_CH2_ST																															SOC_ETM_RTC_EVT_IN_DONE_CH1_ST																															SOC_ETM_RTC_EVT_CMP_ST																															SOC_ETM_RTC_EVT_OVF_ST																															SOC_ETM_ULP_EVT_TICK_ST																															SOC_ETM_ULP_EVT_START_INTR_ST																															SOC_ETM_ULP_EVT_HALT_ST																															SOC_ETM_I2S0_EVT_ERR_INTR_ST																															SOC_ETM_I2S0_EVT_X_WORDS_SENT_ST																															SOC_ETM_I2S0_EVT_TX_DONE_ST																															SOC_ETM_I2S0_EVT_RX_DONE_ST																															(reserved)																															SOC_ETM_ADC_EVT_STARTED_ST																																																																																																																																																																																																																																																																																																																																																																																																																																																			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8																	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

SOC_ETM_ADC_EVT_STARTED_ST Represents ADC_EVT_STARTED0 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_TMPSNSR_EVT_OVER_LIMIT_ST Represents TMPSNSR_EVT_OVER_LIMIT trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_I2SQ_EVT_RX_DONE_ST Represents I2SQ_EVT_RX_DONE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_I2SQ_EVT_TX_DONE_ST Represents I2SQ_EVT_TX_DONE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_I2SQ_EVT_X_WORDS_RECEIVED_ST Represents I2SQ_EVT_X_WORDS_RECEIVED trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_I2SQ_EVT_X_WORDS_SENT_ST Represents I2SQ_EVT_X_WORDS_SENT trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.6. SOC_ETM_EVT_ST3_REG (0x01C0)

Continued from the previous page...

SOC_ETM_ULP_EVT_ERR_INTR_ST Represents ULP_EVT_ERR_INTR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ULP_EVT_HALT_ST Represents ULP_EVT_HALT trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ULP_EVT_START_INTR_ST Represents ULP_EVT_START_INTR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_EVT_TICK_ST Represents RTC_EVT_TICK trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_EVT_OVF_ST Represents RTC_EVT_OVF trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_EVT_CMP_ST Represents RTC_EVT_CMP trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_DONE_CHO_ST Represents GDMA_EVT_IN_DONE_CHO trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_DONE_CH1_ST Represents GDMA_EVT_IN_DONE_CH1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_DONE_CH2_ST Represents GDMA_EVT_IN_DONE_CH2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.6. SOC_ETM_EVT_ST3_REG (0x01C0)

Continued from the previous page...

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH0_ST Represents GDMA_EVT_IN_SUC_EOF_CH0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH1_ST Represents GDMA_EVT_IN_SUC_EOF_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH2_ST Represents GDMA_EVT_IN_SUC_EOF_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH0_ST Represents GDMA_EVT_IN_FIFO_EMPTY_CH0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH1_ST Represents GDMA_EVT_IN_FIFO_EMPTY_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH2_ST Represents GDMA_EVT_IN_FIFO_EMPTY_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH0_ST Represents GDMA_EVT_IN_FIFO_FULL_CH0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.6. SOC_ETM_EVT_ST3_REG (0x01C0)

Continued from the previous page...

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH1_ST Represents GDMA_EVT_IN_FIFO_FULL_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH2_ST Represents GDMA_EVT_IN_FIFO_FULL_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.7. SOC_ETM_EVT_ST4_REG (0x01C8)

(reserved)																SOC_ETM_PMU_EVT_SLEEP_WAKEUP_ST SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH2_ST SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH1_ST SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH0_ST SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH2_ST SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH1_ST SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH0_ST SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH2_ST SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST SOC_ETM_GDMA_EVT_OUT_EOF_CH2_ST SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST SOC_ETM_GDMA_EVT_OUT_EOF_CH2_ST SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST																		
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset									

SOC_ETM_GDMA_EVT_OUT_DONE_CH0_ST Represents GDMA_EVT_OUT_DONE_CH0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_DONE_CH1_ST Represents GDMA_EVT_OUT_DONE_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_DONE_CH2_ST Represents GDMA_EVT_OUT_DONE_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_EOF_CH0_ST Represents GDMA_EVT_OUT_EOF_CH0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST Represents GDMA_EVT_OUT_EOF_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_EOF_CH2_ST Represents GDMA_EVT_OUT_EOF_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.7. SOC_ETM_EVT_ST4_REG (0x01C8)

Continued from the previous page...

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CHO_ST Represents GDMA_EVT_OUT_TOTAL_EOF_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH1_ST Represents GDMA_EVT_OUT_TOTAL_EOF_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH2_ST Represents GDMA_EVT_OUT_TOTAL_EOF_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CHO_ST Represents GDMA_EVT_OUT_FIFO_EMPTY_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH1_ST Represents GDMA_EVT_OUT_FIFO_EMPTY_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH2_ST Represents GDMA_EVT_OUT_FIFO_EMPTY_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CHO_ST Represents GDMA_EVT_OUT_FIFO_FULL_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.7. SOC_ETM_EVT_ST4_REG (0x01C8)

Continued from the previous page...

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH1_ST Represents GDMA_EVT_OUT_FIFO_FULL_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH2_ST Represents GDMA_EVT_OUT_FIFO_FULL_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_PMU_EVT_SLEEP_WEEKUP_ST Represents PMU_EVT_SLEEP_WEEKUP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

[illegible]

0: Not triggered
1: Triggered
(R/WTC/SS)

0: Not triggered
1: Triggered
(R/WTC/SS)

0: Not triggered
1: Triggered
(R/WTC/SS)

0: Not triggered
1: Triggered
(R/WTC/SS)

0: Not triggered
1: Triggered
(R/WTC/SS)

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.8. SOC_ETM_TASK_STO_REG (0x01D0)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH6_SET_ST Represents GPIO_TASK_CH6_SET trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH7_SET_ST Represents GPIO_TASK_CH7_SET trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH0_CLEAR_ST Represents GPIO_TASK_CH0_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH1_CLEAR_ST Represents GPIO_TASK_CH1_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH2_CLEAR_ST Represents GPIO_TASK_CH2_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH3_CLEAR_ST Represents GPIO_TASK_CH3_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH4_CLEAR_ST Represents GPIO_TASK_CH4_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH5_CLEAR_ST Represents GPIO_TASK_CH5_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.8. SOC_ETM_TASK_STO_REG (0x01D0)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH6_CLEAR_ST Represents GPIO_TASK_CH6_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH7_CLEAR_ST Represents GPIO_TASK_CH7_CLEAR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH0_TOGGLE_ST Represents GPIO_TASK_CH0_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH1_TOGGLE_ST Represents GPIO_TASK_CH1_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH2_TOGGLE_ST Represents GPIO_TASK_CH2_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH3_TOGGLE_ST Represents GPIO_TASK_CH3_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH4_TOGGLE_ST Represents GPIO_TASK_CH4_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH5_TOGGLE_ST Represents GPIO_TASK_CH5_TOGGLE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.8. SOC_ETM_TASK_STO_REG (0x01D0)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH6_TOGGLE_ST Represents GPIO_TASK_CH6_TOGGLE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GPIO_TASK_CH7_TOGGLE_ST Represents GPIO_TASK_CH7_TOGGLE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER0_RES_UPDATE_ST Represents LEDC_TASK_TIMER0_RES_UPDATE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER1_RES_UPDATE_ST Represents LEDC_TASK_TIMER1_RES_UPDATE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER2_RES_UPDATE_ST Represents LEDC_TASK_TIMER2_RES_UPDATE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER3_RES_UPDATE_ST Represents LEDC_TASK_TIMER3_RES_UPDATE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CHO_ST Represents LEDC_TASK_DUTY_SCALE_UPDATE_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.8. SOC_ETM_TASK_ST0_REG (0x01D0)

Continued from the previous page...

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH1_ST Represents LEDC_TASK_DUTY_SCALE_UPDATE_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH2_ST Represents LEDC_TASK_DUTY_SCALE_UPDATE_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH3_ST Represents LEDC_TASK_DUTY_SCALE_UPDATE_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.9. SOC_ETM_TASK_ST1_REG (0x01D8)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH1_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH0_ST																																SOC_ETM_LEDC_TASK_TIMER3_PAUSE_ST																																SOC_ETM_LEDC_TASK_TIMER2_PAUSE_ST																																SOC_ETM_LEDC_TASK_TIMER1_PAUSE_ST																																SOC_ETM_LEDC_TASK_TIMER0_PAUSE_ST																																SOC_ETM_LEDC_TASK_TIMER3_RESUME_ST																																SOC_ETM_LEDC_TASK_TIMER2_RESUME_ST																																SOC_ETM_LEDC_TASK_TIMER1_RESUME_ST																																SOC_ETM_LEDC_TASK_TIMER0_RESUME_ST																																SOC_ETM_LEDC_TASK_TIMER3_RST_ST																																SOC_ETM_LEDC_TASK_TIMER2_RST_ST																																SOC_ETM_LEDC_TASK_TIMER1_RST_ST																																SOC_ETM_LEDC_TASK_TIMER0_RST_ST																																SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH5_ST																																SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH4_ST																																SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH3_ST																																SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH2_ST																																SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH1_ST																																SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH0_ST																																SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH5_ST																																SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH4_ST																																SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH3_ST																																SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH2_ST																																SOC_ETM_LEDC_TASK_TIMER3_CAP_ST																																SOC_ETM_LEDC_TASK_TIMER2_CAP_ST																																SOC_ETM_LEDC_TASK_TIMER1_CAP_ST																																SOC_ETM_LEDC_TASK_TIMER0_CAP_ST																																SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH5_ST																																SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH4_ST																																																																																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH4_ST Represents

LEDC_TASK_DUTY_SCALE_UPDATE_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH5_ST Represents

LEDC_TASK_DUTY_SCALE_UPDATE_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER0_CAP_ST Represents LEDC_TASK_TIMER0_CAP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER1_CAP_ST Represents LEDC_TASK_TIMER1_CAP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER2_CAP_ST Represents LEDC_TASK_TIMER2_CAP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER3_CAP_ST Represents LEDC_TASK_TIMER3_CAP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.9. SOC_ETM_TASK_ST1_REG (0x01D8)

Continued from the previous page...

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CHO_ST Represents LEDC_TASK_SIG_OUT_DIS_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH1_ST Represents LEDC_TASK_SIG_OUT_DIS_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH2_ST Represents LEDC_TASK_SIG_OUT_DIS_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH3_ST Represents LEDC_TASK_SIG_OUT_DIS_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH4_ST Represents LEDC_TASK_SIG_OUT_DIS_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH5_ST Represents LEDC_TASK_SIG_OUT_DIS_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CHO_ST Represents LEDC_TASK_OVF_CNT_RST_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.9. SOC_ETM_TASK_ST1_REG (0x01D8)

Continued from the previous page...

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH1_ST Represents LEDC_TASK_OVF_CNT_RST_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH2_ST Represents LEDC_TASK_OVF_CNT_RST_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH3_ST Represents LEDC_TASK_OVF_CNT_RST_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH4_ST Represents LEDC_TASK_OVF_CNT_RST_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH5_ST Represents LEDC_TASK_OVF_CNT_RST_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER0_RST_ST Represents LEDC_TASK_TIMER0_RST trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER1_RST_ST Represents LEDC_TASK_TIMER1_RST trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.9. SOC_ETM_TASK_ST1_REG (0x01D8)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER2_RST_ST Represents LEDC_TASK_TIMER2_RST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER3_RST_ST Represents LEDC_TASK_TIMER3_RST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER0_RESUME_ST Represents LEDC_TASK_TIMER0_RESUME trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER1_RESUME_ST Represents LEDC_TASK_TIMER1_RESUME trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER2_RESUME_ST Represents LEDC_TASK_TIMER2_RESUME trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER3_RESUME_ST Represents LEDC_TASK_TIMER3_RESUME trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER0_PAUSE_ST Represents LEDC_TASK_TIMER0_PAUSE trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.9. SOC_ETM_TASK_ST1_REG (0x01D8)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER1_PAUSE_ST Represents LEDC_TASK_TIMER1_PAUSE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER2_PAUSE_ST Represents LEDC_TASK_TIMER2_PAUSE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_TIMER3_PAUSE_ST Represents LEDC_TASK_TIMER3_PAUSE trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CHO_ST Represents LEDC_TASK_GAMMA_RESTART_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH1_ST Represents LEDC_TASK_GAMMA_RESTART_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.10. SOC_ETM_TASK_ST2_REG (0x01E0)

SOC_ETM_TG1_TASK_CNT_START_TIMER1_ST																																SOC_ETM_TG1_TASK_CNT_CAP_TIMER0_ST																																SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER0_ST																																SOC_ETM_TG1_TASK_CNT_STOP_TIMER0_ST																																SOC_ETM_TG1_TASK_ALARM_START_TIMER0_ST																																SOC_ETM_TG0_TASK_CNT_START_TIMER0_ST																																SOC_ETM_TG0_TASK_CNT_CAP_TIMER0_ST																																SOC_ETM_TG0_TASK_CNT_RELOAD_TIMER0_ST																																SOC_ETM_TG0_TASK_CNT_STOP_TIMER0_ST																																SOC_ETM_TG0_TASK_ALARM_START_TIMER1_ST																																SOC_ETM_TG0_TASK_CNT_START_TIMER1_ST																																SOC_ETM_TG0_TASK_CNT_CAP_TIMER1_ST																																SOC_ETM_TG0_TASK_CNT_RELOAD_TIMER1_ST																																SOC_ETM_TG0_TASK_CNT_STOP_TIMER1_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH5_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH4_ST_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH3_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH2_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH1_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH0_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH5_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH4_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH3_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH2_ST																																SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH1_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH0_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH5_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH4_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH3_ST																																SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH2_ST																																																																																						
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

Reset

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH2_ST Represents LEDC_TASK_GAMMA_RESTART_CH2 trigger status.
 0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH3_ST Represents LEDC_TASK_GAMMA_RESTART_CH3 trigger status.
 0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH4_ST Represents LEDC_TASK_GAMMA_RESTART_CH4 trigger status.
 0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH5_ST Represents LEDC_TASK_GAMMA_RESTART_CH5 trigger status.
 0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH0_ST Represents LEDC_TASK_GAMMA_PAUSE_CH0 trigger status.
 0: Not triggered
 1: Triggered
 (R/WTC/SS)

Continued on the next page...

Register 8.10. SOC_ETM_TASK_ST2_REG (0x01E0)

Continued from the previous page...

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH1_ST Represents LEDC_TASK_GAMMA_PAUSE_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH2_ST Represents LEDC_TASK_GAMMA_PAUSE_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH3_ST Represents LEDC_TASK_GAMMA_PAUSE_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH4_ST Represents LEDC_TASK_GAMMA_PAUSE_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH5_ST Represents LEDC_TASK_GAMMA_PAUSE_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CHO_ST Represents LEDC_TASK_GAMMA_RESUME_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH1_ST Represents LEDC_TASK_GAMMA_RESUME_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.10. SOC_ETM_TASK_ST2_REG (0x01E0)

Continued from the previous page...

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH2_ST Represents LEDC_TASK_GAMMA_RESUME_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH3_ST Represents LEDC_TASK_GAMMA_RESUME_CH3 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH4_ST Represents LEDC_TASK_GAMMA_RESUME_CH4 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH5_ST Represents LEDC_TASK_GAMMA_RESUME_CH5 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_START_TIMER0_ST Represents TGO_TASK_CNT_START_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_ALARM_START_TIMER0_ST Represents TGO_TASK_ALARM_START_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_STOP_TIMER0_ST Represents TGO_TASK_CNT_STOP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.10. SOC_ETM_TASK_ST2_REG (0x01E0)

Continued from the previous page...

SOC_ETM_TGO_TASK_CNT_RELOAD_TIMER0_ST Represents TGO_TASK_CNT_RELOAD_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_CAP_TIMER0_ST Represents TGO_TASK_CNT_CAP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_START_TIMER1_ST Represents TGO_TASK_CNT_START_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_ALARM_START_TIMER1_ST Represents TGO_TASK_ALARM_START_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_STOP_TIMER1_ST Represents TGO_TASK_CNT_STOP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_RELOAD_TIMER1_ST Represents TGO_TASK_CNT_RELOAD_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TGO_TASK_CNT_CAP_TIMER1_ST Represents TGO_TASK_CNT_CAP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.10. SOC_ETM_TASK_ST2_REG (0x01E0)

Continued from the previous page...

SOC_ETM_TG1_TASK_CNT_START_TIMER0_ST Represents TG1_TASK_CNT_START_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_ALARM_START_TIMER0_ST Represents TG1_TASK_ALARM_START_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_STOP_TIMER0_ST Represents TG1_TASK_CNT_STOP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER0_ST Represents TG1_TASK_CNT_RELOAD_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_CAP_TIMER0_ST Represents TG1_TASK_CNT_CAP_TIMER0 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_START_TIMER1_ST Represents TG1_TASK_CNT_START_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.11. SOC_ETM_TASK_ST3_REG (0x01E8)

(reserved) SOC_ETM_ADC_TASK_STOPO_ST SOC_ETM_ADC_TASK_STARTO_ST (reserved) SOC_ETM_MCPWMO_TASK_SAMPLEO_ST SOC_ETM_MCPWMO_TASK_CAP2_ST SOC_ETM_MCPWMO_TASK_CAP1_ST SOC_ETM_MCPWMO_TASK_CAIPO_ST SOC_ETM_MCPWMO_TASK_CLR2_OST_ST SOC_ETM_MCPWMO_TASK_CLR1_OST_ST SOC_ETM_MCPWMO_TASK_CLRO_OST_ST SOC_ETM_MCPWMO_TASK_TZ2_OST_ST SOC_ETM_MCPWMO_TASK_TZ1_OST_ST SOC_ETM_MCPWMO_TASK_TZO_OST_ST SOC_ETM_MCPWMO_TASK_TIMER2_PERIOD_UP_ST SOC_ETM_MCPWMO_TASK_TIMER1_PERIOD_UP_ST SOC_ETM_MCPWMO_TASK_TIMER2_PERIOD_UP_ST SOC_ETM_MCPWMO_TASK_TIMER2_SYN_ST SOC_ETM_MCPWMO_TASK_TIMER1_SYN_ST SOC_ETM_MCPWMO_TASK_TIMER0_SYN_ST SOC_ETM_MCPWMO_TASK_GEN_STOP_ST SOC_ETM_MCPWMO_TASK_CMPR2_B_UP_ST SOC_ETM_MCPWMO_TASK_CMPRI_B_UP_ST SOC_ETM_MCPWMO_TASK_CMPR2_B_UP_ST SOC_ETM_MCPWMO_TASK_CMPR2_A_UP_ST SOC_ETM_TG1_TASK_CMPRI_A_UP_ST SOC_ETM_TG1_TASK_CNT_CAP_TIMER1_A_UP_ST SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER1_ST SOC_ETM_TG1_TASK_ALARM_START_TIMER1_ST																																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Reset

SOC_ETM_TG1_TASK_ALARM_START_TIMER1_ST Represents TG1_TASK_ALARM_START_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_STOP_TIMER1_ST Represents TG1_TASK_CNT_STOP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER1_ST Represents TG1_TASK_CNT_RELOAD_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_TG1_TASK_CNT_CAP_TIMER1_ST Represents TG1_TASK_CNT_CAP_TIMER1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CMPRO_A_UP_ST Represents MCPWMO_TASK_CMPRO_A_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.11. SOC_ETM_TASK_ST3_REG (0x01E8)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_CMPR1_A_UP_ST Represents MCPWMO_TASK_CMPR1_A_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CMPR2_A_UP_ST Represents MCPWMO_TASK_CMPR2_A_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CMPRO_B_UP_ST Represents MCPWMO_TASK_CMPRO_B_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CMPR1_B_UP_ST Represents MCPWMO_TASK_CMPR1_B_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CMPR2_B_UP_ST Represents MCPWMO_TASK_CMPR2_B_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_GEN_STOP_ST Represents MCPWMO_TASK_GEN_STOP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TIMER0_SYN_ST Represents MCPWMO_TASK_TIMER0_SYN trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.11. SOC_ETM_TASK_ST3_REG (0x01E8)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_TIMER1_SYN_ST Represents MCPWMO_TASK_TIMER1_SYN trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TIMER2_SYN_ST Represents MCPWMO_TASK_TIMER2_SYN trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TIMER0_PERIOD_UP_ST Represents MCPWMO_TASK_TIMER0_PERIOD_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TIMER1_PERIOD_UP_ST Represents MCPWMO_TASK_TIMER1_PERIOD_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TIMER2_PERIOD_UP_ST Represents MCPWMO_TASK_TIMER2_PERIOD_UP trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TZ0_OST_ST Represents MCPWMO_TASK_TZ0_OST trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TZ1_OST_ST Represents MCPWMO_TASK_TZ1_OST trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_TZ2_OST_ST Represents MCPWMO_TASK_TZ2_OST trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Continued on the next page...

Register 8.11. SOC_ETM_TASK_ST3_REG (0x01E8)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_CLR0_OST_ST Represents MCPWMO_TASK_CLR0_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CLR1_OST_ST Represents MCPWMO_TASK_CLR1_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CLR2_OST_ST Represents MCPWMO_TASK_CLR2_OST trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CAPO_ST Represents MCPWMO_TASK_CAPO trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CAP1_ST Represents MCPWMO_TASK_CAP1 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_MCPWMO_TASK_CAP2_ST Represents MCPWMO_TASK_CAP2 trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ADC_TASK_SAMPLEO_ST Represents ADC_TASK_SAMPLEO trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.11. SOC_ETM_TASK_ST3_REG (0x01E8)

Continued from the previous page...

SOC_ETM_ADC_TASK_STARTO_ST Represents ADC_TASK_STARTO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_ADC_TASK_STOPO_ST Represents ADC_TASK_STOPO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.12. SOC_ETM_TASK_ST4_REG (0x01F0)

(reserved)												SOC_ETM_PMU_TASK_SLEEP_REQ_ST SOC_ETM_GDMA_TASK_OUT_START_CH2_ST SOC_ETM_GDMA_TASK_OUT_START_CH1_ST SOC_ETM_GDMA_TASK_OUT_START_CH0_ST SOC_ETM_RTC_TASK_TRIGGERFLW_ST (reserved) SOC_ETM_ULP_TASK_WAKEUP_CPU_ST SOC_ETM_I2SO_TASK_STOP_TX_ST SOC_ETM_I2SO_TASK_STOP_RX_ST SOC_ETM_I2SO_TASK_START_TX_ST SOC_ETM_I2SO_TASK_START_RX_ST (reserved)																				
31											21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset	

SOC_ETM_TMPSNSR_TASK_START_SAMPLE_ST Represents TMPSNSR_TASK_START_SAMPLE trigger status.

0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_TMPSNSR_TASK_STOP_SAMPLE_ST Represents TMPSNSR_TASK_STOP_SAMPLE trigger status.

0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_I2SO_TASK_START_RX_ST Represents I2SO_TASK_START_RX trigger status.

0: Not triggered
 1: Triggered
 (R/WTC/SS)

SOC_ETM_I2SO_TASK_START_TX_ST Represents I2SO_TASK_START_TX trigger status.

0: Not triggered
 1: Triggered
 (R/WTC/SS)

Continued on the next page...

Register 8.12. SOC_ETM_TASK_ST4_REG (0x01F0)

Continued from the previous page...

SOC_ETM_I2SO_TASK_STOP_RX_ST Represents I2SO_TASK_STOP_RX trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_I2SO_TASK_STOP_TX_ST Represents I2SO_TASK_STOP_TX trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_ULP_TASK_WAKEUP_CPU_ST Represents ULP_TASK_WAKEUP_CPU trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_TASK_START_ST Represents RTC_TASK_START trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_TASK_STOP_ST Represents RTC_TASK_STOP trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_TASK_CLR_ST Represents RTC_TASK_CLR trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

SOC_ETM_RTC_TASK_TRIGGERFLW_ST Represents RTC_TASK_TRIGGERFLW trigger status.

0: Not triggered
1: Triggered
(R/WTC/SS)

Continued on the next page...

Register 8.12. SOC_ETM_TASK_ST4_REG (0x01F0)

Continued from the previous page...

SOC_ETM_GDMA_TASK_IN_START_CHO_ST Represents GDMA_TASK_IN_START_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_TASK_IN_START_CH1_ST Represents GDMA_TASK_IN_START_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_TASK_IN_START_CH2_ST Represents GDMA_TASK_IN_START_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_TASK_OUT_START_CHO_ST Represents GDMA_TASK_OUT_START_CHO trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_TASK_OUT_START_CH1_ST Represents GDMA_TASK_OUT_START_CH1 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_GDMA_TASK_OUT_START_CH2_ST Represents GDMA_TASK_OUT_START_CH2 trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

SOC_ETM_PMU_TASK_SLEEP_REQ_ST Represents PMU_TASK_SLEEP_REQ trigger status.

0: Not triggered

1: Triggered

(R/WTC/SS)

Register 8.13. SOC_ETM_CH_ENA_ADO_SET_REG (0x0004)

SOC_ETM_CH_ENABLE31	SOC_ETM_CH_ENABLE30	SOC_ETM_CH_ENABLE29	SOC_ETM_CH_ENABLE28	SOC_ETM_CH_ENABLE27	SOC_ETM_CH_ENABLE26	SOC_ETM_CH_ENABLE25	SOC_ETM_CH_ENABLE24	SOC_ETM_CH_ENABLE23	SOC_ETM_CH_ENABLE22	SOC_ETM_CH_ENABLE21	SOC_ETM_CH_ENABLE20	SOC_ETM_CH_ENABLE19	SOC_ETM_CH_ENABLE18	SOC_ETM_CH_ENABLE17	SOC_ETM_CH_ENABLE16	SOC_ETM_CH_ENABLE15	SOC_ETM_CH_ENABLE14	SOC_ETM_CH_ENABLE13	SOC_ETM_CH_ENABLE12	SOC_ETM_CH_ENABLE11	SOC_ETM_CH_ENABLE10	SOC_ETM_CH_ENABLE9	SOC_ETM_CH_ENABLE8	SOC_ETM_CH_ENABLE7	SOC_ETM_CH_ENABLE6	SOC_ETM_CH_ENABLE5	SOC_ETM_CH_ENABLE4	SOC_ETM_CH_ENABLE3	SOC_ETM_CH_ENABLE2	SOC_ETM_CH_ENABLE1	SOC_ETM_CH_ENABLE0
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

SOC_ETM_CH_ENABLE n (n : 0-31) Configures whether or not to enable channel n .

0: Invalid. No effect

1: Enable

(WT)

Register 8.14. SOC_ETM_CH_ENA_ADO_CLR_REG (0x0008)

SOC_ETM_CH_DISABLE31	SOC_ETM_CH_DISABLE30	SOC_ETM_CH_DISABLE29	SOC_ETM_CH_DISABLE28	SOC_ETM_CH_DISABLE27	SOC_ETM_CH_DISABLE26	SOC_ETM_CH_DISABLE25	SOC_ETM_CH_DISABLE24	SOC_ETM_CH_DISABLE23	SOC_ETM_CH_DISABLE22	SOC_ETM_CH_DISABLE21	SOC_ETM_CH_DISABLE20	SOC_ETM_CH_DISABLE19	SOC_ETM_CH_DISABLE18	SOC_ETM_CH_DISABLE17	SOC_ETM_CH_DISABLE16	SOC_ETM_CH_DISABLE15	SOC_ETM_CH_DISABLE14	SOC_ETM_CH_DISABLE13	SOC_ETM_CH_DISABLE12	SOC_ETM_CH_DISABLE11	SOC_ETM_CH_DISABLE10	SOC_ETM_CH_DISABLE9	SOC_ETM_CH_DISABLE8	SOC_ETM_CH_DISABLE7	SOC_ETM_CH_DISABLE6	SOC_ETM_CH_DISABLE5	SOC_ETM_CH_DISABLE4	SOC_ETM_CH_DISABLE3	SOC_ETM_CH_DISABLE2	SOC_ETM_CH_DISABLE1	SOC_ETM_CH_DISABLE0
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

SOC_ETM_CH_DISABLE n (n : 0-31) Configures whether or not to disable channel n .

0: Invalid. No effect

1: Clear

(WT)

Register 8.15. SOC_ETM_CH_ENA_AD1_SET_REG (0x0010)

(reserved)																		SOC_ETM_CH_ENABLE49 SOC_ETM_CH_ENABLE48 SOC_ETM_CH_ENABLE47 SOC_ETM_CH_ENABLE46 SOC_ETM_CH_ENABLE45 SOC_ETM_CH_ENABLE44 SOC_ETM_CH_ENABLE43 SOC_ETM_CH_ENABLE42 SOC_ETM_CH_ENABLE41 SOC_ETM_CH_ENABLE40 SOC_ETM_CH_ENABLE39 SOC_ETM_CH_ENABLE38 SOC_ETM_CH_ENABLE37 SOC_ETM_CH_ENABLE36 SOC_ETM_CH_ENABLE35 SOC_ETM_CH_ENABLE34 SOC_ETM_CH_ENABLE33																				
31																		18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SOC_ETM_CH_ENABLE n (n : 32-49) Configures whether or not to enable channel n .

0: Invalid. No effect

1: Enable

(WT)

Register 8.16. SOC_ETM_CH_ENA_AD1_CLR_REG (0x0014)

(reserved)																		SOC_ETM_CH_DISABLE49 SOC_ETM_CH_DISABLE48 SOC_ETM_CH_DISABLE47 SOC_ETM_CH_DISABLE46 SOC_ETM_CH_DISABLE45 SOC_ETM_CH_DISABLE44 SOC_ETM_CH_DISABLE43 SOC_ETM_CH_DISABLE42 SOC_ETM_CH_DISABLE41 SOC_ETM_CH_DISABLE40 SOC_ETM_CH_DISABLE39 SOC_ETM_CH_DISABLE38 SOC_ETM_CH_DISABLE37 SOC_ETM_CH_DISABLE36 SOC_ETM_CH_DISABLE35 SOC_ETM_CH_DISABLE34 SOC_ETM_CH_DISABLE33																				
31																		18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset					

SOC_ETM_CH_DISABLE n (n : 32-49) Configures whether or not to disable channel n .

0: Invalid. No effect

1: Clear

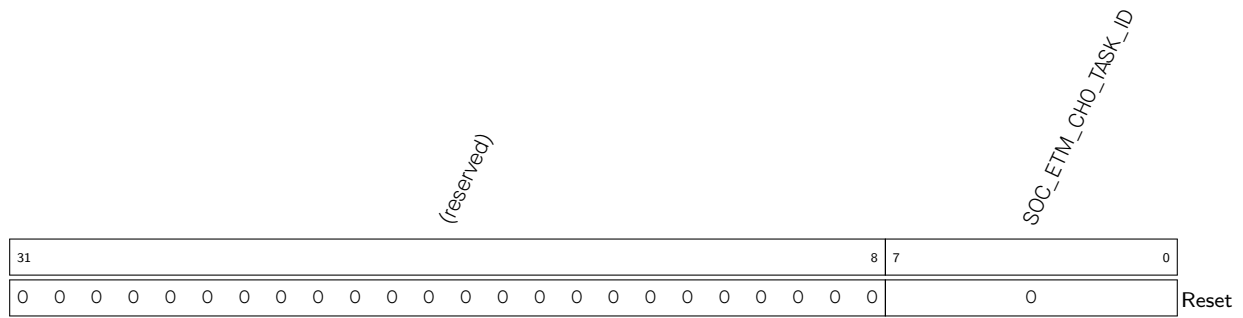
(WT)

Register 8.17. SOC_ETM_CH n _EVT_ID_REG (n : 0-49) (0x0018+0x8* n)

(reserved)																		SOC_ETM_CHO_EVT_ID																		
31																		8	7																	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																Reset

SOC_ETM_CH n _EVT_ID Configures channel n event ID. (R/W)

Register 8.18. SOC_ETM_CH*n*_TASK_ID_REG (*n*: 0-49) (0x001C+0x8**n*)



SOC_ETM_CH n _TASK_ID Configures channel n task ID. (R/W)

Register 8.19. SOC_ETM_EVT_STO_CLR_REG (0x01AC)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH5_ST_CLR																														
SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH4_ST_CLR																														
SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH3_ST_CLR																														
SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH2_ST_CLR																														
SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH1_ST_CLR																														
SOC_ETM_GPIO_EVT_ZERO_DET_NEGO_ST_CLR																														
SOC_ETM_GPIO_EVT_ZERO_DET_CH7_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH6_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH5_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH4_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH3_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH2_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH1_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CHO_ANY_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH7_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH6_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH5_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH4_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH3_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH2_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH1_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CHO_FALL_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH7_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH6_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH5_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH4_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH3_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH2_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CH1_RISE_EDGE_ST_CLR																														
SOC_ETM_GPIO_EVT_CHO_RISE_EDGE_ST_CLR																														
Reset																														

SOC_ETM_GPIO_EVT_CHO_RISE_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CHO_RISE_EDGE trigger status.
 0: Invalid. No effect
 1: Clear
 (WT)

SOC_ETM_GPIO_EVT_CH1_RISE_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH1_RISE_EDGE trigger status.
 0: Invalid. No effect
 1: Clear
 (WT)

SOC_ETM_GPIO_EVT_CH2_RISE_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH2_RISE_EDGE trigger status.
 0: Invalid. No effect
 1: Clear
 (WT)

SOC_ETM_GPIO_EVT_CH3_RISE_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH3_RISE_EDGE trigger status.
 0: Invalid. No effect
 1: Clear
 (WT)

SOC_ETM_GPIO_EVT_CH4_RISE_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH4_RISE_EDGE trigger status.
 0: Invalid. No effect
 1: Clear
 (WT)

Continued on the next page...

Register 8.19. SOC_ETM_EVT_STO_CLR_REG (0x01AC)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH5_RISE_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH5_RISE_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH6_RISE_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH6_RISE_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH7_RISE_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH7_RISE_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH0_FALL_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH0_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH1_FALL_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH1_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH2_FALL_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH2_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH3_FALL_EDGE_ST_CLR Configures whether or not to clear
GPIO_EVT_CH3_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.19. SOC_ETM_EVT_STO_CLR_REG (0x01AC)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH4_FALL_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH4_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH5_FALL_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH5_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH6_FALL_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH6_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH7_FALL_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH7_FALL_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CHO_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CHO_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH1_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH1_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH2_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH2_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.19. SOC_ETM_EVT_STO_CLR_REG (0x01AC)

Continued from the previous page...

SOC_ETM_GPIO_EVT_CH3_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH3_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH4_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH4_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH5_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH5_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH6_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH6_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_CH7_ANY_EDGE_ST_CLR Configures whether or not to clear GPIO_EVT_CH7_ANY_EDGE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_ZERO_DET_POS0_ST_CLR Configures whether or not to clear GPIO_EVT_ZERO_DET_POS0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_EVT_ZERO_DET_NEGO_ST_CLR Configures whether or not to clear GPIO_EVT_ZERO_DET_NEGO trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.19. SOC_ETM_EVT_STO_CLR_REG (0x01AC)

Continued from the previous page...

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CHO_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH1_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH2_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH3_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH4_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_DUTY_CHNG_END_CH5_ST_CLR Configures whether or not to clear LEDC_EVT_DUTY_CHNG_END_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OPI_TEA_ST_CLR
SOC_ETM_MCPWMO_EVT_OPO_TEA_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER2_TEP_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER1_TEP_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER0_TEP_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER2_TEZ_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER1_TEZ_ST_CLR
SOC_ETM_MCPWMO_EVT_TIMER0_TEZ_ST_CLR
SOC_ETM_SYSTIMER_EVT_TIMER1_STOP_ST_CLR
SOC_ETM_SYSTIMER_EVT_TIMER0_STOP_ST_CLR
SOC_ETM_SYSTIMER_EVT_CNT_CMP2_STOP_ST_CLR
SOC_ETM_TG1_EVT_CNT_CMP1_ST_CLR
SOC_ETM_TG0_EVT_CNT_CMP_TIMER1_ST_CLR
SOC_ETM_LEDC_EVT_CMP_TIMER0_ST_CLR
SOC_ETM_LEDC_EVT_TIMER3_CMP_ST_CLR
SOC_ETM_LEDC_EVT_TIMER2_CMP_ST_CLR
SOC_ETM_LEDC_EVT_TIMER0_CMP_ST_CLR
SOC_ETM_LEDC_EVT_TIME_OVF_TIMER3_ST_CLR
SOC_ETM_LEDC_EVT_TIME_OVF_TIMER2_ST_CLR
SOC_ETM_LEDC_EVT_OVF_TIMER0_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH5_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH4_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH3_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH2_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH1_ST_CLR
SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH0_ST_CLR

[illegible]

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CHO_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH1_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH2_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH3_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH4_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.20. SOC_ETM_EVT_ST1_CLR_REG (0x01B4)

Continued from the previous page...

SOC_ETM_LEDC_EVT_OVF_CNT_PLS_CH5_ST_CLR Configures whether or not to clear LEDC_EVT_OVF_CNT_PLS_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER0_ST_CLR Configures whether or not to clear LEDC_EVT_TIME_OVF_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER1_ST_CLR Configures whether or not to clear LEDC_EVT_TIME_OVF_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER2_ST_CLR Configures whether or not to clear LEDC_EVT_TIME_OVF_TIMER2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIME_OVF_TIMER3_ST_CLR Configures whether or not to clear LEDC_EVT_TIME_OVF_TIMER3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIMER0_CMP_ST_CLR Configures whether or not to clear LEDC_EVT_TIMER0_CMP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIMER1_CMP_ST_CLR Configures whether or not to clear LEDC_EVT_TIMER1_CMP trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.20. SOC_ETM_EVT_ST1_CLR_REG (0x01B4)

Continued from the previous page...

SOC_ETM_LEDC_EVT_TIMER2_CMP_ST_CLR Configures whether or not to clear LEDC_EVT_TIMER2_CMP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_EVT_TIMER3_CMP_ST_CLR Configures whether or not to clear LEDC_EVT_TIMER3_CMP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_EVT_CNT_CMP_TIMER0_ST_CLR Configures whether or not to clear TGO_EVT_CNT_CMP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_EVT_CNT_CMP_TIMER1_ST_CLR Configures whether or not to clear TGO_EVT_CNT_CMP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_EVT_CNT_CMP_TIMER0_ST_CLR Configures whether or not to clear TG1_EVT_CNT_CMP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_EVT_CNT_CMP_TIMER1_ST_CLR Configures whether or not to clear TG1_EVT_CNT_CMP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_SYSTIMER_EVT_CNT_CMPO_ST_CLR Configures whether or not to clear SYS-TIMER_EVT_CNT_CMPO trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.20. SOC_ETM_EVT_ST1_CLR_REG (0x01B4)

Continued from the previous page...

SOC_ETM_SYSTIMER_EVT_CNT_CMP1_ST_CLR Configures whether or not to clear SYS-TIMER_EVT_CNT_CMP1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_SYSTIMER_EVT_CNT_CMP2_ST_CLR Configures whether or not to clear SYS-TIMER_EVT_CNT_CMP2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER0_STOP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER0_STOP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER1_STOP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER1_STOP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER2_STOP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER2_STOP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER0_TEZ_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER0_TEZ trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER1_TEZ_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER1_TEZ trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.20. SOC_ETM_EVT_ST1_CLR_REG (0x01B4)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_TIMER2_TEZ_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER2_TEZ trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER0_TEP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER0_TEP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER1_TEP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER1_TEP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TIMER2_TEP_ST_CLR Configures whether or not to clear MCPWMO_EVT_TIMER2_TEP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OPO_TEA_ST_CLR Configures whether or not to clear MCPWMO_EVT_OPO_TEA trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OP1_TEA_ST_CLR Configures whether or not to clear MCPWMO_EVT_OP1_TEA trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_STOPPED0_ST_CLR
SOC_ETM_ADC_EVT_RESULT_DONE0_ST_CLR
SOC_ETM_ADC_EVT_EQ_BELOW_THRESH1_ST_CLR
SOC_ETM_ADC_EVT_EQ_BELOW_THRESH1_ST_CLR
SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH1_ST_CLR
SOC_ETM_ADC_EVT_CONV_CMP1TO_ST_CLR
SOC_ETM_MCPWMO_EVT_OP2_TEE2_ST_CLR
SOC_ETM_MCPWMO_EVT_OP1_TEE2_ST_CLR
SOC_ETM_MCPWMO_EVT_OP0_TEE2_ST_CLR
SOC_ETM_MCPWMO_EVT_OP2_TEE2_ST_CLR
SOC_ETM_MCPWMO_EVT_OP1_TEE1_ST_CLR
SOC_ETM_MCPWMO_EVT_OP0_TEE1_ST_CLR
SOC_ETM_MCPWMO_EVT_CAP2_ST_CLR
SOC_ETM_MCPWMO_EVT_CAP1_ST_CLR
SOC_ETM_MCPWMO_EVT_CAP0_ST_CLR
SOC_ETM_MCPWMO_EVT_TZ2_OST_ST_CLR
SOC_ETM_MCPWMO_EVT_TZ1_OST_ST_CLR
SOC_ETM_MCPWMO_EVT_TZ0_OST_ST_CLR
SOC_ETM_MCPWMO_EVT_TZ2_OBC_ST_CLR
SOC_ETM_MCPWMO_EVT_TZ0_OBC_ST_CLR
SOC_ETM_MCPWMO_EVT_F2_OBC_ST_CLR
SOC_ETM_MCPWMO_EVT_F1_CLR_ST_CLR
SOC_ETM_MCPWMO_EVT_F0_CLR_ST_CLR
SOC_ETM_MCPWMO_EVT_F2_ST_CLR
SOC_ETM_MCPWMO_EVT_F1_ST_CLR
SOC_ETM_MCPWMO_EVT_F0_ST_CLR
SOC_ETM_MCPWMO_EVT_OP2_TEB_ST_CLR
SOC_ETM_MCPWMO_EVT_OP0_TEB_ST_CLR
SOC_ETM_MCPWMO_EVT_OP2_TEA_ST_CLR

[illegible]

SOC_ETM_MCPWMO_EVT_OP2_TEA_ST_CLR	Configures	whether	or	not	to	clear
MCPWMO_EVT_OP2_TEA trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						

SOC_ETM_MCPWMO_EVT_OPO_TEB_ST_CLR	Configures	whether	or	not	to	clear
MCPWMO_EVT_OPO_TEB trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						

SOC_ETM_MCPWMO_EVT_OP1_TEB_ST_CLR	Configures	whether	or	not	to	clear
MCPWMO_EVT_OP1_TEB trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						

SOC_ETM_MCPWMO_EVT_OP2_TEB_ST_CLR	Configures	whether	or	not	to	clear
MCPWMO_EVT_OP2_TEB trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						

SOC_ETM_MCPWMO_EVT_F0_ST_CLR Configures whether or not to clear MCPWMO_EVT_F0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.21. SOC_ETM_EVT_ST2_CLR_REG (0x01BC)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_F1_ST_CLR Configures whether or not to clear MCPWMO_EVT_F1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_F2_ST_CLR Configures whether or not to clear MCPWMO_EVT_F2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_F0_CLR_ST_CLR Configures whether or not to clear MCPWMO_EVT_F0_CLR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_F1_CLR_ST_CLR Configures whether or not to clear MCPWMO_EVT_F1_CLR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_F2_CLR_ST_CLR Configures whether or not to clear MCPWMO_EVT_F2_CLR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TZ0_CBC_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ0_CBC trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TZ1_CBC_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ1_CBC trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.21. SOC_ETM_EVT_ST2_CLR_REG (0x01BC)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_TZ2_CBC_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ2_CBC trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TZ0_OST_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ0_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TZ1_OST_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ1_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_TZ2_OST_ST_CLR Configures whether or not to clear MCPWMO_EVT_TZ2_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_CAPO_ST_CLR Configures whether or not to clear MCPWMO_EVT_CAPO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_CAP1_ST_CLR Configures whether or not to clear MCPWMO_EVT_CAP1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_CAP2_ST_CLR Configures whether or not to clear MCPWMO_EVT_CAP2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.21. SOC_ETM_EVT_ST2_CLR_REG (0x01BC)

Continued from the previous page...

SOC_ETM_MCPWMO_EVT_OPO_TEE1_ST_CLR Configures whether or not to clear MCPWMO_EVT_OPO_TEE1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OP1_TEE1_ST_CLR Configures whether or not to clear MCPWMO_EVT_OP1_TEE1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OP2_TEE1_ST_CLR Configures whether or not to clear MCPWMO_EVT_OP2_TEE1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OPO_TEE2_ST_CLR Configures whether or not to clear MCPWMO_EVT_OPO_TEE2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OP1_TEE2_ST_CLR Configures whether or not to clear MCPWMO_EVT_OP1_TEE2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_EVT_OP2_TEE2_ST_CLR Configures whether or not to clear MCPWMO_EVT_OP2_TEE2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_CONV_CMPLTO_ST_CLR Configures whether or not to clear ADC_EVT_CONV_CMPLTO trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.21. SOC_ETM_EVT_ST2_CLR_REG (0x01BC)

Continued from the previous page...

SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH0_ST_CLR Configures whether or not to clear ADC_EVT_EQ_ABOVE_THRESH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_EQ_ABOVE_THRESH1_ST_CLR Configures whether or not to clear ADC_EVT_EQ_ABOVE_THRESH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_EQ_BELOW_THRESH0_ST_CLR Configures whether or not to clear ADC_EVT_EQ_BELOW_THRESH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_EQ_BELOW_THRESH1_ST_CLR Configures whether or not to clear ADC_EVT_EQ_BELOW_THRESH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_RESULT_DONE0_ST_CLR Configures whether or not to clear ADC_EVT_RESULT_DONE0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_EVT_STOPPED0_ST_CLR Configures whether or not to clear ADC_EVT_STOPPED0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

[illegible]

0: Invalid. No effect
1: Clear
(WT)

0: Invalid. No effect
1: Clear
(WT)

0: Invalid. No effect
1: Clear
(WT)

0: Invalid. No effect
1: Clear
(WT)

PRELIMINARY

Register 8.22. SOC_ETM_EVT_ST3_CLR_REG (0x01C4)

Continued from the previous page...

SOC_ETM_I2SO_EVT_X_WORDS_RECEIVED_ST_CLR Configures whether or not to clear I2SO_EVT_X_WORDS_RECEIVED trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_I2SO_EVT_X_WORDS_SENT_ST_CLR Configures whether or not to clear I2SO_EVT_X_WORDS_SENT trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ULP_EVT_ERR_INTR_ST_CLR Configures whether or not to clear ULP_EVT_ERR_INTR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ULP_EVT_HALT_ST_CLR Configures whether or not to clear ULP_EVT_HALT trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ULP_EVT_START_INTR_ST_CLR Configures whether or not to clear ULP_EVT_START_INTR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_RTC_EVT_TICK_ST_CLR Configures whether or not to clear RTC_EVT_TICK trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_RTC_EVT_OVF_ST_CLR Configures whether or not to clear RTC_EVT_OVF trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.22. SOC_ETM_EVT_ST3_CLR_REG (0x01C4)

Continued from the previous page...

SOC_ETM_RTC_EVT_CMP_ST_CLR Configures whether or not to clear RTC_EVT_CMP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_DONE_CHO_ST_CLR Configures whether or not to clear GDMA_EVT_IN_DONE_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_DONE_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_IN_DONE_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_DONE_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_IN_DONE_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CHO_ST_CLR Configures whether or not to clear GDMA_EVT_IN_SUC_EOF_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_IN_SUC_EOF_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_SUC_EOF_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_IN_SUC_EOF_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.22. SOC_ETM_EVT_ST3_CLR_REG (0x01C4)

Continued from the previous page...

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CHO_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_EMPTY_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_EMPTY_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_FIFO_EMPTY_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_EMPTY_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CHO_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_FULL_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_FULL_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_IN_FIFO_FULL_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_IN_FIFO_FULL_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.23. SOC_ETM_EVT_ST4_CLR_REG (0x01CC)

(reserved)																SOC_ETM_PMU_EVT_SLEEP_WEEKUP_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH2_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH1_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH2_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH1_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH2_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH1_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_EOF_CH2_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_DONE_CH2_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_DONE_CH1_ST_CLR																		
																SOC_ETM_GDMA_EVT_OUT_DONE_CH0_ST_CLR																		
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset								

SOC_ETM_GDMA_EVT_OUT_DONE_CH0_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_DONE_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_DONE_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_DONE_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_DONE_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_DONE_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_EOF_CH0_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_EOF_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_EOF_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_EOF_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.23. SOC_ETM_EVT_ST4_CLR_REG (0x01CC)

Continued from the previous page...

SOC_ETM_GDMA_EVT_OUT_EOF_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_EOF_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH0_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_TOTAL_EOF_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_TOTAL_EOF_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_TOTAL_EOF_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_TOTAL_EOF_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH0_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_EMPTY_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_EMPTY_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_FIFO_EMPTY_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_EMPTY_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.23. SOC_ETM_EVT_ST4_CLR_REG (0x01CC)

Continued from the previous page...

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CHO_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_FULL_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH1_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_FULL_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_EVT_OUT_FIFO_FULL_CH2_ST_CLR Configures whether or not to clear GDMA_EVT_OUT_FIFO_FULL_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_PMU_EVT_SLEEP_WEEKUP_ST_CLR Configures whether or not to clear PMU_EVT_SLEEP_WEEKUP trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.24. SOC_ETM_TASK_STO_CLR_REG (0x01D4)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH3_ST_CLR																															SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH2_ST_CLR																															SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH1_ST_CLR																															SOC_ETM_LEDC_TASK_TIMER3_RES_UPDATE_CH0_ST_CLR																															SOC_ETM_LEDC_TASK_TIMER2_RES_UPDATE_ST_CLR																															SOC_ETM_LEDC_TASK_TIMER1_RES_UPDATE_ST_CLR																															SOC_ETM_GPIO_TASK_CH7_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH6_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH5_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH4_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH3_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH2_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH1_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH0_TOGGLE_ST_CLR																															SOC_ETM_GPIO_TASK_CH7_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH6_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH5_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH4_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH3_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH2_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH1_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH0_CLEAR_ST_CLR																															SOC_ETM_GPIO_TASK_CH7_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH6_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH5_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH4_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH3_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH2_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH1_SET_ST_CLR																															SOC_ETM_GPIO_TASK_CH0_SET_ST_CLR																																																																																																																								
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SOC_ETM_GPIO_TASK_CH0_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH0_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH1_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH1_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH2_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH2_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH3_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH3_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH4_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH4_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.24. SOC_ETM_TASK_STO_CLR_REG (0x01D4)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH5_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH5_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH6_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH6_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH7_SET_ST_CLR Configures whether or not to clear GPIO_TASK_CH7_SET trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH0_CLEAR_ST_CLR Configures whether or not to clear GPIO_TASK_CH0_CLEAR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH1_CLEAR_ST_CLR Configures whether or not to clear GPIO_TASK_CH1_CLEAR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH2_CLEAR_ST_CLR Configures whether or not to clear GPIO_TASK_CH2_CLEAR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH3_CLEAR_ST_CLR Configures whether or not to clear GPIO_TASK_CH3_CLEAR trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.24. SOC_ETM_TASK_STO_CLR_REG (0x01D4)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH4_CLEAR_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH4_CLEAR trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CH5_CLEAR_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH5_CLEAR trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CH6_CLEAR_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH6_CLEAR trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CH7_CLEAR_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH7_CLEAR trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CHO_TOGGLE_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CHO_TOGGLE trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CH1_TOGGLE_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH1_TOGGLE trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						
SOC_ETM_GPIO_TASK_CH2_TOGGLE_ST_CLR	Configures	whether	or	not	to	clear
GPIO_TASK_CH2_TOGGLE trigger status.						
0: Invalid. No effect						
1: Clear						
(WT)						

Continued on the next page...

Register 8.24. SOC_ETM_TASK_STO_CLR_REG (0x01D4)

Continued from the previous page...

SOC_ETM_GPIO_TASK_CH3_TOGGLE_ST_CLR Configures whether or not to clear GPIO_TASK_CH3_TOGGLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH4_TOGGLE_ST_CLR Configures whether or not to clear GPIO_TASK_CH4_TOGGLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH5_TOGGLE_ST_CLR Configures whether or not to clear GPIO_TASK_CH5_TOGGLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH6_TOGGLE_ST_CLR Configures whether or not to clear GPIO_TASK_CH6_TOGGLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GPIO_TASK_CH7_TOGGLE_ST_CLR Configures whether or not to clear GPIO_TASK_CH7_TOGGLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER0_RES_UPDATE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER0_RES_UPDATE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER1_RES_UPDATE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER1_RES_UPDATE trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.24. SOC_ETM_TASK_STO_CLR_REG (0x01D4)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER2_RES_UPDATE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER2_RES_UPDATE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER3_RES_UPDATE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER3_RES_UPDATE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CHO_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.25. SOC_ETM_TASK_ST1_CLR_REG (0x01DC)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH1_ST_CLR	31	0
SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH0_ST_CLR	30	0
SOC_ETM_LEDC_TASK_TIMER3_PAUSE_ST_CLR	29	0
SOC_ETM_LEDC_TASK_TIMER2_PAUSE_ST_CLR	28	0
SOC_ETM_LEDC_TASK_TIMER1_PAUSE_ST_CLR	27	0
SOC_ETM_LEDC_TASK_TIMER0_PAUSE_ST_CLR	26	0
SOC_ETM_LEDC_TASK_TIMER3_RESUME_ST_CLR	25	0
SOC_ETM_LEDC_TASK_TIMER2_RESUME_ST_CLR	24	0
SOC_ETM_LEDC_TASK_TIMER1_RESUME_ST_CLR	23	0
SOC_ETM_LEDC_TASK_TIMER0_RESUME_ST_CLR	22	0
SOC_ETM_LEDC_TASK_TIMER3_RST_ST_CLR	21	0
SOC_ETM_LEDC_TASK_TIMER2_RST_ST_CLR	20	0
SOC_ETM_LEDC_TASK_TIMER1_RST_ST_CLR	19	0
SOC_ETM_LEDC_TASK_TIMER0_RST_ST_CLR	18	0
SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH5_ST_CLR	17	0
SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH4_ST_CLR	16	0
SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH3_ST_CLR	15	0
SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH2_ST_CLR	14	0
SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH1_ST_CLR	13	0
SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH5_ST_CLR	12	0
SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH4_ST_CLR	11	0
SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH3_ST_CLR	10	0
SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH2_ST_CLR	9	0
SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH1_ST_CLR	8	0
SOC_ETM_LEDC_TASK_TIMER3_CAP_ST_CLR	7	0
SOC_ETM_LEDC_TASK_TIMER2_CAP_ST_CLR	6	0
SOC_ETM_LEDC_TASK_TIMER1_CAP_ST_CLR	5	0
SOC_ETM_LEDC_TASK_TIMER0_CAP_ST_CLR	4	0
SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH5_ST_CLR	3	0
SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH4_ST_CLR	2	0
	1	0
	0	0

Reset

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_DUTY_SCALE_UPDATE_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_DUTY_SCALE_UPDATE_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER0_CAP_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER0_CAP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER1_CAP_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER1_CAP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER2_CAP_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER2_CAP trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.25. SOC_ETM_TASK_ST1_CLR_REG (0x01DC)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER3_CAP_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER3_CAP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CHO_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_SIG_OUT_DIS_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_SIG_OUT_DIS_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.25. SOC_ETM_TASK_ST1_CLR_REG (0x01DC)

Continued from the previous page...

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH0_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_OVF_CNT_RST_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_OVF_CNT_RST_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER0_RST_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER0_RST trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.25. SOC_ETM_TASK_ST1_CLR_REG (0x01DC)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER1_RST_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER1_RST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER2_RST_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER2_RST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER3_RST_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER3_RST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER0_RESUME_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER0_RESUME trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER1_RESUME_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER1_RESUME trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER2_RESUME_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER2_RESUME trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER3_RESUME_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER3_RESUME trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.25. SOC_ETM_TASK_ST1_CLR_REG (0x01DC)

Continued from the previous page...

SOC_ETM_LEDC_TASK_TIMER0_PAUSE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER0_PAUSE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER1_PAUSE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER1_PAUSE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER2_PAUSE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER2_PAUSE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_TIMER3_PAUSE_ST_CLR Configures whether or not to clear LEDC_TASK_TIMER3_PAUSE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CHO_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.26. SOC_ETM_TASK_ST2_CLR_REG (0x01E4)

Soc_ETM_TG1_TASK_CNT_START_TIMER1_ST_CLR																															
Soc_ETM_TG1_TASK_CNT_CAP_TIMER0_ST_CLR																															
Soc_ETM_TG1_TASK_CNT_RELOAD_TIMER0_ST_CLR																															
Soc_ETM_TG1_TASK_CNT_STOP_TIMER0_ST_CLR																															
Soc_ETM_TG1_TASK_ALARM_START_TIMER0_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_START_TIMER0_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_CAP_TIMER0_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_RELOAD_TIMER0_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_STOP_TIMER0_ST_CLR																															
Soc_ETM_TG0_TASK_ALARM_START_TIMER1_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_START_TIMER1_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_CAP_TIMER1_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_RELOAD_TIMER1_ST_CLR																															
Soc_ETM_TG0_TASK_CNT_STOP_TIMER1_ST_CLR																															
Soc_ETM_LEDC_TASK_ALARM_START_TIMER0_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESUME_CH5_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESUME_CH4_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESUME_CH3_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESUME_CH2_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESUME_CH1_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH0_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH5_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH4_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH3_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH2_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_PAUSE_CH1_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESTART_CH0_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESTART_CH5_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESTART_CH4_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESTART_CH3_ST_CLR																															
Soc_ETM_LEDC_TASK_GAMMA_RESTART_CH2_ST_CLR																															
Reset																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESTART_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESTART_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CHO_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.26. SOC_ETM_TASK_ST2_CLR_REG (0x01E4)

Continued from the previous page...

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_PAUSE_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_PAUSE_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH0_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH1_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.26. SOC_ETM_TASK_ST2_CLR_REG (0x01E4)

Continued from the previous page...

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH2_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH3_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH3 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH4_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH4 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_LEDC_TASK_GAMMA_RESUME_CH5_ST_CLR Configures whether or not to clear LEDC_TASK_GAMMA_RESUME_CH5 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_START_TIMER0_ST_CLR Configures whether or not to clear TGO_TASK_CNT_START_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_ALARM_START_TIMER0_ST_CLR Configures whether or not to clear TGO_TASK_ALARM_START_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_STOP_TIMER0_ST_CLR Configures whether or not to clear TGO_TASK_CNT_STOP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.26. SOC_ETM_TASK_ST2_CLR_REG (0x01E4)

Continued from the previous page...

SOC_ETM_TGO_TASK_CNT_RELOAD_TIMER0_ST_CLR Configures whether or not to clear TGO_TASK_CNT_RELOAD_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_CAP_TIMER0_ST_CLR Configures whether or not to clear TGO_TASK_CNT_CAP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_START_TIMER1_ST_CLR Configures whether or not to clear TGO_TASK_CNT_START_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_ALARM_START_TIMER1_ST_CLR Configures whether or not to clear TGO_TASK_ALARM_START_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_STOP_TIMER1_ST_CLR Configures whether or not to clear TGO_TASK_CNT_STOP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_RELOAD_TIMER1_ST_CLR Configures whether or not to clear TGO_TASK_CNT_RELOAD_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TGO_TASK_CNT_CAP_TIMER1_ST_CLR Configures whether or not to clear TGO_TASK_CNT_CAP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.26. SOC_ETM_TASK_ST2_CLR_REG (0x01E4)

Continued from the previous page...

SOC_ETM_TG1_TASK_CNT_START_TIMER0_ST_CLR Configures whether or not to clear TG1_TASK_CNT_START_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_ALARM_START_TIMER0_ST_CLR Configures whether or not to clear TG1_TASK_ALARM_START_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_STOP_TIMER0_ST_CLR Configures whether or not to clear TG1_TASK_CNT_STOP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER0_ST_CLR Configures whether or not to clear TG1_TASK_CNT_RELOAD_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_CAP_TIMER0_ST_CLR Configures whether or not to clear TG1_TASK_CNT_CAP_TIMER0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_START_TIMER1_ST_CLR Configures whether or not to clear TG1_TASK_CNT_START_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.27. SOC_ETM_TASK_ST3_CLR_REG (0x01EC)

(reserved)	SOC_ETM_ADC_Task_STOPO_ST_CLR	SOC_ETM_ADC_Task_STARTO_ST_CLR	(reserved)	SOC_ETM_ADC_Task_SAMPLEO_ST_CLR	SOC_ETM_MCPWMO_Task_CAP2_ST_CLR	SOC_ETM_MCPWMO_Task_CAP1_ST_CLR	SOC_ETM_MCPWMO_Task_CAP0_ST_CLR	SOC_ETM_MCPWMO_Task_CLR2_ST_CLR	SOC_ETM_MCPWMO_Task_CLR1_ST_CLR	SOC_ETM_MCPWMO_Task_CLR0_ST_CLR	SOC_ETM_MCPWMO_Task_TZ2_ST_CLR	SOC_ETM_MCPWMO_Task_TZ1_ST_CLR	SOC_ETM_MCPWMO_Task_TZ0_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER2_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER1_PERIOD_UP_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER0_PERIOD_UP_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER2_SYN_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER1_SYN_ST_CLR	SOC_ETM_MCPWMO_Task_TIMER0_SYN_ST_CLR	SOC_ETM_MCPWMO_Task_GEN_STOP_ST_CLR	SOC_ETM_MCPWMO_Task_CMPR2_B_UP_ST_CLR	SOC_ETM_MCPWMO_Task_CMPR1_B_UP_ST_CLR	SOC_ETM_MCPWMO_Task_CMPRO_B_UP_ST_CLR	SOC_ETM_MCPWMO_Task_CMPR2_A_UP_ST_CLR	SOC_ETM_TG1_Task_CNT_CMPR1_A_UP_ST_CLR	SOC_ETM_TG1_Task_CNT_CMPRO_A_UP_ST_CLR	SOC_ETM_TG1_Task_CNT_RELOAD_TIMER1_ST_CLR	SOC_ETM_TG1_Task_ALARM_START_TIMER1_ST_CLR			
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SOC_ETM_TG1_TASK_ALARM_START_TIMER1_ST_CLR Configures whether or not to clear TG1_TASK_ALARM_START_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_STOP_TIMER1_ST_CLR Configures whether or not to clear TG1_TASK_CNT_STOP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_RELOAD_TIMER1_ST_CLR Configures whether or not to clear TG1_TASK_CNT_RELOAD_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TG1_TASK_CNT_CAP_TIMER1_ST_CLR Configures whether or not to clear TG1_TASK_CNT_CAP_TIMER1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CMPRO_A_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPRO_A_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.27. SOC_ETM_TASK_ST3_CLR_REG (0x01EC)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_CMPR1_A_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPR1_A_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CMPR2_A_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPR2_A_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CMPRO_B_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPRO_B_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CMPR1_B_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPR1_B_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CMPR2_B_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_CMPR2_B_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_GEN_STOP_ST_CLR Configures whether or not to clear MCPWMO_TASK_GEN_STOP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TIMER0_SYN_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER0_SYN trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.27. SOC_ETM_TASK_ST3_CLR_REG (0x01EC)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_TIMER1_SYN_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER1_SYN trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TIMER2_SYN_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER2_SYN trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TIMER0_PERIOD_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER0_PERIOD_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TIMER1_PERIOD_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER1_PERIOD_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TIMER2_PERIOD_UP_ST_CLR Configures whether or not to clear MCPWMO_TASK_TIMER2_PERIOD_UP trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TZO_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_TZO_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_TZ1_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_TZ1_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.27. SOC_ETM_TASK_ST3_CLR_REG (0x01EC)

Continued from the previous page...

SOC_ETM_MCPWMO_TASK_TZ2_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_TZ2_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CLR0_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_CLR0_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CLR1_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_CLR1_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CLR2_OST_ST_CLR Configures whether or not to clear MCPWMO_TASK_CLR2_OST trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CAPO_ST_CLR Configures whether or not to clear MCPWMO_TASK_CAPO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CAP1_ST_CLR Configures whether or not to clear MCPWMO_TASK_CAP1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_MCPWMO_TASK_CAP2_ST_CLR Configures whether or not to clear MCPWMO_TASK_CAP2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_TASK_SAMPLE0_ST_CLR Configures whether or not to clear ADC_TASK_SAMPLE0 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.27. SOC_ETM_TASK_ST3_CLR_REG (0x01EC)

Continued from the previous page...

SOC_ETM_ADC_TASK_STARTO_ST_CLR Configures whether or not to clear ADC_TASK_STARTO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ADC_TASK_STOPO_ST_CLR Configures whether or not to clear ADC_TASK_STOPO trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.28. SOC_ETM_TASK_ST4_CLR_REG (0x01F4)

(reserved)																SOC_ETM_PMU_TASK_SLEEP_REQ_ST_CLR SOC_ETM_GDMA_TASK_OUT_OUT_START_CH2_ST_CLR SOC_ETM_GDMA_TASK_OUT_OUT_START_CH1_ST_CLR SOC_ETM_GDMA_TASK_IN_START_CH2_ST_CLR SOC_ETM_GDMA_TASK_IN_START_CH1_ST_CLR SOC_ETM_RTC_TASK_TRIGGERFLW_ST_CLR (reserved) SOC_ETM_ULP_TASK_WAKEUP_CPU_ST_CLR SOC_ETM_I2SO_TASK_STOP_TX_ST_CLR SOC_ETM_I2SO_TASK_STOP_RX_ST_CLR SOC_ETM_I2SO_TASK_START_TX_ST_CLR SOC_ETM_TMPSNSR_TASK_STOP_SAMPLE_ST_CLR (reserved)															
31								21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset		

SOC_ETM_TMPSNSR_TASK_START_SAMPLE_ST_CLR Configures whether or not to clear TMP-SNSR_TASK_START_SAMPLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_TMPSNSR_TASK_STOP_SAMPLE_ST_CLR Configures whether or not to clear TMP-SNSR_TASK_STOP_SAMPLE trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_I2SO_TASK_START_RX_ST_CLR Configures whether or not to clear I2SO_TASK_START_RX trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.28. SOC_ETM_TASK_ST4_CLR_REG (0x01F4)

Continued from the previous page...

SOC_ETM_I2SO_TASK_START_TX_ST_CLR Configures whether or not to clear I2SO_TASK_START_TX trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_I2SO_TASK_STOP_RX_ST_CLR Configures whether or not to clear I2SO_TASK_STOP_RX trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_I2SO_TASK_STOP_TX_ST_CLR Configures whether or not to clear I2SO_TASK_STOP_TX trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_ULP_TASK_WAKEUP_CPU_ST_CLR Configures whether or not to clear ULP_TASK_WAKEUP_CPU trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_RTC_TASK_START_ST_CLR Configures whether or not to clear RTC_TASK_START trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_RTC_TASK_STOP_ST_CLR Configures whether or not to clear RTC_TASK_STOP trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.28. SOC_ETM_TASK_ST4_CLR_REG (0x01F4)

Continued from the previous page...

SOC_ETM_RTC_TASK_CLR_ST_CLR Configures whether or not to clear RTC_TASK_CLR trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_RTC_TASK_TRIGGERFLW_ST_CLR Configures whether or not to clear RTC_TASK_TRIGGERFLW trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_TASK_IN_START_CHO_ST_CLR Configures whether or not to clear GDMA_TASK_IN_START_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_TASK_IN_START_CH1_ST_CLR Configures whether or not to clear GDMA_TASK_IN_START_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_TASK_IN_START_CH2_ST_CLR Configures whether or not to clear GDMA_TASK_IN_START_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_TASK_OUT_START_CHO_ST_CLR Configures whether or not to clear GDMA_TASK_OUT_START_CHO trigger status.

0: Invalid. No effect

1: Clear

(WT)

SOC_ETM_GDMA_TASK_OUT_START_CH1_ST_CLR Configures whether or not to clear GDMA_TASK_OUT_START_CH1 trigger status.

0: Invalid. No effect

1: Clear

(WT)

Continued on the next page...

Register 8.28. SOC_ETM_TASK_ST4_CLR_REG (0x01F4)

Continued from the previous page...

SOC_ETM_GDMA_TASK_OUT_START_CH2_ST_CLR Configures whether or not to clear GDMA_TASK_OUT_START_CH2 trigger status.

0: Invalid. No effect

1: Clear

(WT)

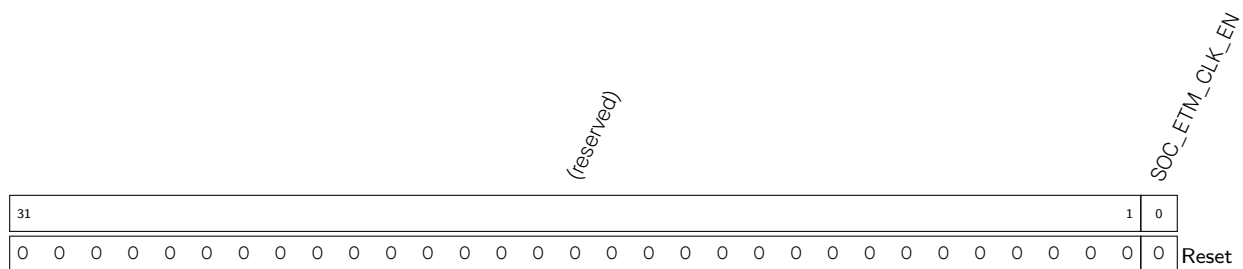
SOC_ETM_PMU_TASK_SLEEP_REQ_ST_CLR Configures whether or not to clear PMU_TASK_SLEEP_REQ trigger status.

0: Invalid. No effect

1: Clear

(WT)

Register 8.29. SOC_ETM_CLK_EN_REG (0x01F8)



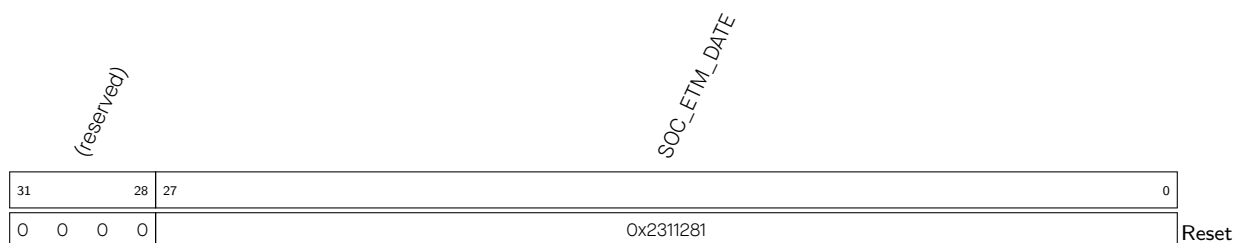
SOC_ETM_CLK_EN Configures whether or not to open register clock gate.

0: Open the clock gate only when application writes registers

1: Force open the clock gate for register

(R/W)

Register 8.30. SOC_ETM_DATE_REG (0x01FC)



SOC_ETM_DATE Version control register. (R/W)

Chapter 9

System Timer

9.1 Overview

ESP32-C5 provides a 52-bit system timer, which can be used to generate tick interrupts for the operating system, or be used as a general timer to generate periodic interrupts or one-time interrupts.

9.2 Features

The system timer has the following features:

- Two 52-bit counters and three 52-bit comparators
- Software accessing registers clocked by APB_CLK
- CNT_CLK used for counting, with an average frequency of 16 MHz in two counting cycles
- 40 MHz/48 MHz XTAL_CLK as the clock source of CNT_CLK
- 52-bit alarm values (t) and 26-bit alarm periods (δt)
- Two modes to generate alarms:
 - Target mode: only a one-time alarm is generated based on the alarm value (t)
 - Period mode: periodic alarms are generated based on the alarm period (δt)
- Three comparators generating three independent interrupts based on configured alarm value (t) or alarm period (δt)
- Software configuring the reference count value. For example, the system timer is able to load back the sleep time recorded by RTC timer via software after Light-sleep
- Able to stall or continue running when CPU stalls or enters the on-chip-debugging mode
- Alarm for Event Task Matrix (ETM) event

9.3 System Timer Structure

The timer consists of two counters: UNIT0 and UNIT1. The count values can be monitored by three comparators, COMP0, COMP1, and COMP2. See the timer block diagram in Figure 9.3-1.

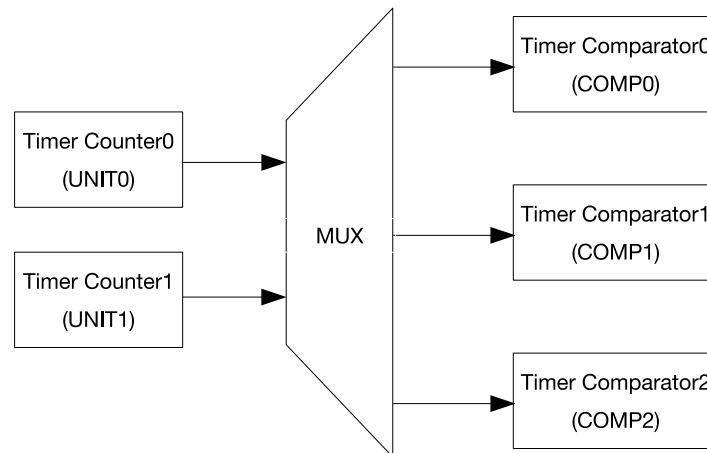


Figure 9.3-1. System Timer Structure

9.4 Clock Source Selection

The counters and comparators use XTAL_CLK or RC_FAST_CLK as the clock sources. The clock source can be selected by configuring field [PCR_SYSTIMER_FUNC_CLK_SEL](#) in register [PCR_SYSTIMER_FUNC_CLK_CONF_REG](#). After XTAL_CLK is scaled, a counter clock signal CNT_CLK clock is generated. The average clock frequency of CNT_CLK is 16 MHz, as shown in Figure 9.5-1. The timer counter is incremented by $1/16 \mu s$ on each CNT_CLK cycle.

Software operation such as configuring registers is clocked by APB_CLK. For more information about APB_CLK, see Chapter 6 [Reset and Clock](#).

The following two bits of system registers are also used to control the system timer:

- Set [PCR_SYSTIMER_CLK_EN](#) in register [PCR_SYSTIMER_CONF_REG](#) to enable APB_CLK signal to the system timer.
- Set [PCR_SYSTIMER_RST_EN](#) in register [PCR_SYSTIMER_CONF_REG](#) to reset the system timer.

Note that if the timer is reset, its registers will be restored to their default values. For more information, please refer to Chapter 6 [Reset and Clock](#).

9.5 Functional Description

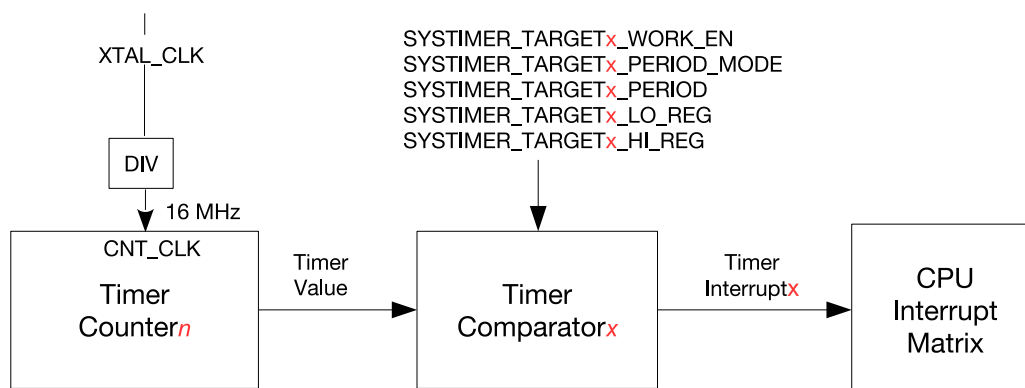


Figure 9.5-1. System Timer Alarm Generation

Figure 9.5-1 shows the procedure to generate alarm in system timer. In this process, one timer counter and one timer comparator are used. An alarm interrupt will be generated accordingly based on the comparison result in the comparator.

9.5.1 Counter

The system timer has two 52-bit timer counters, shown as UNIT n ($n = 0$ or 1). Their counting clock source is a 16 MHz clock, i.e. CNT_CLK. Whether UNIT n works or not is controlled by three bits in register SYSTIMER_CONF_REG:

- SYSTIMER_TIMER_UNIT n _WORK_EN: set this bit to enable the counter UNIT n in the system timer.
- SYSTIMER_TIMER_UNIT n _CORE0_STALL_EN: if this bit is set, the counter UNIT n stops when CPU0 is stalled. The counter continues its counting after the CPU resumes.
- SYSTIMER_TIMER_UNIT n _CORE1_STALL_EN: if this bit is set, the counter UNIT n stops when CPU1 is stalled. The counter continues its counting after the CPU resumes.

The configuration of the bits to control the counter UNIT n is shown below, assuming that CPU is stalled.

Table 9.5-1. UNIT n Configuration Bits

SYSTIMER_TIMER_UNIT n _WORK_EN	SYSTIMER_TIMER_UNIT n _CORE0_STALL_EN	SYSTIMER_TIMER_UNIT n _CORE1_STALL_EN	Counter UNIT n
0	x [*]	x [*]	Not at work
1	x	1	Stop counting, but will continue its counting after the CPU1 resumes
1	1	x	Stop counting, but will continue its counting after the CPU0 resumes
1	0	0	Keep counting

^{*} x: Don't-care.

When the counter UNIT n is at work, the count value is incremented on each counting cycle. When the counter UNIT n is stopped, the count value stops increasing and keeps unchanged.

The lower 32 and higher 20 bits of the initial count value are loaded from the registers SYSTIMER_TIMER_UNIT n _LOAD_LO and SYSTIMER_TIMER_UNIT n _LOAD_HI. Writing 1 to the bit SYSTIMER_TIMER_UNIT n _LOAD will trigger a reload event, and the current count value will be changed immediately. If UNIT n is at work, the counter will continue to count up from the new reloaded value.

Writing 1 to SYSTIMER_TIMER_UNIT n _UPDATE will trigger an update event. The lower 32 and higher 20 bits of the current count value will be locked into the registers SYSTIMER_TIMER_UNIT n _VALUE_LO and SYSTIMER_TIMER_UNIT n _VALUE_HI, and then SYSTIMER_TIMER_UNIT n _VALUE_VALID is asserted. Before the next update event, the values of SYSTIMER_TIMER_UNIT n _VALUE_LO and SYSTIMER_TIMER_UNIT n _VALUE_HI remain unchanged.

9.5.2 Comparator and Alarm

The system timer has three 52-bit comparators, shown as COMP x ($x = 0, 1, \text{ or } 2$). The comparators can generate independent interrupts based on different alarm values (t) or alarm periods (δt).

Configure SYSTIMER_TARGET x _PERIOD_MODE to choose from the two alarm modes for each COMP x :

- 1: period mode
- 0: target mode

In period mode, the alarm period (δt) is provided by the register SYSTIMER_TARGET x _PERIOD. Assuming that current count value is t_1 , when it reaches $(t_1 + \delta t)$, an alarm interrupt will be generated. When the count value reaches $(t_1 + 2 \times \delta t)$, another alarm interrupt also will be generated. By such way, periodic alarms are generated.

In target mode, the lower 32 bits and higher 20 bits of the alarm value (t) are provided by SYSTIMER_TIMER_TARGET x _LO and SYSTIMER_TIMER_TARGET x _HI. Assuming that current count value is t_2 ($t_2 \leq t$), an alarm interrupt will be generated when the count value reaches the alarm value (t). Unlike in period mode, only one alarm interrupt is generated in target mode.

SYSTIMER_TARGET x _TIMER_UNIT_SEL is used to choose the count value from which timer counter to be compared to generate alarms:

- 1: Use the count value from UNIT1
- 0: Use the count value from UNIT0

Finally, set SYSTIMER_TARGET x _WORK_EN and COMP x starts to compare the count value:

- In target mode, COMP x compares it with the alarm value (t).
- In period mode, COMP x compares it with the alarm period ($t_1 + n \times \delta t$).

An alarm is generated when the count value equals to the alarm value (t) in target mode or to the start value + $n \times \delta t$ ($n = 1, 2, 3 \dots$) in period mode. But if the alarm value (t) set in registers is less than the current count value, i.e., the target has already passed, when the current count value is larger than the alarm value (t) within a range ($0 \sim 2^{51} - 1$), an alarm interrupt will also be generated immediately. No matter in target mode or period mode, the low 32 bits and high 20 bits of the real alarm value can always be read from SYSTIMER_TARGET x _LO_RO and SYSTIMER_TARGET x _HI_RO. The alarm trigger point and the relationship between current count value t_c and the alarm value t_t are shown below.

Table 9.5-2. Trigger Point

Relationship Between t_c and t_t	Trigger Point
$t_c - t_t \leq 0$	$t_c = t_t$, an alarm is triggered.
$0 \leq t_c - t_t < 2^{51} - 1$ ($t_c < 2^{51}$ and $t_t < 2^{51}$, or $t_c \geq 2^{51}$ and $t_t \geq 2^{51}$)	An alarm is triggered immediately.
$t_c - t_t \geq 2^{51} - 1$	t_c overflows after counting to its maximum value 52'hffffffffffff, and then starts counting up from 0. When its value reaches t_t , an alarm is triggered.

9.5.3 Event Task Matrix

The system timer on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows the system timer's ETM tasks to be triggered by any peripherals' ETM events, or system timer's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to the system timer. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

The system timer can generate the following ETM event:

- SYSTIMER_EVT_CNT_CMP x : Indicates the alarm events generated by COMP x .

When [SYSTIMER_ETM_EN](#) is set to 1, the alarm events can trigger the ETM event.

9.5.4 Synchronization Operation

The frequency of the clock used for software configuration is different from that of the clock driving the timer counters and comparators. Therefore, synchronization is required for configuring some registers. The steps are as follows:

1. Software writes specific values to configuration fields. See the first column in Table 9.5-3.
2. Software writes 1 to corresponding bits to start synchronization. See the second column in Table 9.5-3.

Table 9.5-3. Synchronization Operation for Configuration Registers

Configuration Fields	Synchronization Enable Bit
SYSTIMER_TIMER_UNIT n _LOAD_LO SYSTIMER_TIMER_UNIT n _LOAD_HI	SYSTIMER_TIMER_UNIT n _LOAD
SYSTIMER_TARGET x _PERIOD SYSTIMER_TIMER_TARGET x _HI SYSTIMER_TIMER_TARGET x _LO	SYSTIMER_TIMER_COMP x _LOAD

The frequency of the clock used for software reading is different from that of the clock driving the timer counters and comparators. Therefore, synchronization is also needed for reading some registers. The steps are as follows:

1. Software writes specific values to the updating register [SYSTIMER_TIMER_UNIT \$n\$ _UPDATE](#).

2. Software reads the corresponding bit `SYSTIMER_TIMER_UNIT $n$ _VALUE_VALID` to be valid to confirm synchronization is done.
3. Software reads the corresponding status registers `SYSTIMER_TIMER_UNIT $n$ _VALUE_HI` and `SYSTIMER_TIMER_UNIT $n$ _VALUE_LO`.

9.6 Interrupts

ESP32-C5's system timer can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- SYSTIMER_TARGET0_INT
- SYSTIMER_TARGET1_INT
- SYSTIMER_TARGET2_INT

There are several internal interrupt sources from the system timer that can generate the above interrupt signals. The interrupt sources from the system timer are listed with their trigger conditions and the resulted interrupt signals in Table 9.6-1.

Table 9.6-1. System Timer's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
SYSTIMER_INT0	When COMPO alarms	SYSTIMER_TARGET0_INT
SYSTIMER_INT1	When COMP1 alarms	SYSTIMER_TARGET1_INT
SYSTIMER_INT2	When COMP2 alarms	SYSTIMER_TARGET2_INT

The above interrupts are level-type alarm interrupts. The interrupt signal is asserted high when the comparator starts to alarm. Until the software clears the interrupt, it remains high. To enable interrupts, set the bit `SYSTIMER_TARGET $x$ _INT_ENA`.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section 9.8 [Register Summary](#).

9.7 Programming Procedure

When configuring COMP x and UNIT n , please ensure the corresponding COMP and UNIT are at work.

9.7.1 Read Current Count Value

1. Set `SYSTIMER_TIMER_UNIT $n$ _UPDATE` to fill the current count value of COMP x into `SYSTIMER_TIMER_UNIT $n$ _VALUE_HI` and `SYSTIMER_TIMER_UNIT $n$ _VALUE_LO`.

2. Poll the reading of `SYSTIMER_TIMER_UNIT $n$ _VALUE_VALID` till it is 1. Then, user can read the count value from `SYSTIMER_TIMER_UNIT $n$ _VALUE_HI` and `SYSTIMER_TIMER_UNIT $n$ _VALUE_LO`.
3. Read the lower 32 bits and higher 20 bits from `SYSTIMER_TIMER_UNIT $n$ _VALUE_LO` and `SYSTIMER_TIMER_UNIT $n$ _VALUE_HI` respectively.

9.7.2 Configure a One-Time Alarm in Target Mode

1. Set `SYSTIMER_TARGET $x$ _TIMER_UNIT_SEL` to select the counter (UNIT 0 or UNIT 1) used for comparison with COMP x .
2. Read the current count value with reference to Section 9.7.1. This value will be used to calculate the alarm value (t) in Step 4.
3. Clear `SYSTIMER_TARGET $x$ _PERIOD_MODE` to enable target mode.
4. Set an alarm value (t), and fill its lower 32 bits into `SYSTIMER_TIMER_TARGET $x$ _LO`, and the higher 20 bits into `SYSTIMER_TIMER_TARGET $x$ _HI`.
5. Set `SYSTIMER_TIMER_COMP $x$ _LOAD` to synchronize the alarm value (t) to COMP x , i.e., load the alarm value (t) to the COMP x .
6. Set `SYSTIMER_TARGET $x$ _WORK_EN` to enable the selected COMP x . COMP x starts comparing the count value with the alarm value (t).
7. Set `SYSTIMER_TARGET $x$ _INT_ENA` to enable the timer interrupt. When UNIT n reaches the alarm value (t), a `SYSTIMER_TARGET $x$ _INT` interrupt is triggered.

9.7.3 Configure Periodic Alarms in Period Mode

1. Set `SYSTIMER_TARGET $x$ _TIMER_UNIT_SEL` to select the counter (UNIT 0 or UNIT 1) used for comparison with COMP x .
2. Set an alarm period (δt), and fill it into `SYSTIMER_TARGET $x$ _PERIOD`.
3. Set `SYSTIMER_TIMER_COMP $x$ _LOAD` to synchronize the alarm period (δt) to COMP x , i.e., load the alarm period (δt) to COMP x .
4. Clear and then set `SYSTIMER_TARGET $x$ _PERIOD_MODE` to configure COMP x into period mode.
5. Set `SYSTIMER_TARGET $x$ _WORK_EN` to enable the selected COMP x . COMP x starts comparing the count value with the sum of (start value + $n \times \delta t$) ($n = 1, 2, 3, \dots$).
6. Set `SYSTIMER_TARGET $x$ _INT_ENA` to enable the timer interrupt. A `SYSTIMER_TARGET $x$ _INT` interrupt is triggered when UNIT n reaches start value + $n \times \delta t$ ($n = 1, 2, 3, \dots$) set in Step 2.

9.7.4 Update After Light-sleep

1. Configure RTC timer before the chip goes to Light-sleep mode, to record the exact sleep time. For more information, see Chapter 4 *Low-power Management [to be added later]*.
2. Read the sleep time from the RTC timer when the chip wakes up from Light-sleep mode.
3. Read the current count value of system timer, see Section 9.7.1.

4. Convert the time value recorded by RTC timer from the clock cycles based on RTC_SLOW_CLK to that based on 16 MHz CNT_CLK. For example, if the frequency of RTC_SLOW_CLK is 32 kHz, the recorded RTC timer value should be converted by multiplying by 500.
5. Add the converted RTC value to the current count value of system timer:
 - Fill the new value into SYSTIMER_TIMER_UNIT n _LOAD_LO (low 32 bits) and SYSTIMER_TIMER_UNIT n _LOAD_HI (high 20 bits).
 - Set SYSTIMER_TIMER_UNIT n _LOAD to load the new timer value into the system timer. By such way, the system timer is updated.

9.8 Register Summary

The addresses in this section are relative to system timer base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Clock Control Register			
SYSTIMER_CONF_REG	Configure system timer clock	0x0000	R/W
UNIT0 Control and Configuration Registers			
SYSTIMER_UNIT0_OP_REG	Read UNIT0 value to registers	0x0004	varies
SYSTIMER_UNIT0_LOAD_HI_REG	High 20 bits to be loaded to UNIT0	0x000C	R/W
SYSTIMER_UNIT0_LOAD_LO_REG	Low 32 bits to be loaded to UNIT0	0x0010	R/W
SYSTIMER_UNIT0_VALUE_HI_REG	UNIT0 value, high 20 bits	0x0040	RO
SYSTIMER_UNIT0_VALUE_LO_REG	UNIT0 value, low 32 bits	0x0044	RO
SYSTIMER_UNIT0_LOAD_REG	UNIT0 synchronization register	0x005C	WT
UNIT1 Control and Configuration Registers			
SYSTIMER_UNIT1_OP_REG	Read UNIT1 value to registers	0x0008	varies
SYSTIMER_UNIT1_LOAD_HI_REG	High 20 bits to be loaded to UNIT1	0x0014	R/W
SYSTIMER_UNIT1_LOAD_LO_REG	Low 32 bits to be loaded to UNIT1	0x0018	R/W
SYSTIMER_UNIT1_VALUE_HI_REG	UNIT1 value, high 20 bits	0x0048	RO
SYSTIMER_UNIT1_VALUE_LO_REG	UNIT1 value, low 32 bits	0x004C	RO
SYSTIMER_UNIT1_LOAD_REG	UNIT1 synchronization register	0x0060	WT
Comparator0 Control and Configuration Registers			
SYSTIMER_TARGET0_HI_REG	Alarm value to be loaded to COMPO, high 20 bits	0x001C	R/W
SYSTIMER_TARGET0_LO_REG	Alarm value to be loaded to COMPO, low 32 bits	0x0020	R/W
SYSTIMER_TARGET0_CONF_REG	Configure COMPO alarm mode	0x0034	R/W
SYSTIMER_COMPO_LOAD_REG	COMPO synchronization register	0x0050	WT
Comparator1 Control and Configuration Registers			
SYSTIMER_TARGET1_HI_REG	Alarm value to be loaded to COMP1, high 20 bits	0x0024	R/W
SYSTIMER_TARGET1_LO_REG	Alarm value to be loaded to COMP1, low 32 bits	0x0028	R/W
SYSTIMER_TARGET1_CONF_REG	Configure COMP1 alarm mode	0x0038	R/W
SYSTIMER_COMP1_LOAD_REG	COMP1 synchronization register	0x0054	WT
Comparator2 Control and Configuration Registers			
SYSTIMER_TARGET2_HI_REG	Alarm value to be loaded to COMP2, high 20 bits	0x002C	R/W
SYSTIMER_TARGET2_LO_REG	Alarm value to be loaded to COMP2, low 32 bits	0x0030	R/W
SYSTIMER_TARGET2_CONF_REG	Configure COMP2 alarm mode	0x003C	R/W
SYSTIMER_COMP2_LOAD_REG	COMP2 synchronization register	0x0058	WT
Interrupt Registers			
SYSTIMER_INT_ENA_REG	Interrupt enable register of system timer	0x0064	R/W
SYSTIMER_INT_RAW_REG	Interrupt raw register of system timer	0x0068	R/WTC/SS
SYSTIMER_INT_CLR_REG	Interrupt clear register of system timer	0x006C	WT

Name	Description	Address	Access
SYSTIMER_INT_ST_REG	Interrupt status register of system timer	0x0070	RO
COMP0 Status Registers			
SYSTIMER_REAL_TARGET0_LO_REG	Actual target value of COMP0, low 32 bits	0x0074	RO
SYSTIMER_REAL_TARGET0_HI_REG	Actual target value of COMP0, high 20 bits	0x0078	RO
COMP1 Status Registers			
SYSTIMER_REAL_TARGET1_LO_REG	Actual target value of COMP1, low 32 bits	0x007C	RO
SYSTIMER_REAL_TARGET1_HI_REG	Actual target value of COMP1, high 20 bits	0x0080	RO
COMP2 Status Registers			
SYSTIMER_REAL_TARGET2_LO_REG	Actual target value of COMP2, low 32 bits	0x0084	RO
SYSTIMER_REAL_TARGET2_HI_REG	Actual target value of COMP2, high 20 bits	0x0088	RO
Version Register			
SYSTIMER_DATE_REG	Version control register	0x00FC	R/W

9.9 Registers

The addresses in this section are relative to system timer base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 9.1. SYSTIMER_CONF_REG (0x0000)

SYSTIMER_CLK_EN										(reserved)										SYSTIMER_ETM_EN										
SYSTIMER_TIMER_UNIT0_WORK_EN										(reserved)										(reserved)										
SYSTIMER_TIMER_UNIT1_WORK_EN										(reserved)										(reserved)										
SYSTIMER_TIMER_UNIT0_CORE0_STALL_EN										(reserved)										(reserved)										
SYSTIMER_TIMER_UNIT1_CORE0_STALL_EN										(reserved)										(reserved)										
SYSTIMER_TIMER_UNIT0_CORE1_STALL_EN										(reserved)										(reserved)										
SYSTIMER_TIMER_UNIT1_CORE1_STALL_EN										(reserved)										(reserved)										
SYSTIMER_TARGET0_WORK_EN										(reserved)										(reserved)										
SYSTIMER_TARGET1_WORK_EN										(reserved)										(reserved)										
SYSTIMER_TARGET2_WORK_EN										(reserved)										(reserved)										
31	30	29	28	27	26	25	24	23	22	21											2	1	0							
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Reset

SYSTIMER_ETM_EN Configures whether to enable generation of ETM events.

0: Disable

1: Enable

(R/W)

SYSTIMER_TARGET2_WORK_EN Configures whether to enable COMP2.

0: Disable

1: Enable

(R/W)

SYSTIMER_TARGET1_WORK_EN Configures whether to enable COMP1. See details in [SYSTIMER_TARGET2_WORK_EN](#). (R/W)

SYSTIMER_TARGET0_WORK_EN Configures whether to enable COMP0. See details in [SYSTIMER_TARGET2_WORK_EN](#). (R/W)

SYSTIMER_TIMER_UNIT1_CORE1_STALL_EN Configures whether UNIT1 is stalled when CORE1 is stalled.

0: UNIT1 is not stalled.

1: UNIT1 is stalled.

(R/W)

SYSTIMER_TIMER_UNIT1_CORE0_STALL_EN Configures whether UNIT1 is stalled when CORE0 is stalled. See details in [SYSTIMER_TIMER_UNIT1_CORE1_STALL_EN](#). (R/W)

SYSTIMER_TIMER_UNIT0_CORE1_STALL_EN Configures whether UNIT0 is stalled when CORE1 is stalled. See details in [SYSTIMER_TIMER_UNIT1_CORE1_STALL_EN](#). (R/W)

SYSTIMER_TIMER_UNIT0_CORE0_STALL_EN Configures whether UNIT0 is stalled when CORE0 is stalled. See details in [SYSTIMER_TIMER_UNIT1_CORE1_STALL_EN](#). (R/W)

Continued on the next page...

Register 9.1. SYSTIMER_CONF_REG (0x0000)

Continued from the previous page...

SYSTIMER_TIMER_UNIT1_WORK_EN Configures whether to enable UNIT1.

0: Disable

1: Enable

(R/W)

SYSTIMER_TIMER_UNITO_WORK_EN Configures whether to enable UNITO.

0: Disable

1: Enable

(R/W)

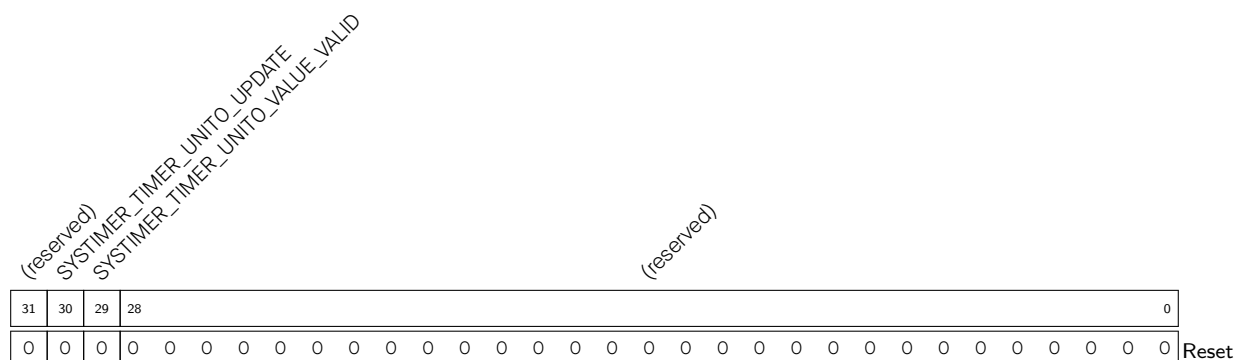
SYSTIMER_CLK_EN Configures register clock gating.

0: Only enable needed clock for register read or write operations.

1: Register clock is always enabled for read and write operations.

(R/W)

Register 9.2. SYSTIMER_UNIT0_OP_REG (0x0004)



SYSTIMER_TIMER_UNITO_VALUE_VALID Represents whether UNITO value is synchronized and valid.

0: UNIT0 value is neither synchronized nor valid

1: UNIT0 value is synchronized and valid

(R/SS/WTC)

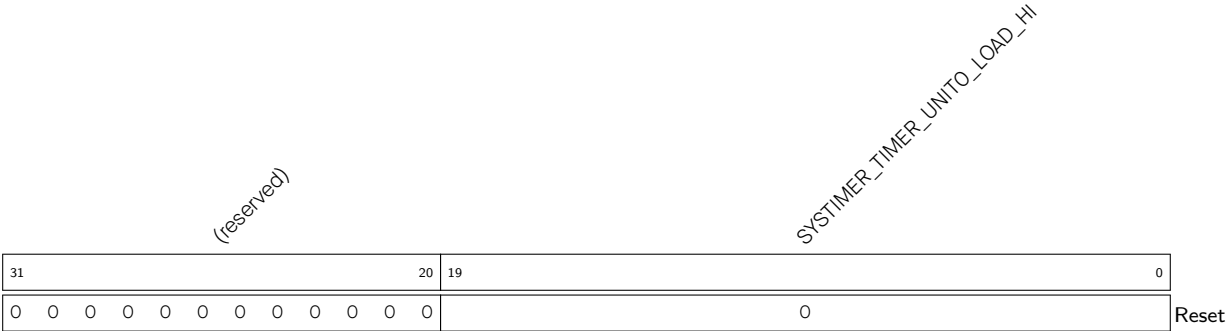
SYSTIMER_TIMER_UNITO_UPDATE Configures whether to update timer UNITO, i.e., reads the UNITO count value to **SYSTIMER_TIMER_UNITO_VALUE_HI** and **SYSTIMER_TIMER_UNITO_VALUE_LO**.

0: No effect

1: Update timer UNIT0

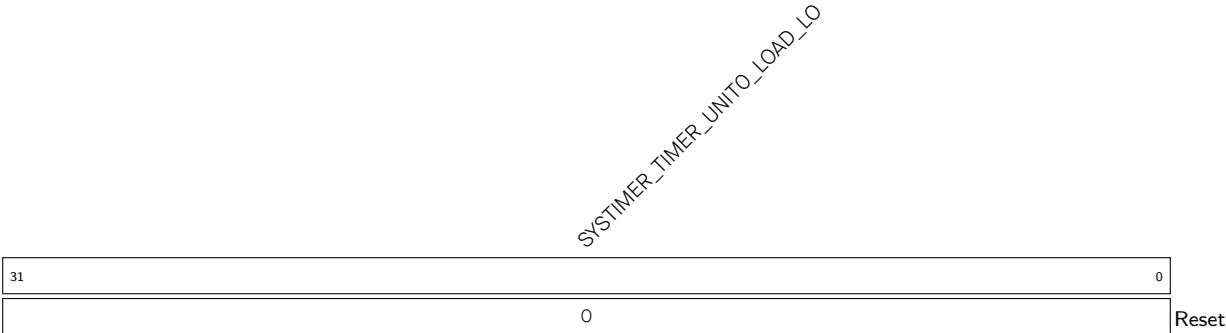
(WT)

Register 9.3. SYSTIMER_UNITO_LOAD_HI_REG (0x000C)



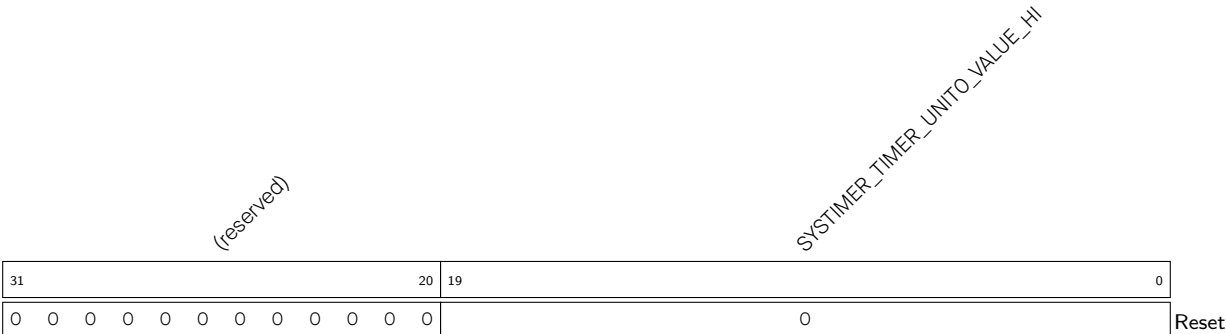
SYSTIMER_TIMER_UNITO_LOAD_HI Configures the value to be loaded to UNITO, high 20 bits. (R/W)

Register 9.4. SYSTIMER_UNITO_LOAD_LO_REG (0x0010)



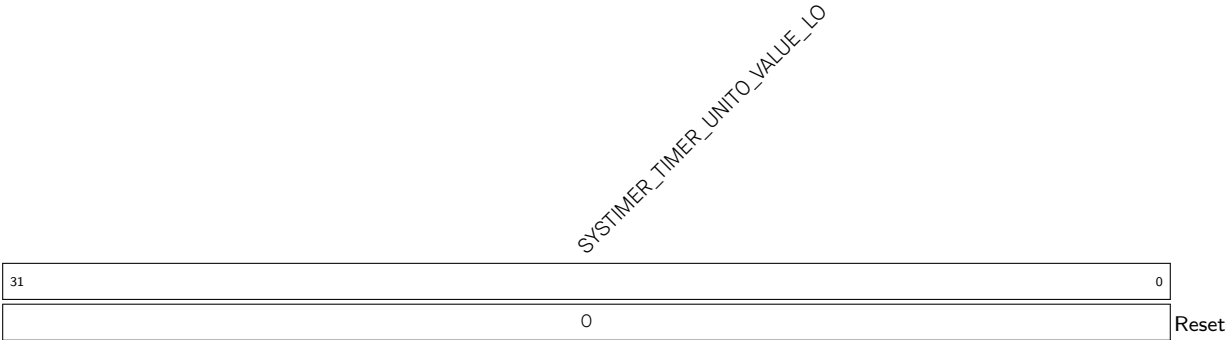
SYSTIMER_TIMER_UNITO_LOAD_LO Configures the value to be loaded to UNITO, low 32 bits. (R/W)

Register 9.5. SYSTIMER_UNITO_VALUE_HI_REG (0x0040)



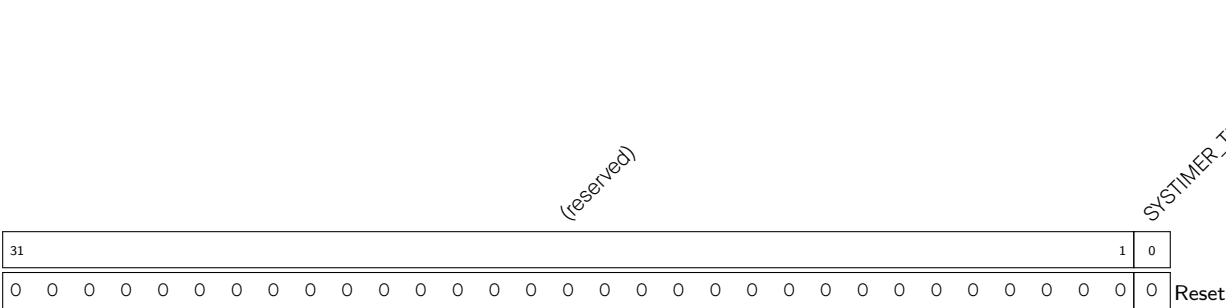
SYSTIMER_TIMER_UNITO_VALUE_HI Represents UNITO read value, high 20 bits. (RO)

Register 9.6. SYSTIMER_UNITO_VALUE_LO_REG (0x0044)



SYSTIMER_TIMER_UNITO_VALUE_LO Represents UNITO read value, low 32 bits. (RO)

Register 9.7. SYSTIMER_UNITO_LOAD_REG (0x005C)



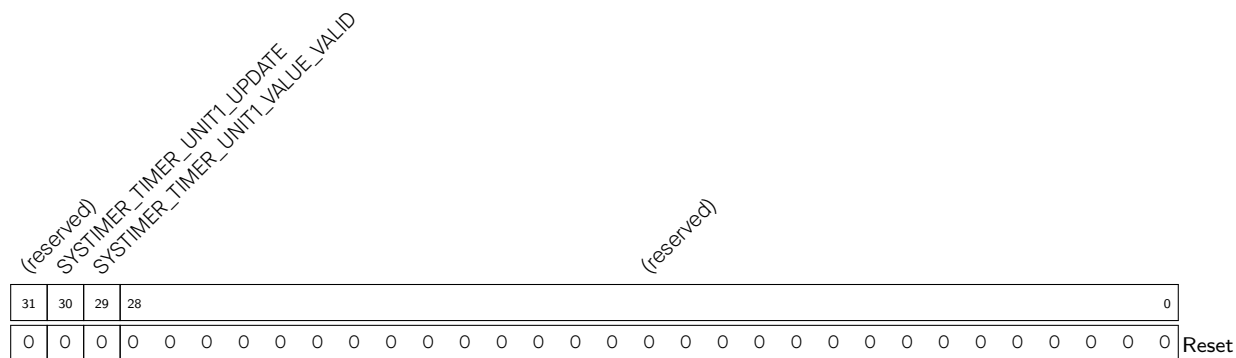
SYSTIMER_TIMER_UNITO_LOAD Configures whether to reload the value of UNITO, i.e., reloads the values of [SYSTIMER_TIMER_UNITO_VALUE_HI](#) and [SYSTIMER_TIMER_UNITO_VALUE_LO](#) to UNITO.

0: No effect

1: Reload the value of UNITO

(WT)

Register 9.8. SYSTIMER_UNIT1_OP_REG (0x0008)



SYSTIMER_TIMER_UNIT1_VALUE_VALID Represents UNIT1 value is synchronized and valid.
(R/SS/WTC)

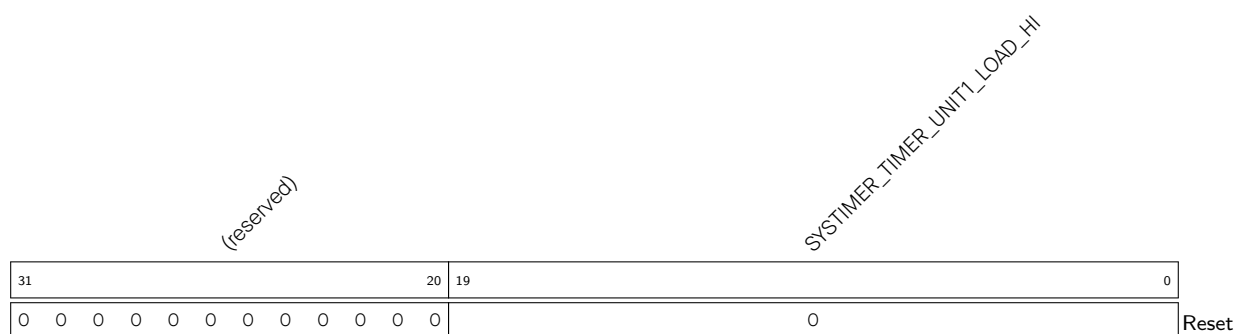
SYSTIMER_TIMER_UNIT1_UPDATE Configures whether to update timer UNIT1, i.e., reads the UNIT1 count value to [SYSTIMER_TIMER_UNIT1_VALUE_HI](#) and [SYSTIMER_TIMER_UNIT1_VALUE_LO](#).

0: No effect

1: Update timer UNIT1

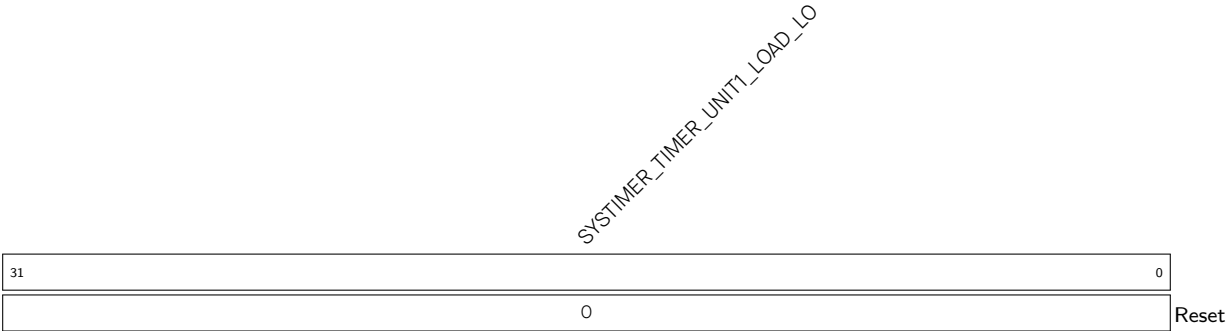
(WT)

Register 9.9. SYSTIMER_UNIT1_LOAD_HI_REG (0x0014)



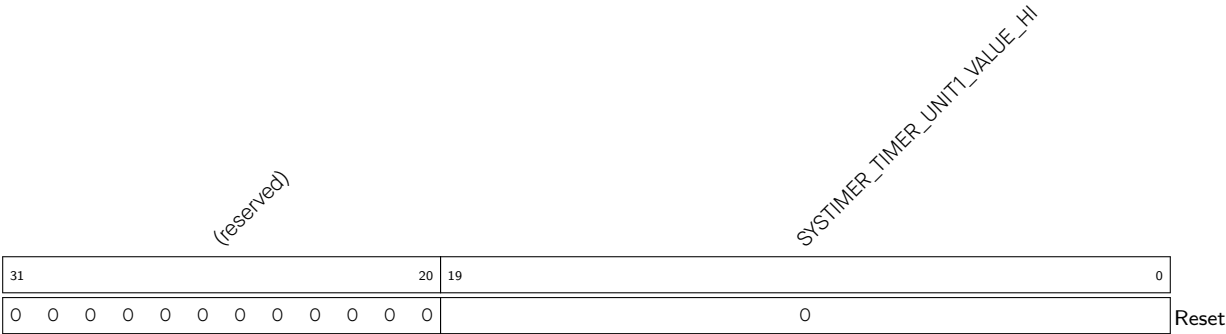
SYSTIMER_TIMER_UNIT1_LOAD_HI Configures the value to be loaded to UNIT1, high 20 bits. (R/W)

Register 9.10. SYSTIMER_UNIT1_LOAD_LO_REG (0x0018)



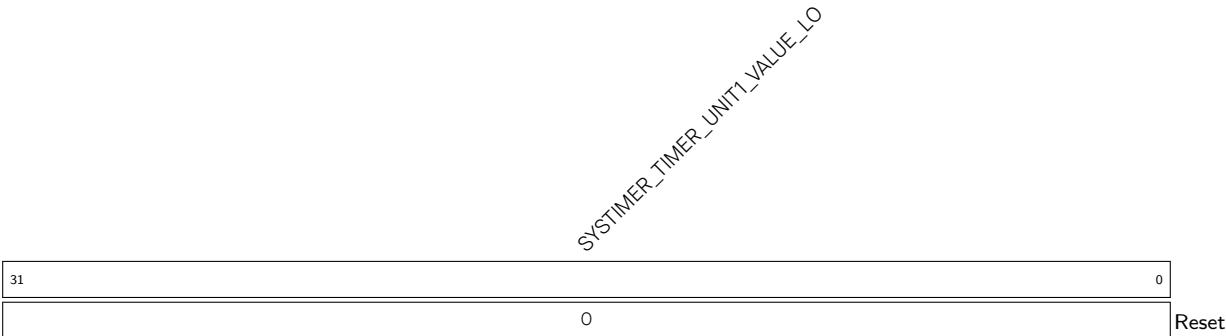
SYSTIMER_TIMER_UNIT1_LOAD_LO Configures the value to be loaded to UNIT1, low 32 bits. (R/W)

Register 9.11. SYSTIMER_UNIT1_VALUE_HI_REG (0x0048)



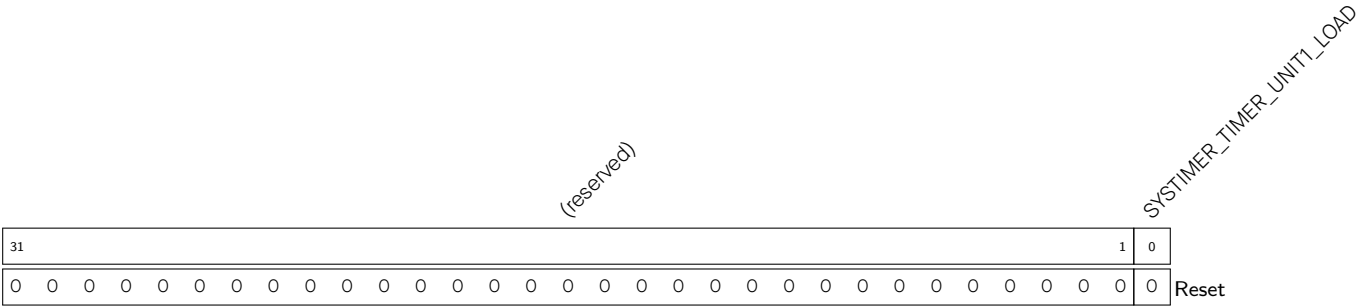
SYSTIMER_TIMER_UNIT1_VALUE_HI Represents UNIT1 read value, high 20 bits. (RO)

Register 9.12. SYSTIMER_UNIT1_VALUE_LO_REG (0x004C)



SYSTIMER_TIMER_UNIT1_VALUE_LO Represents UNIT1 read value, low 32 bits. (RO)

Register 9.13. SYSTIMER_UNIT1_LOAD_REG (0x0060)

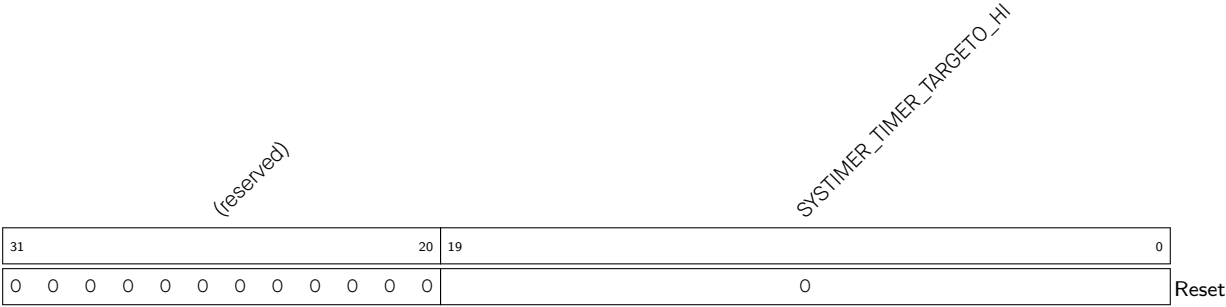


SYSTIMER_TIMER_UNIT1_LOAD Configures whether to reload the value of UNIT1, i.e., reload the values of [SYSTIMER_TIMER_UNIT1_VALUE_HI](#) and [SYSTIMER_TIMER_UNIT1_VALUE_LO](#) to UNIT1.

0: No effect

1: Reload the value of UNIT1 (WT)

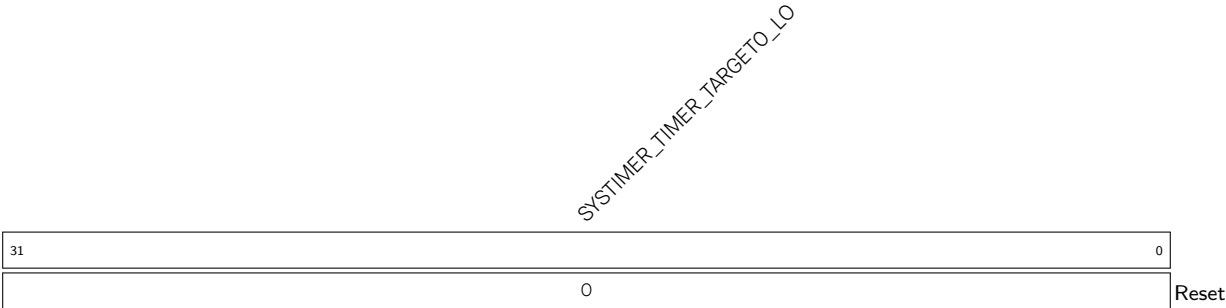
Register 9.14. SYSTIMER_TARGETO_HI_REG (0x001C)



SYSTIMER_TIMER_TARGETO_HI Configures the alarm value to be loaded to COMP0, high 20 bits.

(R/W)

Register 9.15. SYSTIMER_TARGETO_LO_REG (0x0020)



SYSTIMER_TIMER_TARGETO_LO Configures the alarm value to be loaded to COMP0, low 32 bits.

(R/W)

Register 9.16. SYSTIMER_TARGET0_CONF_REG (0x0034)

Diagram illustrating the structure of the SYSTIMER_TARGET0 register. The register is 32 bits wide, divided into two 16-bit fields. The upper 16 bits are labeled 'SYSTIMER_TARGET0_TIMER_UNIT_SEL' and 'SYSTIMER_TARGET0_PERIOD_MODE'. The lower 16 bits are labeled 'SYSTIMER_TARGET0_PERIOD'. The register is shown with a value of 0x000000.

SYSTIMER_TARGET0_PERIOD Configures COMPO alarm period. (R/W)

SYSTIMER TARGET0 PERIOD MODE Selects the two alarm modes for COMP0.

0: Target mode

1: Period mode

(R/W)

SYSTIMER_TARGET0_TIMER_UNIT_SEL Chooses the count value for comparison with COMPO.

0: Use the count value from UNIT0

1: Use the count value from UNIT1

(R/W)

Register 9.17. SYSTIMER_COMPO_LOAD_REG (0x0050)

Diagram of the SYSTIMER_TIMER_LOAD register structure. The register is 32 bits wide. Bits 31 down to 1 are reserved, and bit 0 is the Reset bit.

31																															1	0
Reset																																

SYSTIMER_TIMER_COMPO_LOAD Configures whether to enable COMPO synchronization, i.e., reload the alarm value/period to COMPO.

0: No effect

1: Enable COMPO synchronization

(WT)

Register 9.18. SYSTIMER_TARGET1_HI_REG (0x0024)

(reserved)												SYSTIMER_TIMER_TARGET1_HI																															
31												20												19																			0
0 0 0 0 0 0 0 0 0 0 0 0												0																			Reset												

SYSTIMER_TIMER_TARGET1_HI Configures the alarm value to be loaded to COMP1, high 20 bits.
(R/W)

Register 9.19. SYSTIMER_TARGET1_LO_REG (0x0028)

SYSTIMER_TIMER_TARGET1_LO																															
31																															0
0																															
Reset																															

SYSTIMER_TIMER_TARGET1_LO Configures the alarm value to be loaded to COMP1, low 32 bits.
(R/W)

Register 9.20. SYSTIMER_TARGET1_CONF_REG (0x0038)

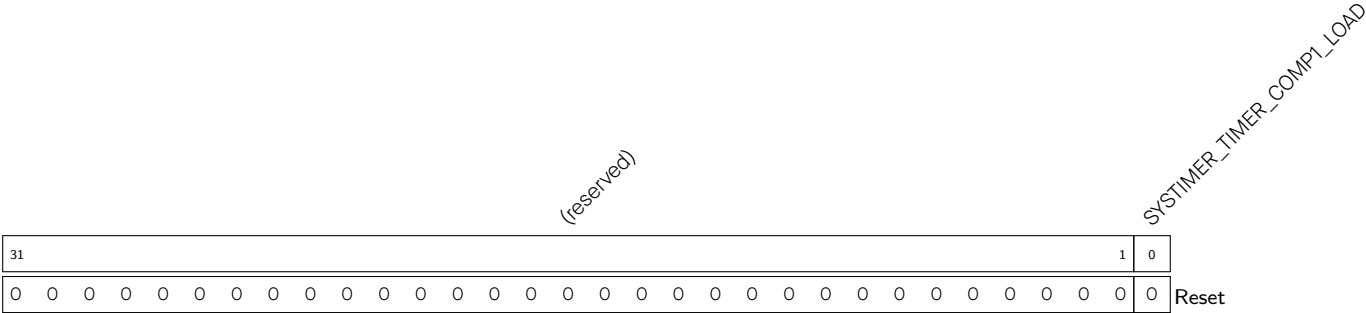
SYSTIMER_TARGET1_TIMER_UNIT_SEL										SYSTIMER_TARGET1_PERIOD_MODE										(reserved)										SYSTIMER_TARGET1_PERIOD									
31	30	29					26	25																						0									
0	0	0	0	0	0	0	0x000000																					Reset											

SYSTIMER_TARGET1_PERIOD Configures COMP1 alarm period. (R/W)

SYSTIMER_TARGET1_PERIOD_MODE Selects the two alarm modes for COMP1. See details in [SYSTIMER_TARGET0_PERIOD_MODE](#). (R/W)

SYSTIMER_TARGET1_TIMER_UNIT_SEL Chooses the count value for comparison with COMP1. See details in [SYSTIMER_TARGET0_TIMER_UNIT_SEL](#). (R/W)

Register 9.21. SYSTIMER_COMP1_LOAD_REG (0x0054)



SYSTIMER_TIMER_COMP1_LOAD Configures whether to enable COMP1 synchronization, i.e., reload the alarm value/period to COMP1.

0: No effect

1: Enable COMP1 synchronization (WT)

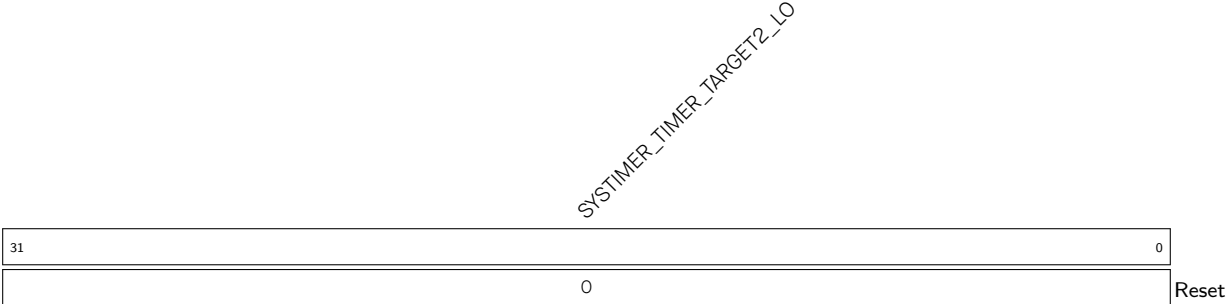
Register 9.22. SYSTIMER_TARGET2_HI_REG (0x002C)



SYSTIMER_TIMER_TARGET2_HI Configures the alarm value to be loaded to COMP2, high 20 bits.

(R/W)

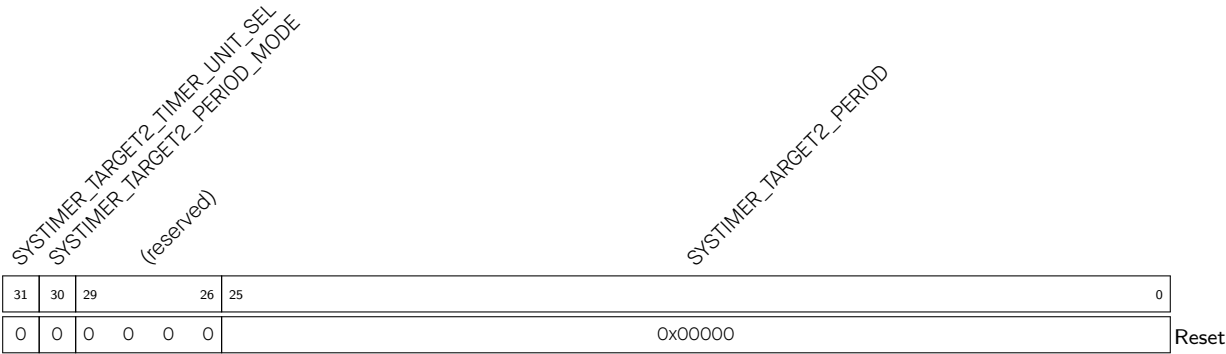
Register 9.23. SYSTIMER_TARGET2_LO_REG (0x0030)



SYSTIMER_TIMER_TARGET2_LO Configures the alarm value to be loaded to COMP2, low 32 bits.

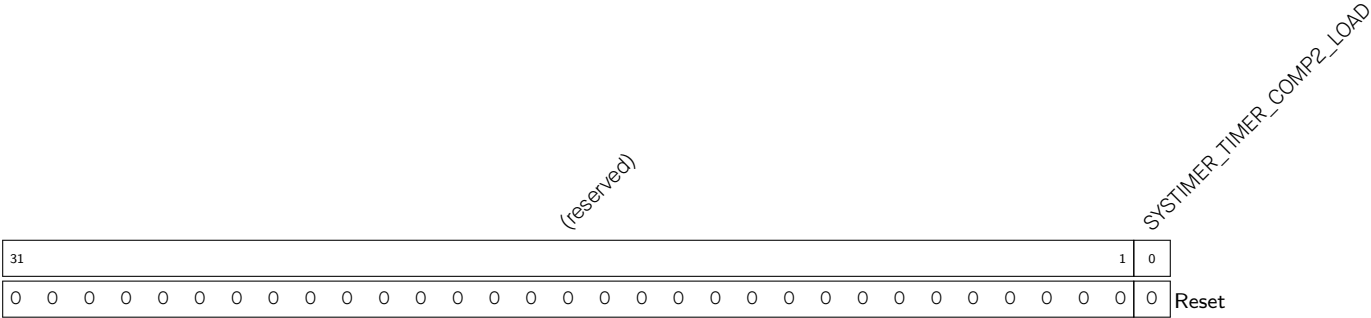
(R/W)

Register 9.24. SYSTIMER_TARGET2_CONF_REG (0x003C)



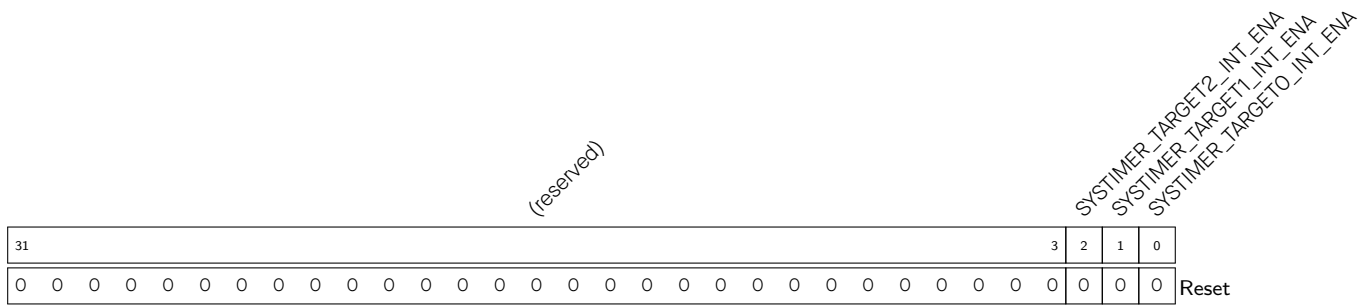
- SYSTIMER_TARGET2_PERIOD** Configures COMP2 alarm period. (R/W)
- SYSTIMER_TARGET2_PERIOD_MODE** Configures Configures the two alarm modes for COMP2. See details in [SYSTIMER_TARGET2_PERIOD_MODE](#). (R/W)
- SYSTIMER_TARGET2_TIMER_UNIT_SEL** Chooses the count value for comparison with COMP2. See details in [SYSTIMER_TARGET2_TIMER_UNIT_SEL](#). (R/W)

Register 9.25. SYSTIMER_COMP2_LOAD_REG (0x0058)



- SYSTIMER_TIMER_COMP2_LOAD** Configures whether to enable COMP2 synchronization, i.e., reload the alarm value/period to COMP2.
0: No effect
1: Enable COMP2 synchronization (WT)

Register 9.26. SYSTIMER_INT_ENA_REG (0x0064)

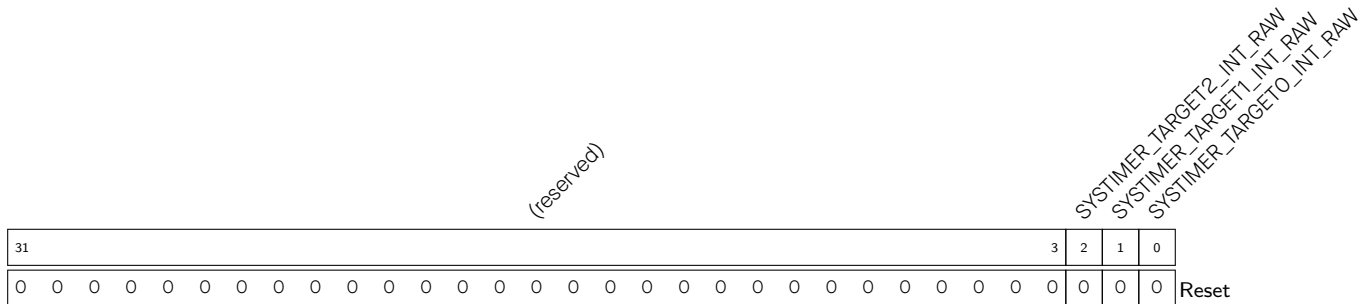


SYSTIMER_TARGET0_INT_ENA Write 1 to enable SYSTIMER_TARGET0_INT. (R/W)

SYSTIMER_TARGET1_INT_ENA Write 1 to enable SYSTIMER_TARGET1_INT. (R/W)

SYSTIMER_TARGET2_INT_ENA Write 1 to enable SYSTIMER_TARGET2_INT. (R/W)

Register 9.27. SYSTIMER_INT_RAW_REG (0x0068)



SYSTIMER_TARGET0_INT_RAW The raw interrupt status of SYSTIMER_TARGET0_INT. (R/WTC/SS)

SYSTIMER_TARGET1_INT_RAW The raw interrupt status of SYSTIMER_TARGET1_INT. (R/WTC/SS)

SYSTIMER_TARGET2_INT_RAW The raw interrupt status of SYSTIMER_TARGET2_INT. (R/WTC/SS)

Register 9.28. SYSTIMER_INT_CLR_REG (0x006C)

Diagram of the 32-bit SYS_TMR_CFG register structure:

Field Name	Width (bits)	Reset Value
(reserved)	28	0
SYS_TIMER_TARGET2_INT_CLR	3	0
SYS_TIMER_TARGET1_INT_CLR	2	0
SYS_TIMER_TARGET0_INT_CLR	1	0

SYSTIMER_TARGET0_INT_CLR Write 1 to clear SYSTIMER_TARGET0_INT. (WT)

SYSTIMER TARGET1 INT CLR Write 1 to clear SYSTIMER TARGET1 INT. (WT)

SYSTIMER TARGET2 INT CLR Write 1 to clear SYSTIMER TARGET2 INT. (WT)

Register 9.29. SYSTIMER_INT_ST_REG (0x0070)

Diagram of the 32-bit SYSTEMER register structure:

- Bits 31-3: (reserved)
- Bits 2-1: SYSTEMER_TARGET2_INT_ST
- Bits 1-0: SYSTEMER_TARGET1_INT_ST
- Bit 0: SYSTEMER_TARGET0_INT_ST

The register is shown with all bits set to 0.

SYSTIMER_TARGET0_INT_ST	The masked interrupt status of SYSTIMER_TARGET0_INT. (RO)

SYSTIMER_TARGET1_INT_ST The masked interrupt status of SYSTIMER_TARGET1_INT. (RO)

SYSTIMER_TARGET2_INT_ST	The masked interrupt status of SYSTIMER_TARGET2_INT. (RO)

Register 9.30. SYSTIMER_REAL_TARGET0_LO_REG (0x0074)

Diagram of the SYSTIMER_TARGET0_LO_R0 register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The label 'SYSTIMER_TARGET0_LO_R0' is written diagonally across the register. A 'Reset' label is at the bottom right.

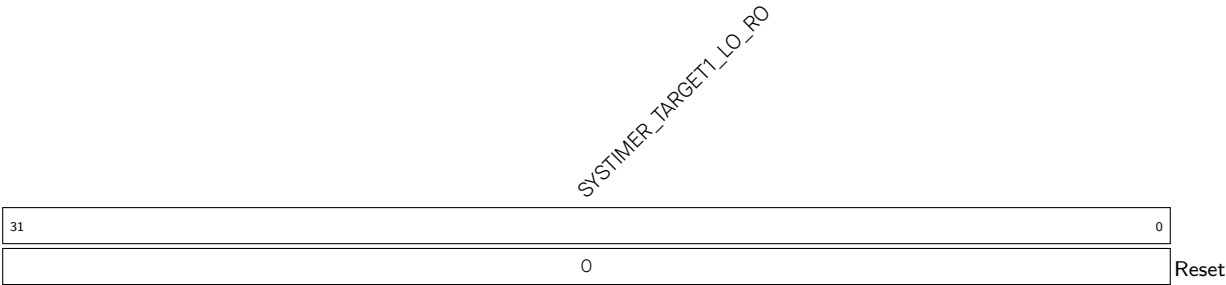
SYSTIMER_TARGET0_LO_RO Represents the actual target value of COMPO, low 32 bits. (RO)

Register 9.31. SYSTIMER_REAL_TARGET0_HI_REG (0x0078)



SYSTIMER_TARGET0_HI_RO Represents the actual target value of COMP0, high 20 bits. (RO)

Register 9.32. SYSTIMER_REAL_TARGET1_LO_REG (0x007C)



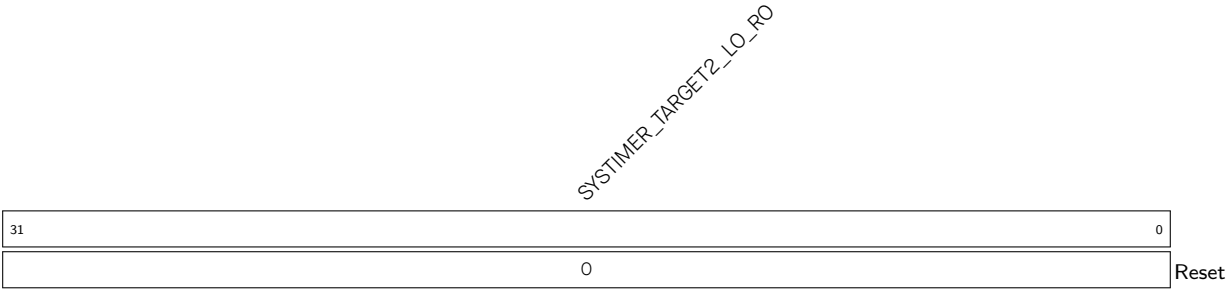
SYSTIMER_TARGET1_LO_RO Represents the actual target value of COMP1, low 32 bits. (RO)

Register 9.33. SYSTIMER_REAL_TARGET1_HI_REG (0x0080)



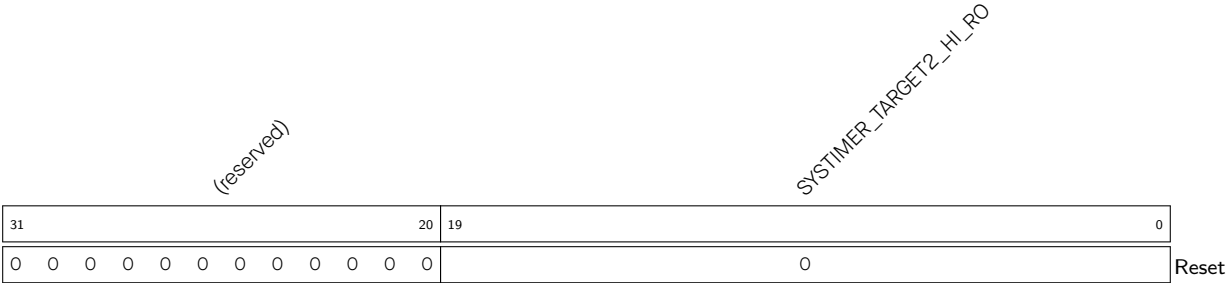
SYSTIMER_TARGET1_HI_RO Represents the actual target value of COMP1, high 20 bits. (RO)

Register 9.34. SYSTIMER_REAL_TARGET2_LO_REG (0x0084)



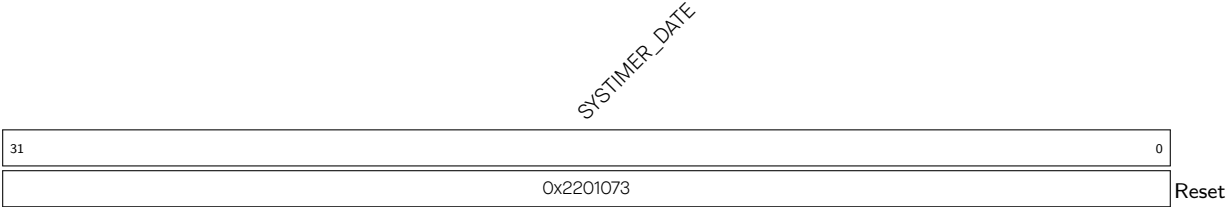
SYSTIMER_TARGET2_LO_RO Represents the actual target value of COMP2, low 32 bits. (RO)

Register 9.35. SYSTIMER_REAL_TARGET2_HI_REG (0x0088)



SYSTIMER_TARGET2_HI_RO Represents the actual target value of COMP2, high 20 bits. (RO)

Register 9.36. SYSTIMER_DATE_REG (0x00FC)



SYSTIMER_DATE Version control register. (R/W)

Chapter 10

Timer Group (TIMG)

10.1 Overview

General-purpose timers can be used to precisely time an interval, trigger an interrupt after a particular interval (periodically and aperiodically), or act as a hardware clock. As shown in Figure 10.1-1, the ESP32-C5 chip contains two timer groups, namely timer group 0 and timer group 1. Each timer group consists of one general-purpose timer referred to as T0 and one Main System Watchdog Timer. The general-purpose timer is based on a 16-bit prescaler and a 54-bit auto-reload-capable up-down counter.

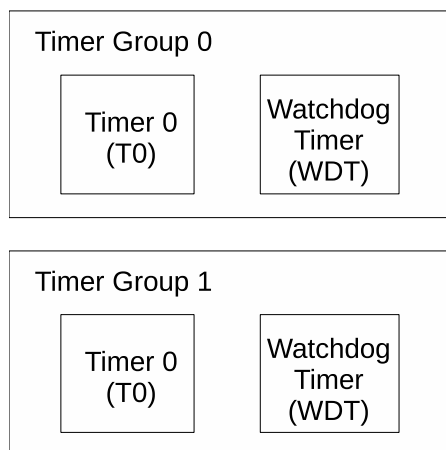


Figure 10.1-1. Timer Group Overview

Note that while the Main System Watchdog Timer registers are described in this chapter, their functional description is included in Chapter 11 *Watchdog Timers (WDT)*. Therefore, the term "timer" within this chapter refers to the general-purpose timer.

10.2 Feature List

The timer's features are summarized as follows:

- A 54-bit time-base counter programmable to incrementing or decrementing
- Three clock sources: PLL_F80M_CLK or XTAL_CLK or RC_FAST_CLK
- A 16-bit clock prescaler, from 2 to 65536
- Able to read real-time value of the time-base counter
- Able to halt and resume the time-base counter
- Programmable alarm generation

- Timer value reload — Auto-reload at alarm or software-controlled instant reload
- Frequency calculation of slow clock for timer group 0
- Level interrupt generation
- Support several ETM tasks and events

10.3 Architectural Overview

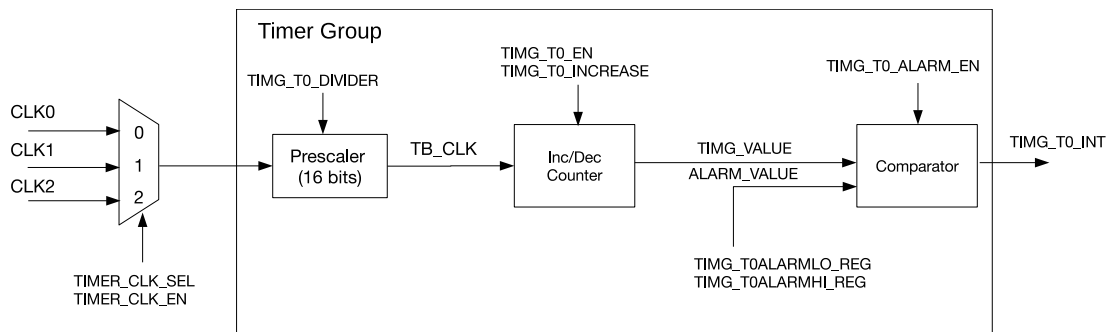


Figure 10.3-1. Timer Group Architecture

Figure 10.3-1 is a diagram of timer T0 in a timer group. T0 contains a 16-bit integer divider as a prescaler, a timer-based counter, and a comparator for alarm generation.

10.4 Functional Description

10.4.1 16-bit Prescaler and Clock Selection

Take the T0 in timer group 0 as an example:

- The timer can select its clock source by setting the `PCR_TGO_TIMER_CLK_SEL` field of the `PCR_TIMERGROUPO_TIMER_CLK_CONF_REG` register. When the field is 0, CLK0 is selected, which indicates XTAL_CLK; when the field is 1, CLK1 is selected, which indicates RC_FAST_CLK; when the field is 2, CLK2 is selected, which indicates PLL_F80M_CLK.
- The selected clock can be switched on by setting `PCR_TGO_TIMER_CLK_EN` field of the `PCR_TIMERGROUPO_TIMER_CLK_CONF_REG` register to 1 and switched off by setting it to 0. This field is indicated as `TIMER_CLK_EN` in Figure 10.3-1. The clock is then divided by a 16-bit prescaler to generate the time-base counter clock (TB_CLK) used by the time-base counter. The divisor of the prescaler can be configured through the `TIMG_T0_DIVIDER` field.

`TIMG_T0_DIVIDER` field can be configured as 0 ~ 65535 for a divisor range of 2 ~ 65536. To be more specific, when `TIMG_T0_DIVIDER` is configured as:

- 0: the divisor is 65536
- 1: the divisor is 2
- 2: the divisor is also 2
- 3 ~ 65525: the divisor is 3 ~ 65535

To modify the 16-bit prescaler, please first configure the [TIMG_TO_DIVIDER](#) field, and then set [TIMG_TO_DIVCNT_RST](#) to 1. Meanwhile, the timer must be disabled (i.e., [TIMG_TO_EN](#) should be cleared). Otherwise, the result can be unpredictable.

10.4.2 54-bit Time-base Counter

The 54-bit time-base counter is based on TB_CLK and can be configured to increment or decrement via the [TIMG_TO_INCREASE](#) field. The time-base counter can be enabled or disabled by setting or clearing the [TIMG_TO_EN](#) field, respectively. When enabled, the time-base counter increments or decrements on each cycle of TB_CLK. When disabled, the time-base counter is essentially frozen. Note that the [TIMG_TO_INCREASE](#) field can be changed no matter whether [TIMG_TO_EN](#) is set or not, and this will cause the time-base counter to change direction instantly.

To read the 54-bit value of the time-base counter, the timer value must be latched to two registers before being read by the CPU (due to the CPU being 32-bit). By writing any value to the [TIMG_TOUUPDATE_REG](#), the current value of the 54-bit timer starts to be latched into the [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#) registers containing the lower 32-bits and higher 22-bits, respectively. When [TIMG_TOUUPDATE_REG](#) is cleared by hardware, it indicates the latch operation has been completed and current timer value can be read from the [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#) registers. [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#) registers will remain unchanged for the CPU to read in its own time until [TIMG_TOUUPDATE_REG](#) is written to again.

10.4.3 Alarm Generation

A timer can be configured to trigger an alarm when the timer's current value matches the alarm value. An alarm will cause an interrupt to occur and (optionally) an automatic reload of the timer's current value (see Section 10.4.4).

The 54-bit alarm value is configured using [TIMG_TOALARMLO_REG](#) and [TIMG_TOALARMHI_REG](#), which represent the lower 32-bits and higher 22-bits of the alarm value, respectively. However, the configured alarm value is ineffective until the alarm is enabled by setting the [TIMG_TO_ALARM_EN](#) field. To avoid alarm being enabled "too late" (i.e., the timer value has already passed the alarm value when the alarm is enabled), the hardware will trigger the alarm immediately if the current timer value is:

- higher than the alarm value (within a defined range) when the up-down counter increments
- lower than the alarm value (within a defined range) when the up-down counter decrements

Table 10.4-1 and Table 10.4-2 show the relationship between the current value of the timer, the alarm value, and when an alarm is triggered. The current time value and the alarm value are defined as follows:

- $TIMG\_VALUE = \{TIMG\_TOHI\_REG, TIMG\_TOLO\_REG\}$
- $ALARM\_VALUE = \{TIMG\_TOALARMHI\_REG, TIMG\_TOALARMLO\_REG\}$

Table 10.4-1. Alarm Generation When Up-Down Counter Increments

Scenario	Range	Alarm
1	$\text{ALARM\_VALUE} - \text{TIMG\_VALUE} > 2^{53}$	Triggered
2	$0 < \text{ALARM\_VALUE} - \text{TIMG\_VALUE} \leq 2^{53}$	Triggered when the up-down counter counts TIMG_VALUE up to ALARM_VALUE
3	$0 \leq \text{TIMG\_VALUE} - \text{ALARM\_VALUE} < 2^{53}$	Triggered
4	$\text{TIMG\_VALUE} - \text{ALARM\_VALUE} \geq 2^{53}$	Triggered when the up-down counter restarts counting up from 0 after reaching the timer's maximum value and counts TIMG_VALUE up to ALARM_VALUE

Table 10.4-2. Alarm Generation When Up-Down Counter Decrements

Scenario	Range	Alarm
5	$\text{TIMG\_VALUE} - \text{ALARM\_VALUE} > 2^{53}$	Triggered
6	$0 < \text{TIMG\_VALUE} - \text{ALARM\_VALUE} \leq 2^{53}$	Triggered when the up-down counter counts TIMG_VALUE down to ALARM_VALUE
7	$0 \leq \text{ALARM\_VALUE} - \text{TIMG\_VALUE} < 2^{53}$	Triggered
8	$\text{ALARM\_VALUE} - \text{TIMG\_VALUE} \geq 2^{53}$	Triggered when the up-down counter restarts counting down from the timer's maximum value after reaching the minimum value and counts TIMG_VALUE down to ALARM_VALUE

When an alarm occurs, the [TIMG_TO_ALARM_EN](#) field is automatically cleared and no alarm will occur again until the [TIMG_TO_ALARM_EN](#) is set next time.

10.4.4 Timer Reload

A timer is reloaded when a timer's current value is overwritten with a reload value stored in the [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#) fields that correspond to the lower 32-bits and higher 22-bits of the timer's new value, respectively. However, writing a reload value to [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#) will not cause the timer's current value to change. Instead, the reload value is ignored by the timer until a reload event occurs. A reload event can be triggered either by a software instant reload or an auto-reload at alarm.

A software instant reload is triggered by the CPU writing any value to [TIMG_TOLOAD_REG](#), which causes the timer's current value to be instantly reloaded. If [TIMG_TO_EN](#) is set, the timer will continue incrementing or decrementing from the new value. In this case, if [TIMG_TO_ALARM_EN](#) is set, the timer will still trigger alarms in scenarios listed in Table 10.4-1 and 10.4-2. If [TIMG_TO_EN](#) is cleared, the timer will remain frozen at the new value until counting is re-enabled.

An auto-reload at alarm will cause a timer reload when an alarm occurs, thus allowing the timer to continue incrementing or decrementing from the reload value. This is generally useful for resetting the timer's value when using periodic alarms. To enable auto-reload at alarm, the [TIMG_TO_AUTORELOAD](#) field should be set. If not enabled, the timer's value will continue to increment or decrement past the alarm value after an alarm.

10.4.5 Frequency Calculation of Slow Clock for Timer Group 0

Using XTAL_CLK as a reference, it is possible to calculate the frequency of slow clock sources provided by ESP32-C5. For the slow clock inputted to the timer group, please refer to [3 Interrupt Matrix \[to be added later\]](#) > [3.1 Interrupt Terminology](#). The calculation method is as follows:

1. Start periodic or one-shot frequency calculation (see Section [10.7.5](#) for details);
2. Once receiving the signal to start calculation, the counter of XTAL_CLK and the counter of slow clock begin to work at the same time. When the counter of slow clock counts to CO, the two counters stop counting simultaneously;
3. Assume the value of XTAL_CLK's counter is C1, and the frequency of slow clock would be calculated as:

$$f_{rtc} = \frac{CO \times f_{XTAL\_CLK}}{C1}$$

Please note that the input frequency should not be larger than $\frac{f_{XTAL\_CLK}}{2}$ when calculating the frequency of slow clock.

10.5 Event Task Matrix Feature

The timer groups on ESP32-C5 support the Event Task Matrix (ETM) function, which allows timer groups' ETM tasks to be triggered by any peripherals' ETM events, or timer groups' ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to timer groups. For more information, please refer to Chapter [8 Event Task Matrix \(ETM\)](#).

The timer groups can receive the following ETM tasks:

- TGO_TASK_CNT_START_TIMER0: When triggered, it will enable the time-base counter.
- TG1_TASK_CNT_START_TIMER0: When triggered, it will enable the time-base counter.
- TGO_TASK_CNT_STOP_TIMER0: When triggered, it will disable the time-base counter.
- TG1_TASK_CNT_STOP_TIMER0: When triggered, it will disable the time-base counter.

Note:

The above two ETM tasks have the same function as the APB configuration [TIMG_TO_EN](#). When these operations occur at the same time, the priority of each operation from high to low is as follows:

1. TGO_TASK_CNT_START_TIMER0 and TG1_TASK_CNT_START_TIMER0: When triggered, it will enable the time-base counter;
2. TGO_TASK_CNT_STOP_TIMER0 and TG1_TASK_CNT_STOP_TIMER0: When triggered, it will disable the time-base counter;
3. APB configuration [TIMG_TO_EN](#): When triggered, it will enable or disable the time-base counter.

- TGO_TASK_ALARM_START_TIMER0: When triggered, it will enable the alarm generation.
- TG1_TASK_ALARM_START_TIMER0: When triggered, it will enable the alarm generation.

Note:

Alarm generation can also be configured through APB method and hardware events. When these operations

occur at the same time, the priority of each operation from high to low is as follows:

1. TGO_TASK_ALARM_START_TIMER0 and TG1_TASK_ALARM_START_TIMER0: When triggered, it will enable the alarm generation;
 2. Alarm events: When triggered, it will disable the alarm generation;
 3. APB configuration TIMG_ALARM_EN: When triggered, it will enable or disable the alarm generation.
- TGO_TASK_CNT_CAP_TIMER0: When triggered, it will update the current counter value to the [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#) registers.
 - TG1_TASK_CNT_CAP_TIMER0: When triggered, it will update the current counter value to the [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#) registers.
 - TGO_TASK_CNT_RELOAD_TIMER0: When triggered, it will overwrite the current counter value with the reload value stored in [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#).
 - TG1_TASK_CNT_RELOAD_TIMER0: When triggered, it will overwrite the current counter value with the reload value stored in [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#).

The timer groups can generate the following ETM events:

- TGO_EVT_CNT_CMP_TIMER0: Indicates the interrupt event of TO in TIMG0.
- TG1_EVT_CNT_CMP_TIMER0: Indicates the interrupt event of TO in TIMG1.

All the ETM tasks and events will not take effect until the [TIMG_ETM_EN](#) is set to 1.

In practical applications, timer groups' ETM events can trigger their own ETM tasks. For example, TGO_TASK_ALARM_START_TIMER0 and TG1_TASK_ALARM_START_TIMER0 can be triggered respectively by TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0 to realize periodic alarm. For configuration steps, please refer to [10.7.4 Timer as Periodic Alarm by ETM](#).

10.6 Interrupts

ESP32-C5's TIMG_n can generate the following interrupt signal(s) that will be sent to the [Interrupt Matrix](#).

- TGN_TO_INTR
- TGN_WDT_INTR

There are several internal interrupt sources from TIMG_n that can generate the above interrupt signals. The interrupt sources from TIMG_n are listed with their trigger conditions and the resulted interrupt signals in Table 10.6-1.

Table 10.6-1. TIMG_n's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
TIMG_TO_INT	TO generates an alarm	TGN_TO_INTR
TIMG_WDT_INT	The watchdog timer generates an alarm	TGN_WDT_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt](#)

Matrix [to be added later] > Section 3.1 Interrupt Terminology.

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [10.8 Register Summary](#).

10.7 Programming Procedures

10.7.1 Timer as a Simple Clock

1. Configure the time-base counter
 - Select clock source by setting or clearing [PCR_TGO_TIMER_CLK_SEL](#) field.
 - Configure the 16-bit prescaler by setting [TIMG_TO_DIVIDER](#).
 - Configure the timer direction by setting or clearing [TIMG_TO_INCREASE](#).
 - Set the timer's starting value by writing the starting value to [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#), then reloading it into the timer by writing any value to [TIMG_TOLOAD_REG](#).
2. Start the timer by setting [TIMG_TO_EN](#).
3. Get the timer's current value.
 - Write any value to [TIMG_TOUUPDATE_REG](#) to latch the timer's current value.
 - Wait until [TIMG_TOUUPDATE_REG](#) is cleared by hardware.
 - Read the latched timer value from [TIMG_TOLO_REG](#) and [TIMG_TOHI_REG](#).

10.7.2 Timer as One-shot Alarm

1. Configure the time-base counter following step 1 of Section [10.7.1](#).
2. Configure the alarm.
 - Configure the alarm value by setting [TIMG_TOALARMLO_REG](#) and [TIMG_TOALARMHI_REG](#).
 - Enable interrupt by setting [TIMG_TO_INT_ENA](#).
3. Disable auto reload by clearing [TIMG_TO_AUTORELOAD](#).
4. Start the alarm by setting [TIMG_TO_ALARM_EN](#).
5. Handle the alarm interrupt.
 - Clear the interrupt by setting the timer's corresponding bit in [TIMG_TO_INT_CLR](#).
 - Disable the timer by clearing [TIMG_TO_EN](#).

10.7.3 Timer as Periodic Alarm by APB

1. Configure the time-base counter following step 1 in Section [10.7.1](#).
2. Configure the alarm following step 2 in Section [10.7.2](#).

3. Enable auto reload by setting [TIMG_TO_AUTORELOAD](#) and configure the reload value via [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#).
4. Start the alarm by setting [TIMG_TO_ALARM_EN](#).
5. Handle the alarm interrupt (repeat on each alarm iteration).
 - Clear the interrupt by setting the timer's corresponding bit in [TIMG_TO_INT_CLR](#).
 - If the next alarm requires a new alarm value and reload value (i.e., different alarm interval per iteration), then [TIMG_TO_ALARMLO_REG](#), [TIMG_TO_ALARMHI_REG](#), [TIMG_TO_LOAD_LO](#), and [TIMG_TO_LOAD_HI](#) should be reconfigured as needed. Otherwise, the aforementioned registers should remain unchanged.
 - Re-enable the alarm by setting [TIMG_TO_ALARM_EN](#).
6. Stop the timer (on final alarm iteration).
 - Clear the interrupt by setting the timer's corresponding bit in [TIMG_TO_INT_CLR](#).
 - Disable the timer by clearing [TIMG_TO_EN](#).

10.7.4 Timer as Periodic Alarm by ETM

1. Enable the ETM module's clock
2. Map ETM event to ETM task (which means using the event to trigger the task)
 - If [TIMG_TO_AUTORELOAD](#) is set to 1, map TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0 to the TGO_TASK_ALARM_START_TIMER0 and TG1_TASK_ALARM_START_TIMER0 respectively by one ETM channel.
 - If [TIMG_TO_AUTORELOAD](#) is set to 0, in addition to mapping TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0 to the TGO_TASK_ALARM_START_TIMER0 and TG1_TASK_ALARM_START_TIMER0, the TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0 should also be mapped to TGO_TASK_CNT_RELOAD_TIMER0 and TG1_TASK_CNT_RELOAD_TIMER0 by another ETM channel.
3. Choose to enable the one or two ETM channels.
4. Set [TIMER_ETM_EN](#) to 1 to enable timer group's ETM events and tasks.
5. Configure the time-base counter following step 1 in Section [10.7.1](#).
6. Configure the alarm following step 2 in Section [10.7.2](#).
7. Configure the reload value via [TIMG_TO_LOAD_LO](#) and [TIMG_TO_LOAD_HI](#).
8. Handle the TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0.
 - When alarm generates, the TGO_EVT_CNT_CMP_TIMER0 and TG1_EVT_CNT_CMP_TIMER0 also generate, and the alarm generation will be disabled by the alarm.
 - If [TIMG_TO_AUTORELOAD](#) is 1, the current counter value is overwritten by the reloaded value. The alarm generation will be reopened by TGO_TASK_ALARM_START_TIMER0 and TG1_TASK_ALARM_START_TIMER0.

- If [TIMG_TO_AUTORELOAD](#) is 0, the current counter value is overwritten by the reloaded value because of the [TGO_TASK_CNT_RELOAD_TIMER0](#) and [TG1_TASK_CNT_RELOAD_TIMER0](#). The alarm generation will be reopened by [TGO_TASK_ALARM_START_TIMER0](#) and [TG1_TASK_ALARM_START_TIMER0](#).

9. Stop the timer (on final alarm iteration).

- Disable the ETM channels used to map timer group's event and task
- Set [TIMER_ETM_EN](#) to 0.
- Clear the interrupt by setting the timer's corresponding bit in [TIMG_TO_INT_CLR](#).
- Disable the timer by clearing [TIMG_TO_EN](#).

10.7.5 Frequency Calculation of Slow Clock

1. One-shot frequency calculation

- Select the clock whose frequency is to be calculated via [PCR_32_SEL](#). For the relationship between the value of this register and the clock source, please refer to Chapter [12 RTC Timer](#).
- To meet the prerequisites for calculating the slow clock frequency, prescale [RC_FAST_CLK](#) via [PCR_FOSC_TICK_NUM](#).
- Configure the time of calculation via [TIMG_RTC_CALI_MAX](#).
- Select one-shot frequency calculation by clearing [TIMG_RTC_CALI_START_CYCLING](#), and enable the two counters via [TIMG_RTC_CALI_START](#).
- Once [TIMG_RTC_CALI_RDY](#) becomes 1, read [TIMG_RTC_CALI_VALUE](#) to get the value of [XTAL_CLK](#)'s counter, and calculate the frequency of slow clock according to the formula in Section [10.4.5](#).

2. Periodic frequency calculation

- Select the clock whose frequency is to be calculated via [PCR_32_SEL](#).
- To meet the prerequisites for calculating the slow clock frequency, prescale [RC_FAST_CLK](#) via [PCR_FOSC_TICK_NUM](#).
- Configure the time of calculation via [TIMG_RTC_CALI_MAX](#).
- Select periodic frequency calculation by enabling [TIMG_RTC_CALI_START_CYCLING](#).
- When [TIMG_RTC_CALI_CYCLING_DATA_VLD](#) is 1, [TIMG_RTC_CALI_VALUE](#) is valid.

3. Timeout

If the counter of slow clock cannot finish counting in [TIMG_RTC_CALI_TIMEOUT_RST_CNT](#) cycles, [TIMG_RTC_CALI_TIMEOUT](#) will be set to indicate a timeout.

10.8 Register Summary

The addresses in this section are relative to **Timer Group** base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
TO Control and configuration registers			
TIMG_TOCONFIG_REG	Timer 0 configuration register	0x0000	varies
TIMG_TOLO_REG	Timer 0 current value, low 32 bits	0x0004	RO
TIMG_TOHI_REG	Timer 0 current value, high 22 bits	0x0008	RO
TIMG_TOUUPDATE_REG	Write to copy current timer value to TIMG_TOLO_REG or TIMG_TOHI_REG	0x000C	R/W/SC
TIMG_TOALARMLO_REG	Timer 0 alarm value, low 32 bits	0x0010	R/W
TIMG_TOALARMHI_REG	Timer 0 alarm value, high 22 bits	0x0014	R/W
TIMG_TOLOADLO_REG	Timer 0 reload value, low 32 bits	0x0018	R/W
TIMG_TOLOADHI_REG	Timer 0 reload value, high 22 bits	0x001C	R/W
TIMG_TOLOAD_REG	Write to reload timer from TIMG_TOLOADLO_REG or TIMG_TOLOADHI_REG	0x0020	WT
WDT Control and configuration registers			
TIMG_WDTCONFIG0_REG	Watchdog timer configuration register	0x0048	varies
TIMG_WDTCONFIG1_REG	Watchdog timer prescaler register	0x004C	varies
TIMG_WDTCONFIG2_REG	Watchdog timer stage 0 timeout value	0x0050	R/W
TIMG_WDTCONFIG3_REG	Watchdog timer stage 1 timeout value	0x0054	R/W
TIMG_WDTCONFIG4_REG	Watchdog timer stage 2 timeout value	0x0058	R/W
TIMG_WDTCONFIG5_REG	Watchdog timer stage 3 timeout value	0x005C	R/W
TIMG_WDTFEED_REG	Write to feed the watchdog timer	0x0060	WT
TIMG_WDTWPROTECT_REG	Watchdog write protect register	0x0064	R/W
RTC CALI Control and configuration registers			
TIMG_RTCCALICFG_REG	RTC calibration configure register	0x0068	varies
TIMG_RTCCALICFG1_REG	RTC calibration configure register 1	0x006C	RO
TIMG_RTCCALICFG2_REG	RTC calibration configure register 2	0x0080	varies
Interrupt registers			
TIMG_INT_ENA_TIMERS_REG	Interrupt enable bits	0x0070	R/W
TIMG_INT_RAW_TIMERS_REG	Raw interrupt status	0x0074	R/SS/WTC
TIMG_INT_ST_TIMERS_REG	Masked interrupt status	0x0078	RO
TIMG_INT_CLR_TIMERS_REG	Interrupt clear bits	0x007C	WT
Version register			
TIMG_NTIMERS_DATE_REG	Timer version control register	0x00F8	R/W
Clock configuration registers			
TIMG_REGCLK_REG	Timer group clock gate register	0x00FC	R/W

10.9 Registers

The addresses in this section are relative to **Timer Group** base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 10.1. TIMG_TOCONFIG_REG (0x0000)

TIMG_TO_EN TIMG_TO_INCREASE TIMG_TO_AUTORELOAD				TIMG_TO_DIVIDER								TIMG_TO_DIVCNT_RST (reserved) TIMG_TO_ALARM_EN				(reserved)							
31	30	29	28	13								12	11	10	9	0							
0	1	1		0x01								0	0	0	0	0	0	0	0	0	0	0	0

Reset

TIMG_TO_ALARM_EN Configures whether or not to enable the timer 0 alarm function. This bit will be automatically cleared once an alarm occurs.

0: Disable

1: Enable

(R/W/SC)

TIMG_TO_DIVCNT_RST Configures whether or not to reset the timer 0 's clock divider counter.

0: No effect

1: Reset

(WT)

TIMG_TO_DIVIDER Represents the timer 0 clock (TO_clk) prescaler value. (R/W)

TIMG_TO_AUTORELOAD Configures whether or not to enable the timer 0 auto-reload function at the time of alarm.

0: No effect

1: Enable

(R/W)

TIMG_TO_INCREASE Configures the counting direction of the timer 0 time-base counter.

0: Decrement

1: Increment

(R/W)

TIMG_TO_EN Configures whether or not to enable the timer 0 time-base counter.

0: Disable

1: Enable

(R/W/SS/SC)

Register 10.2. TIMG_TOLO_REG (0x0004)

TIMG_TO_LO																															
31																															0
0x000000																															
																															Reset

TIMG_TO_LO Represents the low 32 bits of the time-base counter of timer 0. Valid only after writing to TIMG_TOUUPDATE_REG.

Measurement unit: TO_clk
(RO)

Register 10.3. TIMG_TOHI_REG (0x0008)

(reserved)											TIMG_TO_HI																																																
31											22											21																				0																	
0 0 0 0 0 0 0 0 0 0 0											0x0000																																							Reset									

TIMG_TO_HI Represents the high 22 bits of the time-base counter of timer 0. Valid only after writing to TIMG_TOUUPDATE_REG.

Measurement unit: TO_clk
(RO)

Register 10.4. TIMG_TOUUPDATE_REG (0x000C)

TIMG_TO_UPDATE																																	(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31	30																															0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_TO_UPDATE Configures to latch the counter value.

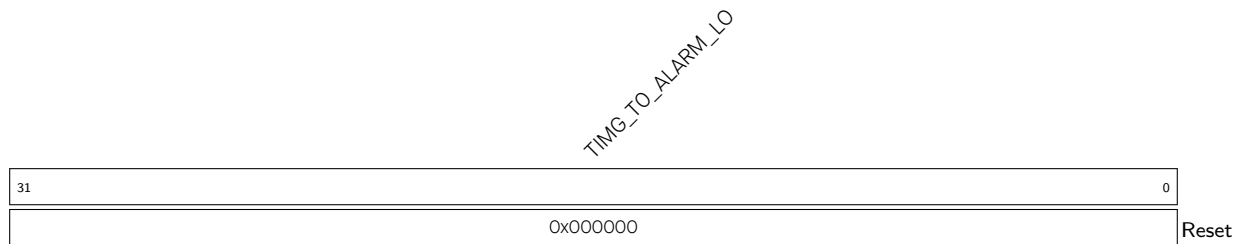
0: Latch

1: Latch

Note that writing either 0 or 1 to this bit will trigger latching of the counter value.

(R/W/SC)

Register 10.5. TIMG_TOALARMLO_REG (0x0010)



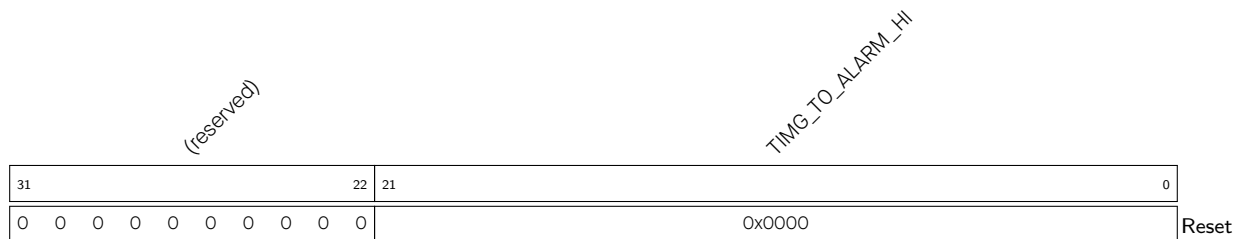
TIMG_TO_ALARM_LO Configures the low 32 bits of timer 0 alarm trigger time-base counter value.

Valid only when TIMG_TO_ALARM_EN is 1.

Measurement unit: T0_clk

(R/W)

Register 10.6. TIMG_TOALARMHI_REG (0x0014)



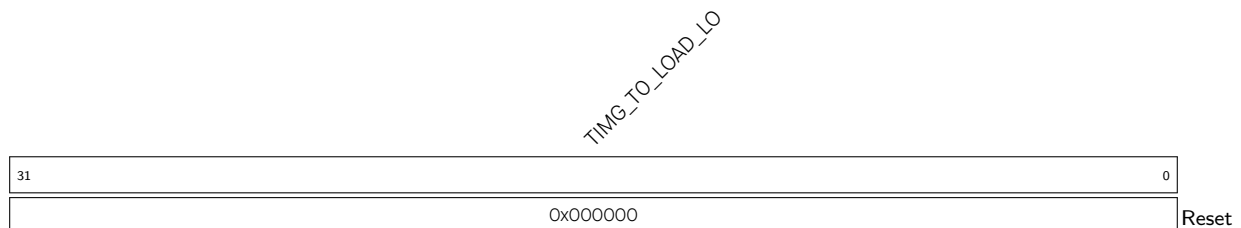
TIMG_TO_ALARM_HI Configures the high 22 bits of timer 0 alarm trigger time-base counter value.

Valid only when TIMG_TO_ALARM_EN is 1.

Measurement unit: TO_clk

(R/W)

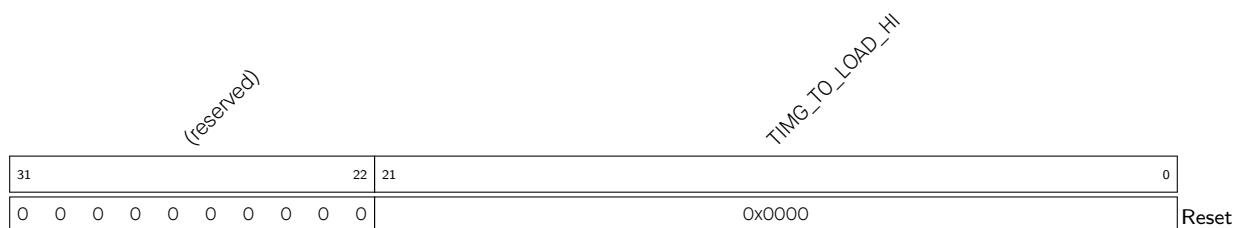
Register 10.7. TIMG_TOLOADLO_REG (0x0018)



TIMG_TO_LOAD_LO Configures low 32 bits of the value that a reload will load onto timer 0 time-base counter.

Measurement unit: TO_clk

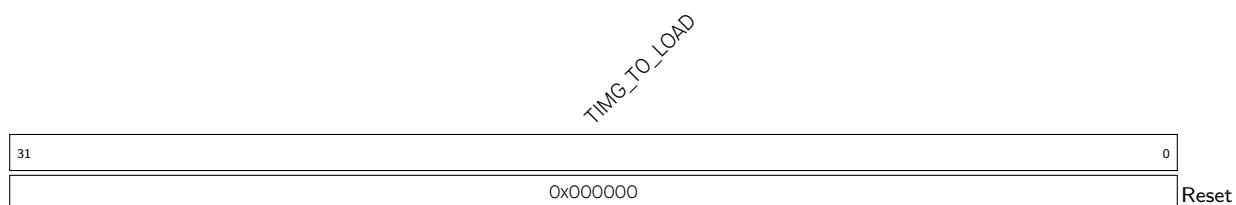
(R/W)

Register 10.8. TIMG_TOLOADHI_REG (0x001C)

TIMG_TO_LOAD_HI Configures high 22 bits of the value that a reload will load onto timer 0 time-base counter.

Measurement unit: TO_clk

(R/W)

Register 10.9. TIMG_TOLOAD_REG (0x0020)

TIMG_TO_LOAD Write any value to trigger a timer 0 time-base counter reload. (WT)

Register 10.10. TIMG_WDTCONFIG0_REG (0x0048)

TIMG_WDT_EN		TIMG_WDT_STG0		TIMG_WDT_STG1		TIMG_WDT_STG2		TIMG_WDT_STG3		(reserved)		TIMG_WDT_CONF_UPDATE_EN		TIMG_WDT_CPU_RESET_LENGTH		TIMG_WDT_SYS_RESET_LENGTH		TIMG_WDT_FLASHBOOT_MOD_EN		TIMG_WDT_PROCPU_RESET_EN		(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
31	30	29	28	27	26	25	24	23	22	21	20	18	17	15	14	13	12	11														0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
0	0	0	0	0	0	0	0	0	0	0	0	Ox1	Ox1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_WDT_PROCPU_RESET_EN Configures whether to mask the CPU reset generated by MWDT.

Valid only when write protection is disabled.

0: Mask

1: Unmask

(R/W)

TIMG_WDT_FLASHBOOT_MOD_EN Configures whether to enable flash boot protection.

0: Disable

1: Enable

(R/W)

TIMG_WDT_SYS_RESET_LENGTH Configures the system reset signal length. Valid only when write protection is disabled.

Measurement unit: mwdt_clk

0: 8

4: 40

1: 16

5: 64

2: 24

6: 128

3: 32

7: 256

(R/W)

TIMG_WDT_CPU_RESET_LENGTH Configures the CPU reset signal length. Valid only when write protection is disabled.

Measurement unit: mwdt_clk

0: 8

4: 40

1: 16

5: 64

2: 24

6: 128

3: 32

7: 256

(R/W)

Continued on the next page...

Register 10.10. USB_SERIAL_JTAG_CONFO_REG (0x0018)

Continued from the previous page...

TIMG_WDT_CONF_UPDATE_EN Configures to update the WDT configuration registers.

0: No effect

1: Update

(WT)

TIMG_WDT_STG3 Configures the timeout action of stage 3. See details in TIMG_WDT_STGO. Valid only when write protection is disabled. (R/W)

TIMG_WDT_STG2 Configures the timeout action of stage 2. See details in TIMG_WDT_STGO. Valid only when write protection is disabled. (R/W)

TIMG_WDT_STG1 Configures the timeout action of stage 1. See details in TIMG_WDT_STGO. Valid only when write protection is disabled. (R/W)

TIMG_WDT_STGO Configures the timeout action of stage 0. Valid only when write protection is disabled.

0: No effect

1: Interrupt

2: Reset CPU

3: Reset system

(R/W)

TIMG_WDT_EN Configures whether or not to enable the MWD. Valid only when write protection is disabled.

0: Disable

1: Enable

(R/W)

Register 10.11. TIMG_WDTCONFIG1_REG (0x004C)

Register structure diagram for TIM16 and TIM17. The register is 32 bits wide, divided into three fields:

- TIM_WDT_CLK_PRESCALE** (bits 31-16): Labeled **0x01**.
- (reserved)** (bits 15-1): Reserved bits.
- TIM_WDT_DIVCNT_RST** (bit 0): Labeled **Reset**.

TIMG_WDT_DIVCNT_RST Configures whether to reset WDT 's clock divider counter.

0: No effect

1: Reset

(WT)

TIMG_WDT_CLK_PRESCALE Configures MWDAT clock prescaler value. Valid only when write protection is disabled.

MWDT clock period = MWDT's clock source period * TIMG_WDT_CLK_PRESCALE.

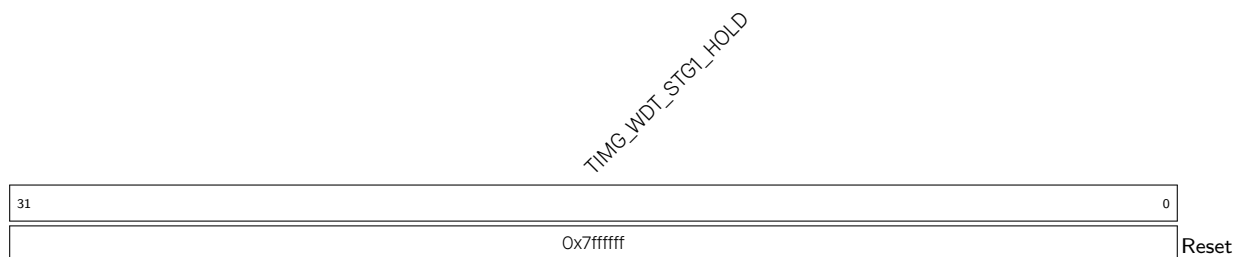
(R/W)

Register 10.12. TIMG_WDTCONFIG2_REG (0x0050)

TIMG_WDT_STGO_HOLD Configures the stage 0 timeout value. Valid only when write protection is disabled.

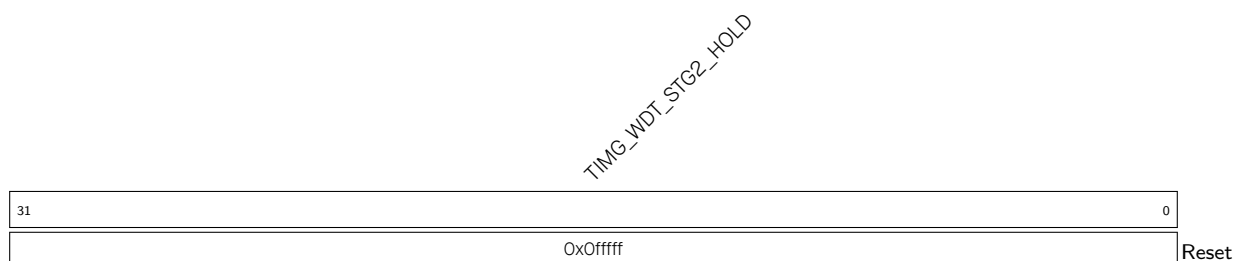
Measurement unit: mwdt_clk

(R/W)

Register 10.13. TIMG_WDTCONFIG3_REG (0x0054)

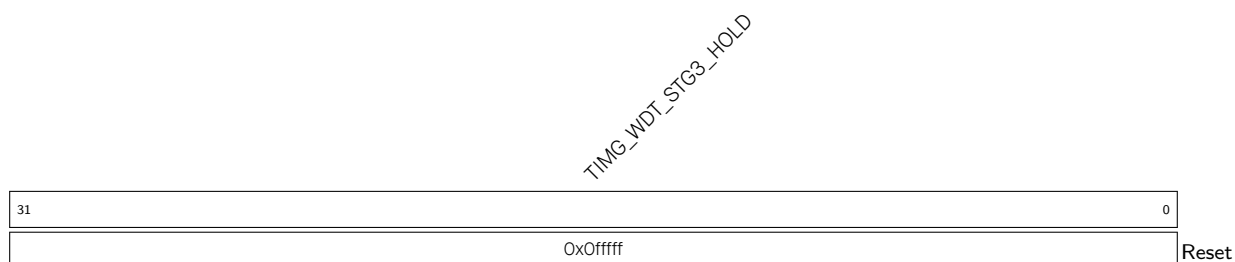
TIMG_WDT_STG1_HOLD Configures the stage 1 timeout value. Valid only when write protection is disabled.

Measurement unit: mwdt_clk
(R/W)

Register 10.14. TIMG_WDTCONFIG4_REG (0x0058)

TIMG_WDT_STG2_HOLD Configures the stage 2 timeout value. Valid only when write protection is disabled.

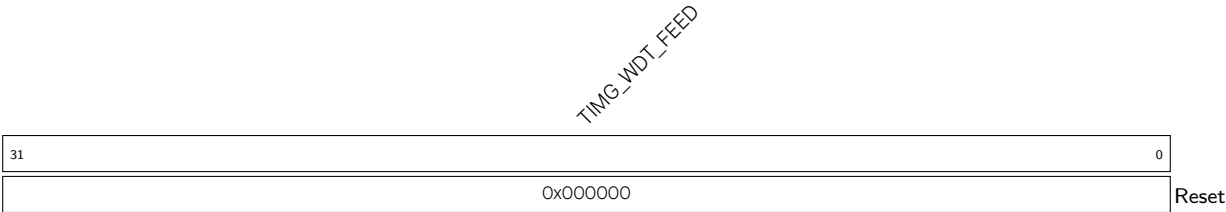
Measurement unit: mwdt_clk
(R/W)

Register 10.15. TIMG_WDTCONFIG5_REG (0x005C)

TIMG_WDT_STG3_HOLD Configures the stage 3 timeout value. Valid only when write protection is disabled.

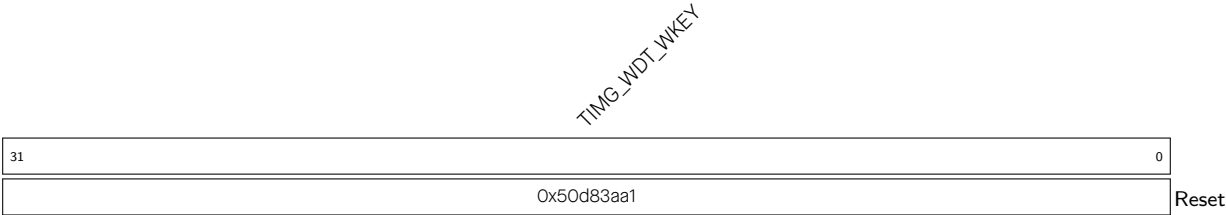
Measurement unit: mwdt_clk
(R/W)

Register 10.16. TIMG_WDTFEED_REG (0x0060)



TIMG_WDT_FEED Write any value to feed the MWDT. Valid only when write protection is disabled. (WT)

Register 10.17. TIMG_WDTWPROTECT_REG (0x0064)



TIMG_WDT_WKEY Configures a different value than its reset value to enable write protection. (R/W)

Register 10.18. TIMG_RTCCALICFG_REG (0x0068)

[illegible]

TIMG_RTC_CALI_START_CYCLING Configures the frequency calculation mode.

0: one-shot frequency calculation

1: periodic frequency calculation

(R/W)

TIMG_RTC_CALI_RDY Represents whether one-shot frequency calculation is done.

0: Not done

1: Done

(RO)

TIMG_RTC_CALI_MAX Configures the time to calculate RTC slow clock's frequency.

Measurement unit: XTAL_CLK

(R/W)

TIMG_RTC_CALI_START Configures whether to enable one-shot frequency calculation.

0: Disable

1: Enable

(R/W)

Register 10.19. TIMG_RTCCALICFG1_REG (0x006C)

TIMG_RTC_CALI_VALUE										(reserved)		TIMG_RTC_CALI_CYCLING_DATA_VLD		
31							7	6	1			0		
0x000000							0	0	0	0	0	0	0	Reset

TIMG_RTC_CALI_CYCLING_DATA_VLD Represents whether periodic frequency calculation is done.

0: Not done

1: Done

(RO)

TIMG_RTC_CALI_VALUE Represents the value countered by XTAL_CLK when one-shot or periodic frequency calculation is done. It is used to calculate RTC slow clock's frequency. (RO)

Register 10.20. TIMG_RTCCALICFG2_REG (0x0080)

TIMG_RTC_CALI_TIMEOUT_THRES										TIMG_RTC_CALI_TIMEOUT_RST_CNT				(reserved)				TIMG_RTC_CALI_TIMEOUT					
31							7	6	3			2	1	0									
0xffffffff									3			0	0	0	0	Reset							

TIMG_RTC_CALI_TIMEOUT Represents whether RTC frequency calculation is timeout.

0: No timeout

1: Timeout

(RO)

TIMG_RTC_CALI_TIMEOUT_RST_CNT Configures the cycles that reset frequency calculation timeout.

Measurement unit: XTAL_CLK

(R/W)

TIMG_RTC_CALI_TIMEOUT_THRES Configures the threshold value for the RTC frequency calculation timer. If the timer's value exceeds this threshold, a timeout is triggered.

Measurement unit: XTAL_CLK

(R/W)

Register 10.21. TIMG_INT_ENA_TIMERS_REG (0x0070)

(reserved)																										TIMG_WDT_INT_ENA (reserved) TIMG_TO_INT_ENA																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																								3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_TO_INT_ENA Write 1 to enable the TIMG_TO_INT interrupt. (R/W)

TIMG_WDT_INT_ENA Write 1 to enable the TIMG_WDT_INT interrupt. (R/W)

Register 10.22. TIMG_INT_RAW_TIMERS_REG (0x0074)

(reserved)																										TIMG_WDT_INT_RAW (reserved) TIMG_TO_INT_RAW																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
31																										3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_TO_INT_RAW The raw interrupt status bit of the TIMG_TO_INT interrupt. (R/SS/WTC)

TIMG_WDT_INT_RAW The raw interrupt status bit of the TIMG_WDT_INT interrupt. (R/SS/WTC)

Register 10.23. TIMG_INT_ST_TIMERS_REG (0x0078)

(reserved)																																TIMG_WDT_INT_ST (reserved) TIMG_TO_INT_ST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_TO_INT_ST The masked interrupt status bit of the TIMG_TO_INT interrupt. (RO)

TIMG_WDT_INT_ST The masked interrupt status bit of the TIMG_WDT_INT interrupt. (RO)

Register 10.24. TIMG_INT_CLR_TIMERS_REG (0x007C)

(reserved)																												TIMG_WDT_INT_CLR (reserved) TIMG_TO_INT_CLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31																												3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMG_TO_INT_CLR Write 1 to clear the TIMG_TO_INT interrupt. (WT)

TIMG_WDT_INT_CLR Write 1 to clear the TIMG_WDT_INT interrupt. (WT)

Register 10.25. TIMG_NTIMERS_DATE_REG (0x00F8)

(reserved)				TIMG_NTIMGS_DATE																																
31				28																												0				
0	0	0	0	0x2209142																												Reset				

TIMG_NTIMGS_DATE Version control register (R/W)

Register 10.26. TIMG_REGCLK_REG (0x00FC)

TIMG_CLK_EN (reserved)				TIMG_ETM_EN (reserved)																												
31	30	29	28	0																												
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

TIMG_ETM_EN Configures whether to enable timer's ETM task and event.

0: Disable

1: Enable

(R/W)

TIMG_CLK_EN Configures whether to enable gate clock signal for registers.

0: Force clock on for registers

1: Support clock only when registers are read or written to by software.

(R/W)

Chapter 11

Watchdog Timers (WDT)

11.1 Overview

Watchdog timers are hardware timers used to detect and recover from malfunctions. They must be periodically fed (reset) to prevent a timeout. A system/software that is behaving unexpectedly (e.g. is stuck in a software loop or in overdue events) will fail to feed the watchdog and thus trigger a watchdog timeout. Therefore, watchdog timers are useful for detecting and handling erroneous system/software behaviors.

As shown in Figure 11.1-1, ESP32-C5 contains three digital watchdog timers: one in each of the two timer groups in Chapter 10 *Timer Group (TIMG)* (called Main System Watchdog Timers, or MWDT) and one in the RTC Module (called the RTC Watchdog Timer, or RWDT). Each digital watchdog timer allows for four separately configurable stages and each stage can be programmed to take one action upon timeout, unless the watchdog is fed or disabled. MWDT supports three timeout actions: interrupt, CPU reset, and core reset, while RWDT supports four timeout actions: interrupt, CPU reset, core reset, and system reset (see details in Section 11.2.2.2 *Stages and Timeout Actions*). A timeout value can be set for each stage individually.

During the flash boot process, RWDT and the MWDT in timer group 0 are enabled automatically in order to detect and recover from booting errors.

ESP32-C5 also has one analog watchdog timer: Super watchdog (SWD). It is an ultra-low-power circuit in analog domain that helps to prevent the system from operating in a sub-optimal state and resets the system if required.

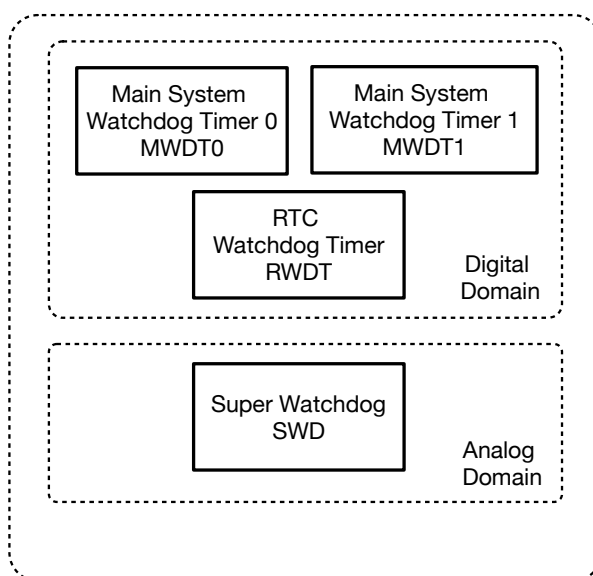


Figure 11.1-1. Watchdog Timers Overview

Note that while this chapter provides the functional descriptions of the watchdog timers, MWDT register descriptions are detailed in Chapter [10 Timer Group \(TIMG\)](#), and the RWDT and SWD register descriptions are detailed in Section [11.5 Register Summary](#).

11.2 Digital Watchdog Timers

11.2.1 Features

Watchdog timers have the following features:

- Four stages, each with a separately programmable timeout value and timeout action
- Timeout actions:
 - MWDT: interrupt, CPU reset, core reset
 - RWDT: interrupt, CPU reset, core reset, system reset
- Flash boot protection at stage 0:
 - MWDT0: core reset upon timeout
 - RWDT: system reset upon timeout
- Write protection that makes WDT register read only unless unlocked
- 32-bit timeout counter
- Clock source:
 - MWDT: PLL_F80M_CLK, RC_FAST_CLK, or XTAL_CLK
 - RWDT: RTC_SLOW_CLK

11.2.2 Functional Description

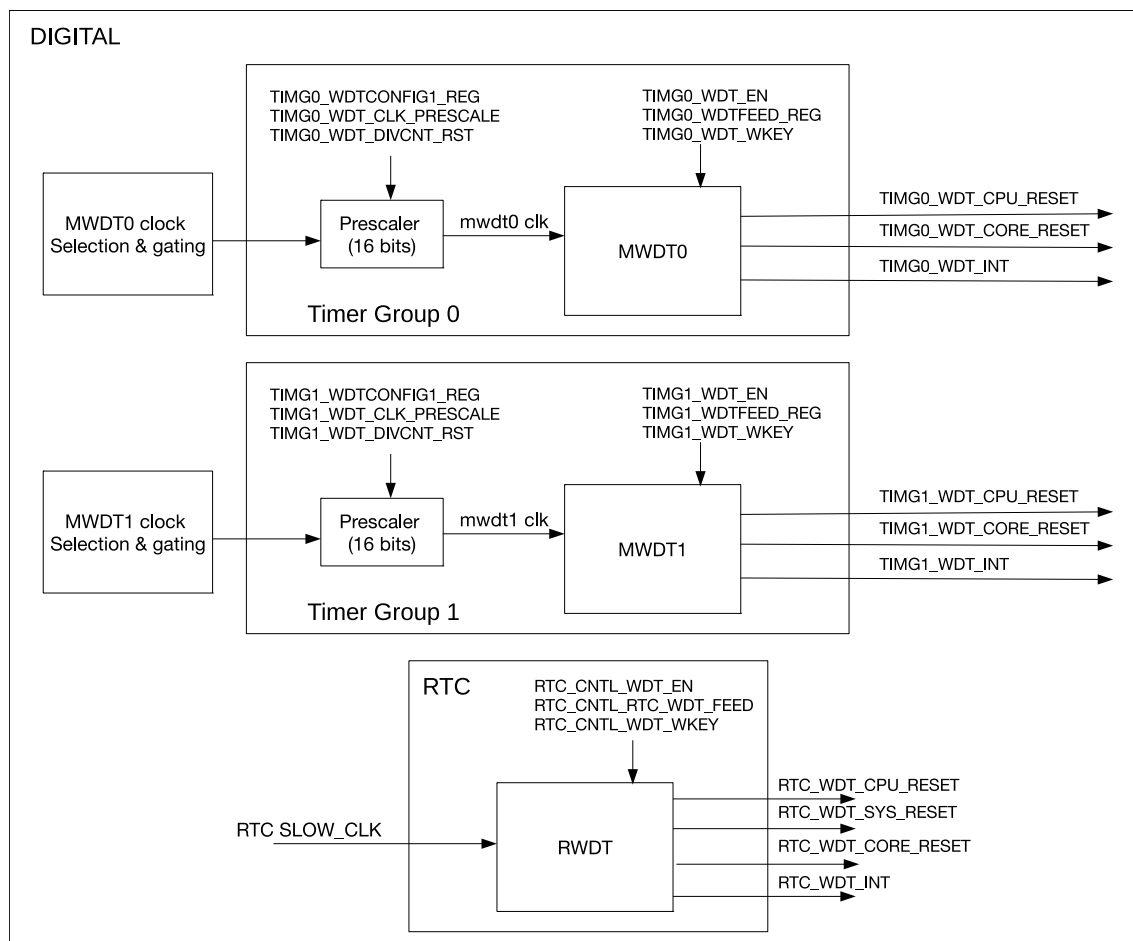


Figure 11.2-1. Digital Watchdog Timers in ESP32-C5

Figure 11.2-1 shows the three watchdog timers in ESP32-C5 digital systems.

11.2.2.1 Clock Source and 32-Bit Counter

At the core of each watchdog timer is a 32-bit counter.

Take MWDTO as an example:

- MWDTO can select between the `PLL_F80M_CLK`, `RC_FAST_CLK`, or `XTAL_CLK` (external) clock as its clock source by setting the `PCR_TGO_WDT_CLK_SEL` field of the `PCR_TIMERGROUPO_WDT_CLK_CONF_REG` register.
- The selected clock is switched on by setting `PCR_TGO_WDT_CLK_EN` field of the `PCR_TIMERGROUPO_WDT_CLK_CONF_REG` register to 1 and switched off by setting it to 0. Then the selected clock is divided by a 16-bit configurable prescaler. See more details in Table 6.2-2 of Chapter 6 Reset and Clock.

The 16-bit prescaler value for MWDT is configured via the `TIMG_WDT_CLK_PRESCALE` field of `TIMG_WDTCONFIG1_REG`. When `TIMG_WDT_DIVCNT_RST` field is set, the prescaler is reset and it can be re-configured at once.

In contrast, the clock source of RWDT is derived directly from RTC_SLOW_CLK (see details in Chapter 6 [Reset and Clock](#)).

MWDT and RWDT are enabled by setting the [TIMG_WDT_EN](#) and [RTC_WDT_EN](#) fields respectively. When enabled, the 32-bit counters of the watchdog will increment on each source clock cycle until the timeout value of the current stage is reached (i.e. timeout of the current stage). When this occurs, the current counter value is reset to zero and the next stage will become active. If a watchdog timer is fed by software, the timer will return to stage 0 and reset its counter value to zero. Software can feed a watchdog timer by writing any value to [TIMG_WDTFEED_REG](#) for MDWT and by writing 1 to [RTC_WDT_FEED](#) for RWDT.

11.2.2.2 Stages and Timeout Actions

Timer stages allow for a timer to have a series of different timeout values and corresponding timeout action. When one stage times out, the timeout action is triggered, the counter value is reset to zero, and the next stage becomes active.

MWDT/RWDT provide four stages (called stages 0 to 3). The watchdog timers will progress through each stage in a loop (i.e. from stage 0 to 3, then back to stage 0).

Timeout values of each stage for MWDT are configured in [TIMG_WDTCONFIG_i_REG](#) (where *i* ranges from 2 to 5), whilst timeout values for RWDT are configured using [RTC_WDT_STG_j_HOLD](#) field (where *j* ranges from 0 to 3).

Please note that the timeout value of stage 0 for RWDT (T_{hold0}) is determined by the combination of the

[EFUSE_WDT_DELAY_SEL](#) field of the eFuse register [EFUSE_RD_REPEAT_DATA0_REG](#) and the [RTC_WDT_STG0_HOLD](#) field. The relationship is as follows:

$$T_{hold0} = \text{RTC\_WDT\_STG0\_HOLD} \ll (\text{EFUSE\_WDT\_DELAY\_SEL} + 1)$$

where $\ll$ is a left-shift operator. For example, if [RTC_WDT_STG0_HOLD](#) is configured as 100 and [EFUSE_WDT_DELAY_SEL](#) is 1, the T_{hold0} will be 400 cycles.

Upon the timeout of each stage, one of the following timeout actions will be executed:

Table 11.2-1. Timeout Actions

Timeout Action	Description
Interrupt	Trigger an interrupt
CPU reset	See Chapter 6 Reset and Clock > Section 6.1.3 Features
Core reset	
System reset	
Disabled	No effect on the system

For MWDT, the timeout action of all stages is configured in [TIMG_WDTCONFIG0_REG](#). Likewise for RWDT, the timeout action is configured in [RTC_WDT_CONFIG0_REG](#).

11.2.2.3 Write Protection

Watchdog timers are critical to detecting and handling erroneous system/software behavior, and thus should not be disabled easily (e.g. due to a misplaced register write). Therefore, MWDT and RWDT incorporate a write protection mechanism that prevents the watchdogs from being disabled or tampered with due to an accidental write.

The write protection mechanism is implemented using a write-key field for each timer ([TIMG_WDT_WKEY](#) for MWDT, [RTC_WDT_WKEY](#) for RWDT). The value 0x50D83AA1 must be written to the watchdog timer's write-key field before any other register of the same watchdog timer can be changed. Any attempts to write to a watchdog timer's registers (other than the write-key field itself) whilst the write-key field's value is not 0x50D83AA1 will be ignored. The recommended procedure for accessing a watchdog timer is as follows:

1. Disable the write protection by writing the value 0x50D83AA1 to the timer's write-key field.
2. Make the required modification of the watchdog such as feeding or changing its configuration.
3. Re-enable write protection by writing any value other than 0x50D83AA1 to the timer's write-key field.

11.2.2.4 Flash Boot Protection

During flash booting process, MWDT0 as well as RWDT, are automatically enabled. Stage 0 for the enabled MWDT0 is automatically configured as core reset action upon timeout, known as core reset. Likewise, stage 0 for RWDT is configured to system reset, which resets the main system and RTC when it times out. After booting, [TIMG_WDT_FLASHBOOT_MOD_EN](#) and [RTC_WDT_FLASHBOOT_MOD_EN](#) should be cleared to stop the flash boot protection procedure for both MWDT0 and RWDT respectively. After this, MWDT0 and RWDT can be configured by software.

11.3 Super Watchdog

Super watchdog (SWD) is an ultra-low-power circuit in analog domain that helps to prevent the system from operating in a sub-optimal state and resets the system (system reset) if required. SWD contains a watchdog circuit that needs to be fed for at least once during its timeout period, which is slightly less than one second. About 100 ms before watchdog timeout, it will also send out a WD_INTR signal as a request to remind the system to feed the watchdog.

If the system doesn't respond to SWD feed request and watchdog finally times out, SWD will generate a system level signal SWD_RSTB to reset whole digital circuits on the chip (system reset) .

The source of the clock for SWD is constant and can not be selected.

11.3.1 Features

SWD has the following features:

- An analog watchdog that operates independently of the digital circuit and can function in environments where the digital circuit's clock and voltage are abnormal
- Interrupt to indicate that the SWD is about to time out

- Various dedicated methods for software to feed SWD, which enables SWD to monitor the working state of the whole operating system

11.3.2 Super Watchdog Controller

11.3.2.1 Structure

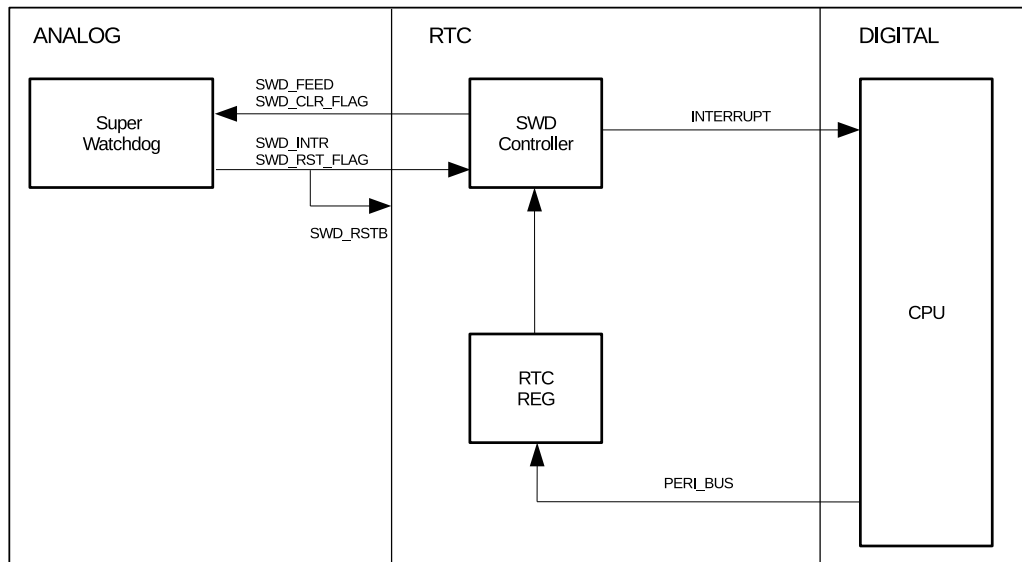


Figure 11.3-1. Super Watchdog Controller Structure

11.3.2.2 Workflow

In normal state:

- SWD controller receives feed request from SWD.
- SWD controller sends an interrupt to HP CPU.
- HP CPU feeds SWD directly by setting `RTC_WDT_SWD_FEED`.
- When trying to feed SWD, CPU needs to disable SWD controller's write protection by writing `0x50D83AA1` to `RTC_WDT_SWD_WKEY`. This prevents SWD from being fed by mistake when the system is operating in sub-optimal state.
- If setting `RTC_WDT_SWD_AUTO_FEED_EN` to 1, SWD controller can also receive interrupts from the SWD for feeding the watchdog itself without any interaction with CPU.

After reset:

- Check `LP_CLKRST_RESET_CAUSE[4:0]` for the cause of CPU reset.
If `LP_CLKRST_RESET_CAUSE[4:0] == 0x12`, it indicates that the cause is SWD reset.
- Set `RTC_WDT_SWD_RST_FLAG_CLR` to clear the SWD reset flag.

11.4 Interrupts

For watchdog timer interrupts, please refer to Section [10.6 Interrupts](#) in Chapter [10 Timer Group \(TIMG\)](#).

11.5 Register Summary

The addresses in this section are relative to RTC_WDT base address provided in Table [4.3-2](#) in Chapter [4 System and Memory](#).

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
configuration register			
RTC_WDT_CONFIG0_REG	Configure the RWDT operation	0x0000	R/W
RTC_WDT_CONFIG1_REG	Configure the RWDT timeout time of stage0	0x0004	R/W
RTC_WDT_CONFIG2_REG	Configure the RWDT timeout time of stage1	0x0008	R/W
RTC_WDT_CONFIG3_REG	Configure the RWDT timeout time of stage2	0x000C	R/W
RTC_WDT_CONFIG4_REG	Configure the RWDT timeout time of stage3	0x0010	R/W
RTC_WDT_FEED_REG	Configure the feed function of RWDT	0x0014	WT
RTC_WDT_WPROTECT_REG	Configure the lock function of RWDT	0x0018	R/W
RTC_WDT_SWD_CONFIG_REG	Configure the SWD operation	0x001C	varies
RTC_WDT_SWD_WPROTECT_REG	Configure the lock function of SWD	0x0020	R/W
RTC_WDT_INT_RAW_REG	The interrupt raw register of WDT	0x0024	R/WTC/SS
RTC_WDT_INT_ST_REG	The interrupt status register of WDT	0x0028	RO
RTC_WDT_INT_ENA_REG	The interrupt enable register of WDT	0x002C	R/W
RTC_WDT_INT_CLR_REG	The interrupt clear register of WDT	0x0030	WT
RTC_WDT_DATE_REG	Version control register	0x03FC	R/W

11.6 Registers

MWDT registers are part of the timer submodule and are described in Section [10.8 Register Summary](#) in Chapter [10 Timer Group \(TIMG\)](#).

The addresses of RWDT and SWD registers in this section are relative to RTC_WDT base address provided in Table [4.3-2](#) in Chapter [4 System and Memory](#).

Register 11.1. RTC_WDT_CONFIG0_REG (0x0000)

RTC_WDT_EN		RTC_WDT_STG0		RTC_WDT_STG1		RTC_WDT_STG2		RTC_WDT_STG3		RTC_WDT_CPU_RESET_LENGTH		RTC_WDT_SYS_RESET_LENGTH		RTC_WDT_FLASHBOOT_MOD_EN		RTC_WDT_PROCPU_RESET_EN		RTC_WDT_PAUSE_IN_SLP		(reserved)									
31	30	28	27	25	24	22	21	19	18	16	15	13	12	11	10	9	8	0											
0	0x0		0x0		0x0		0x0		0x1		0x1		1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RTC_WDT_PAUSE_IN_SLP Configures whether or not to pause RWDT when chip is in Light-sleep or Deep-sleep mode.

0: Enable

1: Disable

(R/W)

RTC_WDT_PROCPU_RESET_EN Configures whether or not to enable RWDT to reset CPU.

0: Disable

1: Enable

(R/W)

RTC_WDT_FLASHBOOT_MOD_EN Configures whether or not to enable RWDT when chip is in SPI boot mode.

0: Disable

1: Enable

(R/W)

RTC_WDT_SYS_RESET_LENGTH Configures the core reset time.

Measurement unit: LP_DYN_FAST_CLK cycles

(R/W)

RTC_WDT_CPU_RESET_LENGTH Configures the CPU reset time.

Measurement unit: LP_DYN_FAST_CLK cycles

(R/W)

RTC_WDT_STG3 Configures the timeout action of stage3.

0: No operation

1: Generate interrupt

2: Generate CPU reset

3: Generate core reset

4: Generate system reset

(R/W)

Continued on the next page...

Register 11.1. RTC_WDT_CONFIG0_REG (0x0000)

Continued from the previous page...

RTC_WDT_STG2 Configures the timeout action of stage2.

- 0: No operation
 - 1: Generate interrupt
 - 2: Generate CPU reset
 - 3: Generate core reset
 - 4: Generate system reset
- (R/W)

RTC_WDT_STG1 Configures the timeout action of stage1.

- 0: No operation
 - 1: Generate interrupt
 - 2: Generate CPU reset
 - 3: Generate core reset
 - 4: Generate system reset
- (R/W)

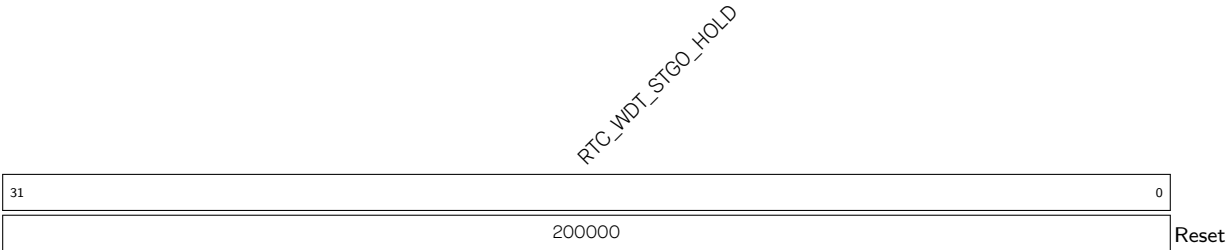
RTC_WDT_STG0 Configures the timeout action of stage0.

- 0: No operation
 - 1: Generate interrupt
 - 2: Generate CPU reset
 - 3: Generate core reset
 - 4: Generate system reset
- (R/W)

RTC_WDT_EN Configures whether or not enable RWDT.

- 0: Disable RWDT
 - 1: Enable RWDT
- (R/W)

Register 11.2. RTC_WDT_CONFIG1_REG (0x0004)



RTC_WDT_STGO_HOLD Configures the timeout time for stage0.

- Measurement unit: LP_DYN_SLOW_CLK cycles
- (R/W)

Register 11.3. RTC_WDT_CONFIG2_REG (0x0008)*RTC_WDT_STG1_HOLD*

31	0
80000	
Reset	

RTC_WDT_STG1_HOLD Configures the timeout time for stage1.
 Measurement unit: LP_DYN_SLOW_CLK cycles
 (R/W)

Register 11.4. RTC_WDT_CONFIG3_REG (0x000C)*RTC_WDT_STG2_HOLD*

31	0
0x000fff	
Reset	

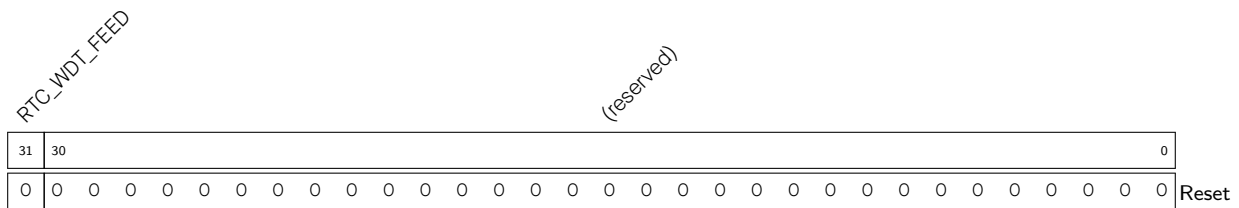
RTC_WDT_STG2_HOLD Configures the timeout time for stage2.
 Measurement unit: LP_DYN_SLOW_CLK cycles
 (R/W)

Register 11.5. RTC_WDT_CONFIG4_REG (0x0010)*RTC_WDT_STG3_HOLD*

31	0
0x000fff	
Reset	

RTC_WDT_STG3_HOLD Configures the timeout time for stage3.
 Measurement unit: LP_DYN_SLOW_CLK cycles
 (R/W)

Register 11.6. RTC_WDT_FEED_REG (0x0014)



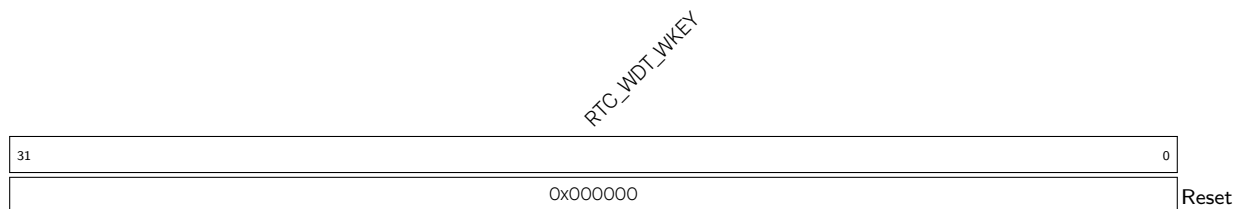
RTC_WDT_FEED Configures this bit to feed the RWDT.

0: Invalid

1: Feed RWDT

(WT)

Register 11.7. RTC_WDT_WPROTECT_REG (0x0018)



RTC_WDT_WKEY Configures this field to lock or unlock RWDT's configuration registers.

0x50D83AA1: unlock the RWDt configuration register

Other values: lock the RWDT configuration register which can't be modified by software.

(R/W)

Register 11.8. RTC_WDT_SWD_CONFIG_REG (0x001C)

RTC_WDT_SWD_FEED		RTC_WDT_SWD_DISABLE		RTC_WDT_SWD_SIGNAL_WIDTH																RTC_WDT_SWD_RST_FLAG_CLR																RTC_WDT_SWD_AUTO_FEED_EN																(reserved)																RTC_WDT_SWD_RESET_FLAG																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31	30	29																	20	19	18	17																	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0	0	300																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RTC_WDT_SWD_RESET_FLAG Represents whether the SWD generates a reset signal or not.

0: No

1: Yes

(RO)

RTC_WDT_SWD_AUTO_FEED_EN Configures this bit to enable to feed SWD automatically by hardware.

0: Disable

1: Enable

(R/W)

RTC_WDT_SWD_RST_FLAG_CLR Configures this bit to clear SWD reset flag.

0: Invalid

1: Clear the reset flag

(WT)

RTC_WDT_SWD_SIGNAL_WIDTH Configure the SWD signal length that output to analog circuit.
Measurement unit: LP_DYN_FAST_CLK (R/W)

RTC_WDT_SWD_DISABLE Configures this bit to disable the SWD.

0: Enable the SWD

1: Disable the SWD

(R/W)

RTC_WDT_SWD_FEED Configures this bit to feed the SWD.

0: Invalid

1: Feed SWD

(WT)

Register 11.9. RTC_WDT_SWD_WPROTECT_REG (0x0020)

Diagram illustrating the 32-bit register structure for `RTC_WDT_SWKEY`. The register is shown as a horizontal bar with bit positions 31 on the left and 0 on the right. The value `0x00000000` is displayed in the center, and the label "Reset" is at the bottom right.

RTC_WDT_SWD_WKEY Configures this field to lock or unlock SWD's configuration registers.

0x50D83AA1: unlock the SWD configuration register.

Other values: lock the SWD configuration register which can't be modified by the software.

(R/W)

Register 11.10. RTC_WDT_INT_RAW_REG (0x0024)

Diagram illustrating the 32-bit register structure:

- Bits 31-30: RTC_WDT_INT_RAW
- Bits 29-0: RTC_WDT_SWD_INT_RAW
- Bit 0: (reserved)

The register is labeled "Reset" at the bottom right.

RTC_WDT_SWD_INT_RAW The raw interrupt status of RTC_WDT_SWD_INT interrupt.(R/WTC/SS)

RTC_WDT_INT_RAW The raw interrupt status of RTC_WDT_INT interrupt. (R/WTC/SS)

Register 11.11. RTC_WDT_INT_ST_REG (0x0028)

Diagram illustrating the structure of the RCR register (32 bits):

- Bits 31-30: RTC_WDT_INT_ST (2 bits)
- Bits 29-0: RTC_WDT_SWD_INT_ST (30 bits)
- Bits 31-0: (reserved) (32 bits)

RTC_WDT_SWD_INT_ST The masked interrupt status of RTC_WDT_SWD_INT interrupt.(RO)

RTC_WDT_INT_ST The masked interrupt status of RTC_WDT_INT interrupt.(RO)

RTC_WDT_INT_ENA
RTC_WDT_SWD_INT_ENA

(reserved)

RTC_WDT_INT_ENA Write 1 to enable RTC_WDT_INT. (R/W)

RTC_WDT_INT_CLR
RTC_WDT_SWD_INT_CLR

(reserved)

RTC_WDT_INT_CLR Write 1 to clear RTC_WDT_INT. (WT)

RTC_WDT_CLK_EN

RTC_WDT_DATE

RTC_WDT_CLK_EN Reserved. (R/W)

Chapter 12

RTC Timer

12.1 Introduction

The RTC Timer is a 48-bit readable counter that can operate in any power mode. It is used as a system timer when the timers in the HP system is unavailable. It also allows for configuring timer interrupts and logging the time when specific events happen in the system.

12.2 Feature List

- 48-bit counter
- Time logging when one of the following events happens:
 - HP system reset
 - CPU enters stall state
 - CPU exits stall state
 - Crystal powers up
 - Crystal powers down
- Time logging through register configuration
- Occurrence time cached of the most recent two specific events
- Generation of interrupts at target times, which are configurable. It is also possible to configure two target times simultaneously
- Uninterrupted operation during any reset or sleep mode, except for power-on reset of LP system
- Can work as the wake-up source (see Chapter [4 Low-power Management \[to be added later\]](#))

12.3 Functional Description

- 48-bit counter
 - The implementation of the RTC Timer is based on a 48-bit counter, driven by the RC_SLOW_CLK in the Always-On power domain (for details about power domains, see Chapter [4 Low-power Management \[to be added later\]](#)). It uses continuous loop counting (except during LP system reset), and an overflow interrupt is generated when the 48-bit counter overflows.
- Log the occurrence time of specific events
 - The RTC Timer supports logging the occurrence time of three types of events:

- * HP system reset
 - * CPU enters or exits stall state
 - * Crystal powers up or down
- The above three types of events have their own independent configuration registers. When the configuration registers are enabled, if the corresponding events occur, the hardware pulse generated will trigger the RTC Timer to log the current time. The corresponding configuration registers are shown in the table below.

Table 12.3-1. Configure RTC Timer to Log the Occurrence Time of Specific Events

Configuration Register	Description
RTC_TIMER_MAIN_TIMER_SYS_RST	Enable the RTC Timer to log the time of system reset.
RTC_TIMER_MAIN_TIMER_SYS_STALL	Enable the RTC Timer to log the time when the CPU enters or exits the stall state.
RTC_TIMER_MAIN_TIMER_XTAL_OFF	Enable the RTC Timer to log the time when the crystal powers up or down.

- Log the current time through software configuration.
 - In addition to the above three specific events that can trigger the RTC Timer to log the time, the RTC Timer can also log the current time through the configuration register [RTC_TIMER_MAIN_TIMER_UPDATE](#).
 - A pulse signal will be generated by configuring the [RTC_TIMER_MAIN_TIMER_UPDATE](#), which will trigger the RTC Timer to log the current time.
- Generate counting interrupts at target times
 - The RTC Timer supports configuring two target times simultaneously, referred to as target time 0 and target time 1. When the timer reaches target time 0, it will trigger the [RTC_TIMER_CMPO_INT](#) interrupt. Similarly, when the timer reaches Target Moment 1, it will trigger the [RTC_TIMER_CMP1_INT](#) interrupt.
 - Configure the target times through the registers below:

Table 12.3-2. Target Time Configuration

Configuration Register	Description
RTC_TIMER_MAIN_TIMER_TAR_ENn	$n = 0$ or 1 , enable target time configuration.
RTC_TIMER_MAIN_TIMER_TAR_LOWn	$n = 0$ or 1 , configure the lower 32 bits of the target time.
RTC_TIMER_MAIN_TIMER_TAR_HIGHn	$n = 0$ or 1 , configure the higher 16 bits of the target time.

- Read the time cached in the RTC Timer
 - There are two sets of registers used to cache the occurrence time of specific events, namely register group 0 and register group 1.
 - Register group 0 includes register [RTC_TIMER_MAIN_BUFO_LOW_REG](#) and [RTC_TIMER_MAIN_BUF1_HIGH_REG](#) which are used to cache the count value of the RTC Timer under the current trigger.

- Register group 1 includes register [RTC_TIMER_MAIN_BUF1_LOW_REG](#) and [RTC_TIMER_MAIN_BUF1_HIGH_REG](#), which are used to cache the count value of the RTC Timer from the last trigger.
- On a new trigger, the record from the previous trigger will be moved from register group 0 to register group 1 (and the previous record in register group 1 will be overwritten), and the record of this trigger will be stored in register group 0. Therefore, only the last two triggers can be logged at any time.

12.4 Event Task Matrix Feature

The RTC Timer on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows RTC Timer's ETM events to trigger any peripherals' ETM tasks. The ETM tasks of RTC timer are not supported. This section introduces the ETM events related to RTC Timer. For more information, please refer to Chapter [8 Event Task Matrix \(ETM\)](#).

The RTC Timer can generate the following ETM events:

- [RTC_EVT_CMP](#): Indicates that the RTC Timer reaches the target time 0 configured by [RTC_TIMER_MAIN_TIMER_TAR_LOW0](#) and [RTC_TIMER_MAIN_TIMER_TAR_HIGH0](#).
- [RTC_EVT_OVF](#): Indicates that the 48-bit counter overflows.
- [RTC_EVT_TICK](#): Indicates the event that occurs when the RTC timer increments by 1.

12.5 Interrupts

ESP32-C5's RTC Timer can generate the following interrupt signal(s) that will be sent to the [Interrupt Matrix](#).

- [LP_TIMER_REG_O_INTR](#) (only for HP CPU)
- [LP_TIMER_REG_1_INTR](#) (only for LP CPU)

There are several internal interrupt sources from the RTC Timer that can generate the above interrupt signal(s). The interrupt sources from the RTC Timer are listed with their trigger conditions and the resulted interrupt signal(s) in Table [12.5-1](#).

Table 12.5-1. RTC Timer's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
RTC_TIMER_CMPO_INT	RTC Timer reaches the target time 0	LP_TIMER_REG_O_INTR
RTC_TIMER_CMP1_INT	RTC Timer reaches the target time 1	LP_TIMER_REG_1_INTR
RTC_TIMER_OVERFLOW_INT	48-bit counter overflows	LP_TIMER_REG_O_INTR LP_TIMER_REG_1_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter [3 Interrupt Matrix \[to be added later\]](#) > Section [3.1 Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [12.6 Register Summary](#).

12.6 Register Summary

The addresses in this section are relative to RTC Timer base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

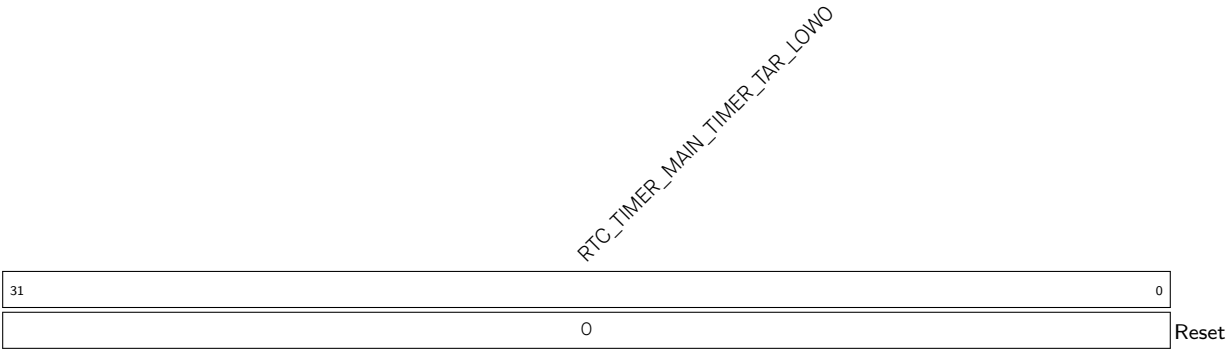
Name	Description	Address	Access
Configuration Registers			
RTC_TIMER_TARO_LOW_REG	Configures the low 32 bits of the target time 0	0x0000	R/W
RTC_TIMER_TARO_HIGH_REG	Configures the high 16 bits of the target time 0	0x0004	varies
RTC_TIMER_TAR1_LOW_REG	Configures the low 32 bits of the target time 1	0x0008	R/W
RTC_TIMER_TAR1_HIGH_REG	Configures the high 16 bits of the target time 1	0x000C	varies
RTC_TIMER_UPDATE_REG	Configures to enable the RTC Timer to latch current value	0x0010	varies
RTC_TIMER_MAIN_BUFO_LOW_REG	The low 32 bits of the cached value 0 in RTC Timer	0x0014	RO
RTC_TIMER_MAIN_BUFO_HIGH_REG	The high 16 bits of the cached value 0 in RTC Timer	0x0018	RO
RTC_TIMER_MAIN_BUF1_LOW_REG	The low 32 bits of the cached value 1 in RTC Timer	0x001C	RO
RTC_TIMER_MAIN_BUF1_HIGH_REG	The high 16 bits of the cached value 1 in RTC Timer	0x0020	RO
Interrupt Registers			
RTC_TIMER_INT_RAW_REG	The interrupt raw status register for the RTC Timer reaching the target time 0	0x0028	R/WTC/SS
RTC_TIMER_INT_ST_REG	The interrupt status register for the RTC Timer reaching the target time 0	0x002C	RO
RTC_TIMER_INT_ENA_REG	The interrupt enable register for the RTC Timer reaching the target time 0	0x0030	R/W
RTC_TIMER_INT_CLR_REG	The interrupt clear register for the RTC Timer reaching the target time 0	0x0034	WT
RTC_TIMER_LP_INT_RAW_REG	The interrupt raw status register for RTC Timer reaching the target time 1	0x0038	R/WTC/SS
RTC_TIMER_LP_INT_ST_REG	The interrupt status register for the RTC Timer reaching the target time 1	0x003C	RO
RTC_TIMER_LP_INT_ENA_REG	The interrupt enable register for the RTC Timer reaching the target time 1	0x0040	R/W
RTC_TIMER_LP_INT_CLR_REG	The interrupt clear register for the RTC Timer reaching the target time 1	0x0044	WT
Version Register			
RTC_TIMER_DATE_REG	Version control register	0x03FC	R/W

12.7 Registers

The addresses in this section are relative to RTC Timer base address provided in Table 4.3-2 in Chapter 4 [System and Memory](#).

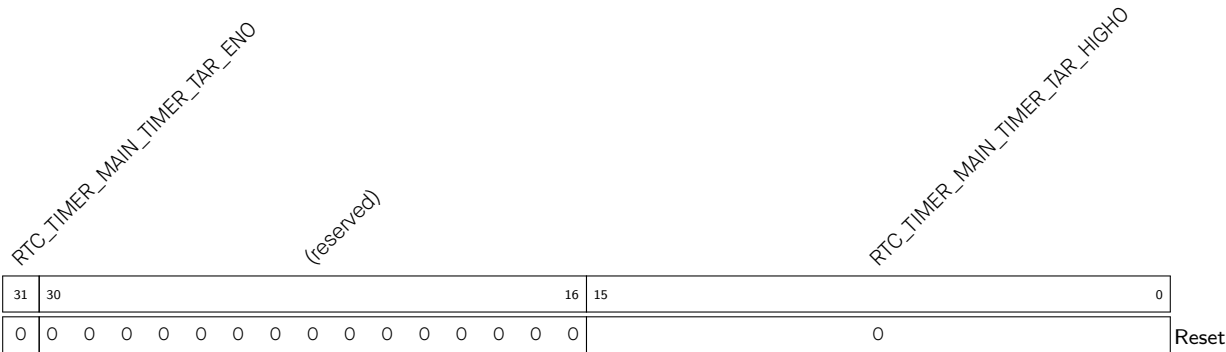
For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 12.1. RTC_TIMER_TARO_LOW_REG (0x0000)



RTC_TIMER_MAIN_TIMER_TAR_LOWO Configures the low 32 bits of the target time 0 of the RTC Timer. (R/W)

Register 12.2. RTC_TIMER_TARO_HIGH_REG (0x0004)



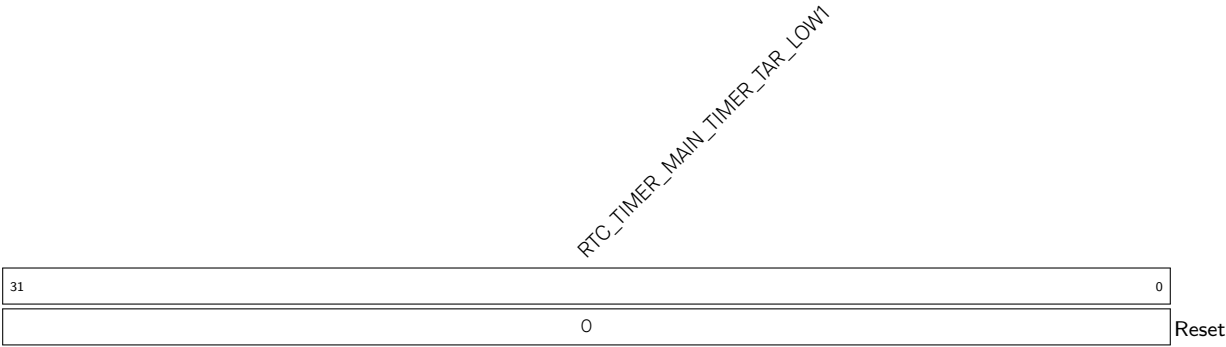
RTC_TIMER_MAIN_TIMER_TAR_HIGHO Configures the high 16 bits of the target time 0. (R/W)

RTC_TIMER_MAIN_TIMER_TAR_ENO Configures to enable the target time 0 of the RTC Timer.

- 0: Disable
- 1: Enable

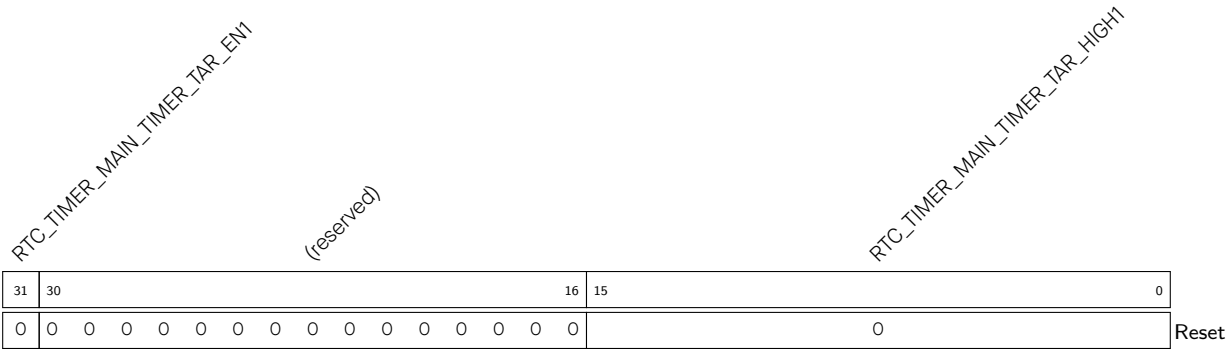
(WT)

Register 12.3. RTC_TIMER_TAR1_LOW_REG (0x0008)



RTC_TIMER_MAIN_TIMER_TAR_LOW1 Configures the low 32 bits of the target time 1. (R/W)

Register 12.4. RTC_TIMER_TAR1_HIGH_REG (0x000C)



RTC_TIMER_MAIN_TIMER_TAR_HIGH1 Configures the high 16 bits of the target time 1. (R/W)

RTC_TIMER_MAIN_TIMER_TAR_EN1 Configures to enable the target time 1 of the RTC Timer.

- 0: Disable
 - 1: Enable
- (WT)

Register 12.5. RTC_TIMER_UPDATE_REG (0x0010)

Diagram illustrating the structure of the **RTC_TIMER_MAIN_TIMER_SYS_STALL** register. The register is 32 bits wide, with bits 31, 30, 29, 28, and 27 labeled. Bit 31 is **RTC_TIMER_MAIN_TIMER_SYS_RST**, bit 30 is **RTC_TIMER_MAIN_TIMER_SYS_STALL**, bit 29 is **RTC_TIMER_MAIN_TIMER_XTAL_OFF**, and bit 28 is **RTC_TIMER_MAIN_TIMER_UPDATE**. Bits 27 through 0 are reserved.

RTC_TIMER_MAIN_TIMER_UPDATE Configures whether to log the current time through software configuration.

0: Not log

1: Log

(WT)

RTC_TIMER_MAIN_TIMER_XTAL_OFF Configures whether to enable RTC Timer to log the time when the crystal powers up or down.

0: Disable

1: Enable

(R/W)

RTC_TIMER_MAIN_TIMER_SYS_STALL Configures whether to enable the RTC Timer to log the time when the CPU enters or exits the stall state.

0: Disable

1: Enable

(R/W)

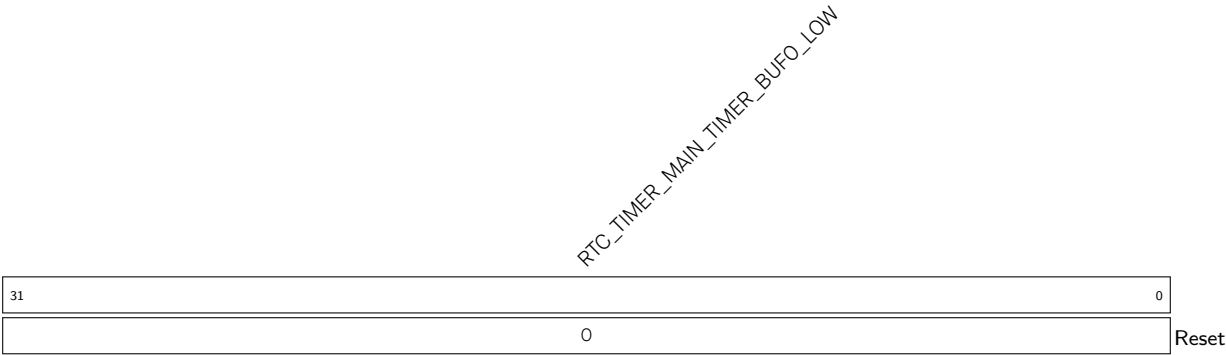
RTC_TIMER_MAIN_TIMER_SYS_RST Configures whether to enable RTC Timer to log the time of system reset.

0: Disable

1: Enable

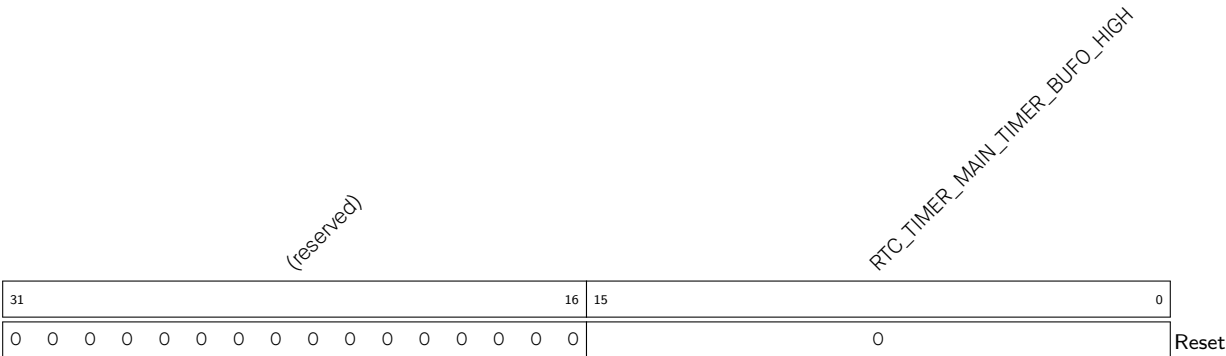
(R/W)

Register 12.6. RTC_TIMER_MAIN_BUFO_LOW_REG (0x0014)



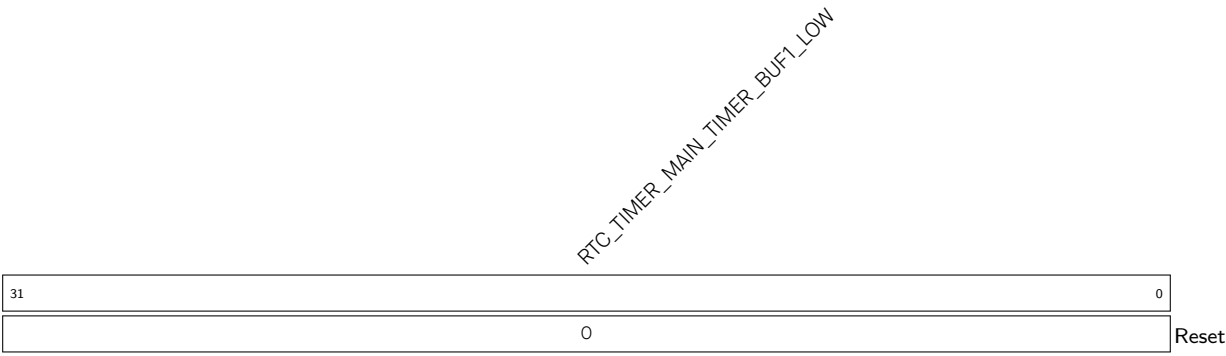
RTC_TIMER_MAIN_TIMER_BUFO_LOW Represents the low 32 bits of the cached value 0 in RTC Timer. (RO)

Register 12.7. RTC_TIMER_MAIN_BUFO_HIGH_REG (0x0018)



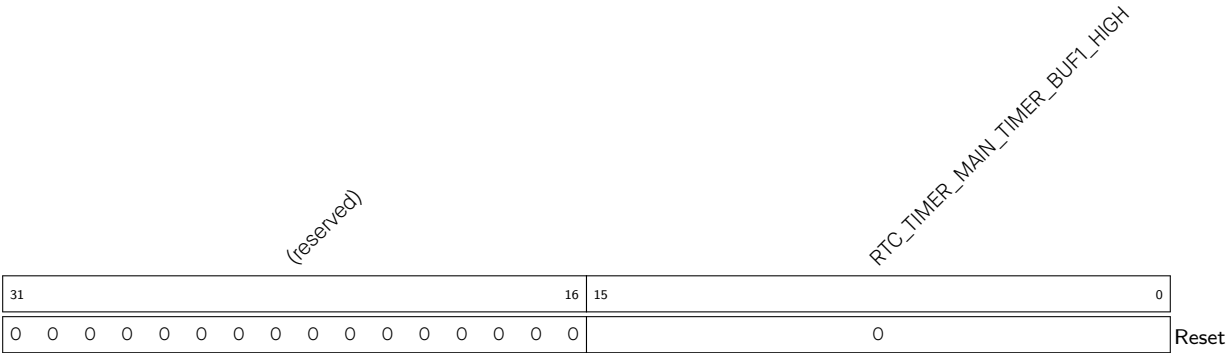
RTC_TIMER_MAIN_TIMER_BUFO_HIGH Represents the high 16 bits of the cached value 0 in RTC Timer. (RO)

Register 12.8. RTC_TIMER_MAIN_BUF1_LOW_REG (0x001C)



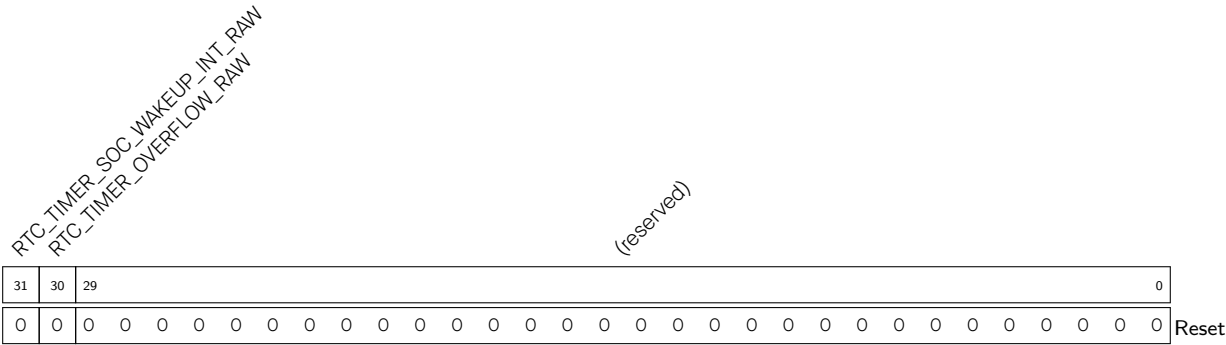
RTC_TIMER_MAIN_TIMER_BUF1_LOW Represents the low 32 bits of the cached value 1 in RTC Timer. (RO)

Register 12.9. RTC_TIMER_MAIN_BUF1_HIGH_REG (0x0020)



RTC_TIMER_MAIN_TIMER_BUF1_HIGH Represents the high 16 bits of the cached value 1 in RTC Timer. (RO)

Register 12.10. RTC_TIMER_INT_RAW_REG (0x0028)



RTC_TIMER_OVERFLOW_RAW The raw interrupt status of RTC_TIMER_OVERFLOW_INT (HP CPU). (R/WTC/SS)

RTC_TIMER_SOC_WAKEUP_INT_RAW The raw interrupt status of RTC_TIMER_CMPO_INT. (R/WTC/SS)

Register 12.11. RTC_TIMER_INT_ST_REG (0x002C)

RTC_TIMER_OVERFLOW_ST The masked interrupt status of RTC_TIMER_OVERFLOW_INT (HP CPU).
(RO)

RTC_TIMER_SOC_WAKEUP_INT_ST	The masked interrupt status of RTC_TIMER_CMPO_INT. (RO)

Register 12.12. RTC_TIMER_INT_ENA_REG (0x0030)

Diagram illustrating the structure of the RTC_TMR register (32 bits):

- Bit 31: RTC_TIMER_SOC_WAKEUP_INT_ENA
- Bit 30: RTC_TIMER_OVERFLOW_ENA
- Bits 29-0: (reserved)

The register is shown as a horizontal bar with bit positions 31, 30, and 29 marked at the left end, and a 0 at the right end. Below the bar, the bits are shown as a sequence of 0s and 1s, with the last bit being 0.

RTC_TIMER_OVERFLOW_ENA Write 1 to enable RTC_TIMER_OVERFLOW_INT (HP CPU). (R/W)

RTC_TIMER_SOC_WAKEUP_INT_ENA Write 1 to enable RTC_TIMER_CMPO_INT. (R/W)

Register 12.13. RTC_TIMER_INT_CLR_REG (0x0034)

Diagram illustrating the structure of the **RTC_TMR_OVERFLOW_CLR** register (32 bits):

- RTC_TIMER_SOC_WAKEUP_INT_CLR** (bits 31-30, 2 bits)
- RTC_TIMER_OVERFLOW_CLR** (bits 29-0, 30 bits)
- (reserved)** (bits 0-0, 1 bit)

The register is labeled **Reset** on the right.

RTC_TIMER_OVERFLOW_CLR Write 1 to clear RTC_TIMER_OVERFLOW_INT (HP CPU). (WT)

RTC_TIMER_SOC_WAKEUP_INT_CLR Write 1 to clear RTC_TIMER_CMPO_INT. (WT)

Register 12.14. RTC_TIMER_LP_INT_RAW_REG (0x0038)

Diagram of the RTC_TIMER_MAIN_TIMER_LP_INT_RAW register. The register is 32 bits wide. Bits 31 and 30 are labeled 'RTC_TIMER_MAIN_TIMER_LP_INT_RAW'. Bit 29 is labeled 'RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_RAW'. The remaining bits (0-28) are labeled '(reserved)'. The register is shown as a horizontal bar with a 'Reset' button on the right.

RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_RAW The raw interrupt status of RTC TIMER OVERFLOW INT (LP CPU). (R/WTC/SS)

RTC_TIMER_MAIN_TIMER_LP_INT_RAW The raw interrupt status of RTC_TIMER_CMP1_INT.
(R/WTC/SS)

Register 12.15. RTC_TIMER_LP_INT_ST_REG (0x003C)

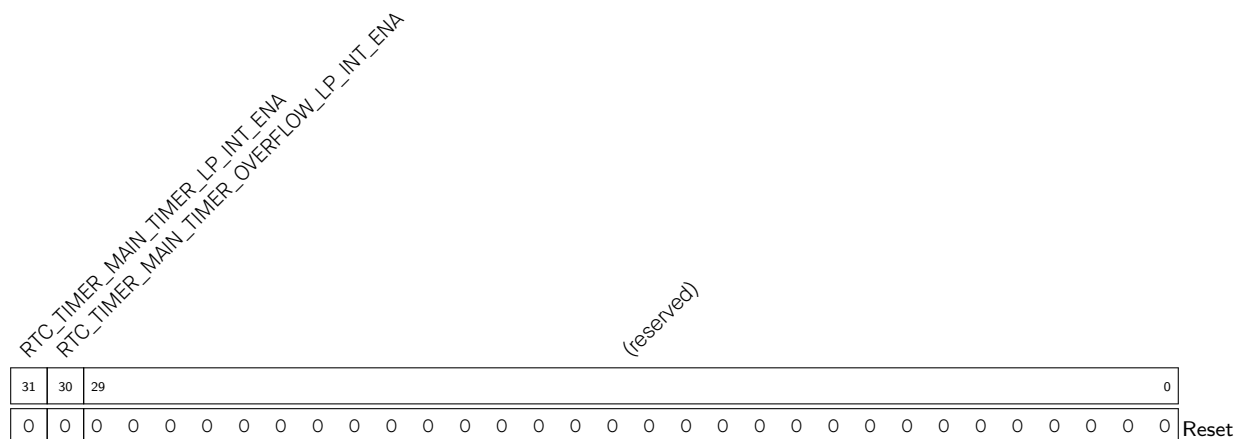
Diagram illustrating the structure of the RTC_TMR register (32 bits wide):

- Bit 31: RTC_TIMER_MAIN_TIMER_LP_INT_ST
- Bit 30: RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_ST
- Bits 29-0: (reserved)

RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_ST The masked interrupt status of RTC_TIMER_OVERFLOW_INT (LP CPU). (RO)

RTC_TIMER_MAIN_TIMER_LP_INT_ST	The masked interrupt status of RTC_TIMER_CMP1_INT. (RO)

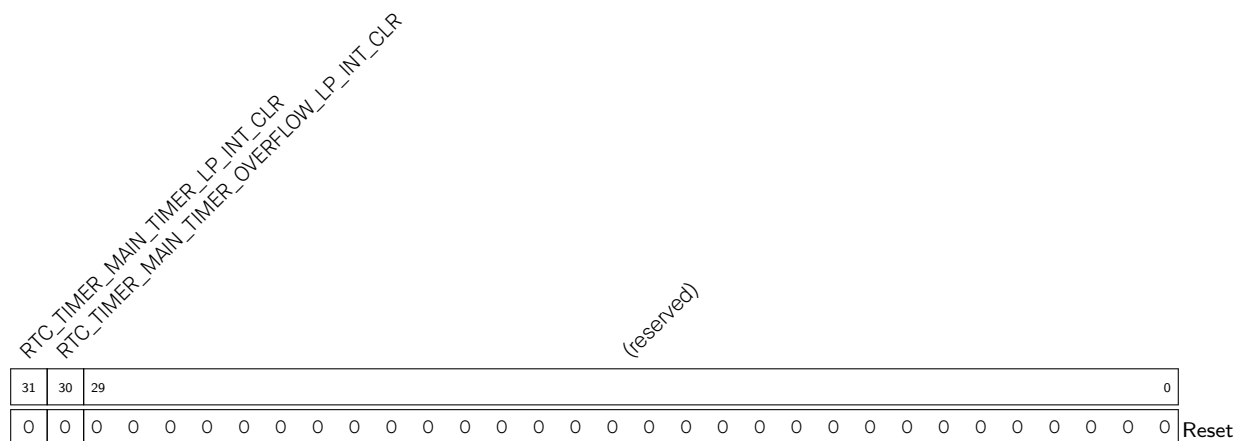
Register 12.16. RTC_TIMER_LP_INT_ENA_REG (0x0040)



RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_ENA Write 1 to enable RTC_TIMER_OVERFLOW_INT (LP CPU). (R/W)

RTC_TIMER_MAIN_TIMER_LP_INT_ENA Write 1 to enable RTC_TIMER_CMP1_INT. (R/W)

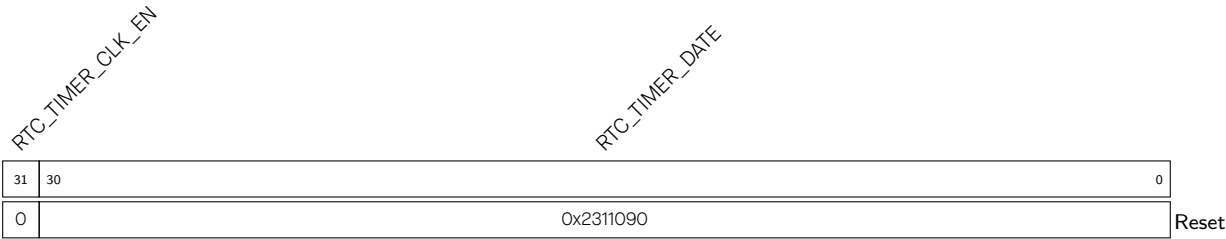
Register 12.17. RTC_TIMER_LP_INT_CLR_REG (0x0044)



RTC_TIMER_MAIN_TIMER_OVERFLOW_LP_INT_CLR Write 1 to clear RTC_TIMER_OVERFLOW_INT (LP CPU). (WT)

RTC_TIMER_MAIN_TIMER_LP_INT_CLR Write 1 to clear RTC_TIMER_CMP1_INT. (WT)

Register 12.18. RTC_TIMER_DATE_REG (0x03FC)



RTC_TIMER_DATE Version control register. (R/W)

RTC_TIMER_CLK_EN Configures RTC timer clock gating.

- 0: Support clock only when the application writes registers.
- 1: Always force the clock on for registers.

(R/W)

Chapter 13

Permission Control (PMS)

13.1 Overview

The permission control of ESP32-C5 can be divided into two parts: PMP (Physical Memory Protection) and APM (Access Permission Management).

PMP is located inside the HP CPU and can control the HP CPU's access to all address spaces. APM is an access permission management module located at the bus port. APM, together with PMP, manages permissions for the HP CPU to access parts of the memory and the peripheral registers. APM can manage the LP CPU independently to access parts of the internal and external memory. Additionally, APM manages DMA masters such as GDMA and TCM_MEM_MONITOR to access parts of internal and external memory.

The distribution of management areas by PMP and APM is shown in Table 13.1-1. For the HP CPU, the control relationship between PMP and APM is shown in Figure 13.1-1.

Table 13.1-1. Management Areas by PMP and AMP

Masters \ Slaves	ROM	HP SRAM	LP SRAM	HP CPU_PERI¹	HP_PERI²	LP_PERI³	EXT MEM⁴
HP CPU	PMP	PMP+ CPU_APM	PMP + LP_APM/ LP_APMO	PMP + HP_APM + PERI_APM	PMP + HP_APM + PERI_APM	PMP + LP_APM + PERI_APM	PMP
LP CPU	N/A	HP_APM	LP_APM/ LP_APMO	N/A	HP_APM + PERI_APM	LP_APM + PERI_APM	N/A
GDMA	N/A	HP_APM	LP_APM/ LP_APMO	N/A	N/A	N/A	HP_APM
Other masters⁵	N/A	HP_APM	LP_APM/ LP_APMO	N/A	N/A	N/A	N/A

¹ Peripheral registers in the HP CPU, address range: 0x600C_0000 – 0x600C_FFFF

² Peripheral registers in the high-performance system, address range: 0x6000_0000 – 0x600A_FFFF

³ Peripheral registers in the low-power system, address range: 0x600B_0000 – 0x600B_FFFF

⁴ External memory, e.g., flash, PSRAM.

⁵ Masters that can request access to the bus, such as TCM_MEM_MONITOR, PSRAM_MEM_MONITOR. For a complete list of masters, please refer to Table 13.5-1.

⁶ For more information on CPU_APM, HP_APM, LP_APM, LP_APMO, and PERI_APM, please refer to Section 13.5.2.

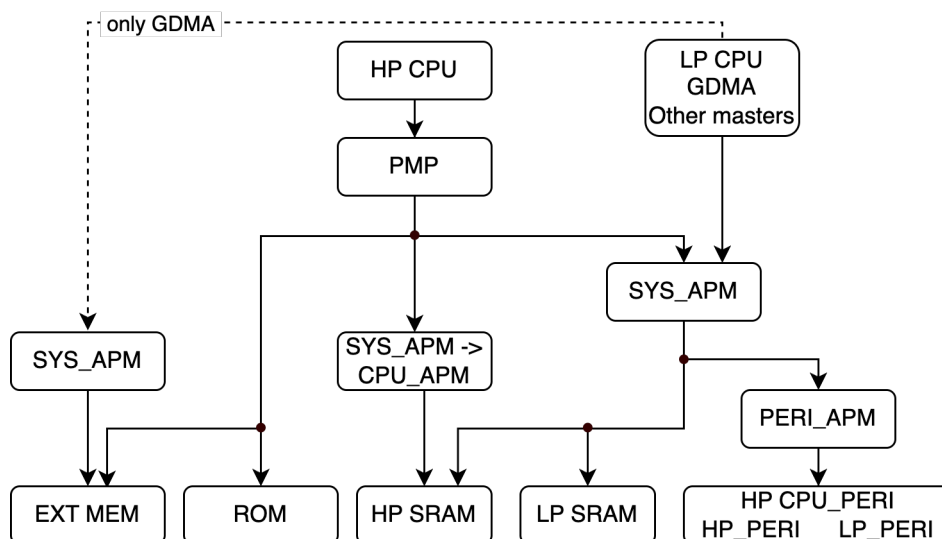


Figure 13.1-1. PMP-APM Management Relation

The diagram illustrates that when the HP CPU accesses ROM and EXT_MEM, the access paths are solely managed by PMP. On the other hand, when the HP CPU accesses the other address spaces, the access paths are controlled by both PMP and APM. If the PMP check fails, the APM permission control will not be triggered. For peripheral registers, there are two levels of APM permission control: the first level manages access permissions of address regions, and the second level manages access permissions of peripherals.

PMP-related registers are located inside the HP CPU and can be read or configured with special instructions. For how to configure PMP, please refer to Chapter 1 *High-Performance CPU [to be added later] > Memory Protection [to be added later]*.

The following sections will provide a detailed introduction to the APM module including its features, functions, and configurations.

13.2 Introduction to APM

The APM module contains two parts: the TEE (Trusted Execution Environment) controller and the APM controller. Each module contains its own registers: TEE registers and APM registers.

- The TEE controller configures the security mode of a particular master in ESP32-C5 (such as GDMA) to access memory or peripheral registers. It supports four security modes: TEE, REEO (Rich Execution Environment), REE1, and REE2.
- The APM controller has two types: SYS_APM controller and PERI_APM controller. The registers of SYS_APM controller are in the APM registers, and the registers of PERI_APM controller are in the TEE registers.
 - The SYS_APM controller manages a master's access permissions when accessing memory and peripheral registers. By comparing the pre-configured address ranges and corresponding access permissions with the information carried on the bus, such as ID number (please refer to Table 13.5-1), security mode, access address, access permissions, etc, the SYS_APM controller determines whether the access is allowed.
 - The PERI_APM controller manages the master's access permissions to peripheral registers. It

compares the pre-configured read and write permission registers for each peripheral with the information carried on the bus, such as secure mode, the peripheral to which the access address belongs, and access permissions, to determine whether access is allowed.

The TEE registers configure the security mode of each master. The APM registers specify the access permission and access address range of each master in the configured security mode. With the TEE controller and APM controller, ESP32-C5 can precisely control the access permission of all masters to the memory and peripheral registers.

13.3 Features

ESP32-C5 has two TEE controllers: HP_TEE and LP_TEE.

- HP_TEE has the following features:
 - Supports four security modes for the masters
 - Supports configurable security mode for 32 masters
- LP_TEE has the following features:
 - Supports four security modes for the masters
 - Supports configurable security mode for one master (i.e., the LP CPU)

ESP32-C5 has four SYS_APM controllers: HP_APM_CTRL, LP_APM_CTRL, LP_APMO_CTRL, and CPU_APM_CTRL.

- HP_APM_CTRL has the following features:
 - Supports access permission configuration for up to 16 address ranges
 - Supports access management to HP SRAM, HP CPU peripheral registers, HP system peripheral registers, and external memory
 - Interrupt function on illegal access
 - Exception information record
- LP_APM_CTRL has the following features:
 - Supports access permission configuration for up to eight address ranges
 - Supports access management to LP SRAM (in low-speed mode) and LP peripheral registers
 - Interrupt function on illegal access
 - Exception information record
- LP_APMO_CTRL has the following features:
 - Supports access permission configuration for up to eight address ranges
 - Supports access management to LP SRAM (in high-speed mode)
 - Interrupt function on illegal access
 - Exception information record
- CPU_APM_CTRL has the following features:

- Supports access permission configuration for up to eight address ranges
- Supports HP CPU's access management to HP SRAM
- Interrupt function on illegal access
- Exception information record

ESP32-C5 has two PERI_APM controllers: HP_PERI_APM_CTRL and LP_PERI_APM_CTRL.

- HP_PERI_APM_CTRL and LP_PERI_APM_CTRL share the following features:
 - Support separate permission configuration for each peripheral register
 - Support returning error status to the master via the system bus
 - Exception information record

13.4 TEE and REE Terminology

TEE Stands for Trusted Execution Environment, which is a secure area that is isolated from the main operating system and provides a secure environment for executing sensitive operations.

REE Stands for Rich Execution Environment, which is the main operating system and environment in which most applications run.

Table 13.4-2. Comparison Between TEE and REE

Aspect	TEE	REE
Security	Enhanced security	Normal security
Access permission	Masters in TEE mode always have read, write, and execute permissions in the address range.	Different levels of access permissions of REE0/1/2 are configurable by software.

13.5 Functional Description

13.5.1 TEE Controller Functional Description

ESP32-C5 supports four security modes: TEE, REE0, REE1, and REE2.

For the HP CPU to access memory or peripheral registers, first select the machine mode or user mode for it, then configure its security mode with HP_TEE registers. For the configuration of machine mode and user mode, please refer to RISC-V Instruction Set Manual, Volume II: Privileged Architecture, Version 1.10.

- When the HP CPU is in machine mode, its security mode is TEE mode.
- When the HP CPU is in user mode, its security mode is REE mode, defined by [TEE_MO_MODE](#) as follows:
 - If [TEE_MO_MODE](#) is 0, which is TEE mode, the valid mode that actually takes effect in the HP CPU user mode is REE0.
 - If [TEE_MO_MODE](#) is 1, 2, or 3, the security mode is REE0, REE1, and REE2 respectively.

For the LP CPU to access memory or peripheral registers, set its secure mode with [LP_TEE_MO_MODE](#).

As for other masters, configure their security mode with [TEE_Mn_MODE](#). *n* represents the ID number of master as listed in Table 13.5-1.

Table 13.5-1. Master Access Source from HP System

Value of <i>n</i>	Source
0	HP CPU
1	Reserved
2	Reserved
3	Reserved
4	Reserved
5	TCM_MEM_MONITOR
6	TRACE
7	Reserved
8	PSRAM_MEM_MONITOR
9 ~ 15	Reserved
16 ~ 31	See the peripherals corresponding to the values 0 ~ 15 in Chapter 3 GDMA Controller (GDMA) > Table 3.4-1 GDMA Selecting Peripherals via Register Configuration . For example, 16 corresponds to the peripheral with value 0 in Table 3.4-1 GDMA Selecting Peripherals via Register Configuration , and 17 corresponds to the peripheral with value 1 in the table, and so on.

You can configure [TEE_Mn_LOCK](#) and [LP_TEE_MO_LOCK](#) to lock the configuration of the master's secure mode. Only chip reset, system reset, and core reset can unlock the configurations and restore all configurations in TEE registers to their default values.

13.5.2 APM Controller Functional Description

13.5.2.1 Architecture

Figure 13.5-1 shows the architecture of the APM controller and the access paths managed by it.

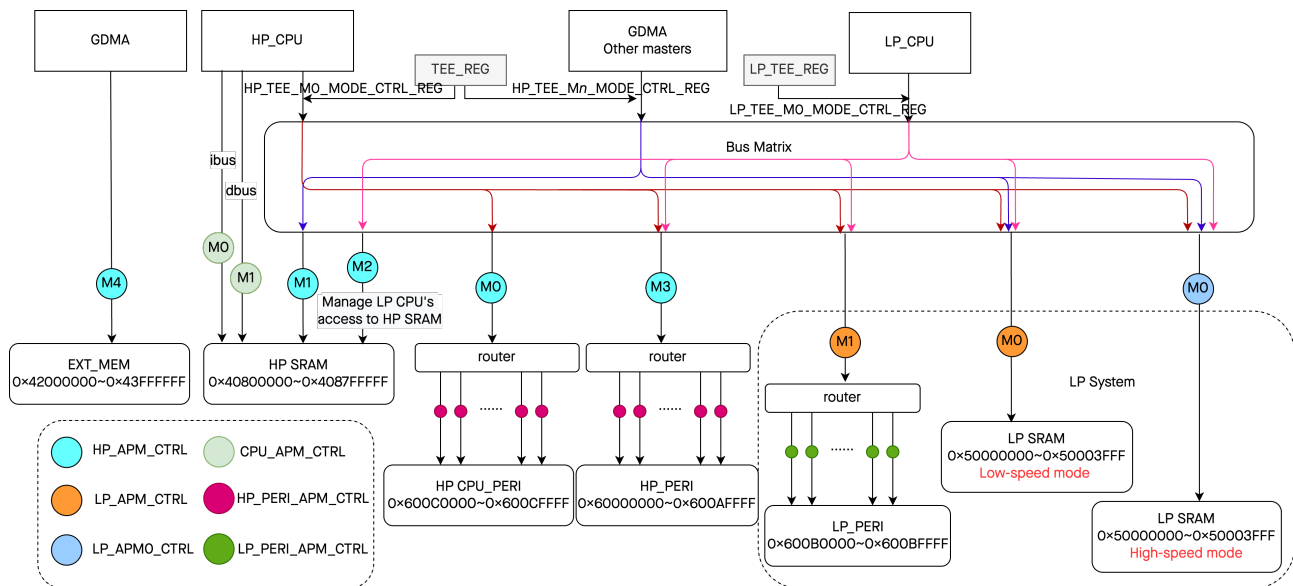


Figure 13.5-1. APM Controller Architecture

Note:

For more information about the “Low-speed mode” and “High-speed mode” in the figure, please refer to [4 System and Memory](#).

As shown in the figure, SYS_APM has four controllers:

- HP_APM_CTRL, configured by the [HP_APM_REG](#) register module.
- LP_APMO_CTRL, configured by the [LP_APMO_REG](#) register module.
- LP_APM_CTRL, configured by the [LP_APM_REG](#) register module.
- CPU_APM_CTRL, configured by the [CPU_APM_REG](#) register module.

PERI_APM has two controllers:

- HP_PERI_APM_CTRL, configured by the [TEE_REG](#) register module.
- LP_PERI_APM_CTRL, configured by the [LP_TEE_REG](#) register module.

The access paths and permission management configuration for each APM controller are as follows.

- HP_APM_CTRL manages five access paths, namely M0 – M4 in the figure. Permission management of each path can be enabled by configuring [HP_APM_FUNC_CTRL_REG](#) (enabled by default).
- LP_APMO_CTRL manages one access path, namely M0 in the figure. Permission management of this path can be enabled by configuring [LP_APMO_FUNC_CTRL_REG](#) (enabled by default).
- LP_APM_CTRL manages two access paths, namely M0 – M1 in the figure. Permission management of this path can be enabled by configuring [LP_APM_FUNC_CTRL_REG](#) (enabled by default).
- CPU_APM_CTRL manages four access paths, namely M0 – M3 highlighted in red in the figure. Permission management of this path can be enabled by configuring [CPU_APM_FUNC_CTRL_REG](#) (enabled by default).

- HP_PERI_APM_CTRL controls all peripheral register access paths in the high-performance system, and permission management of each path can be enabled by configuring [TEE_PERI_CTRL_REG](#).
- LP_PERI_APM_CTRL controls all peripheral register access paths in the low-power system, and permission management of each path can be enabled by configuring [LP_TEE_PERI_CTRL_REG](#).

Table 13.5-2 below provides detailed information about the APM controllers:

Table 13.5-2. APM Controllers Configuration Registers

APM Controllers	Register Modules	Number of Access Paths	Enable Permission Management	Number of Configurable Address Ranges	Enable Address Ranges
HP_APM_CTRL	HP_APM_REG	5	HP_APM_FUNC_CTRL_REG	16	HP_APM_REGION_FILTER_EN_REG
LP_APMO_CTRL	LP_APMO_REG	1	LP_APMO_FUNC_CTRL_REG	8	LP_APMO_REGION_FILTER_EN_REG
LP_APM_CTRL	LP_APM_REG	2	LP_APM_FUNC_CTRL_REG	8	LP_APM_REGION_FILTER_EN_REG
CPU_APM_CTRL	CPU_APM_REG	4	CPU_APM_FUNC_CTRL_REG	8	CPU_APM_REGION_FILTER_EN_REG
HP_PERI_APM_CTRL	TEE_REG	Same as the number of HP system register modules	N/A	Same as the number of HP system register modules	N/A
LP_PERI_APM_CTRL	LP_TEE_REG	Same as the number of LP system register modules	N/A	Same as the number of LP system register modules	N/A

13.5.2.2 SYS_APM Controller

13.5.2.2.1 Address Ranges

- [HP_APM_REG](#) register module can configure up to 16 address ranges (i.e., region n , $n=0-15$) for HP_APM_CTRL. The start and end addresses of each region (address range) are configured by [HP_APM_REGION \$n\$ _ADDR_START](#) and [HP_APM_REGION \$n\$ _ADDR_END](#), respectively. Configure the bit n of [HP_APM_REGION_FILTER_EN_REG](#) to enable region n . Region 0 (i.e., the first address range) is enabled by default.
- [LP_APMO_REG](#) register module can configure up to eight address ranges (i.e., region n , $n=0-7$) for LP_APMO_CTRL. The start and end addresses of each region are configured by [LP_APMO_REGION \$n\$ _ADDR_START](#) and [LP_APMO_REGION \$n\$ _ADDR_END](#). Configure the bit n of [LP_APMO_REGION_FILTER_EN_REG](#) to enable region n . Region 0 (i.e., the first address range) is enabled by default.
- [LP_APM_REG](#) register module can configure up to eight address ranges (i.e., region n , $n=0-7$) for LP_APM_CTRL. The start and end addresses of each region are configured by [LP_APM_REGION \$n\$ _ADDR_START](#) and [LP_APM_REGION \$n\$ _ADDR_END](#). Configure the bit n of [LP_APM_REGION_FILTER_EN_REG](#) to enable region n . Region 0 (i.e., the first address range) is enabled by default.
- [CPU_APM_REG](#) register module can configure up to eight address ranges (i.e., region n , $n=0-7$) for

CPU_APM_CTRL. The start and end addresses of each region are configured by [CPU_APM_REGION \$n\$ _ADDR_START](#) and [CPU_APM_REGION \$n\$ _ADDR_END](#). Configure the bit n of [CPU_APM_REGION_FILTER_EN_REG](#) to enable region n . Region 0 (i.e., the first address range) is enabled by default.

When configuring the address ranges, the address requires a 4-byte alignment (the lower two bits of the address are 0). For example, the address range could be set to 0x4080000C ~ 0x40808774 or 0x600C0008 ~ 0x600CFF70.

13.5.2.2.2 Access Permissions of Address Ranges

For each address range, the access permissions (read/write/execute) are configurable in different security modes:

- Masters in TEE mode always have read, write, and execute permissions in the address range.
- For masters in REE0, REE1 or REE2 mode, access permissions can be configured with [HP_APM_REGION \$n\$ _ATTR_REG](#), [LP_APM_REGION \$n\$ _ATTR_REG](#), or [LP_APMO_REGION \$n\$ _ATTR_REG](#) based on the access path.

Different access paths managed by the same register module share the configuration of address ranges and access permissions. For example, the permission management of data paths HP_APM_CTRL M0-M4 shown in Figure 13.5-1 should follow the address ranges and access permissions of the 16 address ranges configured in the register module [HP_APM_REG](#). Likewise, the permission management of data path LP_APM_CTRL M0-M1 shown in Figure 13.5-1 should follow the address ranges and access permissions of the four address ranges configured in the register module [LP_APM_REG](#).

Take HP SRAM as an example. All masters access it through HP_APM_CTRL M1, except that the HP CPU accesses it through PMP and CPU_APM_CTRL, and the LP CPU accesses it through HP_APM M2. Suppose that [HP_APM_M1_FUNC_EN](#) is enabled and a master in REE1 mode needs to access HP SRAM. The whole process is as follows:

1. HP_APM_CTRL M1 will first determine whether the address requested to access is within the 16 address ranges configured in the [HP_APM_REG](#) register module.
2. Assuming that the address requested to access is within the second address range, then HP_APM_CTRL M1 determines whether the address range is enabled, that is, whether bit 1 of [HP_APM_REGION_FILTER_EN](#) is 1.
3. If the address range is enabled, HP_APM_CTRL M1 checks whether the master has read permission for the second address range, that is, whether [HP_APM_REGION1_R1_R](#) in [HP_APM_REGION1_ATTR_REG](#) is 1. If valid, the read request will be allowed, otherwise, 0 will be returned.

When the HP power domain (please refer to Chapter 4 [Low-power Management \[to be added later\]](#)) restarts after power-down, the LP CPU does not have access to HP SRAM by default. In TEE mode, you can configure [LP_TEE_FORCE_ACC_HPMEM_EN](#) to 1 for the LP CPU to access HP SRAM without requiring permission checks from the SYS_APM controller.

Note:

- When the chip is powered up, only the HP CPU is in TEE mode by default, and other masters are in REE2 mode. By default, the SYS_APM controller blocks access requests from all masters in REE0, REE1, and REE2 modes.

- All registers listed in [13.8 Register Summary](#) can only be configured by the masters that are in TEE security mode.
- The address ranges configured may overlap. For example, if one region is set to be unreadable and another region is readable, then the overlapping area of the two regions is readable. The same rules apply for write and execute permissions.
- You can configure [HP_APM_REGION \$n\$ _LOCK](#), [LP_APM_REGION \$n\$ _LOCK](#), and [LP_APMO_REGION \$n\$ _LOCK](#) to lock the configuration of the start and end address of REGION n and its access permission. Only chip reset, system reset, and core reset can unlock the configurations and restore all configurations in APM registers to their default values.

13.5.2.3 PERI_APM Controller

PERI_APM does not perform permission control by address ranges, but on a per-peripheral register basis. For each peripheral register [PERI](#), there are the following eight configuration fields:

- HP/LP_TEE_WRITE_TEE_[PERI](#)
- HP/LP_TEE_WRITE_REEO_[PERI](#)
- HP/LP_TEE_WRITE_REE1_[PERI](#)
- HP/LP_TEE_WRITE_REE2_[PERI](#)
- HP/LP_TEE_READ_TEE_[PERI](#)
- HP/LP_TEE_READ_REEO_[PERI](#)
- HP/LP_TEE_READ_REE1_[PERI](#)
- HP/LP_TEE_READ_REE2_[PERI](#)

Through the above eight fields, the read/write permissions of slave [PERI](#) in the four security modes TEE, REEO, REE1, and REE2 are respectively controlled. A value of 1 indicates having the corresponding read/write permission under that security mode.

After reset, by default, for each peripheral register, only HP/LP_TEE_WRITE_TEE_[PERI](#) and HP/LP_TEE_READ_TEE_[PERI](#) are set to 1; other fields are all 0, indicating that only the master in TEE security mode can configure peripheral registers.

The six register modules mentioned in this chapter are also peripherals whose access permissions are controlled by PERI_APM. The six register modules and their corresponding permission control registers in PERI_APM are as follows:

- [HP_APM_REG](#), with the permission control register [TEE_HP_APM_CTRL_REG](#)
- [LP_APMO_REG](#), with the permission control register [TEE_HP_APM_CTRL_REG](#)
- [LP_APM_REG](#), with the permission control register [LP_TEE_LP_APM_CTRL_REG](#)
- [CPU_APM_REG](#), with the permission control register [TEE_CPU_APM_CTRL_REG](#)
- [TEE_REG](#), with the permission control register [TEE_TEE_CTRL_REG](#)
- [LP_TEE_REG](#), with the permission control register [LP_TEE_LP_TEE_CTRL_REG](#)

In these permission control registers, only the HP/LP_TEE_WRITE/READ_TEE_[PERI](#) fields are configurable; the HP/LP_TEE_WRITE/READ_REEO/1/2_[PERI](#) fields are not configurable and fixed to 0, indicating that only masters in TEE security mode can access these registers.

13.6 Illegal Access and Interrupts

13.6.1 SYS_APM Controller

When the information carried on the bus does not match the configuration, ESP32-C5 treats it as an illegal access and handles it as follows:

- Rejects the access request and returns a default value, specifically:
 - Returns 0 for read and execute operations
 - Ignores write operations
- Triggers interrupt

The SYS_APM controller will automatically record information related to the illegal access, including the master ID, security mode, access address, reasons for illegal access (address out of bounds or permission restrictions), and permission management result of each access path. All this information can be obtained from relevant registers listed in Section [13.8 Register Summary](#).

Take the access path HP_APM_CTRL MO as an example. When an illegal access occurs:

- [HP_APM_MO_EXCEPTION_ID](#) records the master ID.
- [HP_APM_MO_EXCEPTION_MODE](#) records the security mode.
- [HP_APM_MO_EXCEPTION_ADDR](#) records the access address.
- [HP_APM_MO_EXCEPTION_STATUS](#) records the reason for illegal access.
 - If the address requested to access is not within the enabled address ranges, bit 1 will be set to 1, indicating the address is out of bounds.
 - If the address requested to access is within the enabled address ranges, but the master does not have the read/write/execute permission within this region, then bit 0 will be set to 1, indicating permission restrictions.
- [HP_APM_MO_EXCEPTION_REGION](#) records the permission management result of each address range. This register has a total of 16 bits, corresponding to 16 address ranges, and bit 0 corresponds to the first address range. When the address to access is within a particular enabled address range, but the master does not have the corresponding read/write/execute permission within this address range, the corresponding bit of this register will be set to 1.

The SYS_APM controller can generate 12 interrupt signals, which will be sent to [Interrupt Matrix](#):

- HP_APM_MO_INTR
- HP_APM_M1_INTR
- HP_APM_M2_INTR
- HP_APM_M3_INTR
- HP_APM_M4_INTR
- LP_APM_MO_INTR
- LP_APM_M1_INTR

- LP_APMO_M0_INTR
- CPU_APM_M0_INTR
- CPU_APM_M1_INTR
- CPU_APM_M2_INTR
- CPU_APM_M3_INTR

These interrupt signals correspond to the controlled access paths shown in Figure 13.5-1.

Each interrupt is configured with an interrupt enable register APM_INT_EN and an interrupt clear register EXCEPTION_STATUS_CLR. When the interrupt enable register is set to 1, if an illegal access occurs on a controlled access path, the corresponding interrupt will be generated. Configuring the interrupt clear register clears the interrupt signal until the next illegal access occurs.

13.6.2 PERI_APM Controller

When the information carried on the bus does not match the configuration, ESP32-C5 treats it as an illegal access and handles it as follows:

- Rejects the access request and returns a default value, specifically:
 - Returns 0 for read and execute operations
 - Ignores write operations
- Returns an error message to the master via the bus, causing the master to enter an exception state

Setting HP/LP_BUS_ERR_RESP_EN to 1 enables returning an error message on illegal access.

The PERI_APM controller does not log information related to illegal access.

13.7 Programming Procedure

For a master to access memory or peripheral registers, follow the programming procedures below:

1. Set the HP CPU to the machine mode (i.e., TEE mode).
2. Configure `TEE_Mn_MODE` or `LP_TEE_Mn_MODE` to choose the security mode for the master. The master ID *n* is defined in Table 13.5-1.
3. Configure `HP_APM_REGIONn_ADDR_START` and `HP_APM_REGIONn_ADDR_END`, or `LP_APM_REGIONn_ADDR_START` and `LP_APM_REGIONn_ADDR_END`, or `LP_APMO_REGIONn_ADDR_START` and `LP_APMO_REGIONn_ADDR_END`, or `CPU_APM_REGIONn_ADDR_START` and `CPU_APM_REGIONn_ADDR_END` to define the start and end address of the access address ranges.
4. Configure `HP_APM_REGIONn_ATTR_REG`, or `LP_APM_REGIONn_ATTR_REG`, or `LP_APMO_REGIONn_ATTR_REG`, or `CPU_APM_REGIONn_ATTR_REG` to set the access permissions of each region.
5. Configure `HP_APM_REGION_FILTER_EN_REG`, or `LP_APM_REGION_FILTER_EN_REG`, or `LP_APMO_REGION_FILTER_EN_REG`, or `CPU_APM_REGION_FILTER_EN_REG` to enable region *n*.

- Configure [HP_APM_FUNC_CTRL_REG](#), or [LP_APM_FUNC_CTRL_REG](#), or [LP_APMO_FUNC_CTRL_REG](#), or [CPU_APM_FUNC_CTRL_REG](#) to enable the permission management for different access paths (enabled by default).
- (Required only when accessing peripheral registers) Configure [TEE_PERI_CTRL_REG](#) or [LP_TEE_PERI_CTRL_REG](#) to set the access permissions of the [PERI](#) register.

Take I2S accessing HP SRAM via GDMA as an example, assuming that it is only allowed to read and write in the fourth address range 0x40805000 ~ 0x4080FFFF:

- Configure the HP CPU to machine mode (i.e., TEE mode).
- According to the master ID number in Table 13.5-1, set [TEE_M19_MODE](#) to 1, so that the security mode for I2S access via GDMA is REEO.
- Configure [HP_APM_REGION3_ADDR_START](#) to 0x40805000 and [HP_APM_REGION3_ADDR_END](#) to 0x4080EFFF, respectively.
- Set [HP_APM_REGION3_RO_W](#) and [HP_APM_REGION3_RO_R](#) to 1 to enable the access permissions to the address range.
- Set bit 3 of [HP_APM_REGION_FILTER_EN](#) to 1 to enable region 3 (i.e., the fourth address range).
- Set [HP_APM_M1_FUNC_EN](#) to 1 to enable the permission management for HP_APM_CTRL M1.

Through this configuration, I2S can read and write in the address range of 0x40805000 to 0x4080FFFF in HP SRAM using GDMA.

13.8 Register Summary

13.8.1 HP_APM_REG

The addresses in this section are relative to the HP_APM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
HP_APM_REGION_FILTER_EN_REG	Region enable register	0x0000	R/W
HP_APM_REGION0_ADDR_START_REG	Region address register	0x0004	R/W
HP_APM_REGION0_ADDR_END_REG	Region address register	0x0008	R/W
HP_APM_REGION1_ADDR_START_REG	Region address register	0x0010	R/W
HP_APM_REGION1_ADDR_END_REG	Region address register	0x0014	R/W
HP_APM_REGION2_ADDR_START_REG	Region address register	0x001C	R/W
HP_APM_REGION2_ADDR_END_REG	Region address register	0x0020	R/W
HP_APM_REGION3_ADDR_START_REG	Region address register	0x0028	R/W
HP_APM_REGION3_ADDR_END_REG	Region address register	0x002C	R/W
HP_APM_REGION4_ADDR_START_REG	Region address register	0x0034	R/W
HP_APM_REGION4_ADDR_END_REG	Region address register	0x0038	R/W
HP_APM_REGION5_ADDR_START_REG	Region address register	0x0040	R/W

Name	Description	Address	Access
HP_APM_REGION5_ADDR_END_REG	Region address register	0x0044	R/W
HP_APM_REGION6_ADDR_START_REG	Region address register	0x004C	R/W
HP_APM_REGION6_ADDR_END_REG	Region address register	0x0050	R/W
HP_APM_REGION7_ADDR_START_REG	Region address register	0x0058	R/W
HP_APM_REGION7_ADDR_END_REG	Region address register	0x005C	R/W
HP_APM_REGION8_ADDR_START_REG	Region address register	0x0064	R/W
HP_APM_REGION8_ADDR_END_REG	Region address register	0x0068	R/W
HP_APM_REGION9_ADDR_START_REG	Region address register	0x0070	R/W
HP_APM_REGION9_ADDR_END_REG	Region address register	0x0074	R/W
HP_APM_REGION10_ADDR_START_REG	Region address register	0x007C	R/W
HP_APM_REGION10_ADDR_END_REG	Region address register	0x0080	R/W
HP_APM_REGION11_ADDR_START_REG	Region address register	0x0088	R/W
HP_APM_REGION11_ADDR_END_REG	Region address register	0x008C	R/W
HP_APM_REGION12_ADDR_START_REG	Region address register	0x0094	R/W
HP_APM_REGION12_ADDR_END_REG	Region address register	0x0098	R/W
HP_APM_REGION13_ADDR_START_REG	Region address register	0x00A0	R/W
HP_APM_REGION13_ADDR_END_REG	Region address register	0x00A4	R/W
HP_APM_REGION14_ADDR_START_REG	Region address register	0x00AC	R/W
HP_APM_REGION14_ADDR_END_REG	Region address register	0x00B0	R/W
HP_APM_REGION15_ADDR_START_REG	Region address register	0x00B8	R/W
HP_APM_REGION15_ADDR_END_REG	Region address register	0x00BC	R/W
HP_APM_REGION0_ATTR_REG	Region access permissions configuration register	0x000C	R/W
HP_APM_REGION1_ATTR_REG	Region access permissions configuration register	0x0018	R/W
HP_APM_REGION2_ATTR_REG	Region access permissions configuration register	0x0024	R/W
HP_APM_REGION3_ATTR_REG	Region access permissions configuration register	0x0030	R/W
HP_APM_REGION4_ATTR_REG	Region access permissions configuration register	0x003C	R/W
HP_APM_REGION5_ATTR_REG	Region access permissions configuration register	0x0048	R/W
HP_APM_REGION6_ATTR_REG	Region access permissions configuration register	0x0054	R/W
HP_APM_REGION7_ATTR_REG	Region access permissions configuration register	0x0060	R/W
HP_APM_REGION8_ATTR_REG	Region access permissions configuration register	0x006C	R/W
HP_APM_REGION9_ATTR_REG	Region access permissions configuration register	0x0078	R/W
HP_APM_REGION10_ATTR_REG	Region access permissions configuration register	0x0084	R/W

Name	Description	Address	Access
HP_APM_REGION11_ATTR_REG	Region access permissions configuration register	0x0090	R/W
HP_APM_REGION12_ATTR_REG	Region access permissions configuration register	0x009C	R/W
HP_APM_REGION13_ATTR_REG	Region access permissions configuration register	0x00A8	R/W
HP_APM_REGION14_ATTR_REG	Region access permissions configuration register	0x00B4	R/W
HP_APM_REGION15_ATTR_REG	Region access permissions configuration register	0x00C0	R/W
HP_APM_FUNC_CTRL_REG	APM access path permission management register	0x00C4	R/W
Status Registers			
HP_APM_M0_STATUS_REG	HP_APM_CTRL M0 status register	0x00C8	RO
HP_APM_M0_STATUS_CLR_REG	HP_APM_CTRL M0 status clear register	0x00CC	WT
HP_APM_M0_EXCEPTION_INFO0_REG	HP_APM_CTRL M0 exception information register	0x00D0	RO
HP_APM_M0_EXCEPTION_INFO1_REG	HP_APM_CTRL M0 exception information register	0x00D4	RO
HP_APM_M1_STATUS_REG	HP_APM_CTRL M1 status register	0x00D8	RO
HP_APM_M1_STATUS_CLR_REG	HP_APM_CTRL M1 status clear register	0x00DC	WT
HP_APM_M1_EXCEPTION_INFO0_REG	HP_APM_CTRL M1 exception information register	0x00E0	RO
HP_APM_M1_EXCEPTION_INFO1_REG	HP_APM_CTRL M1 exception information register	0x00E4	RO
HP_APM_M2_STATUS_REG	HP_APM_CTRL M2 status register	0x00E8	RO
HP_APM_M2_STATUS_CLR_REG	HP_APM_CTRL M2 status clear register	0x00EC	WT
HP_APM_M2_EXCEPTION_INFO0_REG	HP_APM_CTRL M2 exception information register	0x00F0	RO
HP_APM_M2_EXCEPTION_INFO1_REG	HP_APM_CTRL M2 exception information register	0x00F4	RO
HP_APM_M3_STATUS_REG	HP_APM_CTRL M3 status register	0x00F8	RO
HP_APM_M3_STATUS_CLR_REG	HP_APM_CTRL M3 status clear register	0x00FC	WT
HP_APM_M3_EXCEPTION_INFO0_REG	HP_APM_CTRL M3 exception information register	0x0100	RO
HP_APM_M3_EXCEPTION_INFO1_REG	HP_APM_CTRL M3 exception information register	0x0104	RO
HP_APM_M4_STATUS_REG	HP_APM_CTRL M4 status register	0x0108	RO
HP_APM_M4_STATUS_CLR_REG	HP_APM_CTRL M4 status clear register	0x010C	WT
HP_APM_M4_EXCEPTION_INFO0_REG	HP_APM_CTRL M4 exception information register	0x0110	RO
HP_APM_M4_EXCEPTION_INFO1_REG	HP_APM_CTRL M4 exception information register	0x0114	RO
Interrupt Registers			

Name	Description	Address	Access
HP_APM_INT_EN_REG	HP_APM_CTRL M0/1/2/3/4 interrupt enable register	0x0118	R/W
Clock Gating Registers			
HP_APM_CLOCK_GATE_REG	Clock gating register	0x07F8	R/W
Version Control Registers			
HP_APM_DATE_REG	Version control register	0x07FC	R/W

13.8.2 LP_APM_REG

The addresses in this section are relative to the LP_APM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
LP_APM_REGION_FILTER_EN_REG	Region enable register	0x0000	R/W
LP_APM_REGION0_ADDR_START_REG	Region address register	0x0004	R/W
LP_APM_REGION0_ADDR_END_REG	Region address register	0x0008	R/W
LP_APM_REGION1_ADDR_START_REG	Region address register	0x0010	R/W
LP_APM_REGION1_ADDR_END_REG	Region address register	0x0014	R/W
LP_APM_REGION2_ADDR_START_REG	Region address register	0x001C	R/W
LP_APM_REGION2_ADDR_END_REG	Region address register	0x0020	R/W
LP_APM_REGION3_ADDR_START_REG	Region address register	0x0028	R/W
LP_APM_REGION3_ADDR_END_REG	Region address register	0x002C	R/W
LP_APM_REGION4_ADDR_START_REG	Region address register	0x0034	R/W
LP_APM_REGION4_ADDR_END_REG	Region address register	0x0038	R/W
LP_APM_REGION5_ADDR_START_REG	Region address register	0x0040	R/W
LP_APM_REGION5_ADDR_END_REG	Region address register	0x0044	R/W
LP_APM_REGION6_ADDR_START_REG	Region address register	0x004C	R/W
LP_APM_REGION6_ADDR_END_REG	Region address register	0x0050	R/W
LP_APM_REGION7_ADDR_START_REG	Region address register	0x0058	R/W
LP_APM_REGION7_ADDR_END_REG	Region address register	0x005C	R/W
LP_APM_REGION0_ATTR_REG	Region access permissions configuration register	0x000C	R/W
LP_APM_REGION1_ATTR_REG	Region access permissions configuration register	0x0018	R/W
LP_APM_REGION2_ATTR_REG	Region access permissions configuration register	0x0024	R/W
LP_APM_REGION3_ATTR_REG	Region access permissions configuration register	0x0030	R/W
LP_APM_REGION4_ATTR_REG	Region access permissions configuration register	0x003C	R/W

Name	Description	Address	Access
LP_APM_REGION5_ATTR_REG	Region access permissions configuration register	0x0048	R/W
LP_APM_REGION6_ATTR_REG	Region access permissions configuration register	0x0054	R/W
LP_APM_REGION7_ATTR_REG	Region access permissions configuration register	0x0060	R/W
LP_APM_FUNC_CTRL_REG	APM access path permission management register	0x00C4	R/W
Status Registers			
LP_APM_M0_STATUS_REG	LP_APM_CTRL M0 status register	0x00C8	RO
LP_APM_M0_STATUS_CLR_REG	LP_APM_CTRL M0 status clear register	0x00CC	WT
LP_APM_M0_EXCEPTION_INFO0_REG	LP_APM_CTRL M0 exception information register	0x00D0	RO
LP_APM_M0_EXCEPTION_INFO1_REG	LP_APM_CTRL M0 exception information register	0x00D4	RO
LP_APM_M1_STATUS_REG	LP_APM_CTRL M1 status register	0x00D8	RO
LP_APM_M1_STATUS_CLR_REG	LP_APM_CTRL M1 status clear register	0x00DC	WT
LP_APM_M1_EXCEPTION_INFO0_REG	LP_APM_CTRL M1 exception information register	0x00E0	RO
LP_APM_M1_EXCEPTION_INFO1_REG	LP_APM_CTRL M1 exception information register	0x00E4	RO
Interrupt Registers			
LP_APM_INT_EN_REG	LP_APM_CTRL M0/1 interrupt enable register	0x00E8	R/W
Clock Gating Registers			
LP_APM_CLOCK_GATE_REG	Clock gating register	0x00EC	R/W
Version Control Registers			
LP_APM_DATE_REG	Version control register	0x00FC	R/W

13.8.3 LP_APM0_REG

The addresses in this section are relative to the LP_APM0 base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
LP_APM0_REGION_FILTER_EN_REG	Region enable register	0x0000	R/W
LP_APM0_REGION0_ADDR_START_REG	Region address register	0x0004	R/W
LP_APM0_REGION0_ADDR_END_REG	Region address register	0x0008	R/W
LP_APM0_REGION1_ADDR_START_REG	Region address register	0x0010	R/W
LP_APM0_REGION1_ADDR_END_REG	Region address register	0x0014	R/W
LP_APM0_REGION2_ADDR_START_REG	Region address register	0x001C	R/W

Name	Description	Address	Access
LP_APMO_REGION2_ADDR_END_REG	Region address register	0x0020	R/W
LP_APMO_REGION3_ADDR_START_REG	Region address register	0x0028	R/W
LP_APMO_REGION3_ADDR_END_REG	Region address register	0x002C	R/W
LP_APMO_REGION4_ADDR_START_REG	Region address register	0x0034	R/W
LP_APMO_REGION4_ADDR_END_REG	Region address register	0x0038	R/W
LP_APMO_REGION5_ADDR_START_REG	Region address register	0x0040	R/W
LP_APMO_REGION5_ADDR_END_REG	Region address register	0x0044	R/W
LP_APMO_REGION6_ADDR_START_REG	Region address register	0x004C	R/W
LP_APMO_REGION6_ADDR_END_REG	Region address register	0x0050	R/W
LP_APMO_REGION7_ADDR_START_REG	Region address register	0x0058	R/W
LP_APMO_REGION7_ADDR_END_REG	Region address register	0x005C	R/W
LP_APMO_REGION0_ATTR_REG	Region access permissions configuration register	0x000C	R/W
LP_APMO_REGION1_ATTR_REG	Region access permissions configuration register	0x0018	R/W
LP_APMO_REGION2_ATTR_REG	Region access permissions configuration register	0x0024	R/W
LP_APMO_REGION3_ATTR_REG	Region access permissions configuration register	0x0030	R/W
LP_APMO_REGION4_ATTR_REG	Region access permissions configuration register	0x003C	R/W
LP_APMO_REGION5_ATTR_REG	Region access permissions configuration register	0x0048	R/W
LP_APMO_REGION6_ATTR_REG	Region access permissions configuration register	0x0054	R/W
LP_APMO_REGION7_ATTR_REG	Region access permissions configuration register	0x0060	R/W
LP_APMO_FUNC_CTRL_REG	APM access path permission management register	0x00C4	R/W
Status Registers			
LP_APMO_MO_STATUS_REG	LP_APMO_CTRL MO status register	0x00C8	RO
LP_APMO_MO_STATUS_CLR_REG	LP_APMO_CTRL MO status clear register	0x00CC	WT
LP_APMO_MO_EXCEPTION_INFO0_REG	LP_APMO_CTRL MO exception information register	0x00D0	RO
LP_APMO_MO_EXCEPTION_INFO1_REG	LP_APMO_CTRL MO exception information register	0x00D4	RO
Interrupt Registers			
LP_APMO_INT_EN_REG	LP_APMO_CTRL interrupt enable register	0x00D8	R/W
Clock Gating Registers			
LP_APMO_CLOCK_GATE_REG	Clock gating register	0x00DC	R/W
Version Control Registers			
LP_APMO_DATE_REG	Version control register	0x07FC	R/W

13.8.4 CPU_APM_REG

The addresses in this section are relative to the CPU_APM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
CPU_APM_REGION_FILTER_EN_REG	Region enable register	0x0000	R/W
Region Address Registers			
CPU_APM_REGION0_ADDR_START_REG	Region address register	0x0004	varies
CPU_APM_REGION0_ADDR_END_REG	Region address register	0x0008	varies
CPU_APM_REGION1_ADDR_START_REG	Region address register	0x0010	varies
CPU_APM_REGION1_ADDR_END_REG	Region address register	0x0014	varies
CPU_APM_REGION2_ADDR_START_REG	Region address register	0x001C	varies
CPU_APM_REGION2_ADDR_END_REG	Region address register	0x0020	varies
CPU_APM_REGION3_ADDR_START_REG	Region address register	0x0028	varies
CPU_APM_REGION3_ADDR_END_REG	Region address register	0x002C	varies
CPU_APM_REGION4_ADDR_START_REG	Region address register	0x0034	varies
CPU_APM_REGION4_ADDR_END_REG	Region address register	0x0038	varies
CPU_APM_REGION5_ADDR_START_REG	Region address register	0x0040	varies
CPU_APM_REGION5_ADDR_END_REG	Region address register	0x0044	varies
CPU_APM_REGION6_ADDR_START_REG	Region address register	0x004C	varies
CPU_APM_REGION6_ADDR_END_REG	Region address register	0x0050	varies
CPU_APM_REGION7_ADDR_START_REG	Region address register	0x0058	varies
CPU_APM_REGION7_ADDR_END_REG	Region address register	0x005C	varies
CPU_APM_REGION0_ATTR_REG	Region access permissions configuration register	0x000C	R/W
CPU_APM_REGION1_ATTR_REG	Region access permissions configuration register	0x0018	R/W
CPU_APM_REGION2_ATTR_REG	Region access permissions configuration register	0x0024	R/W
CPU_APM_REGION3_ATTR_REG	Region access permissions configuration register	0x0030	R/W
CPU_APM_REGION4_ATTR_REG	Region access permissions configuration register	0x003C	R/W
CPU_APM_REGION5_ATTR_REG	Region access permissions configuration register	0x0048	R/W
CPU_APM_REGION6_ATTR_REG	Region access permissions configuration register	0x0054	R/W
CPU_APM_REGION7_ATTR_REG	Region access permissions configuration register	0x0060	R/W
CPU_APM_FUNC_CTRL_REG	APM access path permission management register	0x00C4	R/W
Status Registers			

Name	Description	Address	Access
CPU_APM_MO_STATUS_REG	CPU_APM_CTRL MO status register	0x00C8	RO
CPU_APM_MO_STATUS_CLR_REG	CPU_APM_CTRL MO status clear register	0x00CC	WT
CPU_APM_MO_EXCEPTION_INFO0_REG	CPU_APM_CTRL MO exception information register	0x00D0	RO
CPU_APM_MO_EXCEPTION_INFO1_REG	CPU_APM_CTRL MO exception information register	0x00D4	RO
CPU_APM_M1_STATUS_REG	CPU_APM_CTRL M1 status register	0x00D8	RO
CPU_APM_M1_STATUS_CLR_REG	CPU_APM_CTRL M1 status clear register	0x00DC	WT
CPU_APM_M1_EXCEPTION_INFO0_REG	CPU_APM_CTRL M1 exception information register	0x00E0	RO
CPU_APM_M1_EXCEPTION_INFO1_REG	CPU_APM_CTRL M1 exception information register	0x00E4	RO
Interrupt Registers			
CPU_APM_INT_EN_REG	CPU_APM_CTRL MO/1 interrupt enable register	0x0118	R/W
Clock Gating Registers			
CPU_APM_CLOCK_GATE_REG	Clock gating register	0x07F8	R/W
Version Control Registers			
CPU_APM_DATE_REG	Version control register	0x07FC	R/W

13.8.5 TEE_REG

The addresses in this section are relative to the TEE base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
TEE_MO_MODE_CTRL_REG	Security mode configuration register	0x0000	R/W
TEE_M1_MODE_CTRL_REG	Security mode configuration register	0x0004	R/W
TEE_M2_MODE_CTRL_REG	Security mode configuration register	0x0008	R/W
TEE_M3_MODE_CTRL_REG	Security mode configuration register	0x000C	R/W
TEE_M4_MODE_CTRL_REG	Security mode configuration register	0x0010	R/W
TEE_M5_MODE_CTRL_REG	Security mode configuration register	0x0014	R/W
TEE_M6_MODE_CTRL_REG	Security mode configuration register	0x0018	R/W
TEE_M7_MODE_CTRL_REG	Security mode configuration register	0x001C	R/W
TEE_M8_MODE_CTRL_REG	Security mode configuration register	0x0020	R/W
TEE_M9_MODE_CTRL_REG	Security mode configuration register	0x0024	R/W
TEE_M10_MODE_CTRL_REG	Security mode configuration register	0x0028	R/W
TEE_M11_MODE_CTRL_REG	Security mode configuration register	0x002C	R/W
TEE_M12_MODE_CTRL_REG	Security mode configuration register	0x0030	R/W
TEE_M13_MODE_CTRL_REG	Security mode configuration register	0x0034	R/W
TEE_M14_MODE_CTRL_REG	Security mode configuration register	0x0038	R/W

Name	Description	Address	Access
TEE_M15_MODE_CTRL_REG	Security mode configuration register	0x003C	R/W
TEE_M16_MODE_CTRL_REG	Security mode configuration register	0x0040	R/W
TEE_M17_MODE_CTRL_REG	Security mode configuration register	0x0044	R/W
TEE_M18_MODE_CTRL_REG	Security mode configuration register	0x0048	R/W
TEE_M19_MODE_CTRL_REG	Security mode configuration register	0x004C	R/W
TEE_M20_MODE_CTRL_REG	Security mode configuration register	0x0050	R/W
TEE_M21_MODE_CTRL_REG	Security mode configuration register	0x0054	R/W
TEE_M22_MODE_CTRL_REG	Security mode configuration register	0x0058	R/W
TEE_M23_MODE_CTRL_REG	Security mode configuration register	0x005C	R/W
TEE_M24_MODE_CTRL_REG	Security mode configuration register	0x0060	R/W
TEE_M25_MODE_CTRL_REG	Security mode configuration register	0x0064	R/W
TEE_M26_MODE_CTRL_REG	Security mode configuration register	0x0068	R/W
TEE_M27_MODE_CTRL_REG	Security mode configuration register	0x006C	R/W
TEE_M28_MODE_CTRL_REG	Security mode configuration register	0x0070	R/W
TEE_M29_MODE_CTRL_REG	Security mode configuration register	0x0074	R/W
TEE_M30_MODE_CTRL_REG	Security mode configuration register	0x0078	R/W
TEE_M31_MODE_CTRL_REG	Security mode configuration register	0x007C	R/W
Peripheral Read/Write Control Registers			
TEE_UART0_CTRL_REG	UART0 read/write control register	0x0088	R/W
TEE_UART1_CTRL_REG	UART1 read/write control register	0x008C	R/W
TEE_UHCIO_CTRL_REG	UHCI read/write control register	0x0090	R/W
TEE_I2C_EXT0_CTRL_REG	I2C read/write control register	0x0094	R/W
TEE_I2S_CTRL_REG	I2S read/write control register	0x009C	R/W
TEE_PARL_IO_CTRL_REG	PARL_IO read/write control register	0x00A0	R/W
TEE_PWM_CTRL_REG	MCPWM read/write control register	0x00A4	R/W
TEE_LEDC_CTRL_REG	LEDC read/write control register	0x00AC	R/W
TEE_TWAIO_CTRL_REG	TWAIO read/write control register	0x00B0	R/W
TEE_USB_SERIAL_JTAG_CTRL_REG	USB_SERIAL_JTAG read/write control register	0x00B4	R/W
TEE_RMT_CTRL_REG	RMT read/write control register	0x00B8	R/W
TEE_GDMA_CTRL_REG	GDMA read/write control register	0x00BC	R/W
TEE_ETM_CTRL_REG	SOC_ETM read/write control register	0x00C4	R/W
TEE_INTMTX_CTRL_REG	INTMTX read/write control register	0x00C8	R/W
TEE_APB_ADC_CTRL_REG	SAR ADC read/write control register	0x00D0	R/W
TEE_TIMERGROUP0_CTRL_REG	TIMG0 read/write control register	0x00D4	R/W
TEE_TIMERGROUP1_CTRL_REG	TIMG1 read/write control register	0x00D8	R/W
TEE_SYSTIMER_CTRL_REG	SYSTIMER read/write control register	0x00DC	R/W
TEE_PCNT_CTRL_REG	PCNT read/write control register	0x00F4	R/W
TEE_IOMUX_CTRL_REG	IO MUX read/write control register	0x00F8	R/W
TEE_PSRAM_MEM_MONITOR_CTRL_REG	PSRAM_MEM_MONITOR read/write control register	0x00FC	R/W
TEE_MEM_ACS_MONITOR_CTRL_REG	TCM_MEM_MONITOR read/write control register	0x0100	R/W

Name	Description	Address	Access
TEE_HP_SYSTEM_REG_CTRL_REG	HP_SYSREG read/write control register	0x0104	R/W
TEE_PCR_REG_CTRL_REG	PCR read/write control register	0x0108	R/W
TEE_MSPI_CTRL_REG	SPI01 read/write control register	0x010C	R/W
TEE_HP_APM_CTRL_REG	HP_APM and LP_APMO read/write control register	0x0110	varies
TEE_CPU_APM_CTRL_REG	CPU_APM_REG read/write control register	0x0114	varies
TEE_TEE_CTRL_REG	TEE read/write control register	0x0118	varies
TEE_CRYPT_CTRL_REG	CRYPT read/write control register, including security peripherals from AES to ECDSA address range	0x011C	R/W
TEE_TRACE_CTRL_REG	TRACE read/write control register	0x0120	R/W
TEE_CPU_BUS_MONITOR_CTRL_REG	BUS_MONITOR read/write control register	0x0128	R/W
TEE_INTPRI_REG_CTRL_REG	INTPRI_REG read/write control register	0x012C	R/W
TEE_TWA1_CTRL_REG	TWA1 read/write control register	0x0138	R/W
TEE_SPI2_CTRL_REG	SPI2 read/write control register	0x013C	R/W
TEE_BS_CTRL_REG	BITSCRAMBLER read/write control register	0x0140	R/W
TEE_BUS_ERR_CONF_REG	Error message return configuration register	0xOFF0	R/W
clock gating register			
TEE_CLOCK_GATE_REG	Clock gating register	0xOFF8	R/W
Version Control Registers			
TEE_DATE_REG	Version control register	0xOFFC	R/W

13.8.6 LP_TEE_REG

The addresses in this section are relative to the LP_TEE base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
LP_TEE_MO_MODE_CTRL_REG	Security mode configuration register	0x0000	R/W
Peripheral Read/Write Control Register			
LP_TEE_EFUSE_CTRL_REG	eFuse read/write control register	0x0004	R/W
LP_TEE_PMU_CTRL_REG	PMU read/write control register	0x0008	R/W
LP_TEE_CLKRST_CTRL_REG	LP_CLKRST read/write control register	0x000C	R/W
LP_TEE_LP_AON_CTRL_CTRL_REG	LP_AON read/write control register	0x0010	R/W
LP_TEE_LP_TIMER_CTRL_REG	LP_TIMER read/write control register	0x0014	R/W
LP_TEE_LP_WDT_CTRL_REG	LP_WDT read/write control register	0x0018	R/W
LP_TEE_LP_PERI_CTRL_REG	LPPERI read/write control register	0x001C	R/W

Name	Description	Address	Access
LP_TEE_LP_ANA_PERI_CTRL_REG	LP_ANA_PERI read/write control register	0x0020	R/W
LP_TEE_LP_IO_CTRL_REG	LP_GPIO and LP_IO_MUX read/write control register	0x002C	R/W
LP_TEE_LP_TEE_CTRL_REG	LP_TEE read/write control register	0x0034	varies
LP_TEE_UART_CTRL_REG	LP_UART read/write control register	0x0038	R/W
LP_TEE_I2C_EXT_CTRL_REG	LP_I2C read/write control register	0x0040	R/W
LP_TEE_I2C_ANA_MST_CTRL_REG	I2C_ANA_MST read/write control register	0x0044	R/W
LP_TEE_LP_APM_CTRL_REG	LP_APM read/write control register	0x004C	varies
LP_TEE_FORCE_ACC_HP_REG	Force access to HP SRAM configuration register	0x0090	R/W
LP_TEE_BUS_ERR_CONF_REG	Error message return configuration register	0x00F0	R/W
Clock Gating Registers			
LP_TEE_CLOCK_GATE_REG	Clock gating register	0x00F8	R/W
Version Control Registers			
LP_TEE_DATE_REG	Version control register	0x00FC	R/W

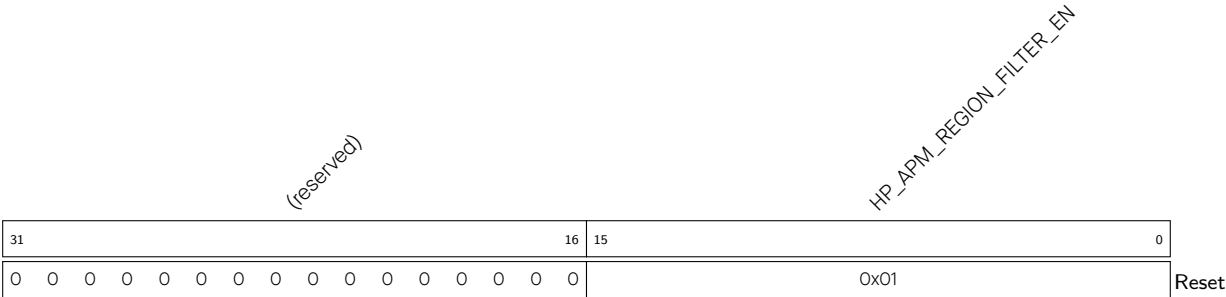
13.9 Registers

13.9.1 HP_APM_REG

The addresses in this section are relative to the HP_APM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

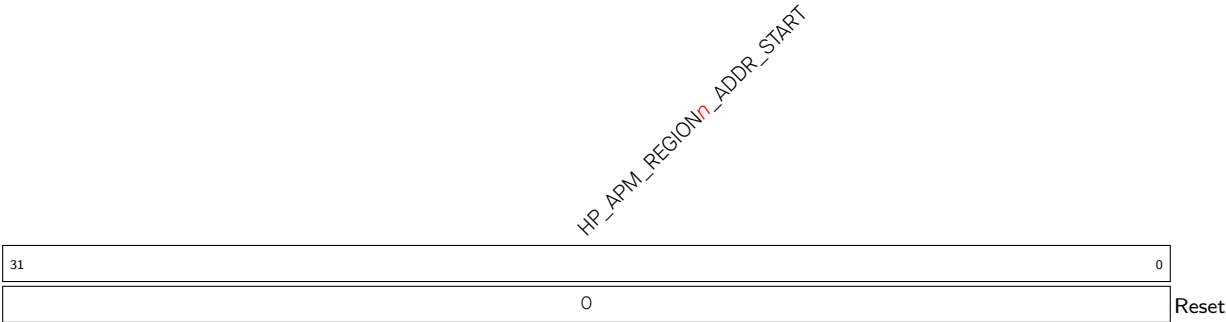
For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 13.1. HP_APM_REGION_FILTER_EN_REG (0x0000)

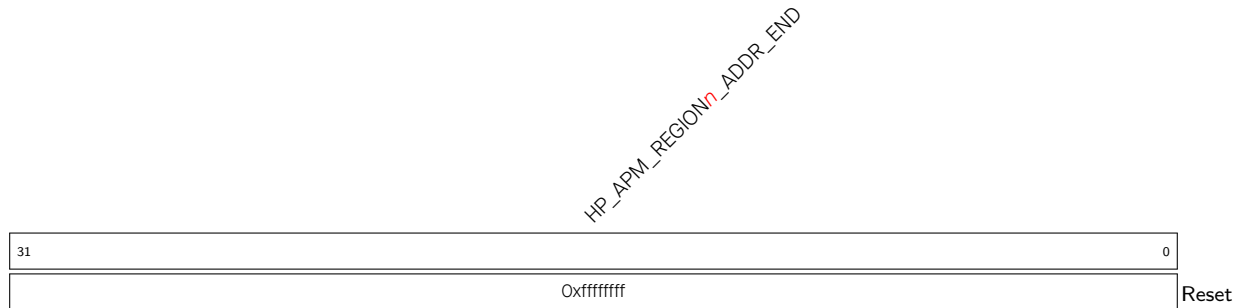


HP_APM_REGION_FILTER_EN Configure bit *n* (0-15) to enable region *n* (0-15).
0: Disable
1: Enable
(R/W)

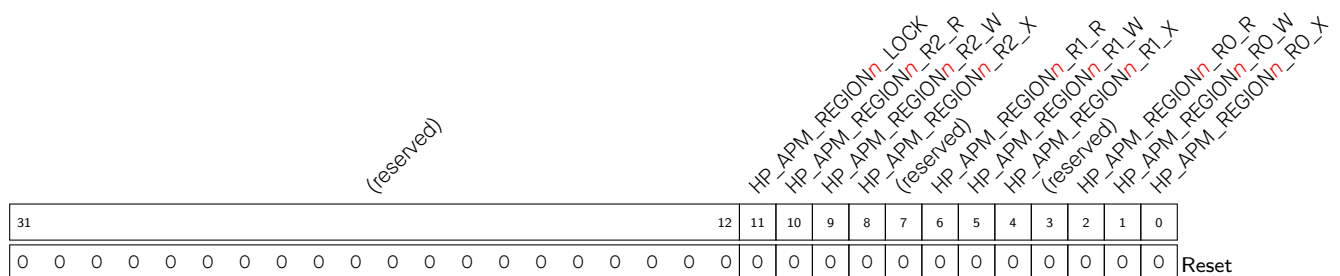
Register 13.2. HP_APM_REGION*n*_ADDR_START_REG (*n*: 0-15) (0x0004+0xC**n*)



HP_APM_REGION*n*_ADDR_START Configures the start address of region *n*. (R/W)

Register 13.3. HP_APM_REGION n _ADDR_END_REG (n : 0-15) (0x0008+0xC* n)

HP_APM_REGION n _ADDR_END Configures the end address of region n . (R/W)

Register 13.4. HP_APM_REGION n _ATTR_REG (n : 0-15) (0x000C+0xC* n)

HP_APM_REGION n _RO_X Configures the execution permission in region n in REE0 mode. (R/W)

HP_APM_REGION n _RO_W Configures the write permission in region n in REE0 mode. (R/W)

HP_APM_REGION n _RO_R Configures the read permission in region n in REE0 mode. (R/W)

HP_APM_REGION n _R1_X Configures the execution permission in region n in REE1 mode. (R/W)

HP_APM_REGION n _R1_W Configures the write permission in region n in REE1 mode. (R/W)

HP_APM_REGION n _R1_R Configures the read permission in region n in REE1 mode. (R/W)

HP_APM_REGION n _R2_X Configures the execution permission in region n in REE2 mode. (R/W)

HP_APM_REGION n _R2_W Configures the write permission in region n in REE2 mode. (R/W)

HP_APM_REGION n _R2_R Configures the read permission in region n in REE2 mode. (R/W)

HP_APM_REGION n _LOCK Configures to lock the value of region n configuration registers (HP_APM_REGION n _ADDR_START_REG, HP_APM_REGION n _ADDR_END_REG and HP_APM_REGION n _ATTR_REG).

0: Do not lock

1: Lock

(R/W)

Register 13.5. HP_APM_FUNC_CTRL_REG (0x00C4)

(reserved)																										HP_APM_M4_FUNC_EN HP_APM_M3_FUNC_EN HP_APM_M2_FUNC_EN HP_APM_M1_FUNC_EN HP_APM_M0_FUNC_EN						
31																										5	4	3	2	1	0	Reset
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																										1	1	1	1	1		

HP_APM_M0_FUNC_EN Configures to enable permission management for HP_APM_CTRL M0.
(R/W)

HP_APM_M1_FUNC_EN Configures to enable permission management for HP_APM_CTRL M1.
(R/W)

HP_APM_M2_FUNC_EN Configures to enable permission management for HP_APM_CTRL M2.
(R/W)

HP_APM_M3_FUNC_EN Configures to enable permission management for HP_APM_CTRL M3.
(R/W)

HP_APM_M4_FUNC_EN Configures to enable permission management for HP_APM_CTRL M4.
(R/W)

Register 13.6. HP_APM_M0_STATUS_REG (0x00C8)

(reserved)																										HP_APM_M0_EXCEPTION_STATUS				
31																										2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

HP_APM_M0_EXCEPTION_STATUS Represents exception status.
bit0: 1 represents permission restrictions
bit1: 1 represents address out of bounds
(RO)

Register 13.7. HP_APM_M0_STATUS_CLR_REG (0x00CC)

[illegible]

HP_APM_MO_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.8. HP_APM_MO_EXCEPTION_INFO0_REG (0x00D0)

(reserved)										HP_APM_MO_EXCEPTION_ID								HP_APM_MO_EXCEPTION_MODE								HP_APM_MO_EXCEPTION_REGION																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31										23										18								17								16								15								0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0										0										0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0								0							

HP_APM_MO_EXCEPTION_REGION Represents the region where an exception occurs. (RO)

HP_APM_MO_EXCEPTION_MODE Represents the master's security mode when an exception occurs. (RO)

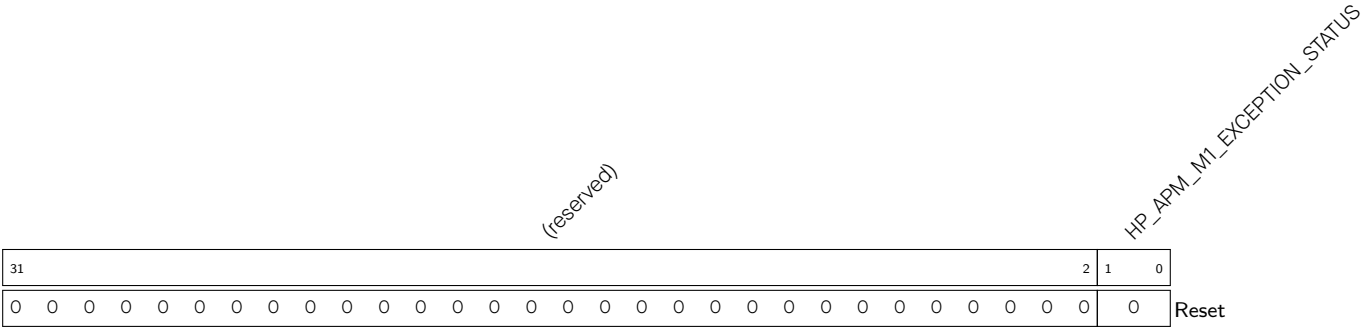
HP_APM_MO_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.9. HP_APM_MO_EXCEPTION_INFO1_REG (0x00D4)

Diagram of the HP-APM_MO_EXCEPTION_ADDR register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The label "HP-APM_MO_EXCEPTION_ADDR" is written diagonally above the register box. The word "Reset" is at the bottom right.

HP_APM_MO_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.10. HP_APM_M1_STATUS_REG (0x00D8)

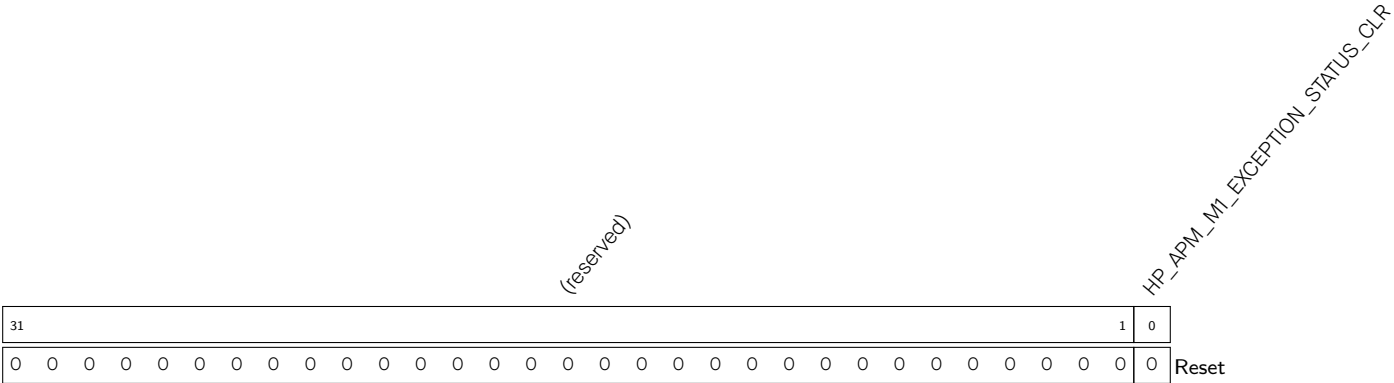


HP_APM_M1_EXCEPTION_STATUS Represents exception status.

bit0: 1 represents permission restrictions

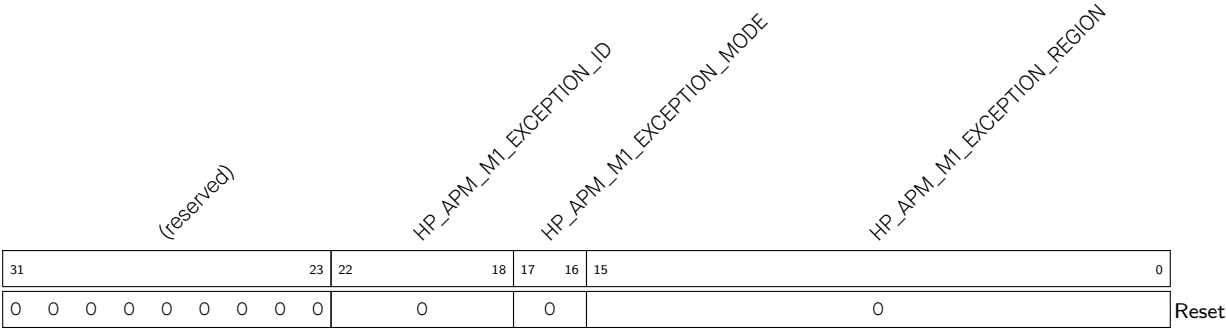
bit1: 1 represents address out of bounds (RO)

Register 13.11. HP_APM_M1_STATUS_CLR_REG (0x00DC)



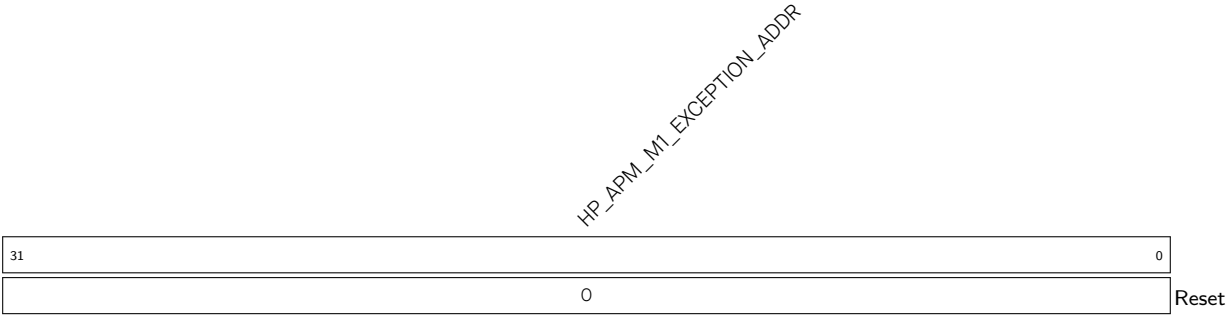
HP_APM_M1_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.12. HP_APM_M1_EXCEPTION_INFO0_REG (0x00E0)



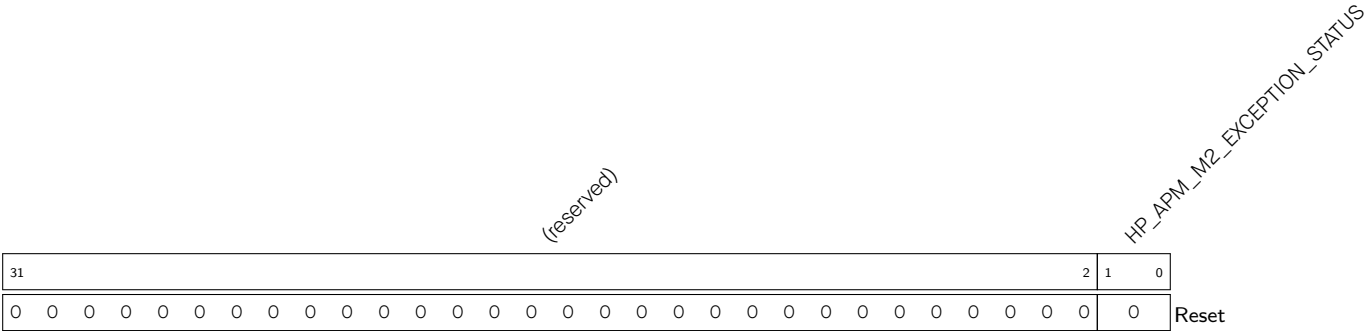
- HP_APM_M1_EXCEPTION_REGION Represents the region where an exception occurs. (RO)
- HP_APM_M1_EXCEPTION_MODE Represents the master’s security mode when an exception occurs. (RO)
- HP_APM_M1_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.13. HP_APM_M1_EXCEPTION_INFO1_REG (0x00E4)



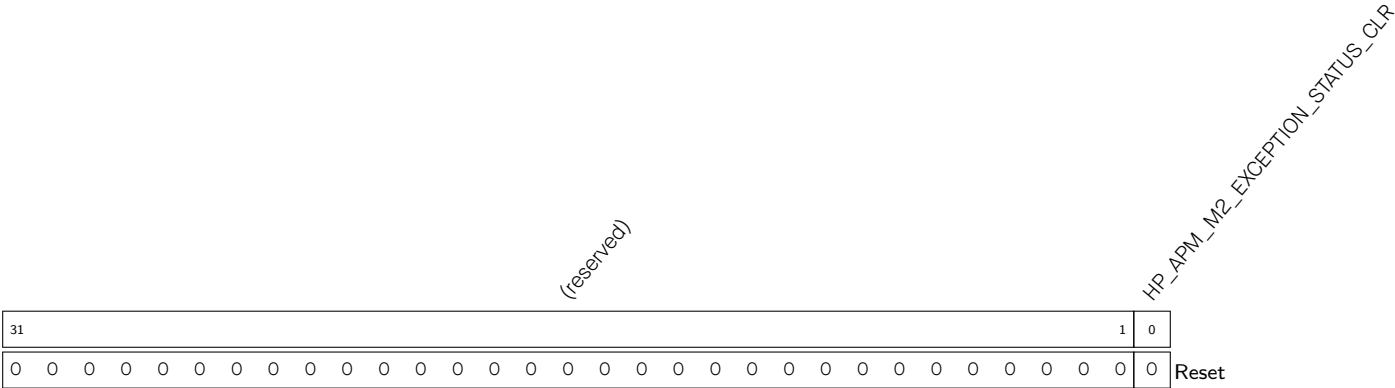
- HP_APM_M1_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.14. HP_APM_M2_STATUS_REG (0x00E8)



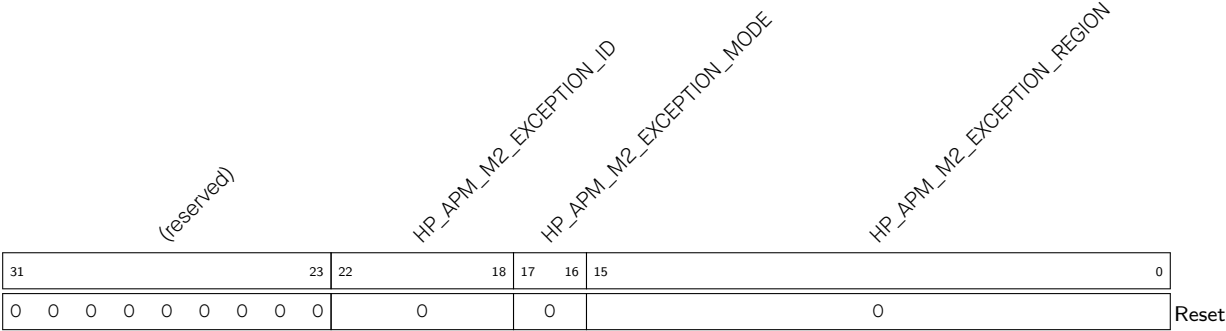
HP_APM_M2_EXCEPTION_STATUS Represents exception status.
bit0: 1 represents permission restrictions
bit1: 1 represents address out of bounds
(RO)

Register 13.15. HP_APM_M2_STATUS_CLR_REG (0x00EC)



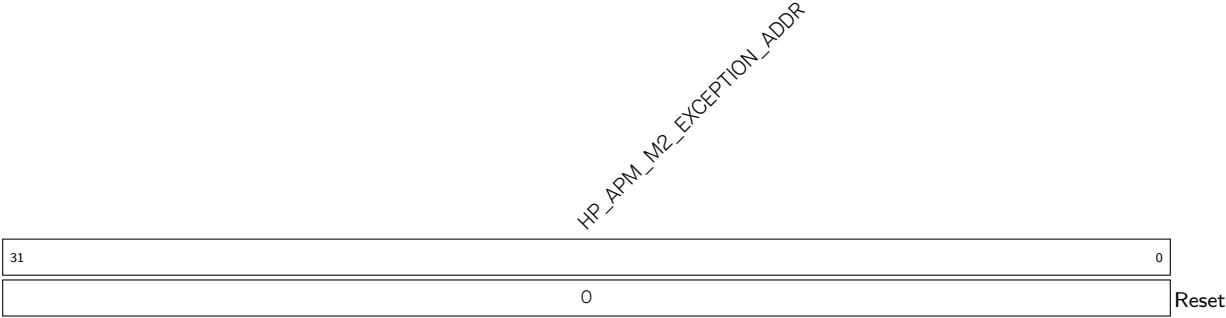
HP_APM_M2_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.16. HP_APM_M2_EXCEPTION_INFO0_REG (0x00F0)



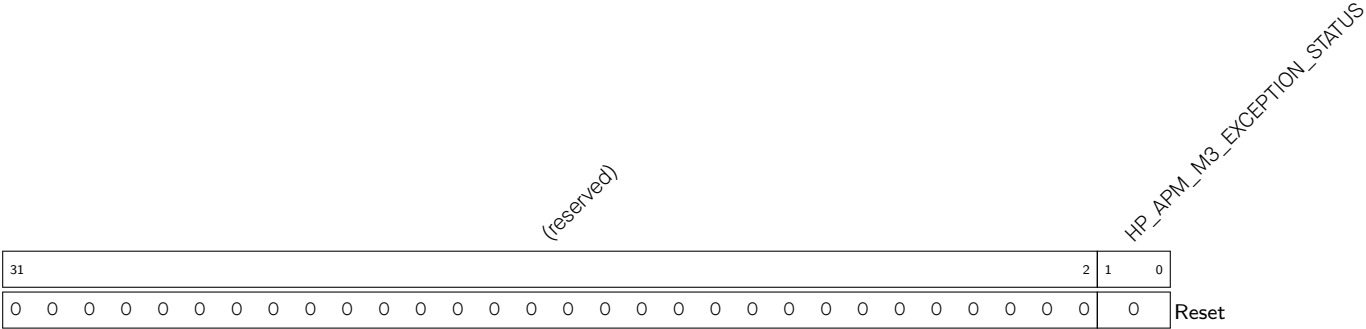
- HP_APM_M2_EXCEPTION_REGION
- Represents the region where an exception occurs. (RO)
- HP_APM_M2_EXCEPTION_MODE
- Represents the master’s security mode when an exception occurs. (RO)
- HP_APM_M2_EXCEPTION_ID
- Represents master ID when an exception occurs. (RO)

Register 13.17. HP_APM_M2_EXCEPTION_INFO1_REG (0x00F4)



- HP_APM_M2_EXCEPTION_ADDR
- Represents the access address when an exception occurs. (RO)

Register 13.18. HP_APM_M3_STATUS_REG (0x00F8)



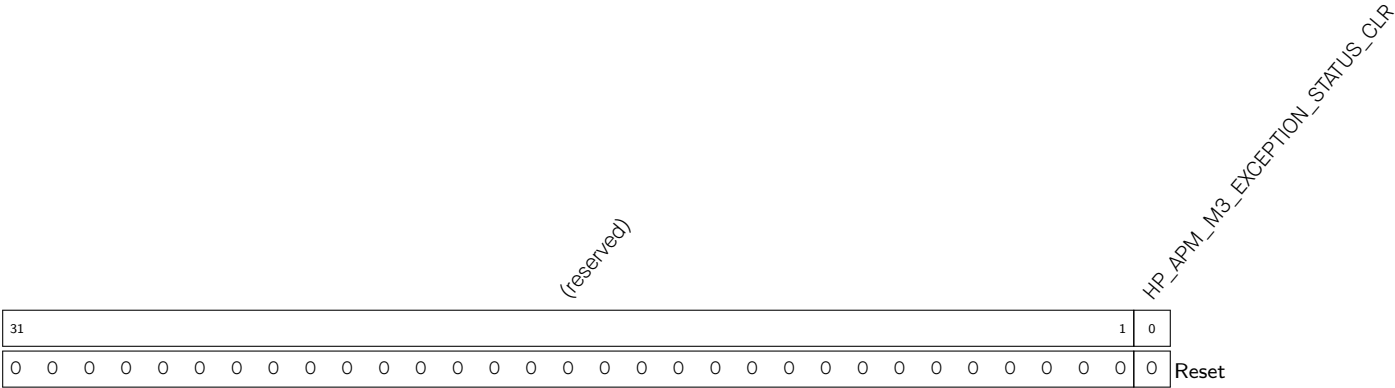
HP_APM_M3_EXCEPTION_STATUS Represents exception status.

bit0: 1 represents permission restrictions

bit1: 1 represents address out of bounds

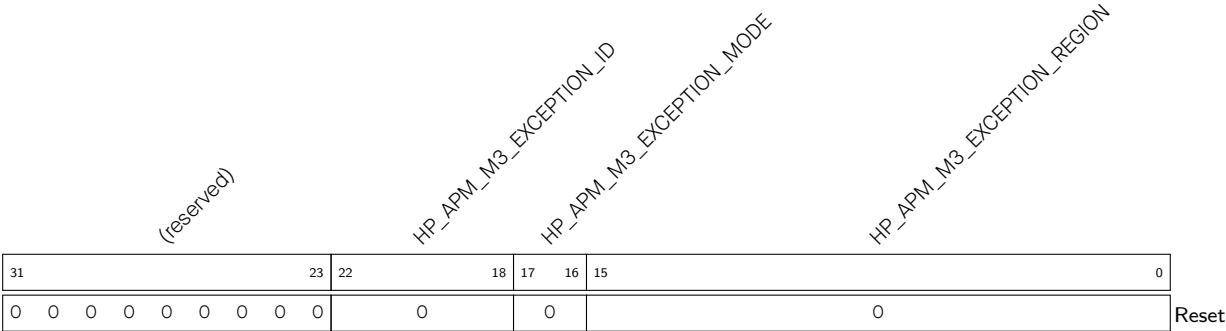
(RO)

Register 13.19. HP_APM_M3_STATUS_CLR_REG (0x00FC)



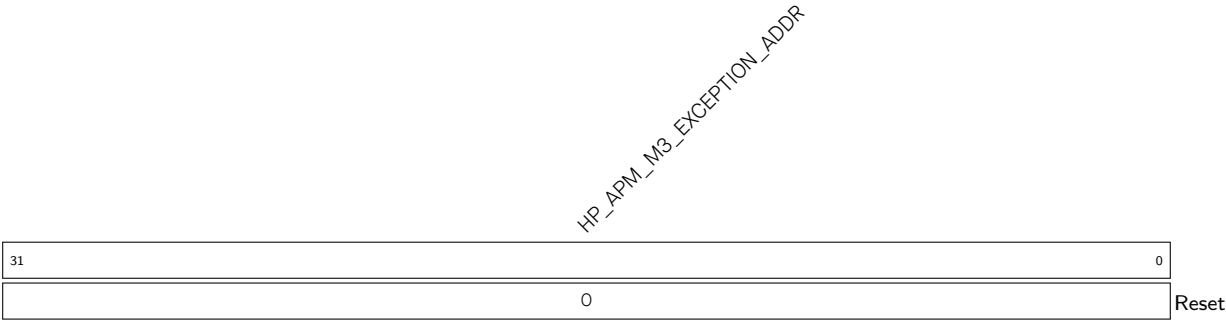
HP_APM_M3_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.20. HP_APM_M3_EXCEPTION_INFO0_REG (0x0100)



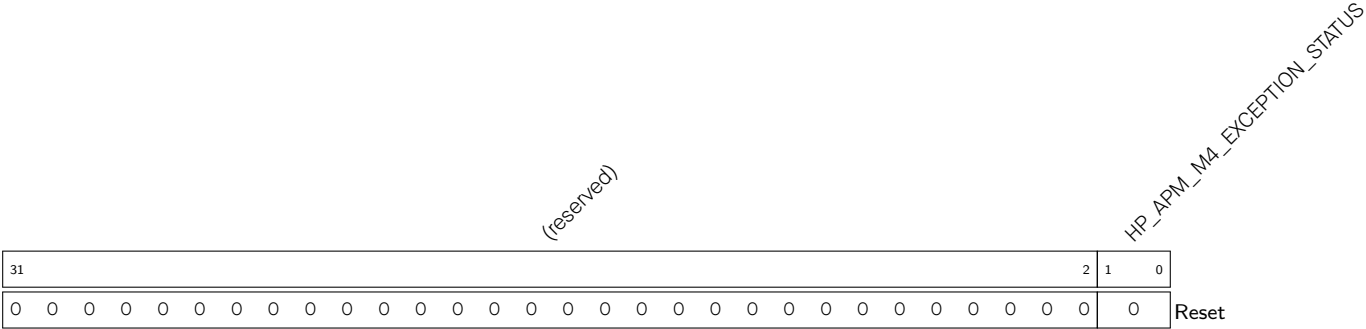
- HP_APM_M3_EXCEPTION_REGION
- Represents the region where an exception occurs. (RO)
- HP_APM_M3_EXCEPTION_MODE
- Represents the master’s security mode when an exception occurs. (RO)
- HP_APM_M3_EXCEPTION_ID
- Represents master ID when an exception occurs. (RO)

Register 13.21. HP_APM_M3_EXCEPTION_INFO1_REG (0x0104)



- HP_APM_M3_EXCEPTION_ADDR
- Represents the access address when an exception occurs. (RO)

Register 13.22. HP_APM_M4_STATUS_REG (0x0108)



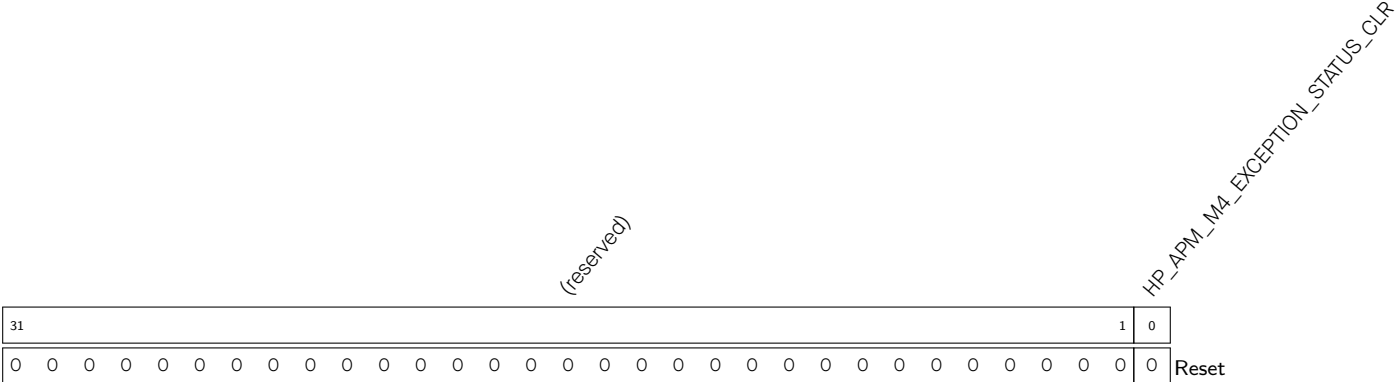
HP_APM_M4_EXCEPTION_STATUS Represents exception status.

bit0: 1 represents permission restrictions

bit1: 1 represents address out of bounds

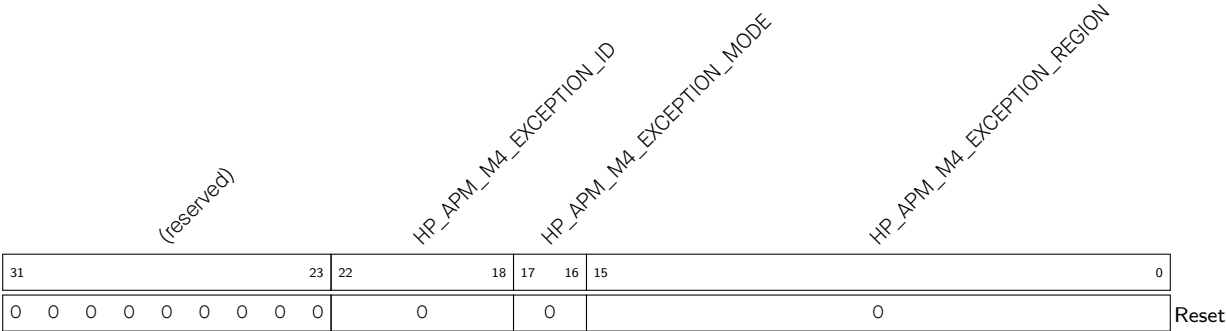
(RO)

Register 13.23. HP_APM_M4_STATUS_CLR_REG (0x010C)



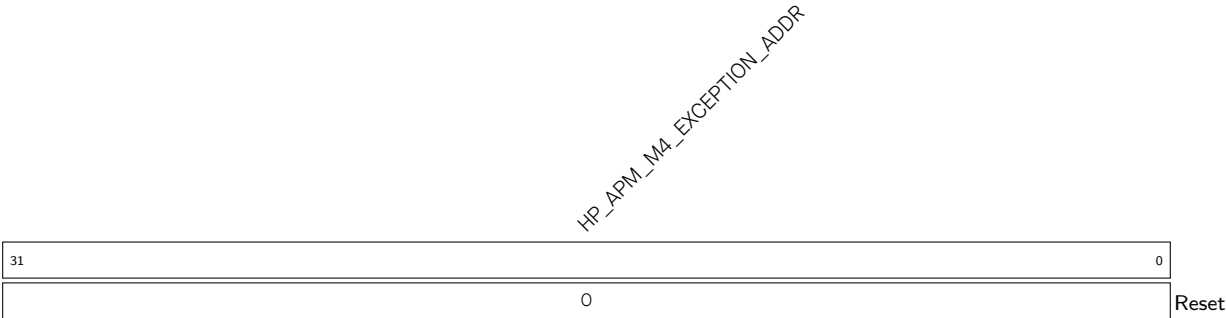
HP_APM_M4_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.24. HP_APM_M4_EXCEPTION_INFO0_REG (0x0110)



- HP_APM_M4_EXCEPTION_REGION Represents the region where an exception occurs. (RO)
- HP_APM_M4_EXCEPTION_MODE Represents the master’s security mode when an exception occurs. (RO)
- HP_APM_M4_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.25. HP_APM_M4_EXCEPTION_INFO1_REG (0x0114)



- HP_APM_M4_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.26. HP_APM_INT_EN_REG (0x0118)

(reserved)																								HP_APM_M4_APM_INT_EN				
																								HP_APM_M3_APM_INT_EN				
																								HP_APM_M2_APM_INT_EN				
																								HP_APM_M1_APM_INT_EN				
																								HP_APM_M0_APM_INT_EN				
31																					5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

HP_APM_M0_APM_INT_EN Configures to enable HP_APM_CTRL M0 interrupt.

0: Disable

1: Enable

(R/W)

HP_APM_M1_APM_INT_EN Configures to enable HP_APM_CTRL M1 interrupt.

0: Disable

1: Enable

(R/W)

HP_APM_M2_APM_INT_EN Configures to enable HP_APM_CTRL M2 interrupt.

0: Disable

1: Enable

(R/W)

HP_APM_M3_APM_INT_EN Configures to enable HP_APM_CTRL M3 interrupt.

0: Disable

1: Enable

(R/W)

HP_APM_M4_APM_INT_EN Configures to enable HP_APM_CTRL M4 interrupt.

0: Disable

1: Enable

(R/W)

Register 13.27. HP_APM_CLOCK_GATE_REG (0x07F8)

(reserved)																												HP_APM_CLK_EN		
31																												1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

HP_APM_CLK_EN Configures whether to keep the clock always on.

0: Enable automatic clock gating

1: Keep the clock always on

(R/W)

Register 13.28. HP_APM_DATE_REG (0x07FC)

(reserved)				HP_APM_DATE																										
31	28	27																												0
0	0	0	0	0x2312010																										

Reset

HP_APM_DATE Version control register. (R/W)

13.9.2 LP_APM_REG

The addresses in this section are relative to the LP_APM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 13.29. LP_APM_REGION_FILTER_EN_REG (0x0000)

(reserved)																LP_APM_REGION_FILTER_EN														
31								8	7																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
																0x1														

Reset

LP_APM_REGION_FILTER_EN Configure bit *n* (0-7) to enable region *n* (0-7).

0: Disable

1: Enable

(R/W)

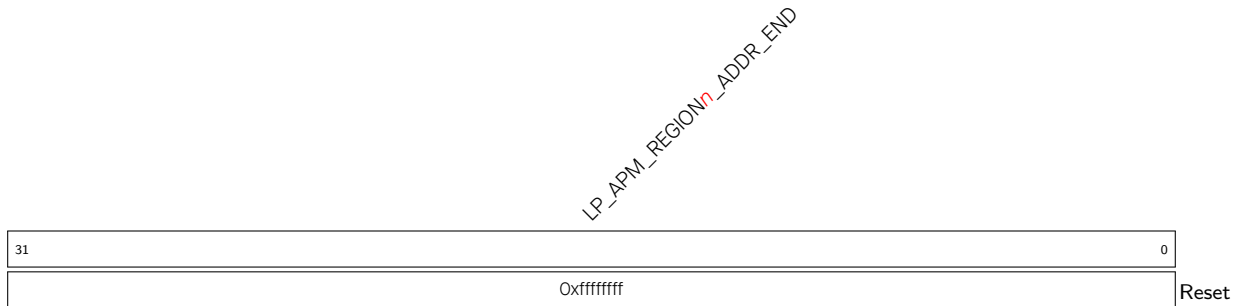
Register 13.30. LP_APM_REGION_{*n*}_ADDR_START_REG (*n*: 0-7) (0x0004+0xC**n*)

LP_APM_REGION _n _ADDR_START																															
31																															0
0																															

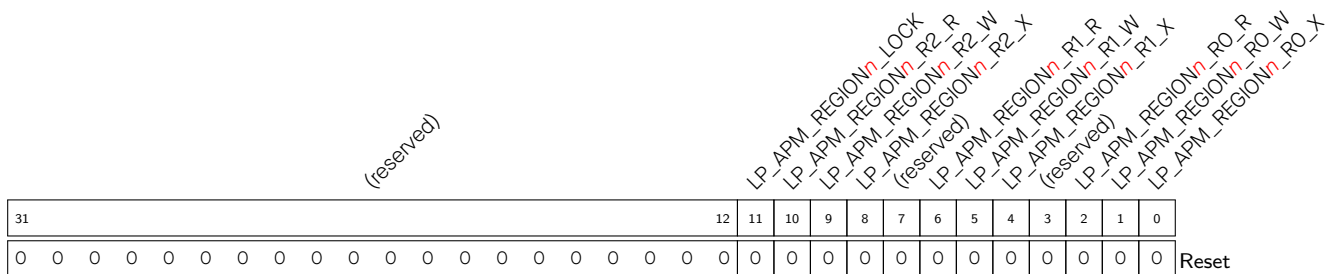
Reset

Reset

LP_APM_REGION_{*n*}_ADDR_START Configures the start address of region *n*. (R/W)

Register 13.31. LP_APM_REGION n _ADDR_END_REG (n : 0-7) (0x0008+0xC* n)

LP_APM_REGION n _ADDR_END Configures the end address of region n . (R/W)

Register 13.32. LP_APM_REGION n _ATTR_REG (n : 0-7) (0x000C+0xC* n)

LP_APM_REGION n _R0_X Configures the execution permission in region n in REE0 mode. (R/W)

LP_APM_REGION n _R0_W Configures the write permission in region n in REE0 mode. (R/W)

LP_APM_REGION n _R0_R Configures the read permission in region n in REE0 mode. (R/W)

LP_APM_REGION n _R1_X Configures the execution permission in region n in REE1 mode. (R/W)

LP_APM_REGION n _R1_W Configures the write permission in region n in REE1 mode. (R/W)

LP_APM_REGION n _R1_R Configures the read permission in region n in REE1 mode. (R/W)

LP_APM_REGION n _R2_X Configures the execution permission in region n in REE1 mode. (R/W)

LP_APM_REGION n _R2_W Configures the write permission in region n in REE2 mode. (R/W)

LP_APM_REGION n _R2_R Configures the read permission in region n in REE2 mode. (R/W)

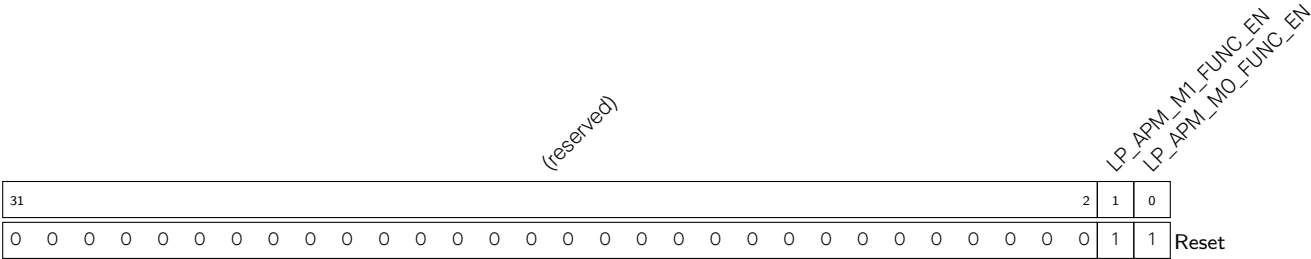
LP_APM_REGION n _LOCK Configures to lock the value of region n configuration registers (LP_APM_REGION n _ADDR_START_REG, LP_APM_REGION n _ADDR_END_REG and LP_APM_REGION n _ATTR_REG).

0: Do not lock

1: Lock

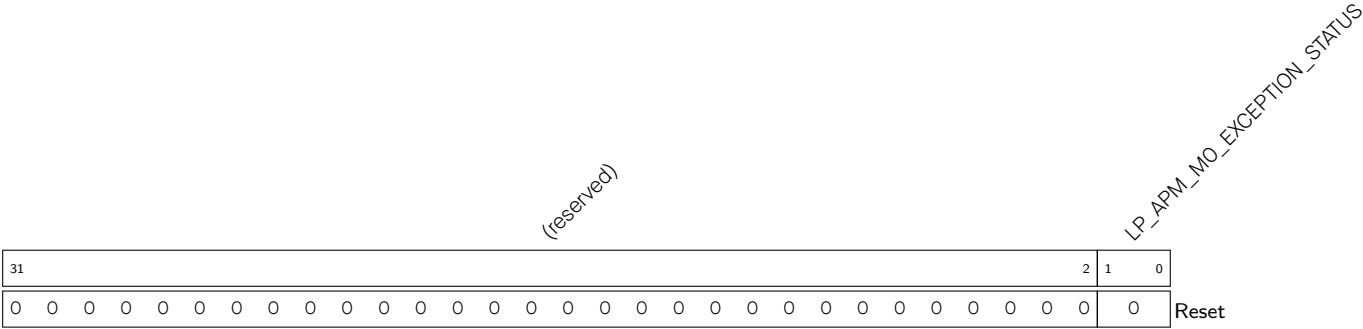
(R/W)

Register 13.33. LP_APM_FUNC_CTRL_REG (0x00C4)



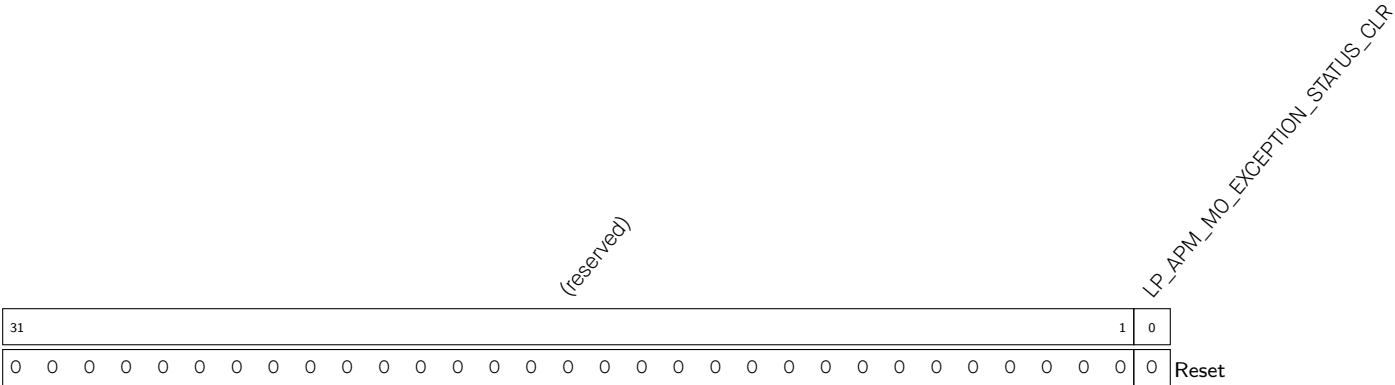
- LP_APM_MO_FUNC_EN Configures to enable permission management for LP_APM_CTRL MO.
(R/W)
- LP_APM_M1_FUNC_EN Configures to enable permission management for LP_APM_CTRL M1.
(R/W)

Register 13.34. LP_APM_MO_STATUS_REG (0x00C8)



- LP_APM_MO_EXCEPTION_STATUS Represents exception status.
bit0: 1 represents permission restrictions
bit1: 1 represents address out of bounds
(RO)

Register 13.35. LP_APM_MO_STATUS_CLR_REG (0x00CC)



- LP_APM_MO_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.36. LP_APM_MO_EXCEPTION_INFO0_REG (0x00D0)

(reserved)								LP_APM_MO_EXCEPTION_ID				(reserved)								LP_APM_MO_EXCEPTION_REGION			
31								23	22	18	17	16	15								4	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

LP_APM_MO_EXCEPTION_REGION Represents the region where an exception occurs. (RO)

LP_APM_MO_EXCEPTION_MODE Represents the master's security mode when an exception occurs. (RO)

LP_APM_MO_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.37. LP_APM_MO_EXCEPTION_INFO1_REG (0x00D4)

LP_APM_MO_EXCEPTION_ADDR																															
31																															0
0																															

Reset

LP_APM_MO_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.38. LP_APM_M1_STATUS_REG (0x00D8)

(reserved)																															LP_APM_M1_EXCEPTION_STATUS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
31																														2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

LP_APM_M1_EXCEPTION_STATUS Represents exception status.

bit0: 1 represents permission restrictions

bit1: 1 represents address out of bounds

(RO)

Register 13.39. LP_APM_M1_STATUS_CLR_REG (0x00DC)

(reserved)																																LP_APM_M1_EXCEPTION_STATUS_CLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																														1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_APM_M1_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.40. LP_APM_M1_EXCEPTION_INFO0_REG (0x00E0)

(reserved)										LP_APM_M1_EXCEPTION_ID							LP_APM_M1_EXCEPTION_MODE							(reserved)							LP_APM_M1_EXCEPTION_REGION																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
31									23	22							18	17	16	15											4	3			0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0			0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_APM_M1_EXCEPTION_REGION Represents the region where an exception occurs. (RO)

LP_APM_M1_EXCEPTION_MODE Represents the master's security mode when an exception occurs. (RO)

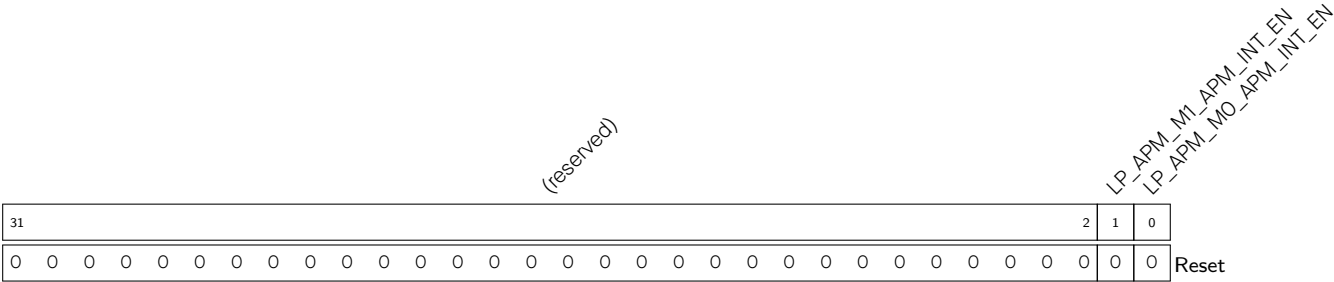
LP_APM_M1_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.41. LP_APM_M1_EXCEPTION_INFO1_REG (0x00E4)

LP_APM_M1_EXCEPTION_ADDR																															
31																															0
0																															
Reset																															

LP_APM_M1_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.42. LP_APM_INT_EN_REG (0x00E8)



LP_APM_M0_APM_INT_EN Configures to enable LP_APM_CTRL M0 interrupt.

0: Disable

1: Enable

(R/W)

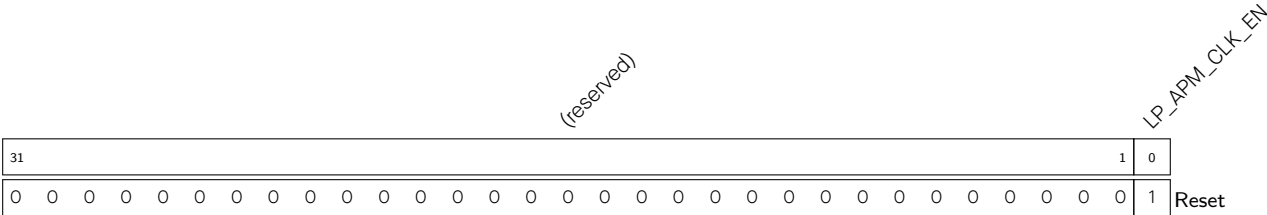
LP_APM_M1_APM_INT_EN Configures to enable LP_APM_CTRL M1 interrupt.

0: Disable

1: Enable

(R/W)

Register 13.43. LP_APM_CLOCK_GATE_REG (0x00EC)



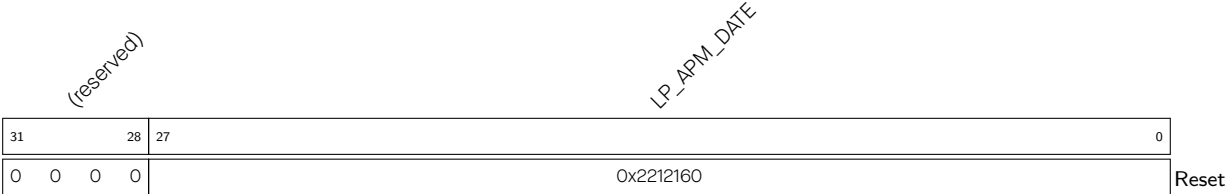
LP_APM_CLK_EN Configures whether to keep the clock always on.

0: Enable automatic clock gating

1: Keep the clock always on

(R/W)

Register 13.44. LP_APM_DATE_REG (0x00FC)



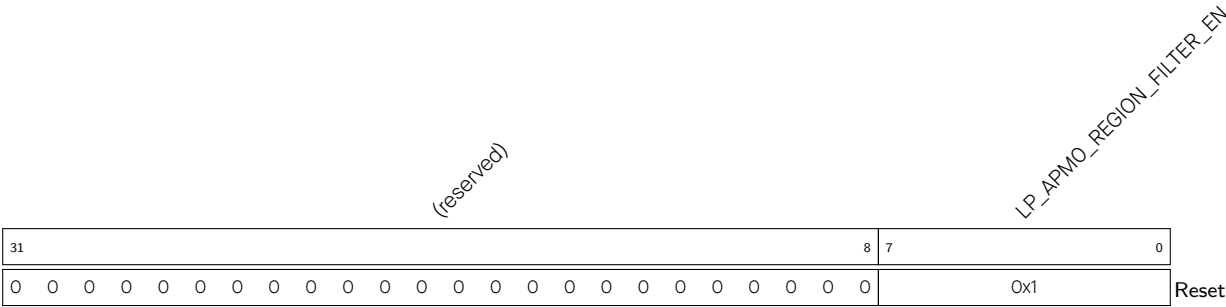
LP_APM_DATE Version control register. (R/W)

13.9.3 LP_APMO_REG

The addresses in this section are relative to the LP_APMO base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

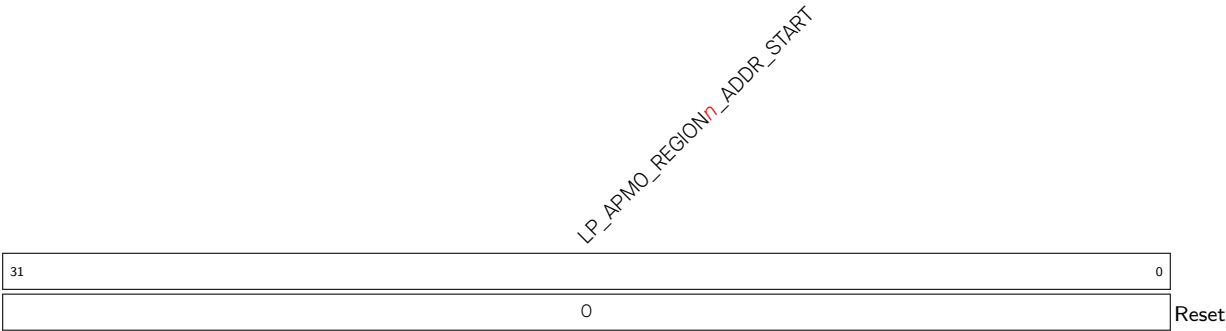
For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 13.45. LP_APMO_REGION_FILTER_EN_REG (0x0000)

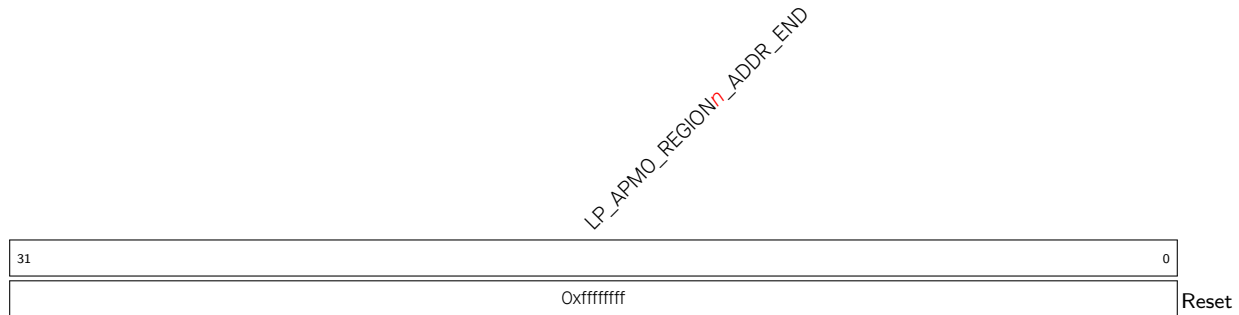


LP_APMO_REGION_FILTER_EN Configure bit *n* (0-7) to enable region *n* (0-7).
0: Disable
1: Enable
(R/W)

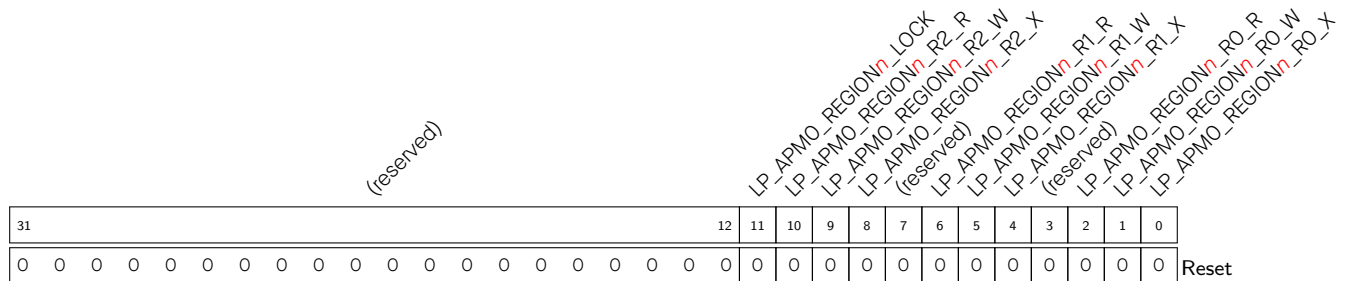
Register 13.46. LP_APMO_REGION*n*_ADDR_START_REG (*n*: 0-7) (0x0004+0xC**n*)



LP_APMO_REGION*n*_ADDR_START Configures the start address of region *n*. (R/W)

Register 13.47. LP_APMO_REGION n _ADDR_END_REG (n : 0-7) (0x0008+0xC* n)

LP_APMO_REGION n _ADDR_END Configures the end address of region n . (R/W)

Register 13.48. LP_APMO_REGION n _ATTR_REG (n : 0-7) (0x000C+0xC* n)

LP_APMO_REGION n _RO_X Configures the execution permission in region n in REEO mode. (R/W)

LP_APMO_REGION n _RO_W Configures the write permission in region n in REEO mode. (R/W)

LP_APMO_REGION n _RO_R Configures the read permission in region n in REEO mode. (R/W)

LP_APMO_REGION n _R1_X Configures the execution permission in region n in REE1 mode. (R/W)

LP_APMO_REGION n _R1_W Configures the write permission in region n in REE1 mode. (R/W)

LP_APMO_REGION n _R1_R Configures the read permission in region n in REE1 mode. (R/W)

LP_APMO_REGION n _R2_X Configures the execution permission in region n in REE2 mode. (R/W)

LP_APMO_REGION n _R2_W Configures the write permission in region n in REE2 mode. (R/W)

LP_APMO_REGION n _R2_R Configures the read permission in region n in REE2 mode. (R/W)

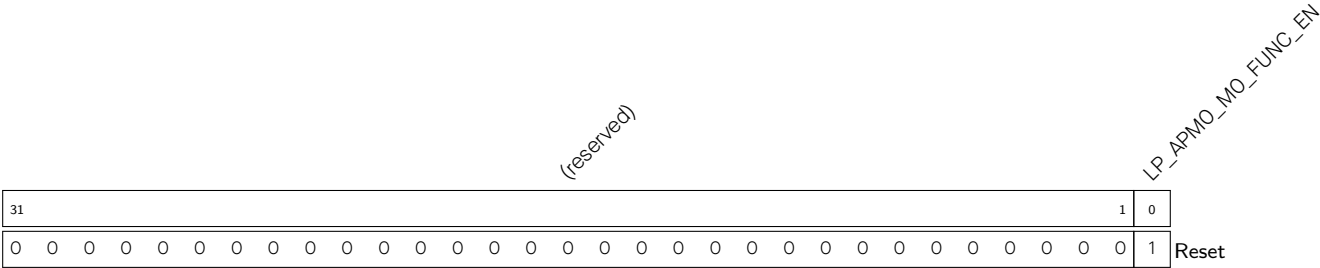
LP_APMO_REGION n _LOCK Configures to lock the value of region n configuration registers (LP_APMO_REGION n _ADDR_START_REG, LP_APMO_REGION n _ADDR_END_REG and LP_APMO_REGION n _ATTR_REG).

0: Do not lock

1: Lock

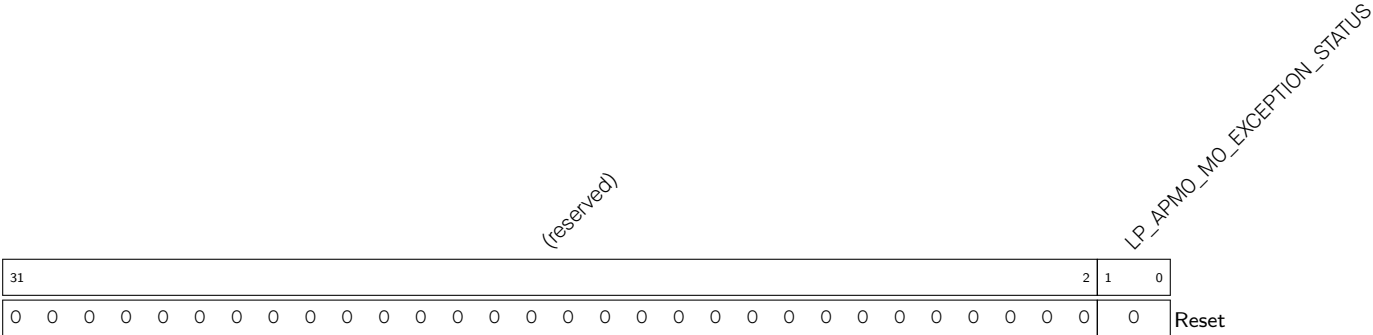
(R/W)

Register 13.49. LP_APMO_FUNC_CTRL_REG (0x00C4)



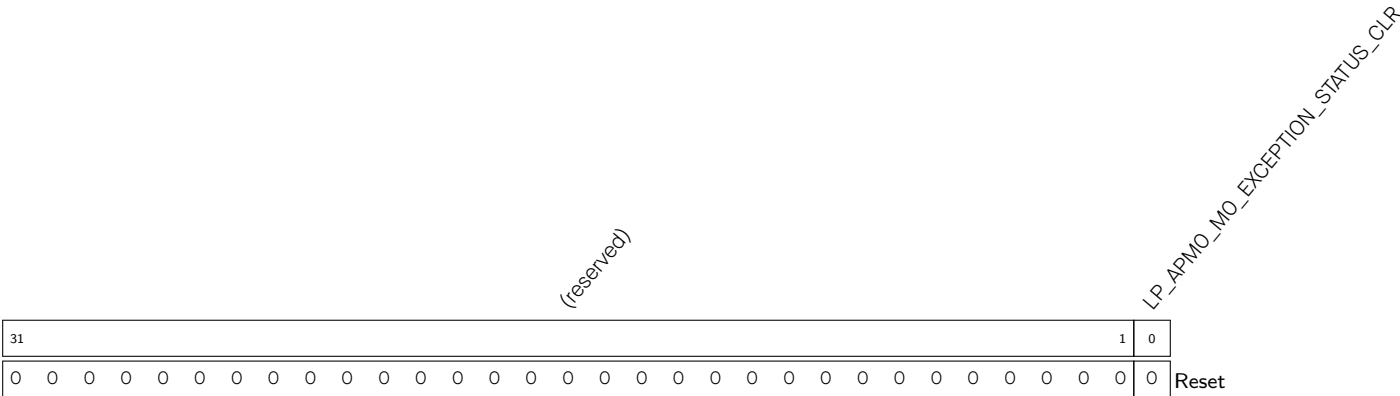
LP_APMO_MO_FUNC_EN Configures to enable permission management for LP_APMO_CTRL MO.
(R/W)

Register 13.50. LP_APMO_MO_STATUS_REG (0x00C8)



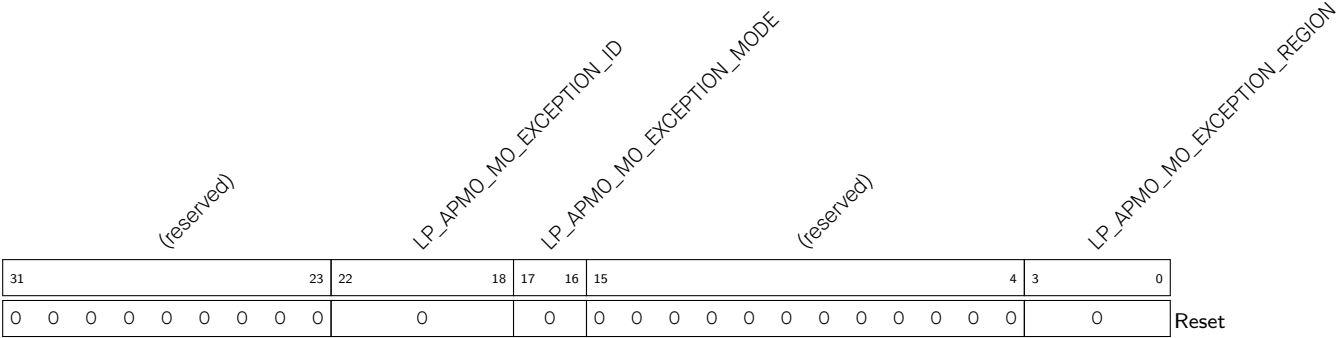
LP_APMO_MO_EXCEPTION_STATUS Represents exception status.
bit0: 1 represents permission restrictions
bit1: 1 represents address out of bounds
(RO)

Register 13.51. LP_APMO_MO_STATUS_CLR_REG (0x00CC)



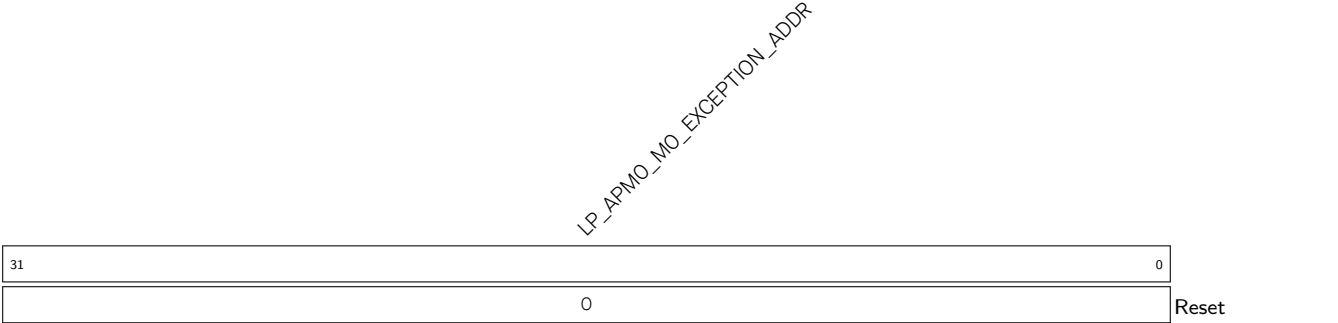
LP_APMO_MO_EXCEPTION_STATUS_CLR Configures to clear exception status. (WT)

Register 13.52. LP_APMO_MO_EXCEPTION_INFO0_REG (0x00D0)



- LP_APMO_MO_EXCEPTION_REGION Represents the region where an exception occurs. (RO)
- LP_APMO_MO_EXCEPTION_MODE Represents the master’s security mode when an exception occurs. (RO)
- LP_APMO_MO_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.53. LP_APMO_MO_EXCEPTION_INFO1_REG (0x00D4)



- LP_APMO_MO_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.54. LP_APMO_INT_EN_REG (0x00D8)

(reserved)																															LP_APMO_MO_APM_INT_EN	
31																															1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															0	Reset

LP_APMO_MO_APM_INT_EN Configures to enable LP_APMO_CTRL MO interrupt.

0: Disable

1: Enable

(R/W)

Register 13.55. LP_APMO_CLOCK_GATE_REG (0x00DC)

(reserved)																												LP_APMO_CLK_EN		
31																												1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

LP_APMO_CLK_EN Configures whether to keep the clock always on.

0: Enable automatic clock gating

1: Keep the clock always on

(R/W)

Register 13.56. LP_APMO_DATE_REG (0x07FC)

(reserved)																												LP_APMO_DATE																											
31				28				27																																0															
0				0				0				0				0x2212160																												Reset											

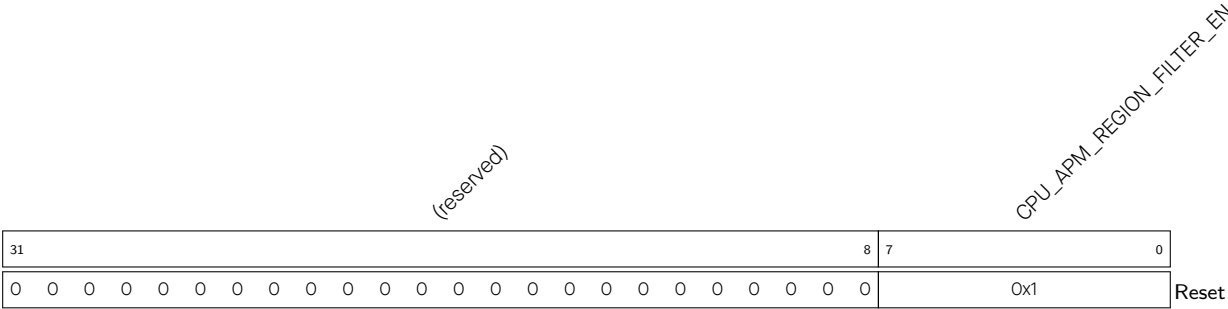
LP_APMO_DATE Version control register. (R/W)

13.9.4 CPU_APM_REG

The addresses in this section are relative to the CPU_APM base address provided in Table 4.3-2 in Chapter 4 [System and Memory](#).

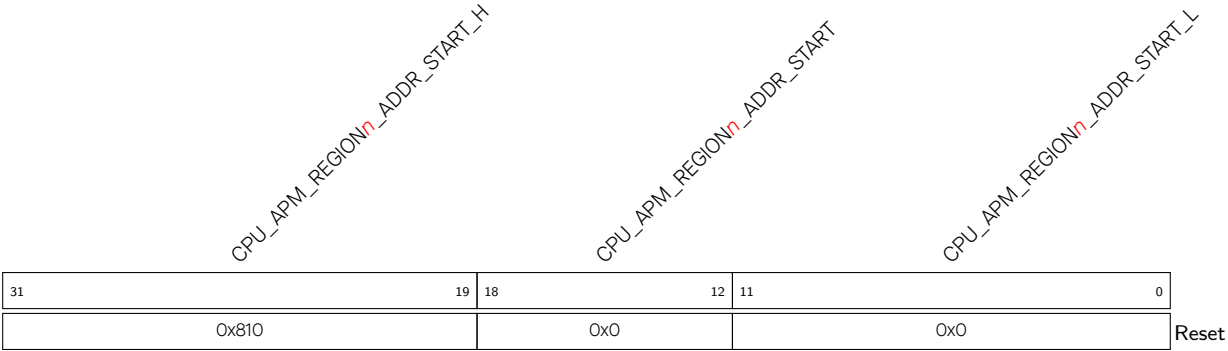
For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 13.57. CPU_APM_REGION_FILTER_EN_REG (0x0000)



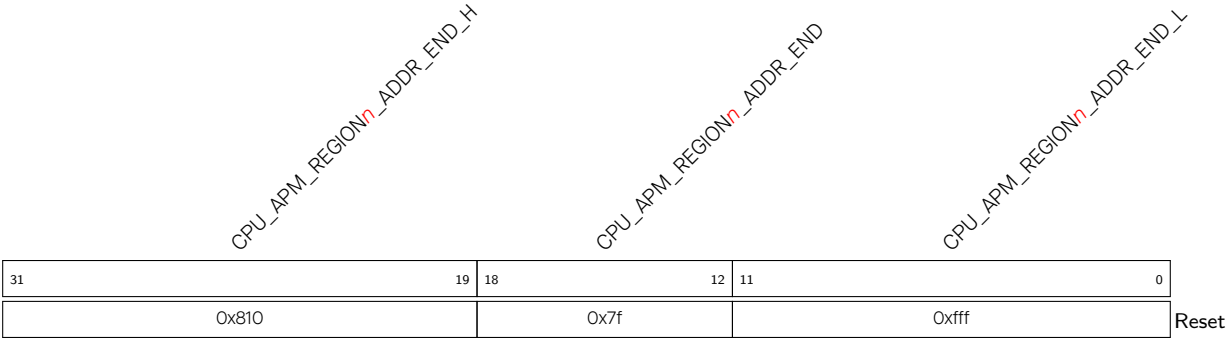
CPU_APM_REGION_FILTER_EN Configures bit *n* (0-7) to enable region *n* (0-7).
0: Disable
1: Enable
(R/W)

Register 13.58. CPU_APM_REGION*n*_ADDR_START_REG (*n*: 0-7) (0x0004+0xC**n*)



CPU_APM_REGION*n*_ADDR_START_L Low 12 bit, start address of region *n*. (HRO)
CPU_APM_REGION*n*_ADDR_START Configures the start address of region *n*. (R/W)
CPU_APM_REGION*n*_ADDR_START_H High 13 bit, start address of region *n*. (HRO)

Register 13.59. CPU_APM_REGION n _ADDR_END_REG (n : 0-7) (0x0008+0xC* n)



CPU_APM_REGION n _ADDR_END_L Low 12 bit, end address of region n . (HRO)

CPU_APM_REGION n _ADDR_END Configures the end address of region n . (R/W)

CPU_APM_REGION n _ADDR_END_H High 13 bit, end address of region n . (HRO)

Register 13.60. CPU_APM_REGION n _ATTR_REG (n : 0-7) (0x000C+0xC* n)

(reserved)																CPU_APM_REGION _n _LOCK CPU_APM_REGION _n _R2_R CPU_APM_REGION _n _R2_W (reserved) CPU_APM_REGION _n _R2_X CPU_APM_REGION _n _R1_R CPU_APM_REGION _n _R1_W (reserved) CPU_APM_REGION _n _R0_R CPU_APM_REGION _n _R0_W CPU_APM_REGION _n _R0_X																												
31																12																11	10	9	8	7	6	5	4	3	2	1	0	Reset
0																0																0	0	0	0	0	0	0	0	0	0	0	0	

CPU_APM_REGION n _R0_X Configures the execution permission in region n in REE0 mode. (R/W)

CPU_APM_REGION n _R0_W Configures the write permission in region n in REE0 mode. (R/W)

CPU_APM_REGION n _R0_R Configures the read permission in region n in REE0 mode. (R/W)

CPU_APM_REGION n _R1_X Configures the execution permission in region n in REE1 mode. (R/W)

CPU_APM_REGION n _R1_W Configures the write permission in region n in REE1 mode. (R/W)

CPU_APM_REGION n _R1_R Configures the read permission in region n in REE1 mode. (R/W)

CPU_APM_REGION n _R2_X Configures the execution permission in region n in REE2 mode. (R/W)

CPU_APM_REGION n _R2_W Configures the write permission in region n in REE2 mode. (R/W)

CPU_APM_REGION n _R2_R Configures the read permission in region n in REE2 mode. (R/W)

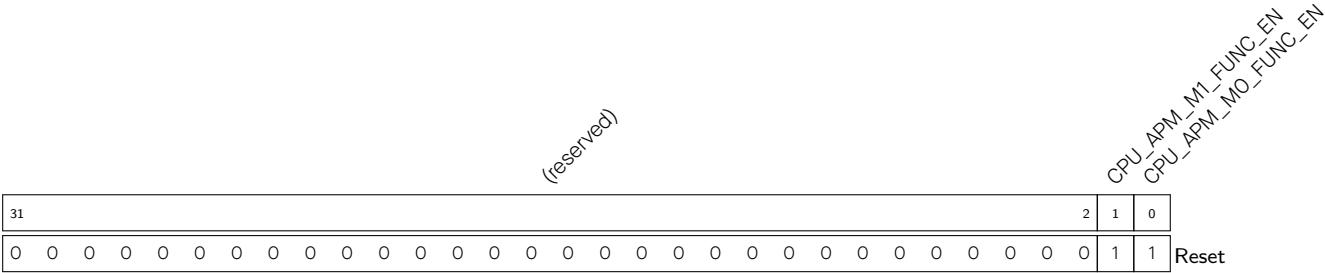
CPU_APM_REGION n _LOCK Configures to lock the value of region n 's configuration registers ([CPU_APM_REGION \$n\$ _ADDR_START_REG](#), [CPU_APM_REGION \$n\$ _ADDR_END_REG](#), and [CPU_APM_REGION \$n\$ _ATTR_REG](#)).

0: Do not lock

1: Lock

(R/W)

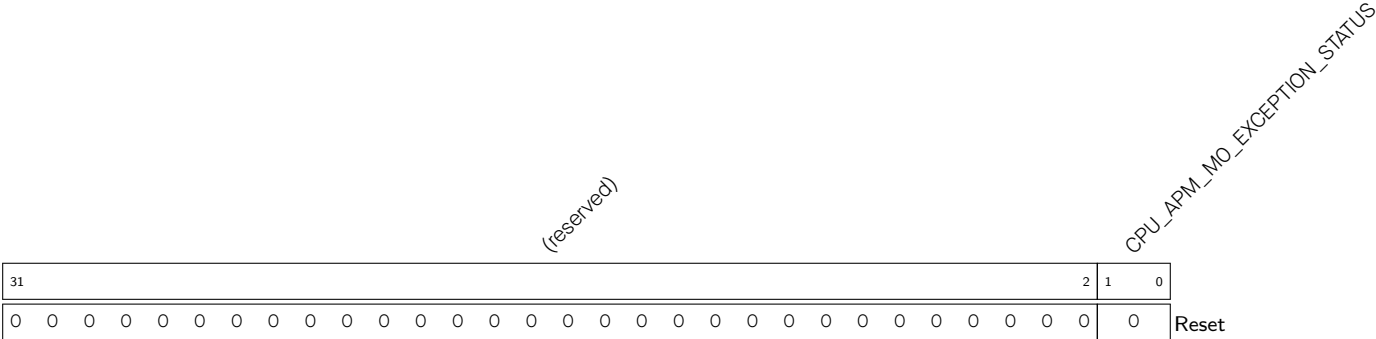
Register 13.61. CPU_APM_FUNC_CTRL_REG (0x00C4)



CPU_APM_MO_FUNC_EN Configures whether to enable permission management for CPU_APM_CTRL M0.
0: Disable
1: Enable
(R/W)

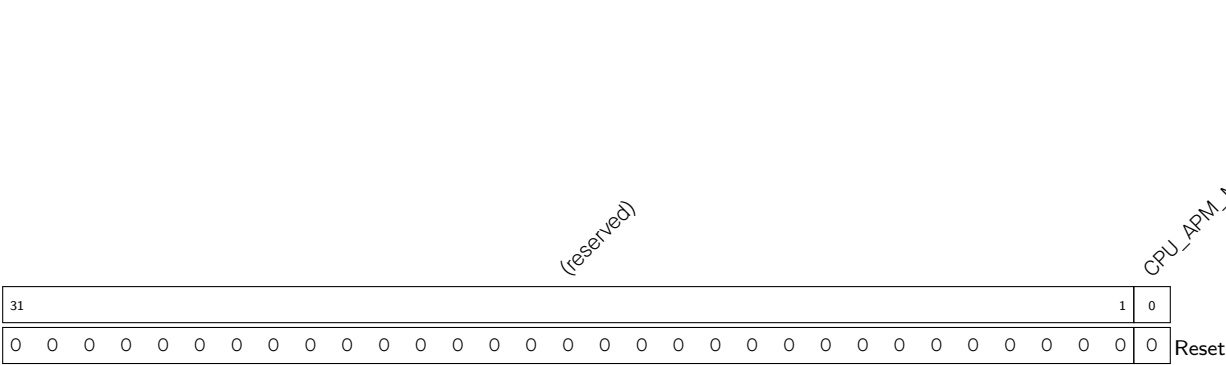
CPU_APM_M1_FUNC_EN Configures whether to enable permission management for CPU_APM_CTRL M1.
0: Disable
1: Enable
(R/W)

Register 13.62. CPU_APM_MO_STATUS_REG (0x00C8)



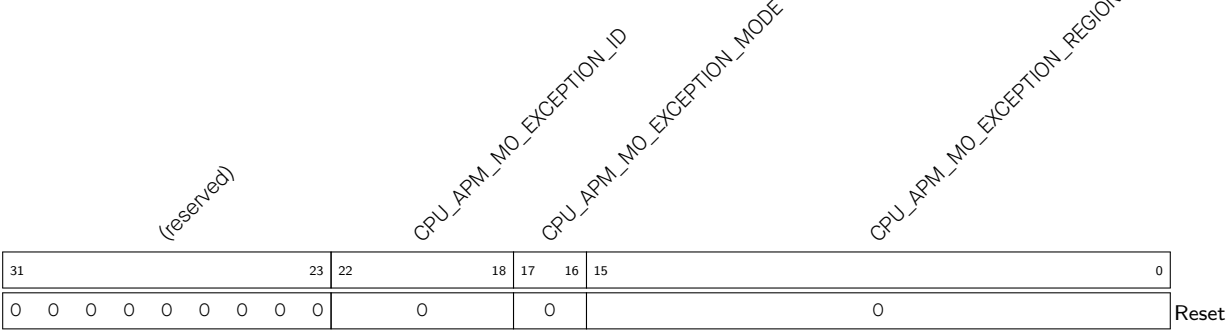
CPU_APM_MO_EXCEPTION_STATUS Represents exception status.
bit0: 1 represents permission restrictions
bit1: 1 represents address out of bounds
(RO)

Register 13.63. CPU_APM_MO_STATUS_CLR_REG (0x00CC)



CPU_APM_MO_EXCEPTION_STATUS_CLR Write 1 to clear exception status. (WT)

Register 13.64. CPU_APM_MO_EXCEPTION_INFO0_REG (0x00D0)

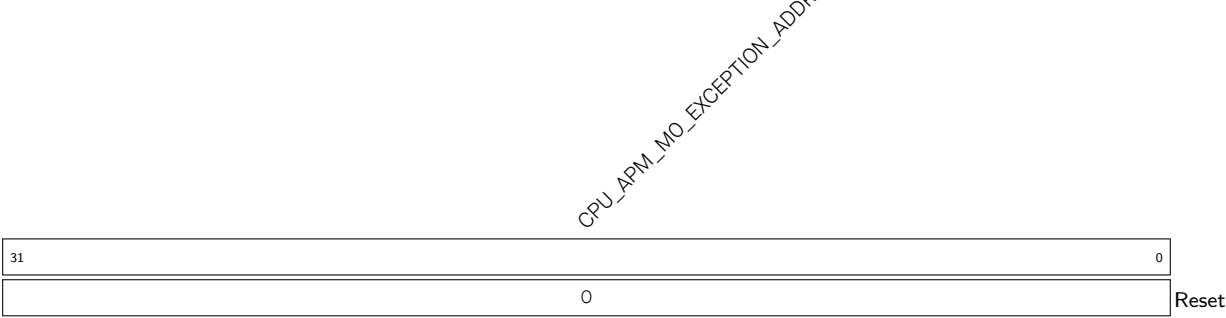


CPU_APM_MO_EXCEPTION_REGION Represents the region where an exception occurs. (RO)

CPU_APM_MO_EXCEPTION_MODE Represents the master's security mode when an exception occurs. (RO)

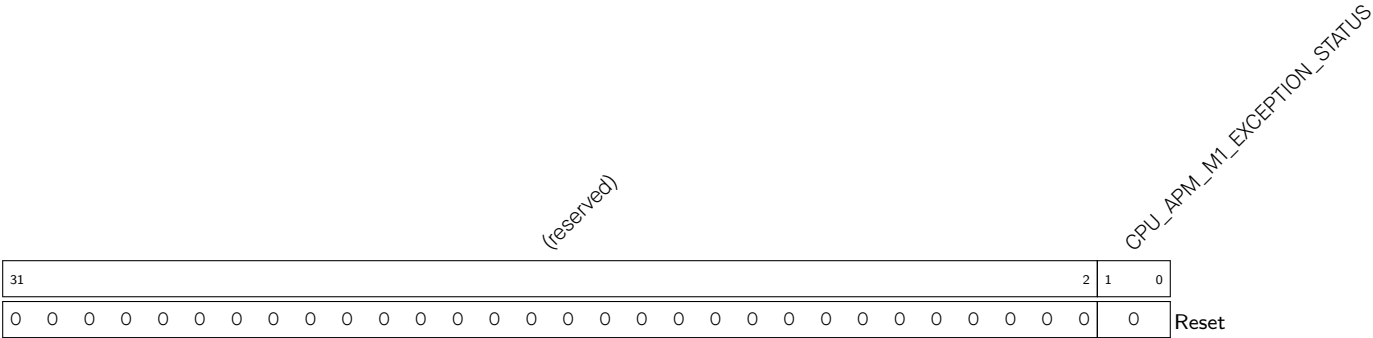
CPU_APM_MO_EXCEPTION_ID Represents master ID when an exception occurs. (RO)

Register 13.65. CPU_APM_MO_EXCEPTION_INFO1_REG (0x00D4)



CPU_APM_MO_EXCEPTION_ADDR Represents the access address when an exception occurs. (RO)

Register 13.66. CPU_APM_M1_STATUS_REG (0x00D8)

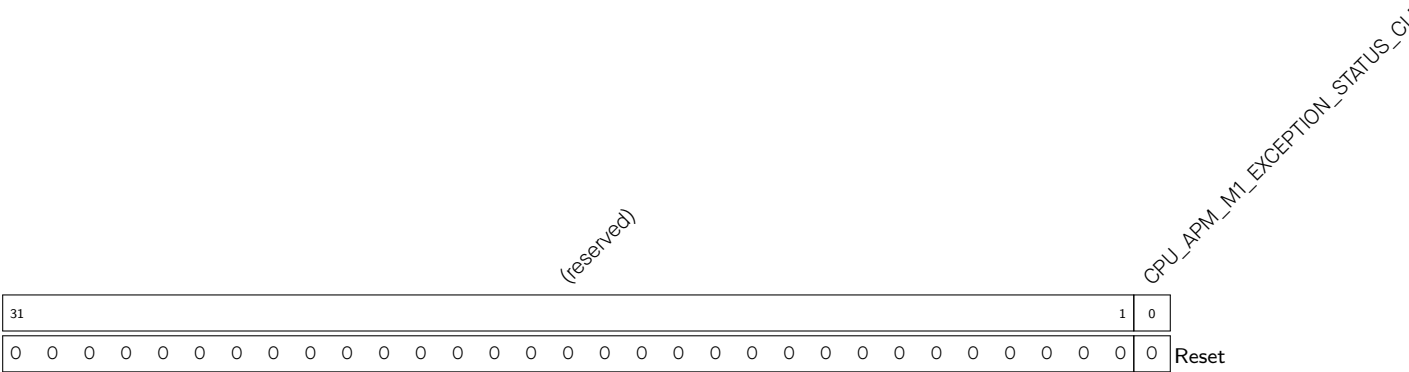


CPU_APM_M1_EXCEPTION_STATUS Represents exception status.

bit0: 1 represents permission restrictions

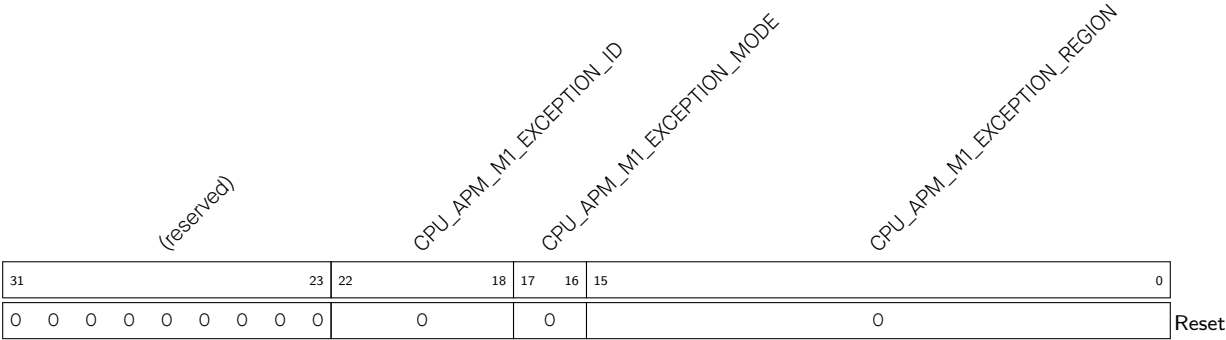
bit1: 1 represents address out of bounds (RO)

Register 13.67. CPU_APM_M1_STATUS_CLR_REG (0x00DC)



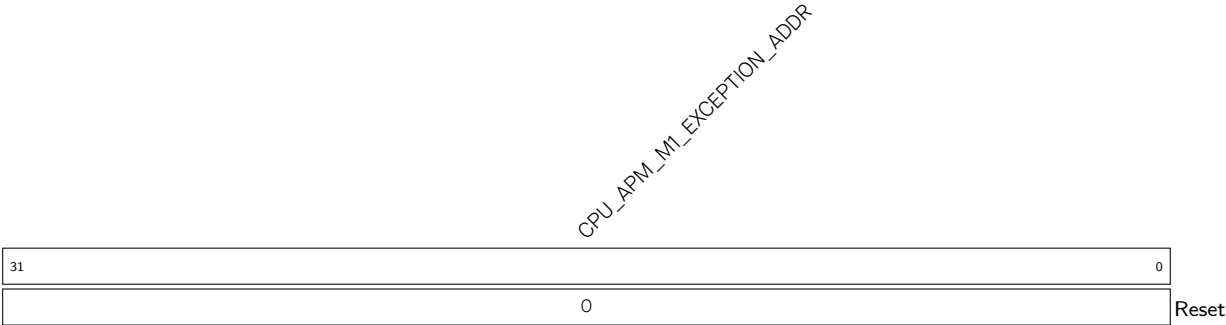
CPU_APM_M1_EXCEPTION_STATUS_CLR Write 1 to clear exception status. (WT)

Register 13.68. CPU_APM_M1_EXCEPTION_INFO0_REG (0x00E0)



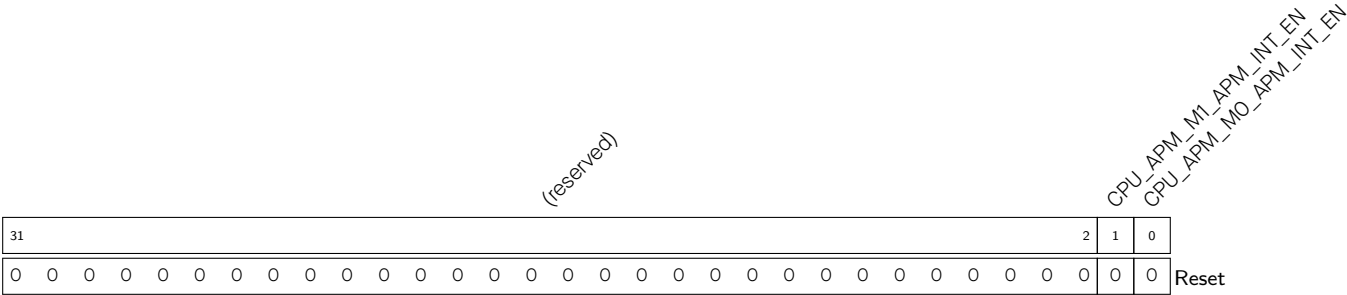
- CPU_APM_M1_EXCEPTION_REGION** Represents the region where an exception occurs. (RO)
- CPU_APM_M1_EXCEPTION_MODE** Represents the master’s security mode when an exception occurs. (RO)
- CPU_APM_M1_EXCEPTION_ID** Represents master ID when an exception occurs. (RO)

Register 13.69. CPU_APM_M1_EXCEPTION_INFO1_REG (0x00E4)



- CPU_APM_M1_EXCEPTION_ADDR** Represents the access address when an exception occurs. (RO)

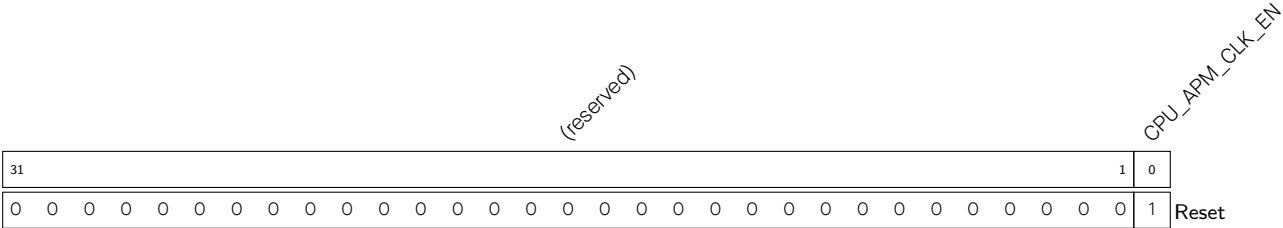
Register 13.70. CPU_APM_INT_EN_REG (0x0118)



CPU_APM_MO_APM_INT_EN Configures whether to enable CPU_APM_CTRL M0 interrupt.
0: Disable
1: Enable
(R/W)

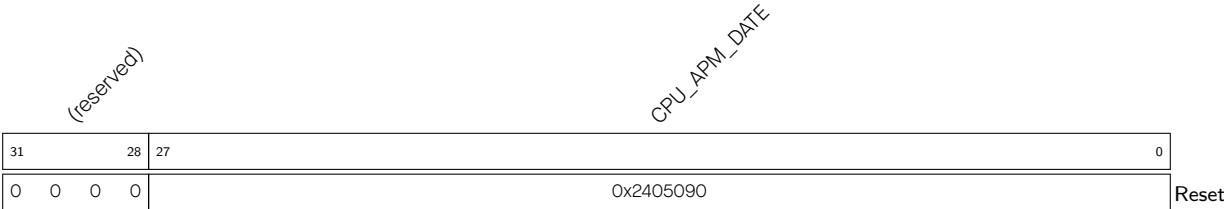
CPU_APM_M1_APM_INT_EN Configures whether to enable CPU_APM_CTRL M1 interrupt.
0: Disable
1: Enable
(R/W)

Register 13.71. CPU_APM_CLOCK_GATE_REG (0x07F8)



CPU_APM_CLK_EN Configures whether to keep the clock always on.
0: Enable automatic clock gating
1: Keep the clock always on
(R/W)

Register 13.72. CPU_APM_DATE_REG (0x07FC)



CPU_APM_DATE Version control register. (R/W)

13.9.5 TEE_REG

The addresses in this section are relative to the TEE base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 13.73. TEE_M n _MODE_CTRL_REG (n : 0-31) (0x0000+0x4* n)

(reserved)																																TEE_Mn_LOCK		TEE_Mn_MODE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

TEE_M n _MODE Configures the security mode for master n .

- 0: TEE
- 1: REEO
- 2: REE1
- 3: REE2
- (R/W)

TEE_M n _LOCK Configures whether to lock the value of TEE_M n _MODE.

- 0: Do not lock
- 1: Lock
- (R/W)

Register 13.74. TEE_UART0_CTRL_REG (0x0088)

Register 13.75. TEE_UART1_CTRL_REG (0x008C)

Register 13.76. TEE_UHCIO_CTRL_REG (0x0090)

Register 13.77. TEE_I2C_EXT0_CTRL_REG (0x0094)

Register 13.78. TEE_I2S_CTRL_REG (0x009C)

Register 13.79. TEE_PARL_IO_CTRL_REG (0x00A0)

Register 13.80. TEE_PWM_CTRL_REG (0x00A4)

Register 13.81. TEE_LEDC_CTRL_REG (0x00AC)

Register 13.82. TEE_TWAIO_CTRL_REG (0x00B0)

Register 13.83. TEE_USB_SERIAL_JTAG_CTRL_REG (0x00B4)

Register 13.84. TEE_RMT_CTRL_REG (0x00B8)

Register 13.85. TEE_GDMA_CTRL_REG (0x00BC)

Register 13.86. TEE_ETM_CTRL_REG (0x00C4)

Register 13.87. TEE_INTMTX_CTRL_REG (0x00C8)

Register 13.88. TEE_APB_ADC_CTRL_REG (0x00D0)

Register 13.110. TEE_BUS_ERR_CONF_REG (0x0FF0)

(reserved)																															TEE_BUS_ERR_RESP_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

TEE_BUS_ERR_RESP_EN Configures whether to return error message to CPU when access is blocked.

0: Disable

1: Enable

(R/W)

Register 13.111. TEE_CLOCK_GATE_REG (0x0FF8)

(reserved)																										TEE_CLK_EN				
31																										1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

Reset

TEE_CLK_EN Configures whether to keep the clock always on.

0: Enable automatic clock gating

1: Keep the clock always on

(R/W)

Register 13.112. TEE_DATE_REG (0x0FFC)

(reserved)				TEE_DATE																										
31	28	27																										0		
0	0	0	0	0x2406200																										Reset

Reset

TEE_DATE Version control register. (R/W)

13.9.6 LP_TEE_REG

The addresses in this section are relative to the LP_TEE base address provided in Table 4.3-2 in Chapter 4 [System and Memory](#).

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 13.113. LP_TEE_MO_MODE_CTRL_REG (0x0000)

The diagram shows two adjacent 32-bit registers. The left register is labeled '(reserved)' and has bits 31 down to 0. The right register is split into two fields: 'LP_TEE_MO_LOOK' (bits 3 down to 2) and 'LP_TEE_MO_MODE' (bits 1 down to 0). Below the registers, bit values are shown: the first 31 bits are all 0; bit 3 is 0; bit 2 is 3; bit 1 is 3; and bit 0 is Reset.

LP_TEE_MO_MODE Configures the security mode for LP CPU.

O: TEE

1: REEO

2: REE1

3: REE2

(R/W)

LP_TEE_M0_LOCK Configures to lock the value of **LP_TEE_M0_MODE**.

0: Do not lock

1: Lock

(R/W)

Register 13.114. LP_TEE_EFUSE_CTRL_REG (0x0004)

Register 13.115. LP_TEE_PMU_CTRL_REG (0x0008)

Register 13.116. LP_TEE_CLKRST_CTRL_REG (0x000C)

Register 13.117. LP TEE LP AON CTRL CTRL REG (0x0010)

Register 13.118. LP_TEE_LP_TIMER_CTRL_REG (0x0014)

Register 13.119. LP_TEE_LP_WDT_CTRL_REG (0x0018)

Register 13.120. LP_TEE_LP_PERI_CTRL_REG (0x001C)

Register 13.121. LP_TEE_LP_ANA_PERI_CTRL_REG (0x0020)

Register 13.122. LP TEE LP IO CTRL REG (0x002C)

Register 13.123. LP_TEE_LP_TEE_CTRL_REG (0x0034)

Register 13.124. LP_TEE_UART_CTRL_REG (0x0038)

Register 13.125. LP_TEE_I2C_EXT_CTRL_REG (0x0040)

Register 13.126. LP_TEE_I2C_ANA_MST_CTRL_REG (0x0044)

Register 13.127. LP TEE LP APM CTRL REG (0x004C)

Register 13.128. LP_TEE_ *PERI*_CTRL_REG (0x0004-0x004C)

[illegible]

LP_TEE_READ_TEE_*PERI* Configures the read permission of *PERI* in TEE mode. (R/W)

LP_TEE_READ_REEO_*PERI* Configures the read permission of *PERI* in REEO mode. (R/W)

LP_TEE_READ_REE1_*PERI* Configures the read permission of *PERI* in REE1 mode. (R/W)

LP_TEE_READ_REE2_PERI Configures the read permission of *PERI* in REE2 mode. (R/W)

LP_TEE_WRITE_TEE_*PERI* Configures the write permission of *PERI* in TEE mode. (R/W)

LP_TEE_WRITE_REEO_*PERI* Configures the write permission of *PERI* in REEO mode. (R/W)

LP_TEE_WRITE_REE1_*PERI* Configures the write permission of *PERI* in REE1 mode. (R/W)

LP_TEE_WRITE_REE2_*PERI* Configures the write permission of *PERI* in REE2 mode. (R/W)

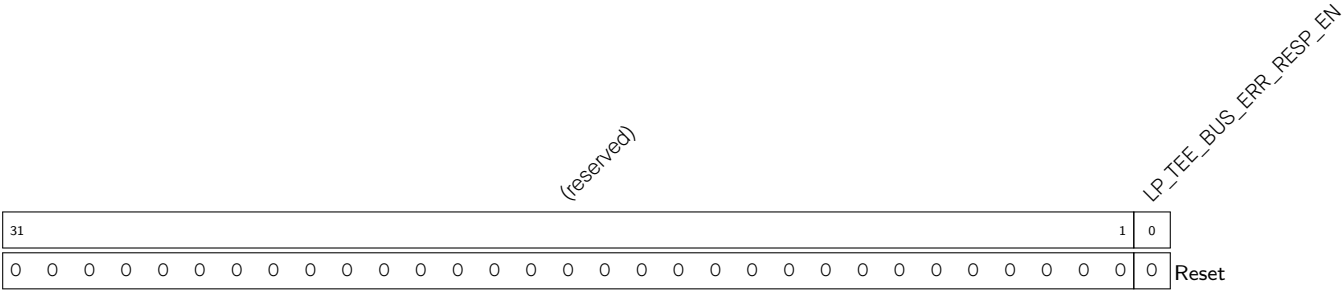
Register 13.129. LP_TEE_FORCE_ACC_HP_REG (0x0090)

Diagram of the LP_TEE_FORCE_ACC_HPMEM_EN register structure:

- Bit 31: (reserved)
- Bits 31-1: (reserved)
- Bit 0: Reset

LP_TEE_FORCE_ACC_HPMEM_EN Configures whether to allow LP CPU to forcibly access HP SRAM regardless of permission management.
0: Disable force access to HP SRAM
1: Enable force access to HP SRAM
(R/W)

Register 13.130. LP_TEE_BUS_ERR_CONF_REG (0x00F0)



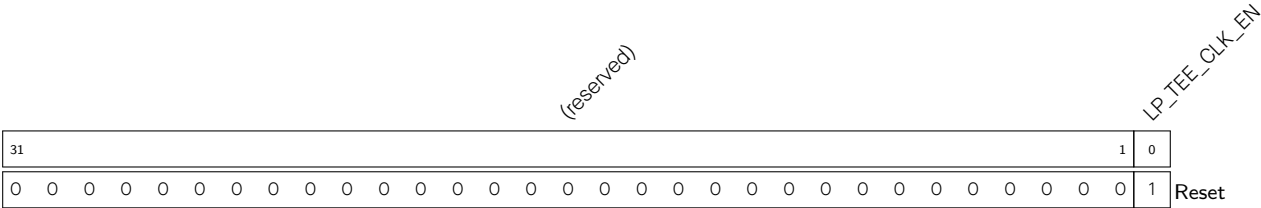
LP_TEE_BUS_ERR_RESP_EN Configures whether to return error message to CPU when access is blocked.

0: Disable

1: Enable

(R/W)

Register 13.131. LP_TEE_CLOCK_GATE_REG (0x00F8)



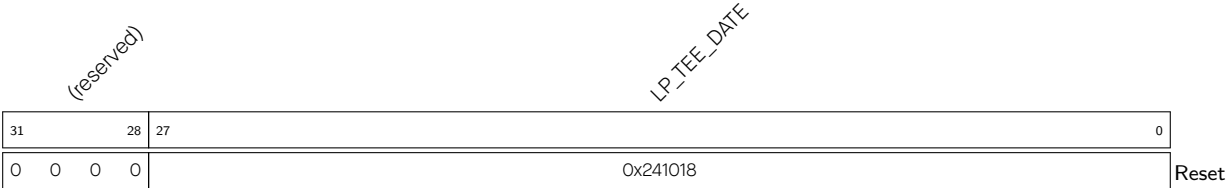
LP_TEE_CLK_EN Configures whether to keep the clock always on.

0: Enable automatic clock gating

1: Keep the clock always on

(R/W)

Register 13.132. LP_TEE_DATE_REG (0x00FC)



LP_TEE_DATE Version control register. (R/W)

Chapter 14

Debug Assistant

14.1 Overview

Debug Assistant is an auxiliary module that features a set of functions to help locate bugs and issues during software debugging.

14.2 Features

The Debug Assistant module has the following features:

- **Region read/write monitoring:** Monitors whether the High-Performance CPU (HP CPU) bus reads from or writes to a specified memory address space. A detected read or write in the monitored address space will trigger an interrupt.
- **Stack pointer (SP) monitoring:** Monitors whether the SP exceeds the specified address space. A bounds violation will trigger an interrupt.
- **Program counter (PC) logging:** Records the PC value. The last PC value at the most recent reset of HP CPU can be fetched.
- **Bus access logging:** Records the information about bus access. When the HP CPU, Low-Power CPU (LP CPU), or the Direct Memory Access controller (DMA) writes a specified value, the Debug Assistant module will record the data type, address of this write operation, and additionally the PC value when the write is performed by HP CPU, and push such information to the HP SRAM.

14.3 Functional Description

14.3.1 Region Read/Write Monitoring

The Debug Assistant module can monitor reads/writes performed by HP CPU bus in a certain address space, i.e., memory region. Whenever the bus reads or writes to the specified address space, an interrupt will be triggered. Please refer to Table 1 in Chapter 1 [High-Performance CPU \[to be added later\]](#).

In this chapter, when HP CPU bus accesses the data space, it is referred to as the **Data bus**, and when HP CPU bus accesses the AHB peripheral space, it is referred to as the **Peripheral bus**. The Debug Assistant module can simultaneously monitor two memory regions (assuming they are HP CPU region 0 and HP CPU region 1) on the Data bus, as well as on the Peripheral bus. The region 0 and 1 should be defined based on the developer's needs. The supported region read/write monitoring modes are listed below.

- Monitoring the read/write operations performed by the Data bus

- HP CPU Data bus reads from HP CPU region 0
- HP CPU Data bus writes to HP CPU region 0
- HP CPU Data bus reads from HP CPU region 1
- HP CPU Data bus writes to HP CPU region 1
- Monitoring the read/write operations performed by the Peripheral bus
 - HP CPU Peripheral bus reads from HP CPU region 0
 - HP CPU Peripheral bus writes to HP CPU region 0
 - HP CPU Peripheral bus reads from HP CPU region 1
 - HP CPU Peripheral bus writes to HP CPU region 1

14.3.2 SP Monitoring

The Debug Assistant module can monitor the SP to prevent stack overflow or erroneous push/pop in HP CPU. When the HP CPU SP exceeds the monitored region's lower or upper bounds, the module will record the PC and generate an interrupt. The bound is configured by software.

The supported SP monitoring modes are listed below.

- HP CPU SP goes beyond the upper bound of the HP CPU SP monitored region
- HP CPU SP goes below the lower bound of the HP CPU SP monitored region

14.3.3 PC Logging

In some cases, software developers may need to identify the PC value at the last reset of HP CPU. For instance, when the program is stuck and requires a reset, the developer may want to know where the program got stuck in order to debug. The Debug Assistant module can record the PC at the last reset of HP CPU, which can be then read for software debugging.

14.3.4 CPU/DMA Bus Access Logging

The Debug Assistant module can record the write operations of the HP CPU Data bus, LP CPU bus, and DMA bus in real time. When a write operation occurs in or a specific value is written to a specified address space, the module will record the bus type, the address, PC (only when the write is performed by the HP CPU), and other information. It will then store the data in the HP SRAM in a certain format. The specified address range must fall within the permitted memory regions of HP SRAM or external RAM.

14.4 Interrupts

The Debug Assistant can generate the BUS_MONITOR_INT interrupt signal that will be sent to the [Interrupt Matrix](#). The following interrupt sources can generate this interrupt signal.

- BUS_MONITOR_CORE_O_AREA_DRAMO_O_RD_INT: Triggered when the HP CPU Data bus reads from HP CPU region 0.

- `BUS_MONITOR_CORE_O_AREA_DRAMO_O_WR_INT`: Triggered when the HP CPU Data bus writes to HP CPU region 0.
- `BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_INT`: Triggered when the HP CPU Data bus reads from HP CPU region 1.
- `BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_INT`: Triggered when the HP CPU Data bus writes to HP CPU region 1.
- `BUS_MONITOR_CORE_O_AREA_PIF_O_RD_INT`: Triggered when the HP CPU Peripheral bus reads from HP CPU region 0.
- `BUS_MONITOR_CORE_O_AREA_PIF_O_WR_INT`: Triggered when the HP CPU Peripheral bus writes to HP CPU region 0.
- `BUS_MONITOR_CORE_O_AREA_PIF_1_RD_INT`: Triggered when the HP CPU Peripheral bus reads from HP CPU region 1.
- `BUS_MONITOR_CORE_O_AREA_PIF_1_WR_INT`: Triggered when the HP CPU Peripheral bus writes to HP CPU region 1.
- `BUS_MONITOR_CORE_O_SP_SPILL_MIN_INT`: Triggered when the HP CPU SP goes below the lower bound of the HP CPU SP monitored region.
- `BUS_MONITOR_CORE_O_SP_SPILL_MAX_INT`: Triggered when the HP CPU SP goes beyond the upper bound of the HP CPU SP monitored region.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix](#) [to be added later] > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [14.6 Register Summary](#).

14.5 Programming Procedures

14.5.1 Region Monitoring and SP Monitoring Configuration

The configuration process for region monitoring and SP monitoring is as follows:

1. Configure the monitored region and SP bounds.
 - Configure the HP CPU Data bus region 0 with `BUS_MONITOR_CORE_O_AREA_DRAMO_O_MIN_REG` and `BUS_MONITOR_CORE_O_AREA_DRAMO_O_MAX_REG`.
 - Configure the HP CPU Data bus region 1 with `BUS_MONITOR_CORE_O_AREA_DRAMO_1_MIN_REG` and `BUS_MONITOR_CORE_O_AREA_DRAMO_1_MAX_REG`.
 - Configure the HP CPU Peripheral bus region 0 with `BUS_MONITOR_CORE_O_AREA_PIF_O_MIN_REG` and `BUS_MONITOR_CORE_O_AREA_PIF_O_MAX_REG`.

- Configure the HP CPU Peripheral bus region 1 with [BUS_MONITOR_CORE_O_AREA_PIF_1_MIN_REG](#) and [BUS_MONITOR_CORE_O_AREA_PIF_1_MAX_REG](#).
 - Configure the HP CPU SP bounds with [BUS_MONITOR_CORE_O_SP_MIN_REG](#) and [BUS_MONITOR_CORE_O_SP_MAX_REG](#).
2. Configure [BUS_MONITOR_CORE_O_INTR_ENA_REG](#) to enable interrupts of a monitoring mode.
 3. Configure [BUS_MONITOR_CORE_O_MONTR_ENA_REG](#) to enable the monitoring mode(s). Various monitoring modes can be enabled at the same time.
 4. Read [BUS_MONITOR_CORE_O_INTR_RAW_REG](#) to get the raw interrupt status of a monitoring mode.
 5. Configure [BUS_MONITOR_CORE_O_INTR_CLR_REG](#) to clear the interrupt of a monitoring mode.

For example, if the Debug Assistant module needs to monitor whether the HP CPU Data bus has written to [A ~ B] address space, the user can enable monitoring in either the HP CPU Data bus region 0 or region 1. The following configuration process is based on region 0:

1. Configure [BUS_MONITOR_CORE_O_RCD_PDEBUGEN](#) to 1 to enable HP CPU to update the PC signals to the Debug Assistant module.
2. Configure [BUS_MONITOR_CORE_O_AREA_DRAMO_O_MIN_REG](#) to Address A.
3. Configure [BUS_MONITOR_CORE_O_AREA_DRAMO_O_MAX_REG](#) to Address B.
4. Configure [BUS_MONITOR_CORE_O_INTR_ENA_REG](#) bit[1] to enable the interrupt for write operations by the HP CPU Data bus in region 0.
5. Configure [BUS_MONITOR_CORE_O_MONTR_ENA_REG](#) bit[1] to enable monitoring write operations by the HP CPU Data bus in region 0.
6. Configure interrupt matrix to map [BUS_MONITOR_INTR](#) into HP CPU interrupt (refer to Chapter 3 *Interrupt Matrix [to be added later]*).
7. After the interrupt is triggered:
 - Read [BUS_MONITOR_CORE_O_INTR_RAW_REG](#) to identify the interrupt source.
 - If the interrupt is triggered by HP CPU region monitoring, read [BUS_MONITOR_CORE_O_AREA_PC](#) for the HP CPU PC value, and [BUS_MONITOR_CORE_O_AREA_SP](#) for the HP CPU SP.
 - If the interrupt is triggered by SP monitoring, read [BUS_MONITOR_CORE_O_SP_PC](#) for the HP CPU PC value.
 - Write 1 to the corresponding bit of [BUS_MONITOR_CORE_O_INTR_CLR_REG](#) to clear the interrupt.

14.5.2 PC Logging Configuration

Configure [BUS_MONITOR_CORE_O_RCD_PDEBUGEN](#) to 1 to enable HP CPU to update the PC signals to the Debug Assistant module. If [BUS_MONITOR_CORE_O_RCD_RECORDEN](#) is also configured to 1, [BUS_MONITOR_CORE_O_RCD_PDEBUGPC_REG](#) will record the HP CPU's PC value and [BUS_MONITOR_CORE_O_RCD_PDEBUGSP_REG](#) will record the HP CPU SP value. Otherwise, these two registers will retain their original values.

When the HP CPU resets, [BUS_MONITOR_CORE_O_RCD_EN_REG](#) will reset, while [BUS_MONITOR_CORE_O_RCD_PDEBUGPC_REG](#) and [BUS_MONITOR_CORE_O_RCD_PDEBUGSP_REG](#) will

not. Therefore, these two registers will retain the PC and SP values at the HP CPU reset.

14.5.3 CPU/DMA Bus Access Logging Configuration

14.5.3.1 Bus Access Logging 1 Configuration

Bus access logging 1 includes HP CPU Bus access logging (HP SRAM and external RAM monitoring), LP CPU Bus access logging (HP SRAM monitoring), and DMA Bus access logging (HP SRAM monitoring). The configuration process is described below.

1. Configure monitored address space: set [MEM_MONITOR_LOG_MIN_REG](#) and [MEM_MONITOR_LOG_MAX_REG](#) to specify monitored address space. For HP CPU, the monitored address space should be in the address range of HP SRAM (0x4080_0000 ~ 0x4085_FFFF) or external RAM (0x4200_0000 ~ 0x43FF_FFFF). For LP CPU or DMA, the monitored address space should be in the address range of HP SRAM (0x4080_0000 ~ 0x4085_FFFF).
2. Configure the monitoring mode with [MEM_MONITOR_LOG_MODE](#):
 - Write monitoring (detects bus write operations)
 - Word monitoring (detects writes of a specific word)
 - Halfword monitoring (detects writes of a specific halfword)
 - Byte monitoring (detects writes of a specific byte)
3. Configure the specific values to be monitored.
 - In word monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG](#) specifies the monitored word.
 - In halfword monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG](#)[15:0] specifies the monitored halfword.
 - In byte monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG](#)[7:0] specifies the monitored byte.
 - [MEM_MONITOR_LOG_DATA_MASK_REG](#) is used to mask the byte(s) specified in [MEM_MONITOR_LOG_CHECK_DATA_REG](#). A masked byte can be any value. For example, in word monitoring, if [MEM_MONITOR_LOG_CHECK_DATA_REG](#) is configured to 0x01020304 and [MEM_MONITOR_LOG_DATA_MASK_REG](#) is configured to 0x1, then any writes of the data matching the 0x010203XX pattern by the bus will be recorded.
4. Configure the storage space for recorded data.
 - [MEM_MONITOR_LOG_MEM_START_REG](#) and [MEM_MONITOR_LOG_MEM_END_REG](#) specify the storage space for recorded data. The storage space must be in the range of 0x4080_0000 ~ 0x4085_FFFF.
 - Set [MEM_MONITOR_LOG_MEM_ADDR_UPDATE_REG](#) to update the value in [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) to [MEM_MONITOR_LOG_MEM_START_REG](#).
 - Configure the permission of the Debug Assistant module to access the HP SRAM. The Debug Assistant module can only access HP SRAM when the access permission is enabled. For more information, please refer to Chapter 13 *Permission Control (PMS)*.
5. Configure the writing mode for the recorded data for loop mode or non-loop mode.

- In loop mode, writing to the specified address space is performed in loops. When writing reaches the end address, it returns to the starting address and continues, overwriting the previously recorded data. Set [MEM_MONITOR_LOG_MEM_LOOP_ENABLE](#) to 1 to enable loop mode. For example, there are 10 write operations (1 ~ 10) to address space 0 ~ 4 during bus access. After the 5th operation writes to address 4, the 6th operation will start writing from address 0. The 6th to 10th operations will overwrite the previous data written by the 1st to 5th operations.
- In non-loop mode, when writing reaches the end address, it stops at the end address and dumps the remaining data. The previously recorded data will not be overwritten. Clear [MEM_MONITOR_LOG_MEM_LOOP_ENABLE](#) to enable non-loop mode. For example, there are 10 write operations (1 ~ 10) to address space 0 ~ 4 during bus access. After the 5th operation writes to address 4, the 6th to 10th write operations will stop at address 4 and will not be performed any more. Therefore, the address 0 ~ 4 stores the values written by the 1st to 5th operations and the values of the 6th to 10th operations are dumped.

6. Configure bus enable registers.

- Enable HP CPU, LP CPU, and DMA bus access logging respectively with [MEM_MONITOR_LOG_CORE_ENA](#), [MEM_MONITOR_LOG_DMA_1_ENA](#), and [MEM_MONITOR_LOG_DMA_0_ENA](#). They can be enabled at the same time.

The Debug Assistant module first writes the recorded data to the internal buffer A, and then fetches the data from the buffer and writes it to the configured memory space. When the monitored behaviors are triggered continuously, the generated recording packets may fill the buffer, preventing it from accepting new packets. In such cases, the module dumps the incoming packets and buffers a LOST packet instead before the buffer reaches full capacity. However, it is only possible to know the bus type of the dumped packets when the LOST packet was generated, but not the number of the dumped packets before the generation.

When bus access logging is finished, the recorded data can be read from memory for decoding. The recorded data is in four packet formats, namely, HP CPU packet (corresponding to the HP CPU Data bus), LP CPU packet (corresponding to LP CPU Data bus), DMA packet (corresponding to DMA Data bus), and LOST packet. The packet formats are shown in Tables [14.5-1](#), [14.5-2](#), [14.5-3](#), and [14.5-4](#).

Table 14.5-1. HP CPU Packet Format

Bit[63:33]	Bit[32]	Bit[31:3]	Bit[2:1]	Bit[0]
pc_offset	anchored(1)	addr_offset	format	anchored(0)

Table 14.5-2. LP CPU Packet Format

Bit[31:8]	Bit[7:3]	Bit[2:1]	Bit[0]
addr_offset	reserved	format	anchored(0)

Table 14.5-3. DMA Packet Format

Bit[31:8]	Bit[7:3]	Bit[2:1]	Bit[0]
addr_offset	dma_source	format	anchored(0)

Table 14.5-4. LOST Packet Format

Bit[31:7]	Bit[6:3]	Bit[2:1]	Bit[0]
reserved	lost_source	format	anchored(0)

The data packet formats show that the HP CPU packet has 64 bits, and other packets have 32 bits each. These packets have the following fields:

- **format** – the packet type:
 - 0: HP CPU packet
 - 1: DMA packet
 - 2: LP CPU packet
 - 3: LOST packet
- **pc_offset** - the offset of the HP CPU PC register at the time of access. Actual PC = pc_offset + 0x0000_0000.
- **addr_offset** - the address offset of a write operation. Actual address = addr_offset + {[MEM_MONITOR_LOG_MIN_REG](#)[31:2], 2'b0}.
- **dma_source** - the source of DMA access. Details can be found in [Table 14.5-5](#), where sources corresponding to value 16 ~ 31 are detailed in [3 GDMA Controller \(GDMA\)](#).
- **lost_source** - the dumped packets when the LOST packet was generated.
 - Bit[4:3]:
 - * 0: HP CPU packets were not dumped
 - * 3: HP CPU packets were dumped
 - * Other values: reserved
 - Bit[5]:
 - * 0: DMA packets were not dumped
 - * 1: DMA packets were dumped
 - Bit[6]:
 - * 0: LP CPU packets were not dumped
 - * 1: LP CPU packets were dumped
- **anchored** - the location of the 32 bits in the data packet:
 - 0: Lower 32 bits
 - 1: Higher 32 bits

Table 14.5-5. DMA Access Source

Value	Source
0	HP CPU
1	LP CPU

Value	Source
2	Reserved
3	Reserved
4	Reserved
5	MEM_MONITOR
6	TRACE
7	Reserved
8	PSRAM_MEM_MONITOR
9 ~ 15	Reserved
16 ~ 31	Refer to the peripheral corresponding to the value 0 ~ 15 in Chapter 3 <i>GDMA Controller (GDMA)</i> > Table 3.4-1. For example, a value of 16 matches the peripheral corresponding to 0 in this table, a value of 17 matches the peripheral corresponding to 1

The internal buffer of the module is 32-bit wide. If HP CPU, LP CPU, and DMA bus access loggings are enabled at the same time, and the record data is generated at the same time, the HP CPU packets are cached into the buffer first, then the DMA packets, and finally the LP CPU packets. The Debug Assistant module will automatically fetch the buffered data and store it in 32-bit data width into the specified memory space.

In loop mode, data looping several times in the storage memory may cause residual data, which can interfere with packet parsing. For example, the lower 32 bits of an HP CPU packet are overwritten, thus making its higher 32 bits residual data. Therefore, users need to filter out the possible residual data when determining the starting position of the first valid packet. Read [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) to identify the starting position of the packet, then check the anchored bit value of the packet. If it is 0, retain the data. If it is 1, discard it.

The process of packet parsing is described below.

- Read [MEM_MONITOR_LOG_MEM_FULL_FLAG](#) to determine whether there is a data overflow.
 - If no, the address space to read is [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) – 4.
 - If yes and the loop mode is enabled, the address space is [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) ~ [MEM_MONITOR_LOG_MEM_END_REG](#) and [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) – 4.
 - If yes and loop mode is not enabled, the address space is [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_END_REG](#).
- Read and parse data from the starting address. Read 32 bits at a time.

After packet parsing is completed, clear the [MEM_MONITOR_LOG_MEM_FULL_FLAG](#) flag bit by setting [MEM_MONITOR_CLR_LOG_MEM_FULL_FLAG](#) to 1.

14.5.3.2 Bus Access Logging 2 Configuration

Bus access logging 2 includes DMA Bus access logging (external RAM monitoring). The configuration process is described below.

1. Configure monitored address space: Configure [MEM_MONITOR_LOG_MIN_REG](#) and [MEM_MONITOR_LOG_MAX_REG](#) to specify monitored address space. The monitored address space should range from 0x4200_0000 to 0x43FF_FFFF.
2. Configure the monitoring mode with [MEM_MONITOR_LOG_MODE](#):
 - Write monitoring (detects bus write operations)
 - Word monitoring (detects writes of a specific word)
 - Halfword monitoring (detects writes of a specific halfword)
 - Byte monitoring (detects writes of a specific byte)
3. Configure the specific values to be monitored.
 - In word monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG](#) specifies the monitored word.
 - In halfword monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG\[15:0\]](#) specifies the monitored halfword.
 - In byte monitoring mode, [MEM_MONITOR_LOG_CHECK_DATA_REG\[7:0\]](#) specifies the monitored byte.
 - [MEM_MONITOR_LOG_DATA_MASK_REG](#) is used to mask the byte specified in [MEM_MONITOR_LOG_CHECK_DATA_REG](#). A masked byte can be any value. For example, in word monitoring, if [MEM_MONITOR_LOG_CHECK_DATA_REG](#) is configured to 0x01020304 and [MEM_MONITOR_LOG_DATA_MASK_REG](#) is configured to 0x1, then any writes of the data matching the 0x010203XX pattern by the bus will be recorded.
4. Configure the storage space for recorded data.
 - [MEM_MONITOR_LOG_MEM_START_REG](#) and [MEM_MONITOR_LOG_MEM_END_REG](#) specify the storage space for recorded data. The storage space must be in the range of 0x4080_0000 ~ 0x4085_FFFF.
 - Set [MEM_MONITOR_LOG_MEM_ADDR_UPDATE_REG](#) to update the value in [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) to [MEM_MONITOR_LOG_MEM_START_REG](#).
 - Configure the permission for the Debug Assistant module to access the HP SRAM. Only when the access permission is enabled can the Debug Assistant module access the HP SRAM. For more information, please refer to Chapter 13 *Permission Control (PMS)*.
5. Configure the writing mode for the recorded data in loop mode or non-loop mode.
 - In loop mode, writing to the specified address space is performed in loops. When writing reaches the end address, it will return to the starting address and continue, overwriting the previously recorded data. Set [MEM_MONITOR_LOG_MEM_LOOP_ENABLE](#) to enable loop mode.
 - In non-loop mode, when writing reaches the end address, it will stop at the end address and dump the remaining data, not overwriting the previously recorded data. Clear [MEM_MONITOR_LOG_MEM_LOOP_ENABLE](#) to enable non-loop mode.
 - See examples in Section 14.5.3.1 > step 5.
6. Configure [MEM_MONITOR_LOG_DMA_O_ENA](#) to enable DMA_0 channel groups bus access logging.

The Debug Assistant module first writes the DMA recorded data to the internal buffer B, and then fetches the data from the buffer and writes it to the configured memory space. When the monitored behaviors are triggered continuously, the generated recording packets may fill the buffer, preventing it from accepting new packets. In such cases, the module dumps the incoming packets and buffers an DMA LOST packet instead before the buffer reaches full capacity. However, it is only possible to know the bus type of the dumped packets when the DMA LOST packet was generated, but not the number of the dumped packets before the generation.

When bus access logging is finished, the recorded data can be read from memory for decoding. The recorded data is in two packet formats, namely, DMA_0 packet (corresponding to DMA_0 Data bus) and DMA LOST packet. The packet formats are shown in Table 14.5-6 and 14.5-7.

Table 14.5-6. DMA_0 Packet Format

Bit[31:7]	Bit[6:2]	Bit[1]	Bit[0]
addr_offset	dma_source	format	anchored(0)

Table 14.5-7. DMA LOST Packet Format

Bit[31:3]	Bit[2]	Bit[1]	Bit[0]
reserved	lost_source	format	anchored(0)

The data packet formats show that the DMA_0 and DMA LOST packets both have 32 bits. These packets contain the following fields:

- **format** – the packet type:
 - 0: DMA_0 packet
 - 1: DMA LOST packet
- **addr_offset0** - the address offset of a write operation. Actual address = addr_offset + {[MEM_MONITOR_LOG_MIN_REG](#)[31:2], 2'b0}.
- **dma_source** - the source of DMA access. Details can be found in Table 14.5-5.
- **lost_source** - the dumped packets when the DMA LOST packet was generated:
 - 0: DMA_0 packets were not dumped
 - 1: DMA_0 packets were dumped
- **anchored** - the location of the 32 bits in the data packet:
 - 0: Lower 32 bits
 - 1: Higher 32 bits

The internal buffer of the module is 32 bits wide. When the bus access logging of DMA_0 channel groups is enabled and the record data is generated, the DMA_0 data packets are buffered. The Debug Assistant module will automatically fetch the buffered data and store it in 32-bit data width into the specified memory space.

The process of packet parsing is described below.

- Read [MEM_MONITOR_LOG_MEM_FULL_FLAG](#) to determine whether there is a data overflow.

- If no, the address space to read is [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) – 4.
- If yes and the loop mode is enabled, the address space is [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) ~ [MEM_MONITOR_LOG_MEM_END_REG](#) and [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG](#) – 4.
- If yes and loop mode is not enabled, the address space is [MEM_MONITOR_LOG_MEM_START_REG](#) ~ [MEM_MONITOR_LOG_MEM_END_REG](#).
- Read and parse data from the starting address. Read 32 bits at a time.

After packet parsing is completed, clear the [MEM_MONITOR_LOG_MEM_FULL_FLAG](#) flag bit by setting [MEM_MONITOR_CLR_LOG_MEM_FULL_FLAG](#) to 1.

14.6 Register Summary

The addresses of **bus logging configuration registers** in Section 14.6.1 are relative to the **TCM_MEM_MONITOR** base address and the **PSRAM_MEM_MONITOR**. The addresses of other registers in Section 14.6.2 are relative to the **BUS_MONITOR** base address. All base addresses are provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

14.6.1 Bus Logging Configuration Register Summary

Name	Description	Address	Access
Bus logging configuration registers			
MEM_MONITOR_LOG_SETTING_REG	Configures bus access logging	0x0000	R/W
MEM_MONITOR_LOG_SETTING1_REG	Configures bus access logging	0x0004	R/W
MEM_MONITOR_LOG_CHECK_DATA_REG	Configures monitored data in bus access logging	0x0008	R/W
MEM_MONITOR_LOG_DATA_MASK_REG	Configures masked data in bus access logging	0x000C	R/W
MEM_MONITOR_LOG_MIN_REG	Configures the monitored lower address for bus access logging	0x0010	R/W
MEM_MONITOR_LOG_MAX_REG	Configures the monitored upper address for bus access logging	0x0014	R/W
MEM_MONITOR_LOG_MON_ADDR_UPDATE_0_REG	Configures whether to update the monitored address space for HP CPU bus access logging	0x0018	WT
MEM_MONITOR_LOG_MON_ADDR_UPDATE_1_REG	Configures whether to update the monitored address space for DMA_0 bus access logging	0x001C	WT
MEM_MONITOR_LOG_MEM_START_REG	Configures the starting address of the storage memory for recorded data	0x0020	R/W
MEM_MONITOR_LOG_MEM_END_REG	Configures the end address of the storage memory for recorded data	0x0024	R/W
MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG	Represents the address for the next write	0x0028	RO
MEM_MONITOR_LOG_MEM_ADDR_UPDATE_REG	Updates the address for the next write with the starting address for the recorded data	0x002C	WT
MEM_MONITOR_LOG_MEM_FULL_FLAG_REG	Logging overflow status register	0x0030	varies
Clock control register			
MEM_MONITOR_CLOCK_GATE_REG	Clock control register	0x0034	R/W
Version control register			

Name	Description	Address	Access
MEM_MONITOR_DATE_REG	Version control register	0x03FC	R/W

14.6.2 Summary of Other Registers

Name	Description	Address	Access
Monitor configuration registers			
BUS_MONITOR_CORE_O_MONTR_ENA_REG	Configures whether to enable HP CPU monitoring	0x0000	R/W
BUS_MONITOR_CORE_O_AREA_DRAMO_O_MIN_REG	Configures lower boundary address of region 0 monitored on Data bus	0x0010	R/W
BUS_MONITOR_CORE_O_AREA_DRAMO_O_MAX_REG	Configures upper boundary address of region 0 monitored on Data bus	0x0014	R/W
BUS_MONITOR_CORE_O_AREA_DRAMO_1_MIN_REG	Configures lower boundary address of region 1 monitored on Data bus	0x0018	R/W
BUS_MONITOR_CORE_O_AREA_DRAMO_1_MAX_REG	Configures upper boundary address of region 1 monitored on Data bus	0x001C	R/W
BUS_MONITOR_CORE_O_AREA_PIF_O_MIN_REG	Configures lower boundary address of region 0 monitored on Peripheral bus	0x0020	R/W
BUS_MONITOR_CORE_O_AREA_PIF_O_MAX_REG	Configures upper boundary address of region 0 monitored on Peripheral bus	0x0024	R/W
BUS_MONITOR_CORE_O_AREA_PIF_1_MIN_REG	Configures lower boundary address of region 1 monitored on Peripheral bus	0x0028	R/W
BUS_MONITOR_CORE_O_AREA_PIF_1_MAX_REG	Configures upper boundary address of region 1 monitored on Peripheral bus	0x002C	R/W
BUS_MONITOR_CORE_O_AREA_PC_REG	Represents the PC status under HP CPU region monitoring	0x0030	RO
BUS_MONITOR_CORE_O_AREA_SP_REG	Represents the SP status under HP CPU region monitoring	0x0034	RO
BUS_MONITOR_CORE_O_SP_MIN_REG	Configures SP monitoring lower boundary address	0x0038	R/W
BUS_MONITOR_CORE_O_SP_MAX_REG	Configures SP monitoring upper boundary address	0x003C	R/W
BUS_MONITOR_CORE_O_SP_PC_REG	Represents the PC status under HP CPU SP monitoring	0x0040	RO
Interrupt configuration registers			
BUS_MONITOR_CORE_O_INTR_RAW_REG	HP CPU monitor raw interrupt status register	0x0004	RO
BUS_MONITOR_CORE_O_INTR_ENA_REG	HP CPU monitor interrupt enable register	0x0008	R/W
BUS_MONITOR_CORE_O_INTR_CLR_REG	HP CPU monitor interrupt clear register	0x000C	WT
PC recording configuration register			
BUS_MONITOR_CORE_O_RCD_EN_REG	HP CPU PC logging enable register	0x0044	R/W
PC recording status registers			
BUS_MONITOR_CORE_O_RCD_PDEBUGPC_REG	HP CPU PC logging register	0x0048	RO

Name	Description	Address	Access
BUS_MONITOR_CORE_0_RCD_PDEBUGSP_REG	HP CPU SP logging register	0x004C	RO
CPU status registers			
BUS_MONITOR_CORE_0_LASTPC_BEFORE_EXCEPTION_REG	PC of the last command before HP CPU enters exception	0x0070	RO
BUS_MONITOR_CORE_0_DEBUG_MODE_REG	HP CPU debug mode status register	0x0074	RO
Clock control register			
BUS_MONITOR_CLOCK_GATE_REG	Clock control register	0x0108	R/W
Version control register			
BUS_MONITOR_DATE_REG	Version control register	0x03FC	R/W

14.7 Registers

The addresses of **bus logging configuration registers** in Section 14.7.1 are relative to the **TCM_MEM_MONITOR** base address and the **PSRAM_MEM_MONITOR**. The addresses of other registers in Section 14.7.2 are relative to the **BUS_MONITOR** base address. All base addresses are provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

14.7.1 Bus Logging Configuration Registers

Register 14.1. MEM_MONITOR_LOG_SETTING_REG (0x0000)

MEM_MONITOR_LOG_DMA_1_ENA								MEM_MONITOR_LOG_DMA_0_ENA								MEM_MONITOR_LOG_CORE_ENA								(reserved)				MEM_MONITOR_LOG_MEM_LOOP_ENABLE								MEM_MONITOR_LOG_MODE																																																			
31								24								23								16								15								8								7								5								4								3								0							
0																0																0								0								0								0								1								0								Reset							

MEM_MONITOR_LOG_MODE Configures monitoring modes.

- 1: Enable write monitoring
- 2: Enable word monitoring
- 4: Enable halfword monitoring
- 8: Enable byte monitoring.

Other values: Invalid

(R/W)

MEM_MONITOR_LOG_MEM_LOOP_ENABLE Configures the writing mode for recorded data.

- 0: Non-loop mode
- 1: Loop mode

(R/W)

MEM_MONITOR_LOG_CORE_ENA Configures whether to enable HP CPU bus access logging.

- 0: Disable
- 1: Enable

Other values: Invalid

(R/W)

Continued on the next page...

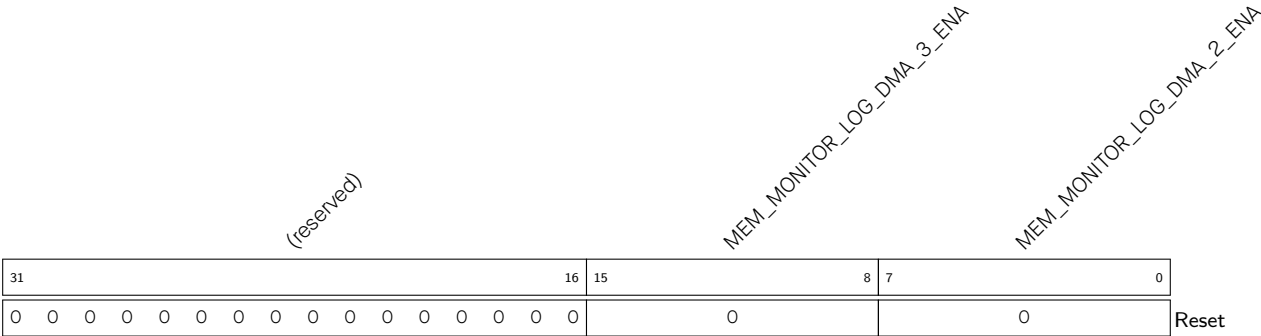
Register 14.1. MEM_MONITOR_LOG_SETTING_REG (0x0000)

Continued from the previous page...

MEM_MONITOR_LOG_DMA_0_ENA Configures whether to enable DMA_0 bus access logging.
0: Disable
1: Enable
Other values: Invalid
(R/W)

MEM_MONITOR_LOG_DMA_1_ENA Configures whether to enable DMA_1 bus access logging.
0: Disable
1: Enable
Other values: Invalid
(R/W)

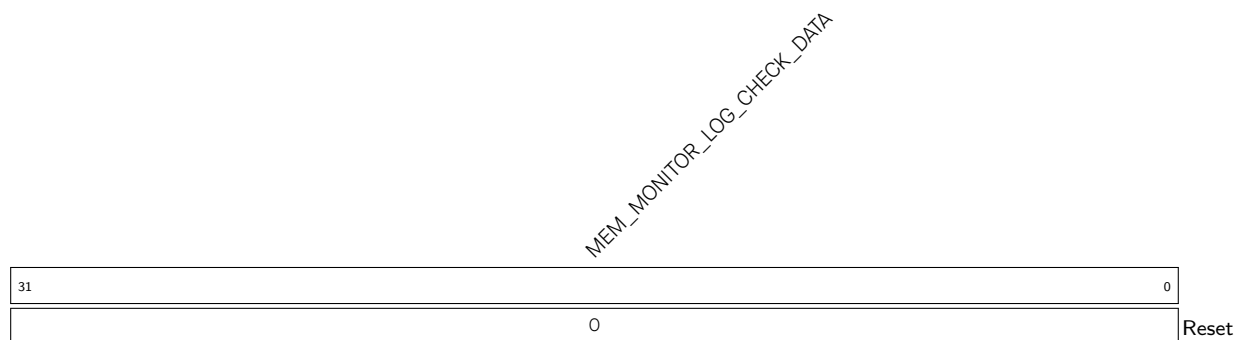
Register 14.2. MEM_MONITOR_LOG_SETTING1_REG (0x0004)



MEM_MONITOR_LOG_DMA_2_ENA Configures whether to enable DMA_2 bus access logging.
0: Disable
1: Enable
Other values: Invalid
(R/W)

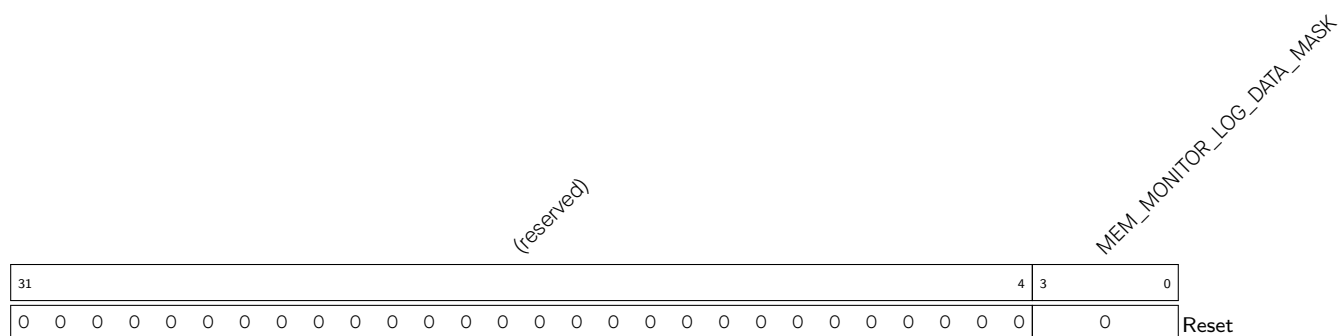
MEM_MONITOR_LOG_DMA_3_ENA Configures whether to enableDMA_3 bus access logging.
0: Disable
1: Enable
Other values: Invalid
(R/W)

Register 14.3. MEM_MONITOR_LOG_CHECK_DATA_REG (0x0008)



MEM_MONITOR_LOG_CHECK_DATA Configures the data to be monitored during bus accessing.
(R/W)

Register 14.4. MEM_MONITOR_LOG_DATA_MASK_REG (0x000C)



MEM_MONITOR_LOG_DATA_MASK Configures which byte(s) in **MEM_MONITOR_LOG_CHECK_DATA_REG** to mask. Multiple bytes can be masked at the same time.

bit[0]: Configures whether to mask the least significant byte of
MEM_MONITOR_LOG_CHECK_DATA_REG.

0: Not mask

1: Mask

bit[1]: Configures whether to mask the second least significant byte of MEM_MONITOR_LOG_CHECK_DATA_REG.

0: Not mask

1: Mask

bit[2]: Configures whether to mask the second most significant byte of MEM MONITOR LOG CHECK DATA REG.

0: Not mask

1: Mask

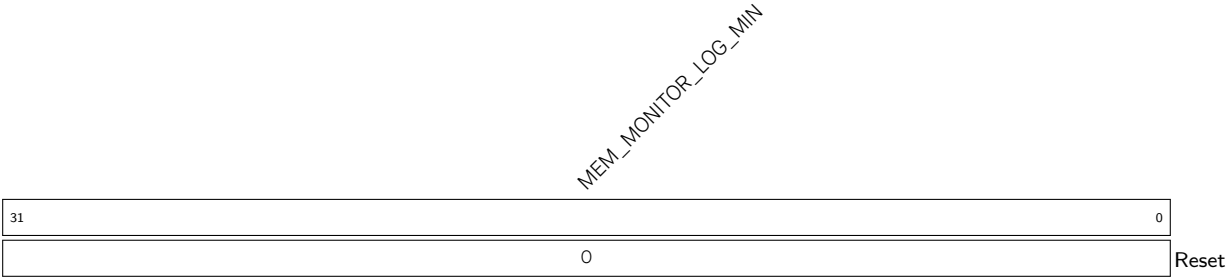
bit[3]: Configures whether to mask the most significant byte of
MEM_MONITOR_LOG_CHECK_DATA_REG.

0: Not mask

1: Mask

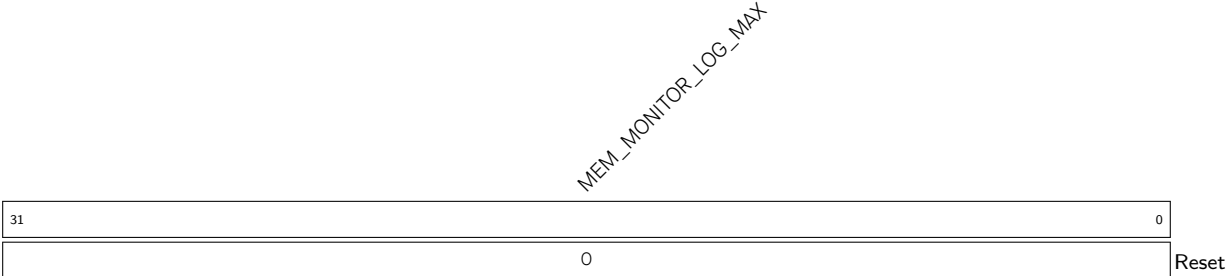
(R/W)

Register 14.5. MEM_MONITOR_LOG_MIN_REG (0x0010)



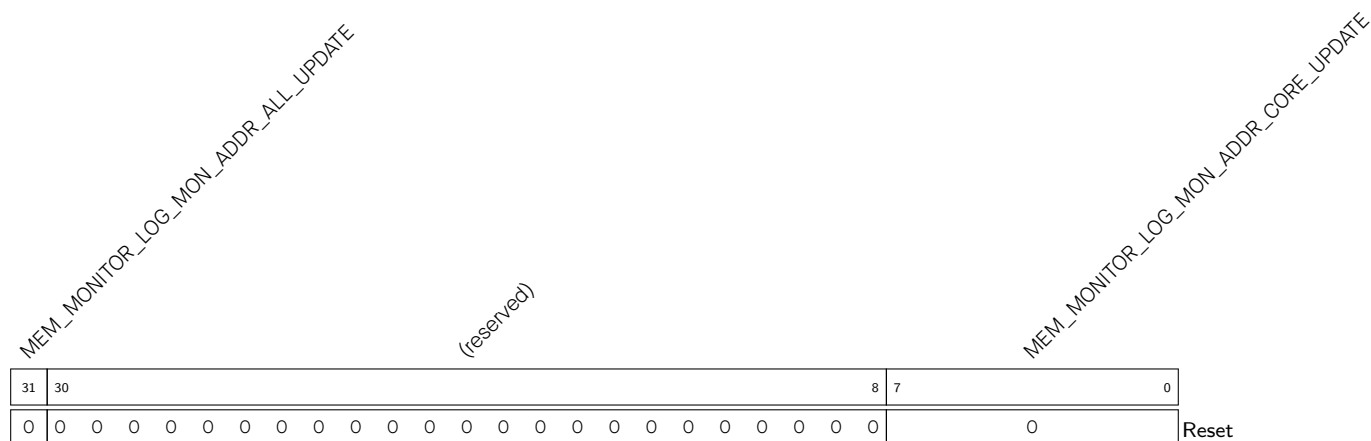
MEM_MONITOR_LOG_MIN Configures the lower bound address of the monitored address space.
(R/W)

Register 14.6. MEM_MONITOR_LOG_MAX_REG (0x0014)



MEM_MONITOR_LOG_MAX Configures the upper bound address of the monitored address space.
(R/W)

Register 14.7. MEM_MONITOR_LOG_MON_ADDR_UPDATE_0_REG (0x0018)



MEM_MONITOR_LOG_MON_ADDR_CORE_UPDATE Configures whether to update the monitored address space of the HP CPU bus as the address space from **MEM_MONITOR_LOG_MIN_REG** to **MEM_MONITOR_LOG_MAX_REG**.

0: Not update

1: Update

Other values: Invalid

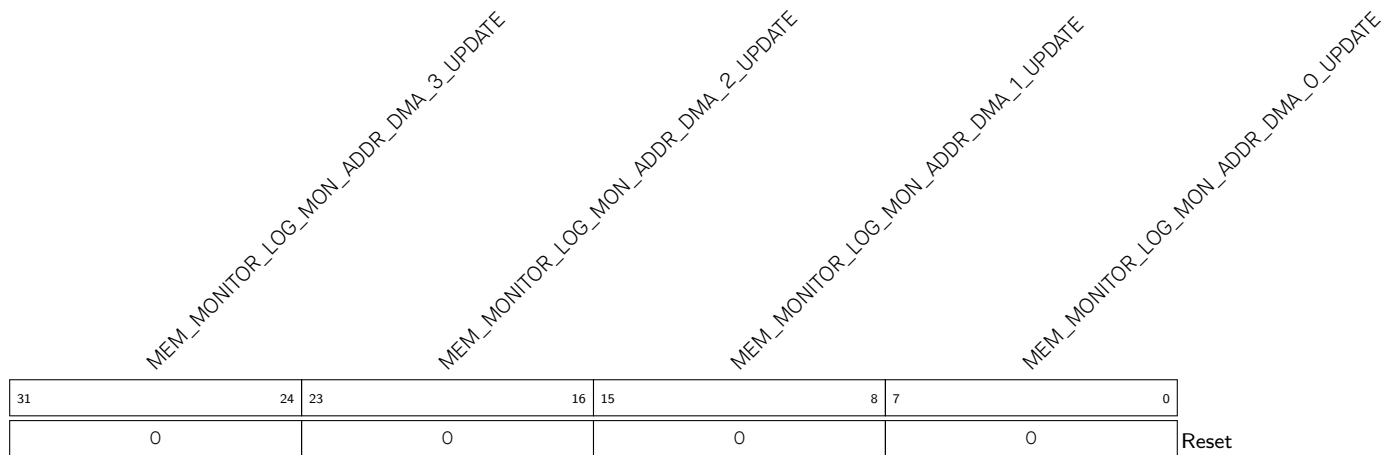
(WT)

MEM_MONITOR_LOG_MON_ADDR_ALL_UPDATE Configures whether to update the monitored address space of all masters as the address space from [MEM_MONITOR_LOG_MIN_REG](#) to [MEM_MONITOR_LOG_MAX_REG](#).

0: Not update

1: Update

(WT)

Register 14.8. MEM_MONITOR_LOG_MON_ADDR_UPDATE_1_REG (0x001C)

MEM_MONITOR_LOG_MON_ADDR_DMA_0_UPDATE Configures whether to update the monitored address space of the DMA_0 bus as the address space from [MEM_MONITOR_LOG_MIN_REG](#) to [MEM_MONITOR_LOG_MAX_REG](#).

0: Not update

1: Update

Other values: Invalid
(WT)

MEM_MONITOR_LOG_MON_ADDR_DMA_1_UPDATE Configures whether to update the monitored address space of the DMA_1 bus as the address space from [MEM_MONITOR_LOG_MIN_REG](#) to [MEM_MONITOR_LOG_MAX_REG](#).

0: Not update

1: Update

Other values: Invalid
(WT)

MEM_MONITOR_LOG_MON_ADDR_DMA_2_UPDATE Configures whether to update the monitored address space of the DMA_2 bus as the address space from [MEM_MONITOR_LOG_MIN_REG](#) to [MEM_MONITOR_LOG_MAX_REG](#).

0: Not update

1: Update

Other values: Invalid
(WT)

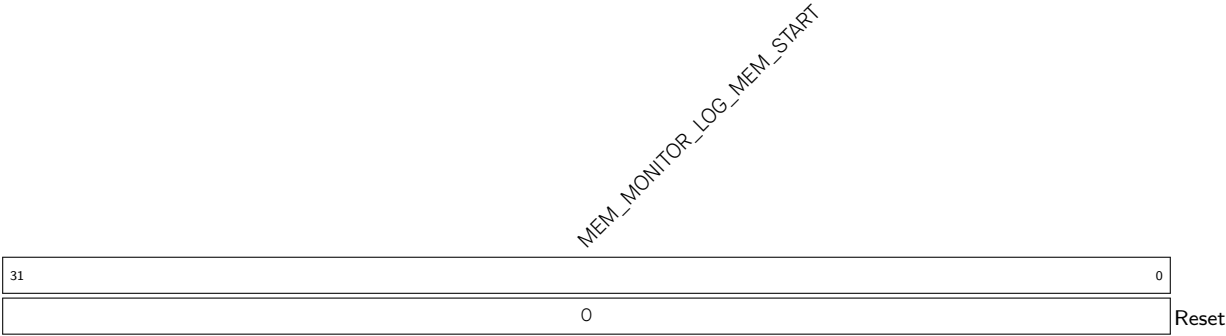
MEM_MONITOR_LOG_MON_ADDR_DMA_3_UPDATE Configures whether to update the monitored address space of the DMA_3 bus as the address space from [MEM_MONITOR_LOG_MIN_REG](#) to [MEM_MONITOR_LOG_MAX_REG](#).

0: Not update

1: Update

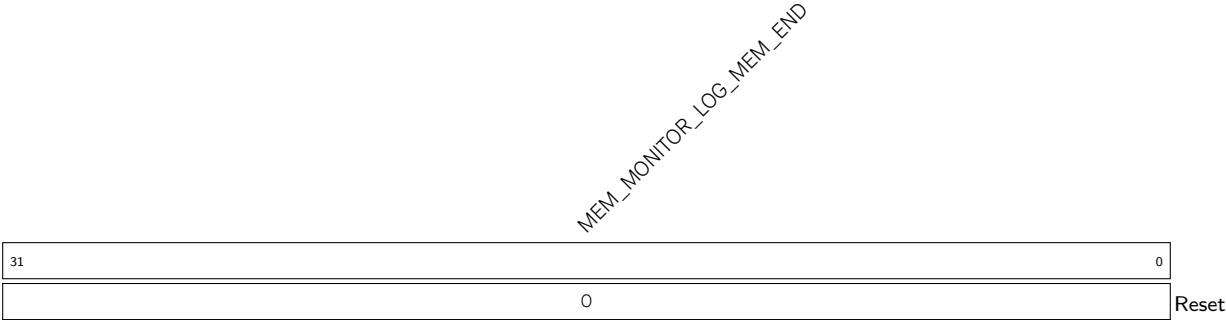
Other values: Invalid
(WT)

Register 14.9. MEM_MONITOR_LOG_MEM_START_REG (0x0020)



MEM_MONITOR_LOG_MEM_START Configures the starting address of the storage space for recorded data. (R/W)

Register 14.10. MEM_MONITOR_LOG_MEM_END_REG (0x0024)



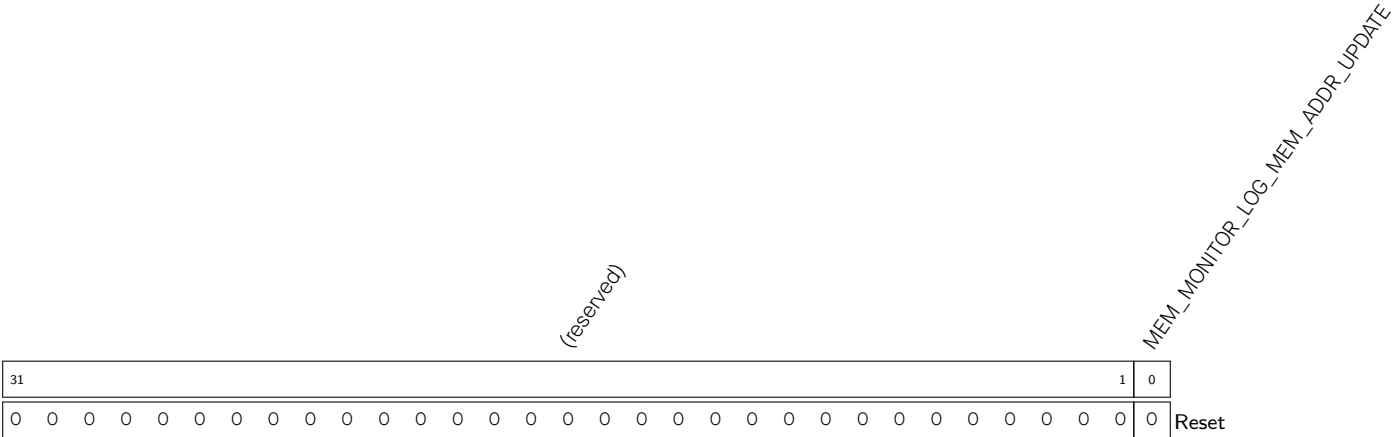
MEM_MONITOR_LOG_MEM_END Configures the ending address of the storage space for recorded data. (R/W)

Register 14.11. MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG (0x0028)



MEM_MONITOR_LOG_MEM_CURRENT_ADDR Represents the address of the next write. (RO)

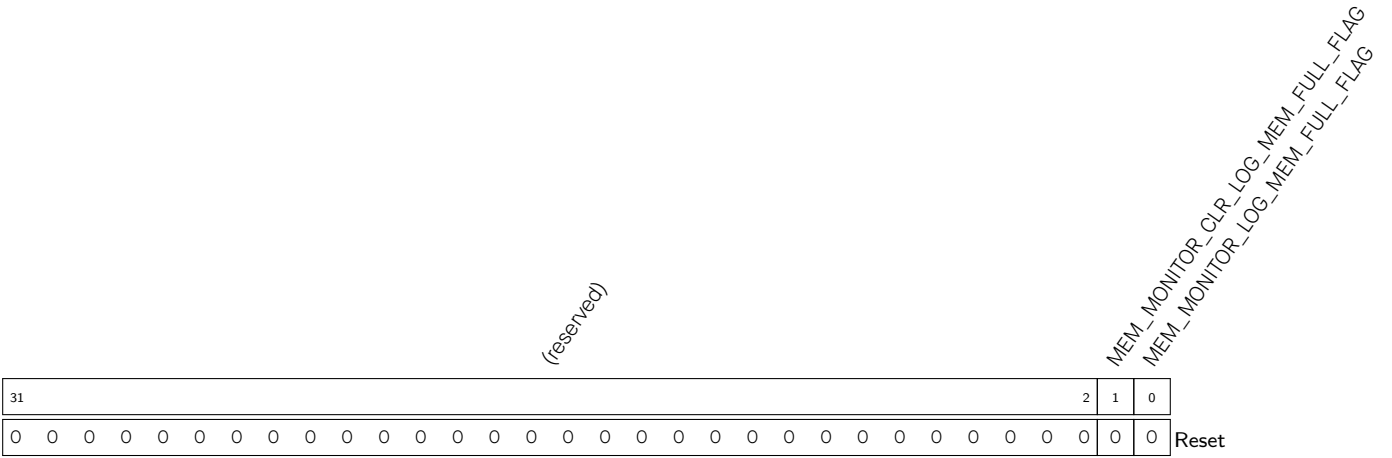
Register 14.12. MEM_MONITOR_LOG_MEM_ADDR_UPDATE_REG (0x002C)



MEM_MONITOR_LOG_MEM_ADDR_UPDATE Configures whether to update the value in **MEM_MONITOR_LOG_MEM_START_REG** to the value of **MEM_MONITOR_LOG_MEM_CURRENT_ADDR_REG**.

0: Not update
1: Update
(WT)

Register 14.13. MEM_MONITOR_LOG_MEM_FULL_FLAG_REG (0x0030)



MEM_MONITOR_LOG_MEM_FULL_FLAG Represents whether data overflows the storage space.

0: Not Overflow
1: Overflow
(RO)

MEM_MONITOR_CLR_LOG_MEM_FULL_FLAG Configures whether to clear the-
MEM_MONITOR_LOG_MEM_FULL_FLAG flag bit.

0: Not clear (default)
1: Clear
(WT)

Register 14.14. MEM_MONITOR_CLOCK_GATE_REG (0x0034)

(reserved)																												MEM_MONITOR_CLK_EN	
31																												1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
																													Reset

MEM_MONITOR_CLK_EN Configures whether to enable the register clock gating.

- 0: Disable
- 1: Enable

(R/W)

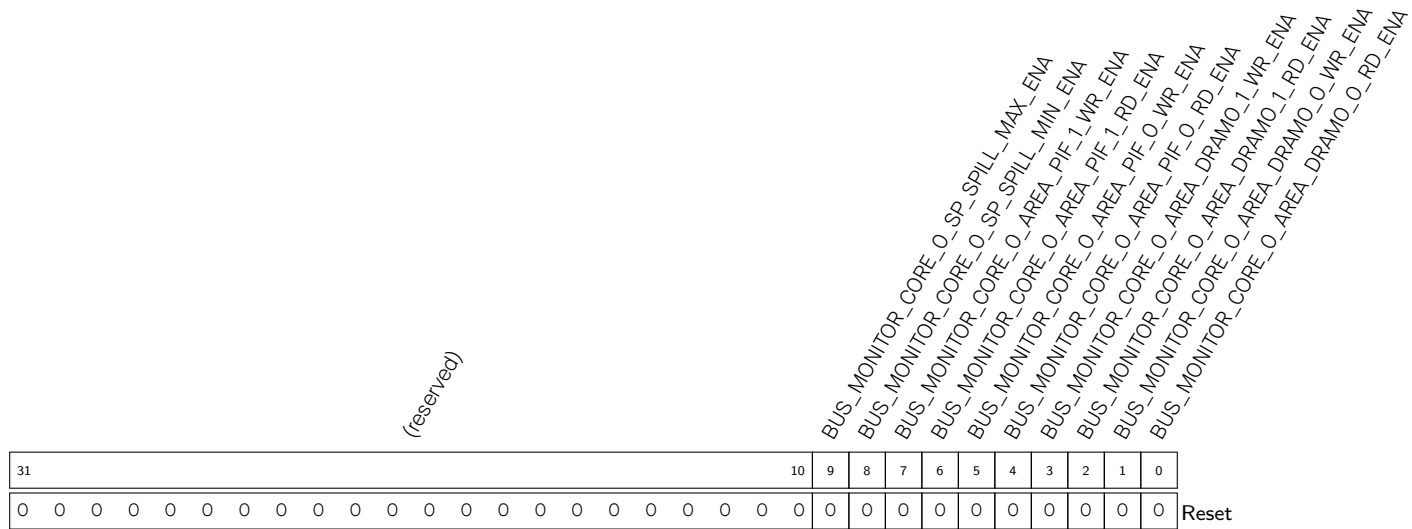
Register 14.15. MEM_MONITOR_DATE_REG (0x03FC)

(reserved)				MEM_MONITOR_DATE																									
31			28	27																									0
0	0	0	0	0x2308140																								Reset	

MEM_MONITOR_DATE Version control register. (R/W)

14.7.2 Other Registers

Register 14.16. BUS_MONITOR_CORE_O_MONTR_ENA_REG (0x0000)



BUS_MONITOR_CORE_0_AREA_DRAMO_0_RD_ENA Configures whether to monitor read operations in region 0 by the Data bus.

0: Not monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_0_AREA_DRAMO_0_WR_ENA Configures whether to monitor write operations in region 0 by the Data bus.

0: Not monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_0_AREA_DRAMO_1_RD_ENA Configures whether to monitor read operations in region 1 by the Data bus.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_0_AREA_DRAMO_1_WR_ENA Configures whether to monitor write operations in region 1 by the Data bus.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_0_AREA_PIF_0_RD_ENA Configures whether to monitor read operations in region 0 by the Peripheral bus.

0: Not Monitor

1: Monitor

(R/W)

Continued on the next page...

Register 14.16. BUS_MONITOR_CORE_O_MONTR_ENA_REG (0x0000)

Continued from the previous page...

BUS_MONITOR_CORE_O_AREA_PIF_0_WR_ENA Configures whether to monitor write operations in region 0 by the Peripheral bus.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_O_AREA_PIF_1_RD_ENA Configures whether to monitor read operations in region 1 by the Peripheral bus.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_O_AREA_PIF_1_WR_ENA Configures whether to monitor write operations in region 1 by the Peripheral bus.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_O_SP_SPILL_MIN_ENA Configures whether to monitor SP exceeding the lower bound address of SP monitored region.

0: Not Monitor

1: Monitor

(R/W)

BUS_MONITOR_CORE_O_SP_SPILL_MAX_ENA Configures whether to monitor SP exceeding the upper bound address of SP monitored region.

0: Not Monitor

1: Monitor

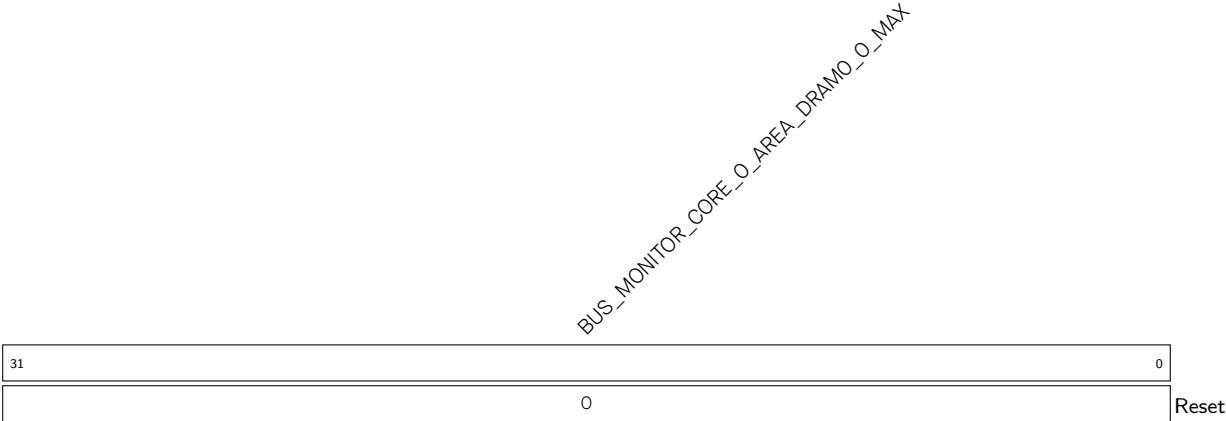
(R/W)

Register 14.17. BUS_MONITOR_CORE_O_AREA_DRAMO_O_MIN_REG (0x0010)



BUS_MONITOR_CORE_O_AREA_DRAMO_O_MIN Configures the lower bound address of Data bus region 0. (R/W)

Register 14.18. BUS_MONITOR_CORE_O_AREA_DRAMO_O_MAX_REG (0x0014)



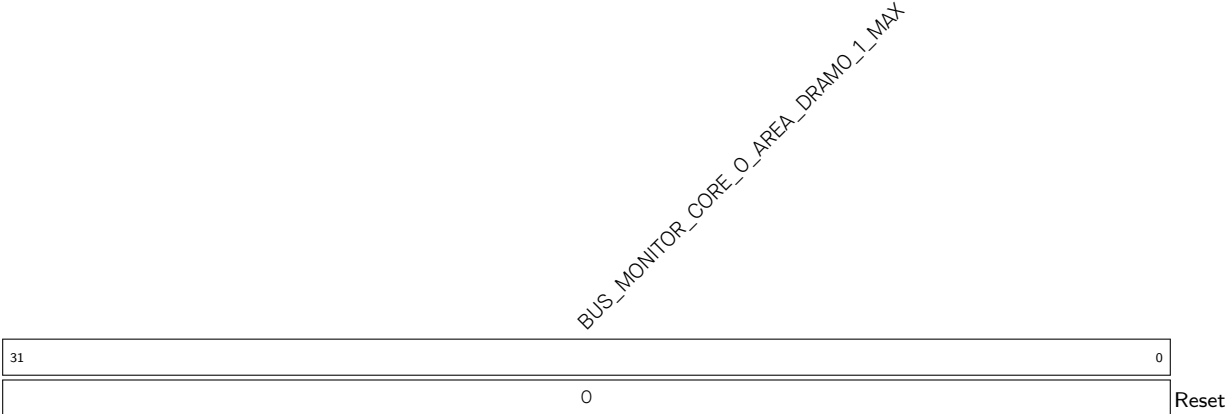
BUS_MONITOR_CORE_O_AREA_DRAMO_O_MAX Configures the upper bound address of Data bus region 0. (R/W)

Register 14.19. BUS_MONITOR_CORE_O_AREA_DRAMO_1_MIN_REG (0x0018)



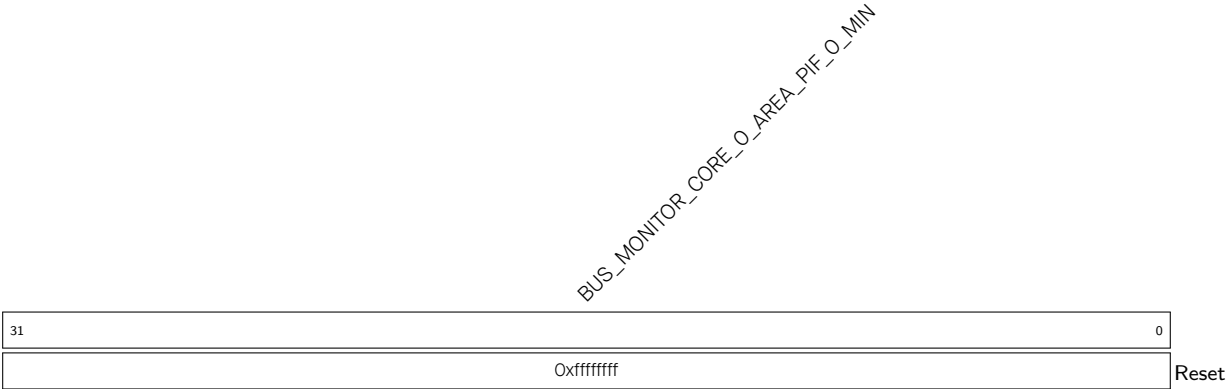
BUS_MONITOR_CORE_O_AREA_DRAMO_1_MIN Configures the lower bound address of Data bus region 1. (R/W)

Register 14.20. BUS_MONITOR_CORE_O_AREA_DRAMO_1_MAX_REG (0x001C)



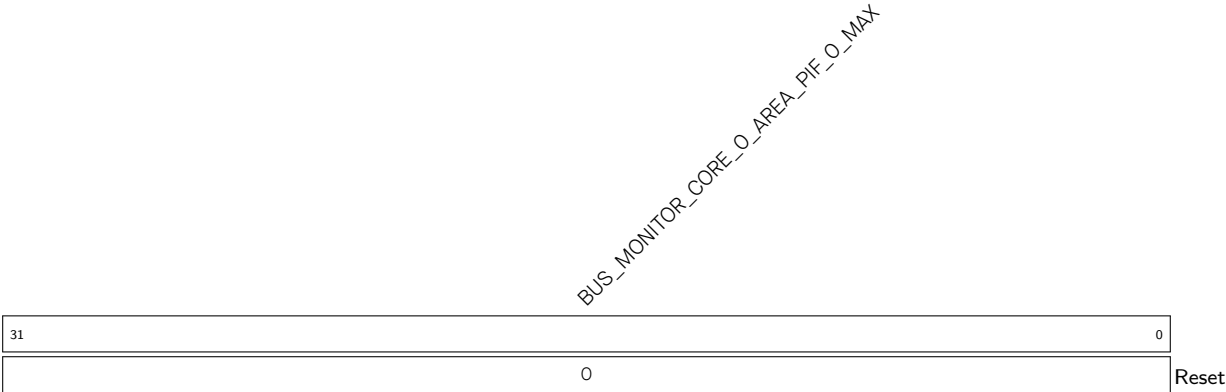
BUS_MONITOR_CORE_O_AREA_DRAMO_1_MAX Configures the upper bound address of Data bus region 1. (R/W)

Register 14.21. BUS_MONITOR_CORE_O_AREA_PIF_O_MIN_REG (0x0020)

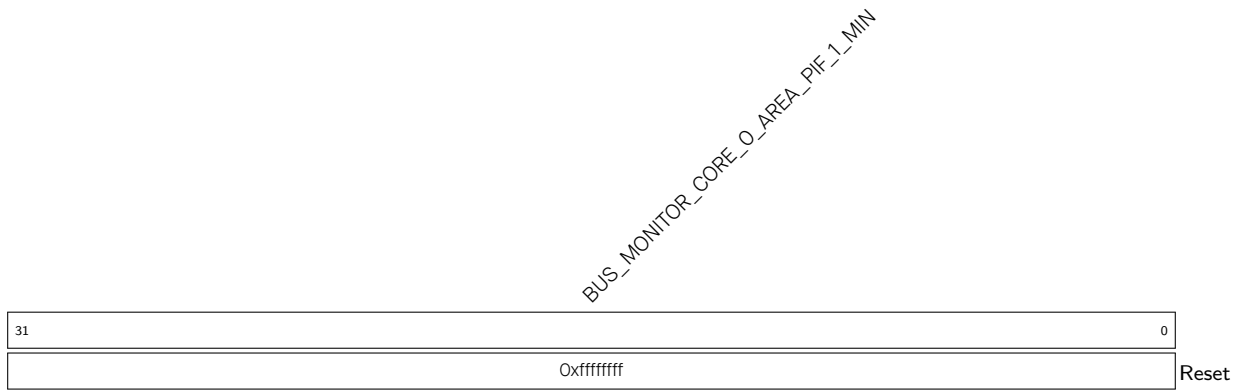


BUS_MONITOR_CORE_O_AREA_PIF_O_MIN Configures the lower bound address of Peripheral bus region 0. (R/W)

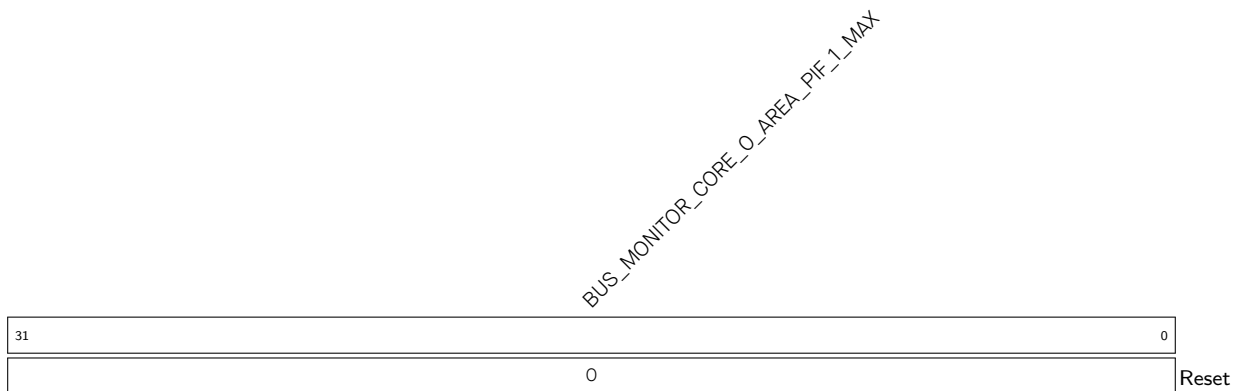
Register 14.22. BUS_MONITOR_CORE_O_AREA_PIF_O_MAX_REG (0x0024)



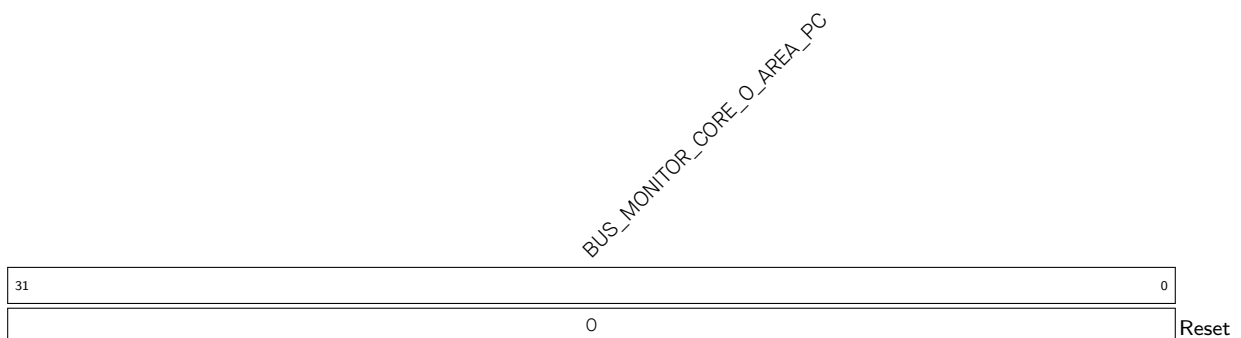
BUS_MONITOR_CORE_O_AREA_PIF_O_MAX Configures the upper bound address of Peripheral bus region 0. (R/W)

Register 14.23. BUS_MONITOR_CORE_O_AREA_PIF_1_MIN_REG (0x0028)

BUS_MONITOR_CORE_O_AREA_PIF_1_MIN Configures the lower bound address of Peripheral bus region 1. (R/W)

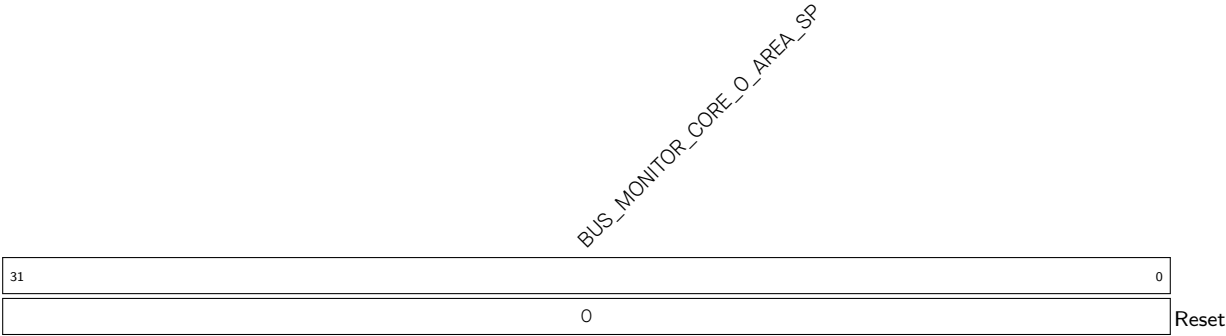
Register 14.24. BUS_MONITOR_CORE_O_AREA_PIF_1_MAX_REG (0x002C)

BUS_MONITOR_CORE_O_AREA_PIF_1_MAX Configures the upper bound address of Peripheral bus region 1. (R/W)

Register 14.25. BUS_MONITOR_CORE_O_AREA_PC_REG (0x0030)

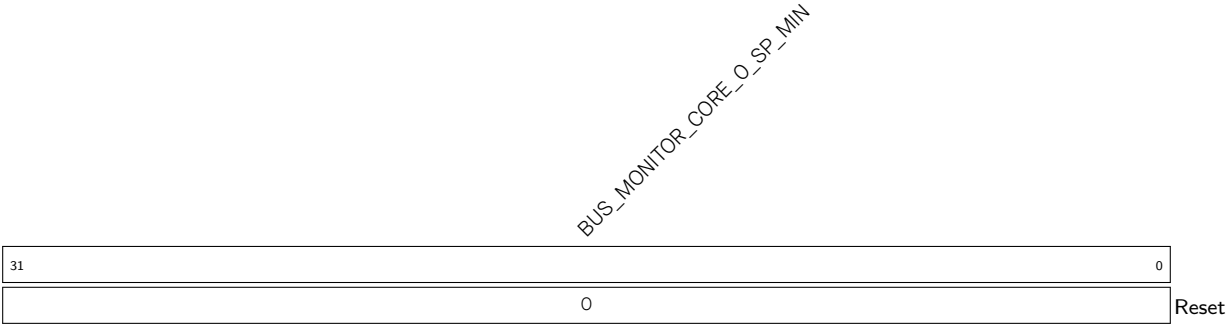
BUS_MONITOR_CORE_O_AREA_PC Represents the PC value when an interrupt is triggered during region monitoring. (RO)

Register 14.26. BUS_MONITOR_CORE_O_AREA_SP_REG (0x0034)



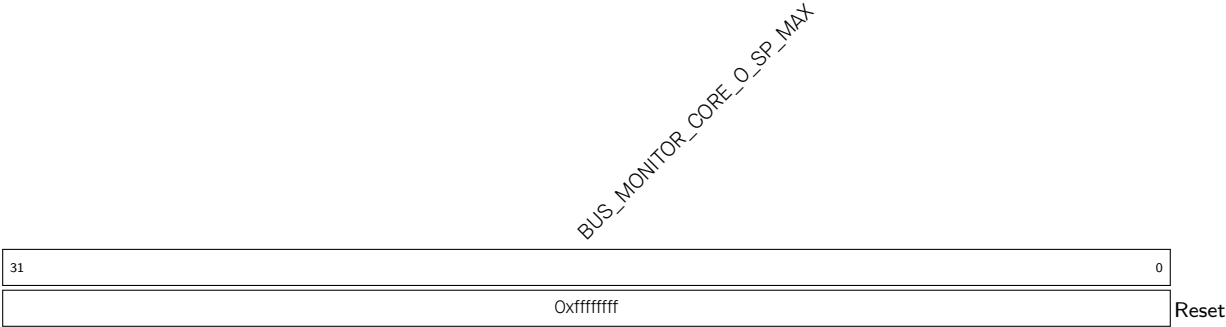
BUS_MONITOR_CORE_O_AREA_SP Represents the SP value when an interrupt is triggered during region monitoring. (RO)

Register 14.27. BUS_MONITOR_CORE_O_SP_MIN_REG (0x0038)



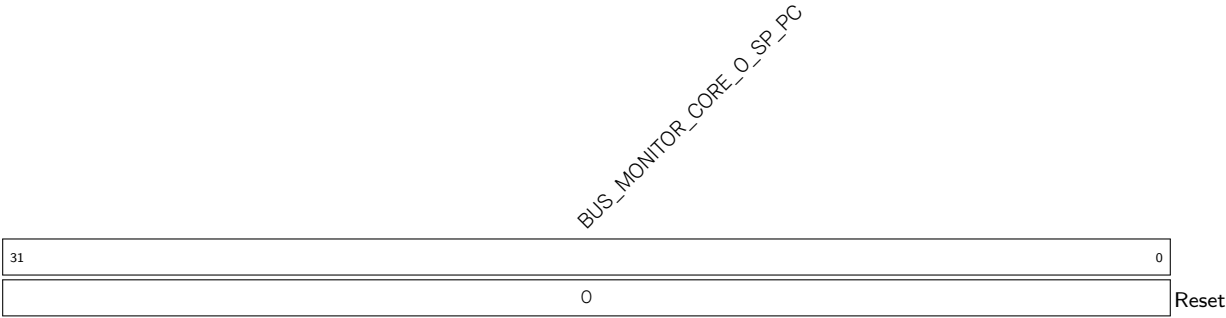
BUS_MONITOR_CORE_O_SP_MIN Configures the lower bound address of SP monitored region. (R/W)

Register 14.28. BUS_MONITOR_CORE_O_SP_MAX_REG (0x003C)



BUS_MONITOR_CORE_O_SP_MAX Configures the upper bound address of SP monitored region. (R/W)

Register 14.29. BUS_MONITOR_CORE_0_SP_PC_REG (0x0040)



BUS_MONITOR_CORE_0_SP_PC Represents the PC value during SP monitoring. (RO)

Register 14.30. BUS_MONITOR_CORE_O_INTR_RAW_REG (0x0004)

(reserved)																								BUS_MONITOR_CORE_O_SP_SPILL_MAX_RAW BUS_MONITOR_CORE_O_SP_SPILL_MIN_RAW BUS_MONITOR_CORE_O_AREA_PIF_1_RD_RAW BUS_MONITOR_CORE_O_AREA_PIF_1_WR_RAW BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_RAW BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_RAW BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_RAW BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_RAW																						
31																								10		9	8	7	6	5	4	3	2	1	0											
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0	0	0	0	0	0	0	0	0	0	0	0	0	Reset									

BUS_MONITOR_CORE_O_AREA_DRAMO_O_RD_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_DRAMO_O_RD_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_DRAMO_O_WR_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_DRAMO_O_WR_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_PIF_O_RD_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_PIF_O_RD_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_PIF_O_WR_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_PIF_O_WR_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_PIF_1_RD_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_PIF_1_RD_INT](#). (RO)

BUS_MONITOR_CORE_O_AREA_PIF_1_WR_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_AREA_PIF_1_WR_INT](#). (RO)

BUS_MONITOR_CORE_O_SP_SPILL_MIN_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_SP_SPILL_MIN_INT](#). (RO)

BUS_MONITOR_CORE_O_SP_SPILL_MAX_RAW The raw interrupt status of [BUS_MONITOR_CORE_O_SP_SPILL_MAX_INT](#). (RO)

Register 14.31. BUS_MONITOR_CORE_O_INTR_ENA_REG (0x0008)

(reserved)																						BUS_MONITOR_CORE_0_SP_SPILL_MAX_INTR_ENA BUS_MONITOR_CORE_0_SP_SPILL_MIN_INTR_ENA BUS_MONITOR_CORE_0_AREA_PIF_1_WR_INTR_ENA BUS_MONITOR_CORE_0_AREA_PIF_1_RD_INTR_ENA BUS_MONITOR_CORE_0_AREA_PIF_0_WR_INTR_ENA BUS_MONITOR_CORE_0_AREA_PIF_0_RD_INTR_ENA BUS_MONITOR_CORE_0_AREA_DRAMO_1_WR_INTR_ENA BUS_MONITOR_CORE_0_AREA_DRAMO_1_RD_INTR_ENA BUS_MONITOR_CORE_0_AREA_DRAMO_0_WR_INTR_ENA BUS_MONITOR_CORE_0_AREA_DRAMO_0_RD_INTR_ENA									
31																				10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset									

BUS_MONITOR_CORE_O_AREA_DRAMO_0_RD_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_DRAMO_0_RD_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_DRAMO_0_WR_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_DRAMO_0_WR_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_PIF_0_RD_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_PIF_0_RD_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_PIF_0_WR_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_PIF_0_WR_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_PIF_1_RD_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_PIF_1_RD_INT. (R/W)				
BUS_MONITOR_CORE_O_AREA_PIF_1_WR_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_AREA_PIF_1_WR_INT. (R/W)				
BUS_MONITOR_CORE_O_SP_SPILL_MIN_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_SP_SPILL_MIN_INT. (R/W)				
BUS_MONITOR_CORE_O_SP_SPILL_MAX_INTR_ENA	Write	1	to	enable
BUS_MONITOR_CORE_O_SP_SPILL_MAX_INT. (R/W)				

Register 14.32. BUS_MONITOR_CORE_O_INTR_CLR_REG (0x000C)

(reserved)																						BUS_MONITOR_CORE_O_SP_SPILL_MAX_CLR BUS_MONITOR_CORE_O_SP_SPILL_MIN_CLR BUS_MONITOR_CORE_O_AREA_PIF_1_WR_CLR BUS_MONITOR_CORE_O_AREA_PIF_1_RD_CLR BUS_MONITOR_CORE_O_AREA_PIF_0_WR_CLR BUS_MONITOR_CORE_O_AREA_PIF_0_RD_CLR BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_CLR BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_CLR BUS_MONITOR_CORE_O_AREA_DRAMO_0_WR_CLR BUS_MONITOR_CORE_O_AREA_DRAMO_0_RD_CLR																		
31																						10	9	8	7	6	5	4	3	2	1	0								
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																						0	0	0	0	0	0	0	0	0	0	0	Reset							

BUS_MONITOR_CORE_O_AREA_DRAMO_0_RD_CLR Write 1 to clear
[BUS_MONITOR_CORE_O_AREA_DRAMO_0_RD_INT.](#) (WT)

BUS_MONITOR_CORE_O_AREA_DRAMO_0_WR_CLR Write 1 to clear
[BUS_MONITOR_CORE_O_AREA_DRAMO_0_WR_INT.](#) (WT)

BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_CLR Write 1 to clear
[BUS_MONITOR_CORE_O_AREA_DRAMO_1_RD_INT.](#) (WT)

BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_CLR Write 1 to clear
[BUS_MONITOR_CORE_O_AREA_DRAMO_1_WR_INT.](#) (WT)

BUS_MONITOR_CORE_O_AREA_PIF_0_RD_CLR Write 1 to clear [BUS_MONITOR_CORE_O_AREA_PIF_0_RD_INT.](#)
(WT)

BUS_MONITOR_CORE_O_AREA_PIF_0_WR_CLR Write 1 to clear [BUS_MONITOR_CORE_O_AREA_PIF_0_WR_INT.](#)
(WT)

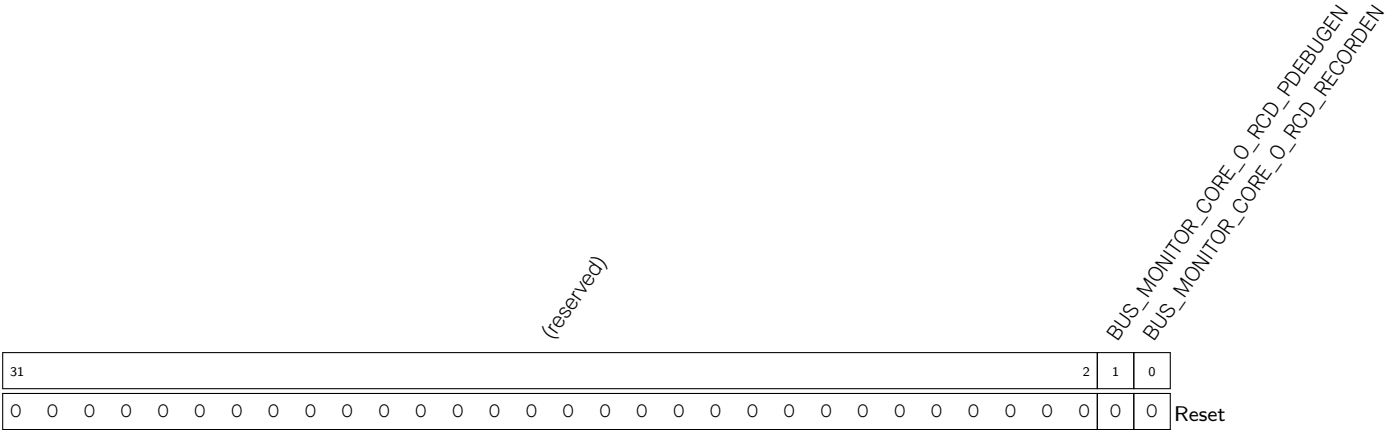
BUS_MONITOR_CORE_O_AREA_PIF_1_RD_CLR Write 1 to clear [BUS_MONITOR_CORE_O_AREA_PIF_1_RD_INT.](#)
(WT)

BUS_MONITOR_CORE_O_AREA_PIF_1_WR_CLR Write 1 to clear [BUS_MONITOR_CORE_O_AREA_PIF_1_WR_INT.](#)
(WT)

BUS_MONITOR_CORE_O_SP_SPILL_MIN_CLR Write 1 to clear [BUS_MONITOR_CORE_O_SP_SPILL_MIN_INT.](#)
(WT)

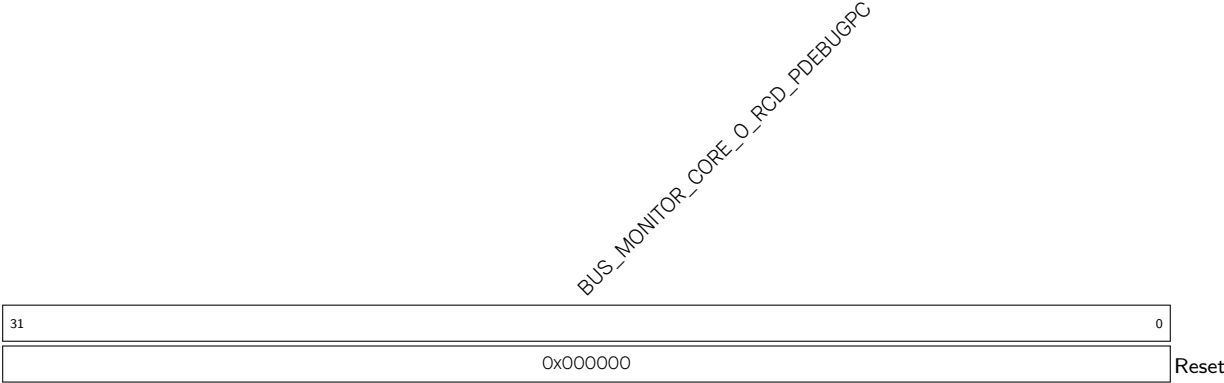
BUS_MONITOR_CORE_O_SP_SPILL_MAX_CLR Write 1 to clear [BUS_MONITOR_CORE_O_SP_SPILL_MAX_INT.](#)
(WT)

Register 14.33. BUS_MONITOR_CORE_O_RCD_EN_REG (0x0044)



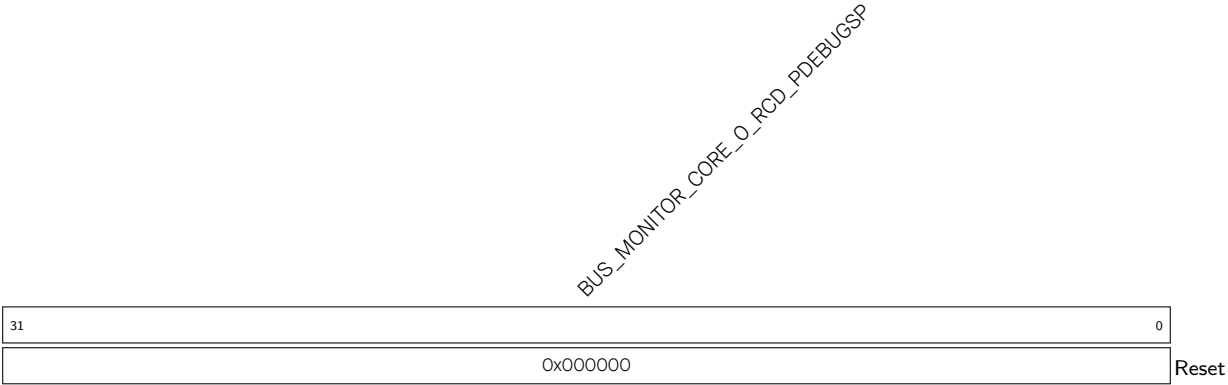
- BUS_MONITOR_CORE_O_RCD_RECORDEN** Configures whether to enable PC logging.
- 0: Disable
 - 1: [BUS_MONITOR_CORE_O_RCD_PDEBUGPC_REG](#) starts to record PC in real time
- (R/W)
- BUS_MONITOR_CORE_O_RCD_PDEBUGEN** Configures whether to enable HP CPU debugging.
- 0: Disable
 - 1: HP CPU outputs PC
- (R/W)

Register 14.34. BUS_MONITOR_CORE_O_RCD_PDEBUGPC_REG (0x0048)



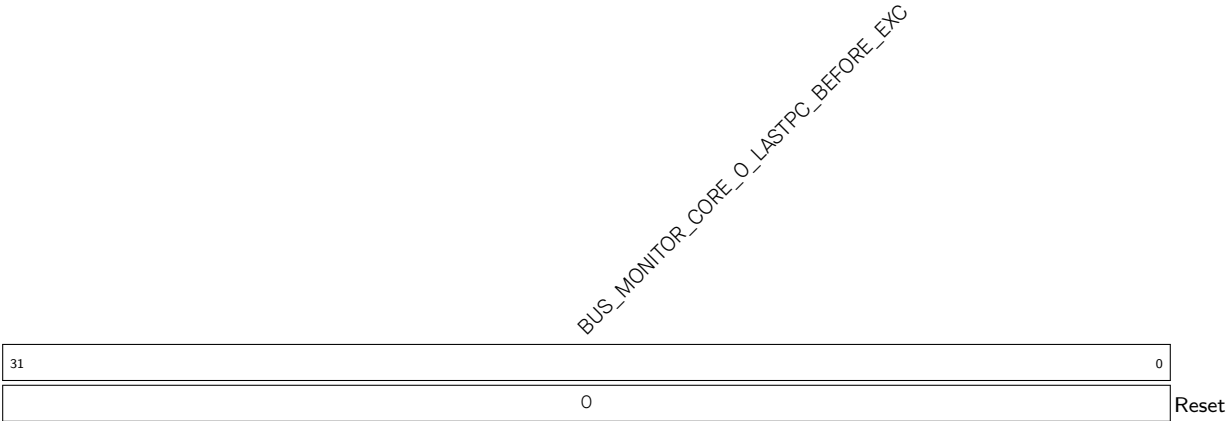
BUS_MONITOR_CORE_O_RCD_PDEBUGPC Represents the PC value at HP CPU reset. (RO)

Register 14.35. BUS_MONITOR_CORE_O_RCD_PDEBUGSP_REG (0x004C)



BUS_MONITOR_CORE_O_RCD_PDEBUGSP Represents SP. (RO)

Register 14.36. BUS_MONITOR_CORE_O_LASTPC_BEFORE_EXCEPTION_REG (0x0070)



BUS_MONITOR_CORE_O_LASTPC_BEFORE_EXC Represents the PC of the last command before the HP CPU enters exception. (RO)

Register 14.37. BUS_MONITOR_CORE_O_DEBUG_MODE_REG (0x0074)

(reserved)																														BUS_MONITOR_CORE_O_DEBUG_MODULE_ACTIVE	
																														BUS_MONITOR_CORE_O_DEBUG_MODE	BUS_MONITOR_CORE_O_DEBUG_MODULE_ACTIVE
31																											2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

- BUS_MONITOR_CORE_O_DEBUG_MODE** Represents whether RISC-V CPU (HP CPU) is in debugging mode.
1: In debugging mode
0: Not in debugging mode
(RO)
- BUS_MONITOR_CORE_O_DEBUG_MODULE_ACTIVE** Represents the status of the RISC-V CPU (HP CPU) debug module.
1: Active status
Other: Inactive status
(RO)

Register 14.38. BUS_MONITOR_CLOCK_GATE_REG (0x0108)

(reserved)																														BUS_MONITOR_CLK_EN	
																														BUS_MONITOR_CLK_EN	
31																											1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

- BUS_MONITOR_CLK_EN** Configures whether to enable the register clock gating.
0: Disable
1: Enable
(R/W)

Register 14.39. BUS_MONITOR_DATE_REG (0x03FC)

(reserved)				BUS_MONITOR_DATE																											
31	28	27																											0		
0	0	0	0	0x2109130																											Reset

BUS_MONITOR_DATE Version control register. (R/W)

Chapter 15

Power Supply Detector

15.1 Overview

ESP32-C5 has a power supply detector that can monitor the voltage on the power pins related to the on-chip clock and digital circuits. The power supply detector includes a brown-out detector and four voltage glitch detectors. The brown-out detector prevents the SoC from brown-out resets, while the voltage glitch detectors protect the SoC against voltage glitch attacks. The power detector operates in the always-on power domain, allowing it to monitor voltage at any time.

15.2 Features

- Brown-out detector supports two brown-out detection modes (Mode 0 and Mode 1)
 - Mode 0 supports:
 - * interrupt generation
 - * RF circuits power-down
 - * flash suspend triggering
 - * system reset
 - Mode 1 supports:
 - * system reset until the voltage returns to normal
- Voltage glitch detector can detect voltage glitches lasting 50 ns or longer and trigger a system reset.

15.3 Functional Description

15.3.1 Architecture

Figure 15.3-1 shows the architecture of the power supply detector. As shown, the power supply detector consists of one brown-out detector and four voltage glitch detectors.

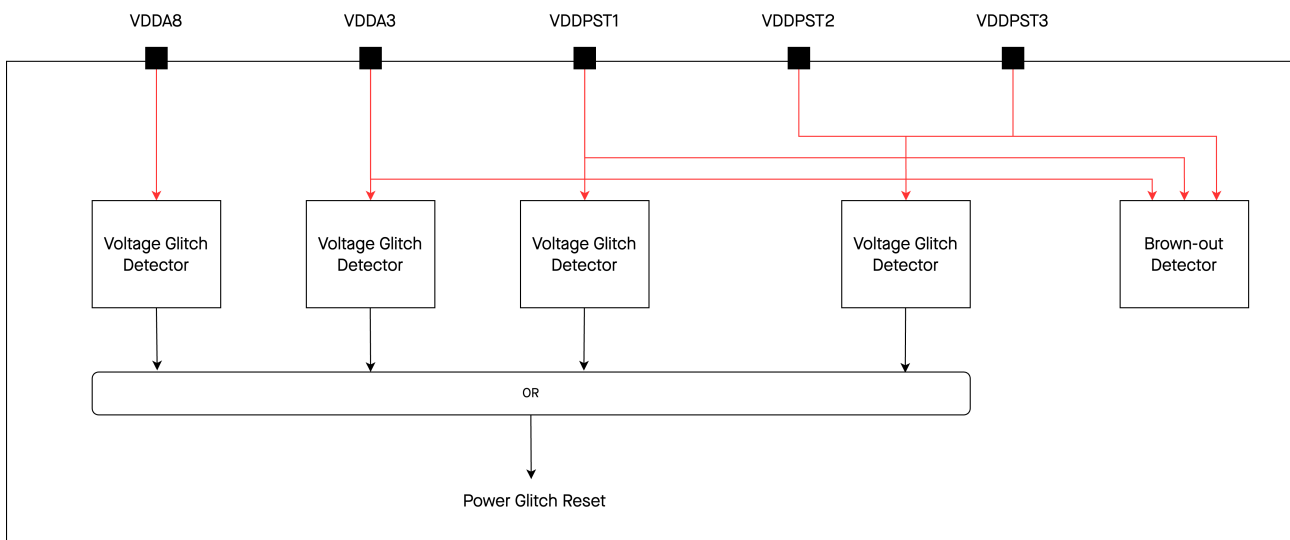


Figure 15.3-1. Architecture of Power Supply Detector

15.3.2 Brown-out Detector

The brown-out detector checks the voltage of pins VDDA3, VDDPST1, VDDPST2, and VDDPST3, about every 280 μ s. When the voltage on these pins falls below a predefined threshold of 2.7 V (default setting), the detector activates a signal to power down the power-hungry RF circuits. This action provides additional time for the digital system to save and transfer important data. The brown-out detector consumes very little power and remains active as long as the chip is powered on.

[LP_ANA_BOD_MODE0_INT_RAW](#) indicates the detection result from the brown-out detector. By default, this register reads a value of 0. It changes to 1 when the voltage on the monitored pin falls below a predefined threshold.

When a brown-out signal is detected, the brown-out detector handles it in one of the following two modes (Mode 1 is the default):

- Mode 0:
 - Mode 0 is enabled by setting the `bod_mode0_en` ([LP_ANA_BOD_MODE0_INTR_ENA](#)) signal.
 - The brown-out detector triggers an interrupt when the counter counts to the thresholds predefined in Int Comparer ([LP_ANA_BOD_MODE0_INTR_WAIT](#)). If [LP_ANA_BOD_MODE0_CLOSE_FLASH_ENA](#) is set, flash suspend will be triggered; if [LP_ANA_BOD_MODE0_PD_RF_ENA](#) is set, the RF module will be powered down.
 - The brown-out detector resets the chip based on the configuration of `bod_mode0_rst_sel` ([LP_ANA_BOD_MODE0_RESET_SEL](#)) when the brown-out counter counts to the thresholds predefined in Rst Comparer ([LP_ANA_BOD_MODE0_RESET_WAIT](#)). The reset is enabled by `bod_mode0_rst_en` ([LP_ANA_BOD_MODE0_RESET_ENA](#)).
- Mode 1: Resets the system directly.

The brown-out reset workflow is illustrated in the diagram below.

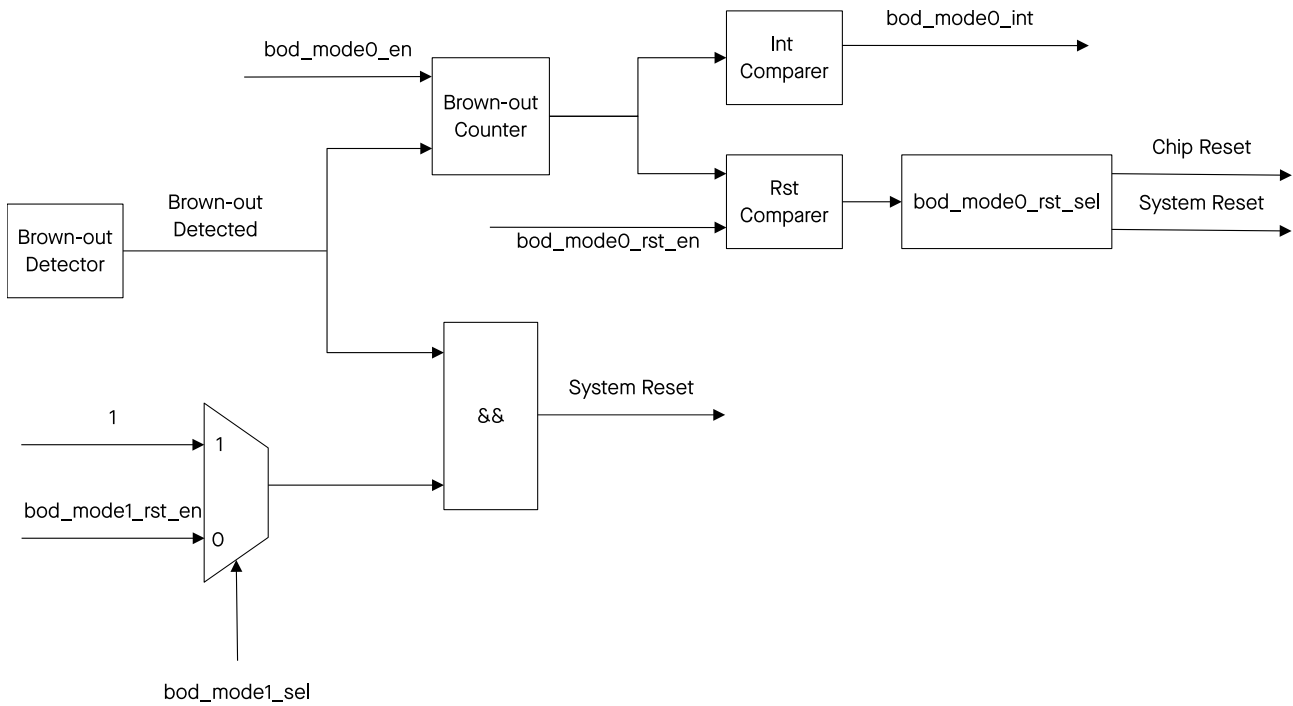


Figure 15.3-2. Brown-out Reset Workflow

Registers for controlling related signals are described below.

- `bod_mode0_en`: [LP_ANA_BOD_MODE0_INTR_ENA](#)
- `bod_mode0_rst_en`: [LP_ANA_BOD_MODE0_RESET_ENA](#)
- `bod_mode0_rst_sel`: [LP_ANA_BOD_MODE0_RESET_SEL](#) configures the reset type:
 - 0: chip reset
 - 1: system reset

For more information regarding chip reset and system reset, please refer to Chapter 6 [Reset and Clock](#).

- `bod_mode1_sel`: The first bit of [LP_ANA_ANA_FIB_ENA](#).
- `bod_mode1_rst_en`: [LP_ANA_BOD_MODE1_RESET_ENA](#)

15.3.3 Voltage Glitch Detectors

Figure 15.3-3 shows the structure of the voltage glitch detectors.

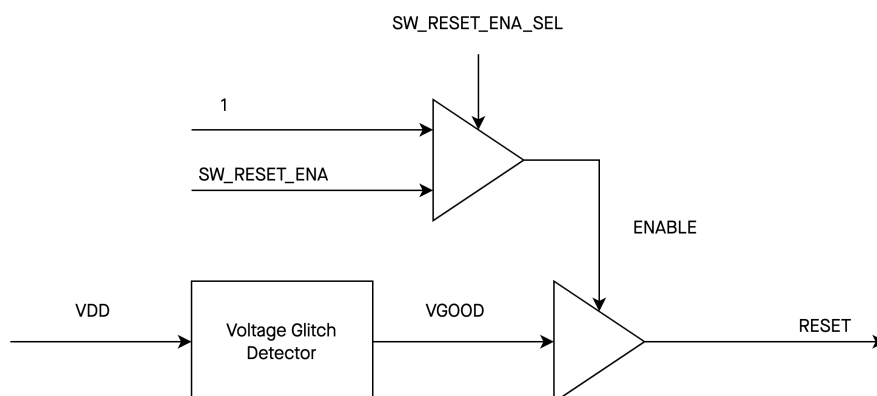


Figure 15.3-3. Structure of Voltage Glitch Detectors

The ESP32-C5 has four voltage glitch detectors, each of which can detect glitches longer than 50 ns on the corresponding power supply pins, including VDDPST1, VDDPST2, VDDPST3, VDDA8, and VDDA3. VDDPST2 and VDDPST3 are connected and thus share one voltage glitch detector. All resets generated by the detectors are combined into one voltage glitch reset.

The voltage glitch detectors' enable control is managed by SW_RESET_ENA_SEL. When set to 1, it is forcibly enabled by hardware, and when set to 0, it is controlled by the software register SW_RESET_ENA. The configuration for detecting different power pins is shown in the table below:

Table 15.3-1. Configuration for Detecting Power Pins

VDD Pins	SW_RESET_ENA	SW_RESET_ENA_SEL
VDDPST2/3	LP_ANA_PWR_GLITCH_RESET_ENA[0]	LP_ANA_ANA_FIB_ENA[2]
VDDPST1	LP_ANA_PWR_GLITCH_RESET_ENA[1]	LP_ANA_ANA_FIB_ENA[3]
VDDA3	LP_ANA_PWR_GLITCH_RESET_ENA[2]	LP_ANA_ANA_FIB_ENA[4]
VDDA8	LP_ANA_PWR_GLITCH_RESET_ENA[3]	LP_ANA_ANA_FIB_ENA[5]

15.4 Interrupts

ESP32-C5's power supply detector can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- LP_ANA_INTR (only for HP CPU)
- LP_ANA_LP_INTR (only for LP CPU)

The interrupt signals are generated by the power supply detector's internal interrupt sources which are listed in [Table 15.4-1](#) with their trigger conditions.

Table 15.4-1. Power Supply Detector's Internal Interrupt Source

Internal Interrupt Source	Trigger Condition	Interrupt Signal
LP_ANA_BOD_MODE0_INT	Triggered when the brown-out detector detects that the voltage is below the threshold.	LP_ANA_INTR
LP_ANA_BOD_MODE0_LP_INT	Triggered when the brown-out detector detects that the voltage is below the threshold.	LP_ANA_LP_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [15.5 Register Summary](#).

15.5 Register Summary

The addresses in this section are relative to the Power Supply Detector base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
LP_ANA_BOD_MODE0_CNTL_REG	Brownout detector mode 0 configuration register	0x0000	R/W
LP_ANA_BOD_MODE1_CNTL_REG	Brownout detector mode 1 configuration register	0x0004	R/W
LP_ANA_POWER_GLITCH_CNTL_REG	Voltage glitch configuration register	0x0008	R/W
LP_ANA_FIB_ENABLE_REG	Voltage glitch detectors' enable control register	0x000C	R/W
LP_ANA_INT_RAW_REG	LP_ANA_BOD_MODE0_INT raw interrupt	0x0010	R/WTC/SS
LP_ANA_INT_ST_REG	LP_ANA_BOD_MODE0_INT state interrupt	0x0014	RO
LP_ANA_INT_ENA_REG	LP_ANA_BOD_MODE0_INT enable register	0x0018	R/W
LP_ANA_INT_CLR_REG	LP_ANA_BOD_MODE0_INT clear register	0x001C	WT
LP_ANA_LP_INT_RAW_REG	LP_ANA_BOD_MODE0_LP_INT raw interrupt	0x0020	R/WTC/SS
LP_ANA_LP_INT_ST_REG	LP_ANA_BOD_MODE0_LP_INT state interrupt	0x0024	RO
LP_ANA_LP_INT_ENA_REG	LP_ANA_BOD_MODE0_LP_INT enable register	0x0028	R/W
LP_ANA_LP_INT_CLR_REG	LP_ANA_BOD_MODE0_LP_INT clear register	0x002C	WT
Version Control Registers			
LP_ANA_DATE_REG	Version control register	0x03FC	R/W

15.6 Registers

The addresses in this section are relative to the Power Supply Detector base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 15.1. LP_ANA_BOD_MODE0_CNTL_REG (0x0000)

LP_ANA_BOD_MODE0_RESET_ENA				LP_ANA_BOD_MODE0_RESET_WAIT				LP_ANA_BOD_MODE0_INTR_WAIT				LP_ANA_BOD_MODE0_PD_RF_ENA				(reserved)								
LP_ANA_BOD_MODE0_RESET_SEL				LP_ANA_BOD_MODE0_INTR_ENA				LP_ANA_BOD_MODE0_CNT_CLR																
31	30	29	28	27	18				17	8				7	6	5	0							
0	0	0	0	0x3ff				1				0				0	0	0	0	0	0	0	0	Reset

LP_ANA_BOD_MODE0_CLOSE_FLASH_ENA Configures whether to enable the brown-out detector to trigger flash suspend.
 0: Disable
 1: Enable
 (R/W)

LP_ANA_BOD_MODE0_PD_RF_ENA Configures whether to enable the brown-out detector to power down the RF module.
 0: Disable
 1: Enable
 (R/W)

LP_ANA_BOD_MODE0_INTR_WAIT Configures the time to generate an interrupt after the brown-out signal is valid. The unit is LP_FAST_CLK cycles. (R/W)

LP_ANA_BOD_MODE0_RESET_WAIT Configures the time to generate a reset after the brown-out signal is valid. The unit is LP_FAST_CLK cycles. (R/W)

LP_ANA_BOD_MODE0_CNT_CLR Configures whether to clear the count value of the brown-out detector.
 0: Do not clear
 1: Clear
 (R/W)

LP_ANA_BOD_MODE0_INTR_ENA Enables the interrupts for the brown-out detector mode 0. [LP_ANA_BOD_MODE0_INT_RAW](#) and [LP_ANA_BOD_MODE0_LP_INT_RAW](#) are valid only when this field is set to 1.
 0: Disable
 1: Enable
 (R/W)

LP_ANA_BOD_MODE0_RESET_SEL Configures the reset type when the brown-out detector is triggered.
 0: Chip reset
 1: System reset
 (R/W)

Continued on the next page...

Register 15.1. LP_ANA_BOD_MODE0_CNTL_REG (0x0000)

Continued from the previous page...

LP_ANA_BOD_MODE0_RESET_ENA Configures whether to enable reset for the brown-out detector.

0: Disable

1: Enable

(R/W)

Register 15.2. LP_ANA_BOD_MODE1_CNTL_REG (0x0004)

Diagram of the LP_ANALOG_BOD_MODEL_RESET_ENA register. The register is 32 bits wide. Bit 31 is labeled LP_ANA_BOD_MODEL_RESET_ENA. Bits 0 through 30 are labeled (reserved). The reset value for the entire register is 0.

LP_ANA_BOD_MODE1_RESET_ENA Configures whether to enable brown-out detector mode 1.

0: Disable

1: Enable

(R/W)

Register 15.3. LP_ANA_POWER_GLITCH_CNTL_REG (0x0008)

Diagram illustrating the structure of the LP_ANA_PWR_GLITCH_RESET_ENA register:

- Bit 31: (reserved)
- Bits 30-28: (reserved)
- Bit 27: LP_ANA_PWR_GLITCH_RESET_ENA
- Bits 26-0: (reserved)

LP_ANA_PWR_GLITCH_RESET_ENA Configures whether to enable the voltage glitch detectors.

Bit0, bit1, bit2, bit3 correspond to VDDPST2/3, VDDPST1, VDDA3, and VDDA8, respectively.

0: Disable

1: Enable

(R/W)

Register 15.4. LP_ANA_FIB_ENABLE_REG (0x000C)

LP_ANA_ANA_FIB_ENA																															
31																															0
0xffffffff																															
Reset																															

LP_ANA_ANA_FIB_ENA Controls the enable of the voltage glitch detectors. Bit2, bit3, bit4, bit5 correspond to VDDPST2/3, VDDPST1, VDDA3, and VDDA8, respectively.

0: Controlled by [LP_ANA_PWR_GLITCH_RESET_ENA](#)

1: Forcibly enabled by hardware

(R/W)

Register 15.5. LP_ANA_INT_RAW_REG (0x0010)

LP_ANA_BOD_MODE0_INT_RAW																															
31	30	(reserved)																													0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset																															

LP_ANA_BOD_MODE0_INT_RAW The raw interrupt status of [LP_ANA_BOD_MODE0_INT](#).
(R/WTC/SS)

Register 15.6. LP_ANA_INT_ST_REG (0x0014)

LP_ANA_BOD_MODE0_INT_ST																															
31	30	(reserved)																													0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Reset																															

LP_ANA_BOD_MODE0_INT_ST The masked interrupt status of [LP_ANA_BOD_MODE0_INT](#). (RO)

Register 15.7. LP_ANA_INT_ENA_REG (0x0018)

Diagram of the LP_ANA_BOD_MODE0_INT_ENA register. The register is 32 bits wide. Bit 31 is labeled LP_ANA_BOD_MODE0_INT_ENA. Bits 30-0 are labeled (reserved). The register is shown with a value of 0.

LP_ANA_BOD_MODE0_INT_ENA Write 1 to enable **LP_ANA_BOD_MODE0_INT**. (R/W)

Register 15.8. LP_ANA_INT_CLR_REG (0x001C)

Diagram illustrating the structure of the LP_ANA_BOD_MODE0_INT_CLR register. The register is 32 bits wide. Bit 31 is labeled LP_ANA_BOD_MODE0_INT_CLR. Bits 30-0 are labeled (reserved). The register is shown as a horizontal bar with bit positions 31, 30, and 0 marked. Below the bar, the bits are shown as a sequence of 32 zeros, with the last bit labeled Reset.

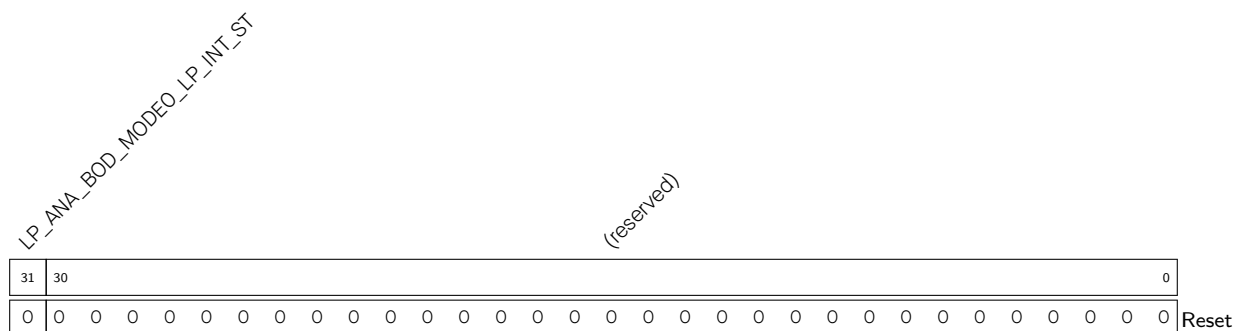
LP_ANA_BOD_MODE0_INT_CLR Write 1 to clear **LP_ANA_BOD_MODE0_INT**. (WT)

Register 15.9. LP_ANA_LP_INT_RAW_REG (0x0020)

Diagram of the LP_ANA_BOD_MODE0_LP_INT_RAW register. The register is 32 bits wide. Bit 31 is labeled 'LP_ANA_BOD_MODE0_LP_INT_RAW'. Bit 30 is labeled '(reserved)'. The register is reset to 0.

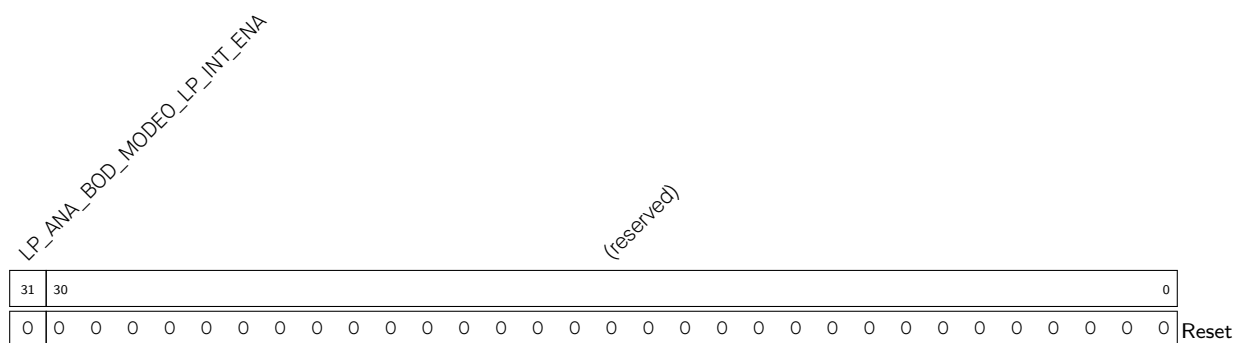
LP_ANA_BOD_MODE0_LP_INT_RAW The raw interrupt status of **LP_ANA_BOD_MODE0_LP_INT**.
(R/WTC/SS)

Register 15.10. LP_ANA_LP_INT_ST_REG (0x0024)



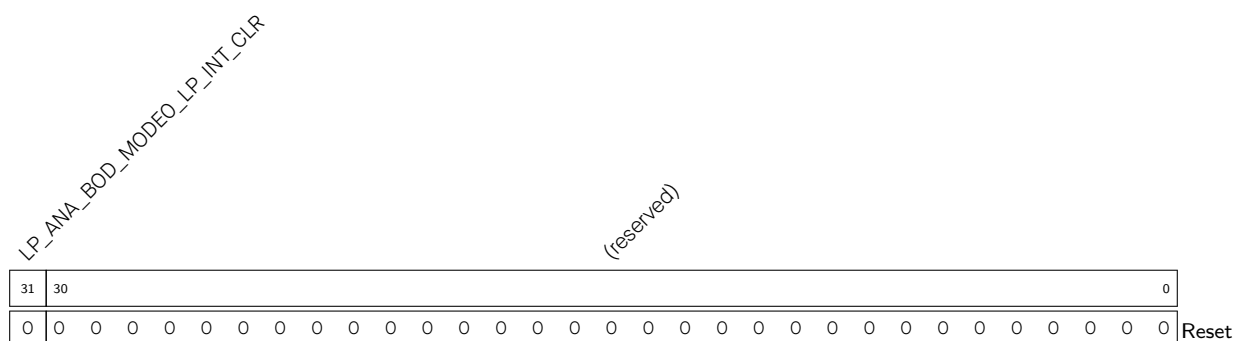
LP_ANA_BOD_MODEO_LP_INT_ST The masked interrupt status of [LP_ANA_BOD_MODEO_LP_INT](#).
(RO)

Register 15.11. LP_ANA_LP_INT_ENA_REG (0x0028)



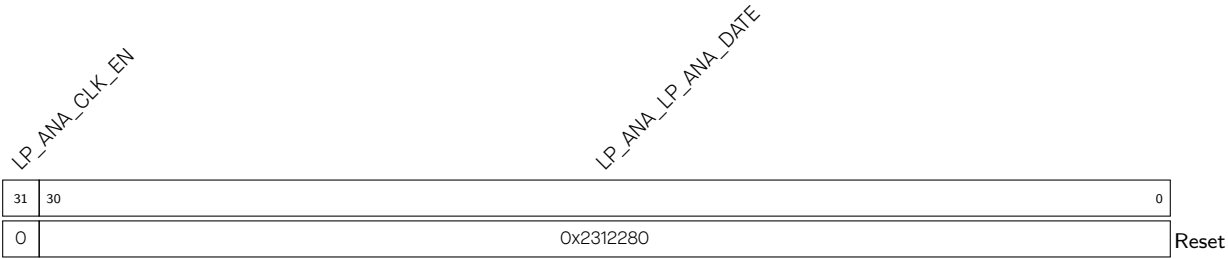
LP_ANA_BOD_MODE0_LP_INT_ENA Write 1 to enable **LP_ANA_BOD_MODE0_LP_INT**. (R/W)

Register 15.12. LP_ANA_LP_INT_CLR_REG (0x002C)



LP_ANA_BOD_MODE0_LP_INT_CLR Write 1 to clear **LP_ANA_BOD_MODE0_LP_INT**. (WT)

Register 15.13. LP_ANA_DATE_REG (0x03FC)



LP_ANA_LP_ANA_DATE Version control register. (R/W)

LP_ANA_CLK_EN Configures whether to force enable register clock.

- 0: Automatic clock gating
- 1: Force enable register clock

The configuration of this field does not effect the access of registers.

(R/W)

Part IV

Cryptography/Security Component

Dedicated to security features, this part explores cryptographic accelerators like AES and ECC. It also covers digital signatures, random number generation, and encryption/decryption algorithms, showcasing the SoC's capabilities in cryptography and secure data processing.

Chapter 16

AES Accelerator (AES)

16.1 Introduction

ESP32-C5 integrates an Advanced Encryption Standard (AES) accelerator, which is a hardware device that speeds up computation using AES algorithm significantly, compared to AES algorithms implemented solely in software. The AES accelerator integrated in ESP32-C5 has two working modes, which are [Typical AES](#) and [DMA-AES](#).

16.2 Features

The following functionality is supported:

- Typical AES working mode
 - AES-128/AES-256 encryption and decryption
- DMA-AES working mode
 - AES-128/AES-256 encryption and decryption
 - Block cipher mode
 - * ECB (Electronic Codebook)
 - * CBC (Cipher Block Chaining)
 - * OFB (Output Feedback)
 - * CTR (Counter)
 - * CFB8 (8-bit Cipher Feedback)
 - * CFB128 (128-bit Cipher Feedback)
 - Interrupt on completion of computation

16.3 Clock and Reset

The AES accelerator is activated by setting the [PCR_AES_CLK_EN](#) bit and clearing the [PCR_AES_RST_EN](#) bit in the [PCR_AES_CONF_REG](#) register. Besides, due to resource reuse between cryptography accelerator modules, users also need to additionally clear the [PCR_DS_RST_EN](#) bit in the [PCR_DS_CONF_REG](#) register.

16.4 AES Working Modes

The AES accelerator integrated in ESP32-C5 has two working modes, which are [Typical AES](#) and [DMA-AES](#).

- Typical AES Working Mode:
 - Supports encryption and decryption using cryptographic keys of 128 and 256 bits, specified in [NIST FIPS 197](#).

In this working mode, the plaintext and ciphertext is written and read via CPU directly.

- DMA-AES Working Mode:
 - Supports encryption and decryption using cryptographic keys of 128 and 256 bits, specified in [NIST FIPS 197](#);
 - Supports block cipher modes ECB, CBC, OFB, CTR, CFB8, and CFB128 under [NIST SP 800-38A](#).

In this working mode, the plaintext and ciphertext are written and read via DMA. An interrupt will be generated when operation completes.

Users can choose the working mode for AES accelerator by configuring the [AES_DMA_ENABLE_REG](#) register according to Table 16.4-1 below.

Table 16.4-1. AES Accelerator Working Mode

AES_DMA_ENABLE_REG	Working Mode
0	Typical AES
1	DMA-AES

Users can choose the length of cryptographic keys and encryption/decryption by configuring the [AES_MODE_REG](#) register according to Table 16.4-2 below.

Table 16.4-2. Key Length and Encryption/Decryption

AES_MODE_REG [2:0]	Key Length and Encryption / Decryption
0	AES-128 encryption
1	reserved
2	AES-256 encryption
3	reserved
4	AES-128 decryption
5	reserved
6	AES-256 decryption
7	reserved

For a detailed introduction to these two working modes, please refer to Section 16.5 and Section 16.6 below.

Notice:

ESP32-C5's [Digital Signature Algorithm \(DSA\)](#) module will call the AES accelerator. Therefore, users cannot access the AES accelerator when [Digital Signature Algorithm \(DSA\)](#) module is working.

16.5 Typical AES Working Mode

In the Typical AES working mode, users can check the working status of the AES accelerator by inquiring the [AES_STATE_REG](#) register and comparing the return value against the Table 16.5-1 below.

Table 16.5-1. Working Status under Typical AES Working Mode

AES_STATE_REG	Status	Description
0	IDLE	The AES accelerator is idle or completed operation.
1	WORK	The AES accelerator is in the middle of an operation.

16.5.1 Key, Plaintext, and Ciphertext

The encryption or decryption key is stored in [AES_KEY_n_REG](#), which is a set of eight 32-bit registers.

- For AES-128 encryption or decryption, the 128-bit key is stored in [AES_KEY_0_REG](#) ~ [AES_KEY_3_REG](#).
- For AES-256 encryption or decryption, the 256-bit key is stored in [AES_KEY_0_REG](#) ~ [AES_KEY_7_REG](#).

The plaintext and ciphertext are stored in [AES_TEXT_IN_m_REG](#) and [AES_TEXT_OUT_m_REG](#), which are two sets of four 32-bit registers.

- For AES-128 or AES-256 encryption, the [AES_TEXT_IN_m_REG](#) registers are initialized with plaintext. Then, the AES accelerator stores the ciphertext into [AES_TEXT_OUT_m_REG](#) after operation.
- For AES-128 or AES-256 decryption, the [AES_TEXT_IN_m_REG](#) registers are initialized with ciphertext. Then, the AES accelerator stores the plaintext into [AES_TEXT_OUT_m_REG](#) after operation.

16.5.2 Endianness

Text Endianness

In Typical AES working mode, the AES accelerator uses cryptographic keys to encrypt and decrypt data in blocks of 128 bits. When filling data into [AES_TEXT_IN_m_REG](#) register or reading result from [AES_TEXT_OUT_m_REG](#) registers, users should follow the text endianness type specified in Table 16.5-2.

Table 16.5-2. Text Endianness Type for Typical AES

Plaintext/Ciphertext					
State ¹		c ²			
		0	1	2	3
r	0	AES_TEXT_x_0_REG[7:0]	AES_TEXT_x_1_REG[7:0]	AES_TEXT_x_2_REG[7:0]	AES_TEXT_x_3_REG[7:0]
	1	AES_TEXT_x_0_REG[15:8]	AES_TEXT_x_1_REG[15:8]	AES_TEXT_x_2_REG[15:8]	AES_TEXT_x_3_REG[15:8]
	2	AES_TEXT_x_0_REG[23:16]	AES_TEXT_x_1_REG[23:16]	AES_TEXT_x_2_REG[23:16]	AES_TEXT_x_3_REG[23:16]
	3	AES_TEXT_x_0_REG[31:24]	AES_TEXT_x_1_REG[31:24]	AES_TEXT_x_2_REG[31:24]	AES_TEXT_x_3_REG[31:24]

¹ The definition of “State (including c and r)” is described in Section 3.4 *The State* in [NIST FIPS 197](#).

² Where **x** = IN or OUT.

Key Endianness

In Typical AES working mode, when filling keys into [AES_KEY_m_REG](#) registers, users should follow the key endianness type specified in [Table 16.5-3](#) and [Table 16.5-4](#).

Table 16.5-3. Key Endianness Type for AES-128 Encryption and Decryption

Bit ¹	w[0]	w[1]	w[2]	w[3] ²
[31:24]	AES_KEY_O_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_3_REG[7:0]
[23:16]	AES_KEY_O_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_3_REG[15:8]
[15:8]	AES_KEY_O_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_3_REG[23:16]
[7:0]	AES_KEY_O_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_3_REG[31:24]

¹ Column “Bit” specifies the bytes of each word stored in w[0] ~ w[3].
² w[0] ~ w[3] are “the first Nk words of the expanded key” as specified in Section 5.2 *Key Expansion* in [NIST FIPS 197](#).

Table 16.5-4. Key Endianness Type for AES-256 Encryption and Decryption

Bit ¹	w[0]	w[1]	w[2]	w[3]	w[4]	w[5]	w[6]	w[7] ²
[31:24]	AES_KEY_O_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_3_REG[7:0]	AES_KEY_4_REG[7:0]	AES_KEY_5_REG[7:0]	AES_KEY_6_REG[7:0]	AES_KEY_7_REG[7:0]
[23:16]	AES_KEY_O_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_3_REG[15:8]	AES_KEY_4_REG[15:8]	AES_KEY_5_REG[15:8]	AES_KEY_6_REG[15:8]	AES_KEY_7_REG[15:8]
[15:8]	AES_KEY_O_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_3_REG[23:16]	AES_KEY_4_REG[23:16]	AES_KEY_5_REG[23:16]	AES_KEY_6_REG[23:16]	AES_KEY_7_REG[23:16]
[7:0]	AES_KEY_O_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_3_REG[31:24]	AES_KEY_4_REG[31:24]	AES_KEY_5_REG[31:24]	AES_KEY_6_REG[31:24]	AES_KEY_7_REG[31:24]

¹ Column “Bit” specifies the bytes of each word stored in w[0] ~ w[7].
² w[0] ~ w[7] are “the first Nk words of the expanded key” as specified in Chapter 5.2 *Key Expansion* in [NIST FIPS 197](#).

16.5.3 Operation Process

Single Operation

1. Write 0 to the [AES_DMA_ENABLE_REG](#) register.
2. Initialize registers [AES_MODE_REG](#), [AES_KEY_n_REG](#), and [AES_TEXT_IN_m_REG](#).
3. Start operation by writing 1 to the [AES_TRIGGER_REG](#) register.
4. Wait till the content of the [AES_STATE_REG](#) register becomes 0, which indicates the operation is completed.
5. Read results from the [AES_TEXT_OUT_m_REG](#) register.

Consecutive Operations

In consecutive operations, primarily the input [AES_TEXT_IN_m_REG](#) and output [AES_TEXT_OUT_m_REG](#) registers (*m*: 0-3) are being written and read, while the content of [AES_DMA_ENABLE_REG](#), [AES_MODE_REG](#), and [AES_KEY_n_REG](#) is kept unchanged. Therefore, the initialization can be simplified during the consecutive operation.

1. Write 0 to the [AES_DMA_ENABLE_REG](#) register before starting the first operation.
2. Initialize registers [AES_MODE_REG](#) and [AES_KEY_n_REG](#) before starting the first operation.
3. Update the content of [AES_TEXT_IN_m_REG](#).
4. Start operation by writing 1 to the [AES_TRIGGER_REG](#) register.
5. Wait till the content of the [AES_STATE_REG](#) register becomes 0, which indicates the operation completes.
6. Read results from the [AES_TEXT_OUT_m_REG](#) register, and return to Step 3 to continue the next operation.

16.6 DMA-AES Working Mode

In the DMA-AES working mode, the AES accelerator supports six block cipher modes: ECB, CBC, OFB, CTR, CFB8, and CFB128. Users can choose the block cipher mode by configuring the [AES_BLOCK_MODE_REG](#) register according to Table 16.6-1 below.

Table 16.6-1. Block Cipher Mode

AES_BLOCK_MODE_REG [2:0]	Block Cipher Mode
0	ECB (Electronic Codebook)
1	CBC (Cipher Block Chaining)
2	OFB (Output Feedback)
3	CTR (Counter)
4	CFB8 (8-bit Cipher Feedback)
5	CFB128 (128-bit Cipher Feedback)
6	reserved
7	reserved

Users can check the working status of the AES accelerator by inquiring the [AES_STATE_REG](#) register and comparing the return value against the Table 16.6-2 below.

Table 16.6-2. Working Status under DMA-AES Working mode

AES_STATE_REG	Status	Description
0	IDLE	The AES accelerator is idle.
1	WORK	The AES accelerator is in the middle of an operation.
2	DONE	The AES accelerator completed operations.

When working in the DMA-AES working mode, the AES accelerator supports interrupt on the completion of computation. To enable this function, write 1 to the [AES_INT_ENA_REG](#) register. By default, the interrupt function is disabled. Also, note that the interrupt should be cleared by software after use.

16.6.1 Key, Plaintext, and Ciphertext

Block Operation

During the block operations, the AES accelerator reads source data from DMA, and writes result data to DMA after the computation.

- For encryption, DMA reads plaintext from memory, then passes it to AES as source data. After computation, AES passes ciphertext as result data back to DMA to write into memory.
- For decryption, DMA reads ciphertext from memory, then passes it to AES as source data. After computation, AES passes plaintext as result data back to DMA to write into memory.

During block operations, the lengths of the source data and result data are the same. The total computation time is reduced because the DMA data operation and AES computation can happen concurrently.

The length of source data for AES accelerator under DMA-AES working mode must be 128 bits or the integral multiples of 128 bits. Otherwise, trailing zeros will be added to the original source data, so the length of source data equals to the nearest integral multiples of 128 bits. Please see details in Table 16.6-3 below.

Table 16.6-3. TEXT-PADDING

Function : TEXT-PADDING()	
Input	: X , bit string.
Output	: $Y = \text{TEXT-PADDING}(X)$, whose length is the nearest integral multiples of 128 bits.
Steps Let us assume that X is a data-stream that can be split into n parts as following: $X = X_1 X_2 \cdots X_{n-1} X_n$ Here, the lengths of $X_1, X_2, \cdots, X_{n-1}$ all equal to 128 bits, and the length of X_n is t ($0 \leq t \leq 127$). If $t = 0$, then $\text{TEXT-PADDING}(X) = X;$ If $0 < t \leq 127$, define a 128-bit block, X_n^*, and let $X_n^* = X_n 0^{128-t}$, then $\text{TEXT-PADDING}(X) = X_1 X_2 \cdots X_{n-1} X_n^* = X 0^{128-t}$	

16.6.2 Endianness

Under the DMA-AES working mode, the transmission of source data and result data for AES accelerator is solely controlled by DMA. Therefore, the AES accelerator cannot control the Endianness of the source data and result data, but does have requirement on how these data should be stored in memory and on the length of the data.

For example, let us assume DMA needs to write the following data into memory at address 0x0280.

- Data represented in hexadecimal:
 - 0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20
- Data Length:
 - Equals to 2 blocks.

Then, this data will be stored in memory as shown in Table 16.6-4 below.

Table 16.6-4. Text Endianness for DMA-AES

Address	Byte	Address	Byte	Address	Byte	Address	Byte
0x0280	0x01	0x0281	0x02	0x0282	0x03	0x0283	0x04
0x0284	0x05	0x0285	0x06	0x0286	0x07	0x0287	0x08
0x0288	0x09	0x0289	0x0A	0x028A	0x0B	0x028B	0x0C
0x028C	0x0D	0x028D	0x0E	0x028E	0x0F	0x028F	0x10
0x0290	0x11	0x0291	0x12	0x0292	0x13	0x0293	0x14
0x0294	0x15	0x0295	0x16	0x0296	0x17	0x0297	0x18
0x0298	0x19	0x0299	0x1A	0x029A	0x1B	0x029B	0x1C
0x029C	0x1D	0x029D	0x1E	0x029E	0x1F	0x029F	0x20

16.6.3 Standard Incrementing Function

AES accelerator provides two Standard Incrementing Functions for the CTR block operation, which are INC₃₂ and INC₁₂₈ Standard Incrementing Functions. By setting the [AES_INC_SEL_REG](#) register to 0 or 1, users can choose the INC₃₂ or INC₁₂₈ functions respectively. For details on the Standard Incrementing Function, please see Chapter B.1 *The Standard Incrementing Function* in [NIST SP 800-38A](#).

16.6.4 Block Number

Register [AES_BLOCK_NUM_REG](#) stores the Block Number of plaintext P or ciphertext C . The length of this register equals to $\text{length}(\text{TEXT-PADDING}(P))/128$ or $\text{length}(\text{TEXT-PADDING}(C))/128$. The AES accelerator only uses this register when working in the DMA-AES mode.

16.6.5 Initialization Vector

[AES_IV_MEM](#) is a 16-byte memory, which is only available for AES accelerator working in block operations. For CBC/OFB/CFB8/CFB128 operations, the [AES_IV_MEM](#) memory stores the Initialization Vector (IV). For the CTR operation, the [AES_IV_MEM](#) memory stores the Initial Counter Block (ICB).

Both IV and ICB are 128-bit strings, which can be divided into Byte0, Byte1, Byte2 $\dots$ Byte15 (from left to right). [AES_IV_MEM](#) stores data following the Endianness pattern presented in Table 16.6-4, i.e. the most significant (i.e., left-most) byte Byte0 is stored at the lowest address while the least significant (i.e., right-most) byte Byte15 at the highest address.

For more details on IV and ICB, please refer to [NIST SP 800-38A](#).

16.6.6 Block Operation Process

1. Select one of DMA channels to connect with AES, configure the DMA linked list, and then start DMA. For details, please refer to Chapter 3 *GDMA Controller (GDMA)*.
2. Initialize the AES accelerator-related registers:
 - Write 1 to the [AES_DMA_ENABLE_REG](#) register.
 - Configure the [AES_INT_ENA_REG](#) register to enable or disable the interrupt function.
 - Initialize registers [AES_MODE_REG](#) and [AES_KEY_n_REG](#).
 - Select block cipher mode by configuring the [AES_BLOCK_MODE_REG](#) register. For details, see Table 16.6-1.
 - Initialize the [AES_BLOCK_NUM_REG](#) register. For details, see Section 16.6.4.
 - Initialize the [AES_INC_SEL_REG](#) register (only needed when AES accelerator is working under CTR block operation).
 - Initialize the [AES_IV_MEM](#) memory (This is always needed except for ECB block operation).
3. Start operation by writing 1 to the [AES_TRIGGER_REG](#) register.
4. Wait for the completion of computation, which happens when the content of [AES_STATE_REG](#) becomes 2 or the AES interrupt occurs.

5. Check if DMA completes data transmission from AES to memory. At this time, DMA had already written the result data in memory, which can be accessed directly. For details on DMA, please refer to Chapter 3 [GDMA Controller \(GDMA\)](#).
6. Clear interrupt by writing 1 to the [AES_INT_CLEAR_REG](#) register, if any AES interrupt occurred during the computation.
7. Release the AES accelerator by writing 1 to the [AES_DMA_EXIT_REG](#) register. After this, the content of the [AES_STATE_REG](#) register becomes 0. Note that, you can release DMA earlier, but only after Step 4 is completed.

16.7 Anti-Attack Pseudo-Round Function

To enhance anti-attack performance, ESP32-C5's AES incorporates a pseudo-round function.

When the pseudo-round function is enabled, AES randomly inserts pseudo-rounds before and after the original operation rounds. During a pseudo-round, AES uses a pseudo-key, automatically generated by the hardware, to perform a round of dummy operations. These operations do not affect the original AES operation results.

Users can enable the pseudo-round function by setting [AES_PSEUDO_EN](#) to 1. The pseudo-round configuration is as follows.

Total number of pseudo-rounds

The total number of pseudo-rounds randomly inserted into an AES operation is controlled by the following register fields:

- `pseudo_base`: Equal to the value of [AES_PSEUDO_BASE](#), which configures the basic number of pseudo-rounds.
- `pseudo_inc`: Equal to the value of [AES_PSEUDO_INC](#), which configures the random incremental number of pseudo-rounds.

The total number of pseudo-rounds will randomly fall into the range:

$$[\text{pseudo\_base}, \text{pseudo\_base} + (2^{\text{pseudo\_inc}} - 1)]$$

Random number update frequency

Users can set the frequency of random key updates in the pseudo-round function by configuring [AES_PSEUDO_RNG_CNT](#). A higher value results in a higher update frequency. It is generally recommended to set this value to the maximum of 7.

16.8 Memory Summary

The addresses in this section are relative to the AES accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Size (byte)	Starting Address	Ending Address	Access
AES_IV_MEM	Memory IV	16 bytes	0x0050	0x005F	R/W

16.9 Register Summary

The addresses in this section are relative to the AES accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

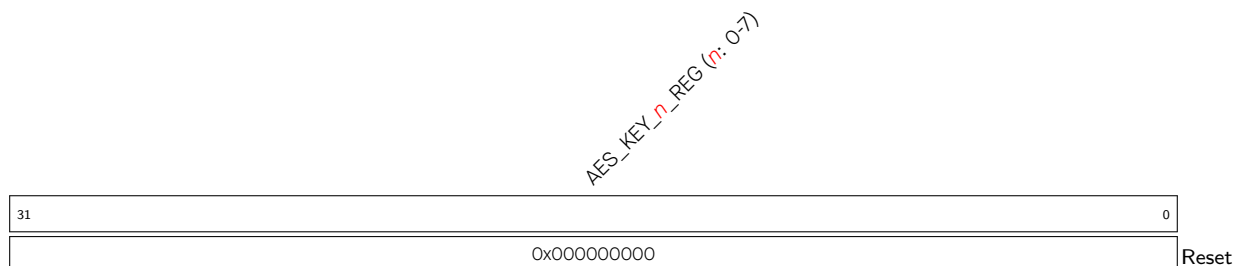
Name	Description	Address	Access
Key Registers			
AES_KEY_0_REG	AES key data register 0	0x0000	R/W
AES_KEY_1_REG	AES key data register 1	0x0004	R/W
AES_KEY_2_REG	AES key data register 2	0x0008	R/W
AES_KEY_3_REG	AES key data register 3	0x000C	R/W
AES_KEY_4_REG	AES key data register 4	0x0010	R/W
AES_KEY_5_REG	AES key data register 5	0x0014	R/W
AES_KEY_6_REG	AES key data register 6	0x0018	R/W
AES_KEY_7_REG	AES key data register 7	0x001C	R/W
TEXT_IN Registers			
AES_TEXT_IN_0_REG	Source text data register 0	0x0020	R/W
AES_TEXT_IN_1_REG	Source text data register 1	0x0024	R/W
AES_TEXT_IN_2_REG	Source text data register 2	0x0028	R/W
AES_TEXT_IN_3_REG	Source text data register 3	0x002C	R/W
TEXT_OUT Registers			
AES_TEXT_OUT_0_REG	Result text data register 0	0x0030	R/W
AES_TEXT_OUT_1_REG	Result text data register 1	0x0034	R/W
AES_TEXT_OUT_2_REG	Result text data register 2	0x0038	R/W
AES_TEXT_OUT_3_REG	Result text data register 3	0x003C	R/W
Control/Configuration Registers			
AES_MODE_REG	Key length and encryption/decryption configuration register	0x0040	R/W
AES_TRIGGER_REG	Operation start control register	0x0048	WT
AES_DMA_ENABLE_REG	Working mode configuration register	0x0090	R/W
AES_BLOCK_MODE_REG	Block cipher mode configuration register	0x0094	R/W
AES_BLOCK_NUM_REG	Block number configuration register	0x0098	R/W
AES_INC_SEL_REG	Standard incrementing function register	0x009C	R/W
AES_DMA_EXIT_REG	Operation exit control register	0x00B8	WT
AES_PSEUDO_REG	Pseudo-round function configuration register	0x00D0	R/W
Status Register			
AES_STATE_REG	Operation status register	0x004C	RO
Interrupt Registers			
AES_INT_CLEAR_REG	DMA-AES interrupt clear register	0x00AC	WT
AES_INT_ENA_REG	DMA-AES interrupt enable register	0x00B0	R/W
Version control register			
AES_DATE_REG	AES version control register	0x00B4	R/W

16.10 Registers

The addresses in this section are relative to the AES accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

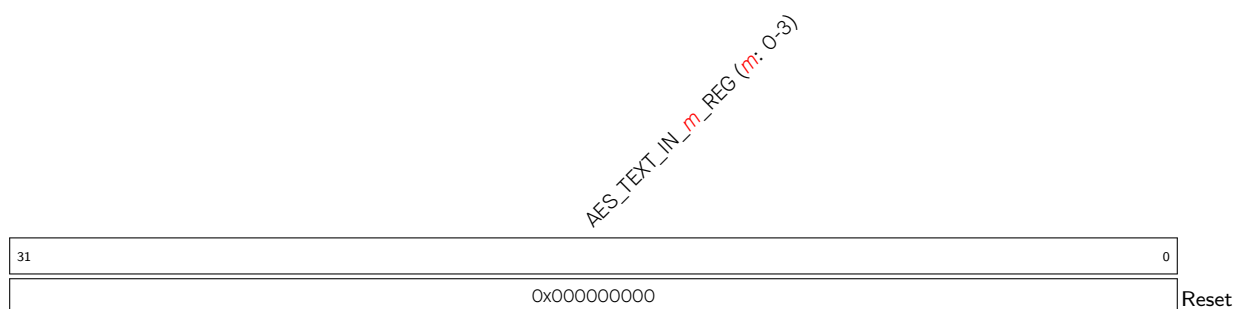
For how to program reserved fields, please refer to Section VII .

Register 16.1. AES_KEY_ *n*_REG (*n*: 0-7) (0x0000+4n*)**



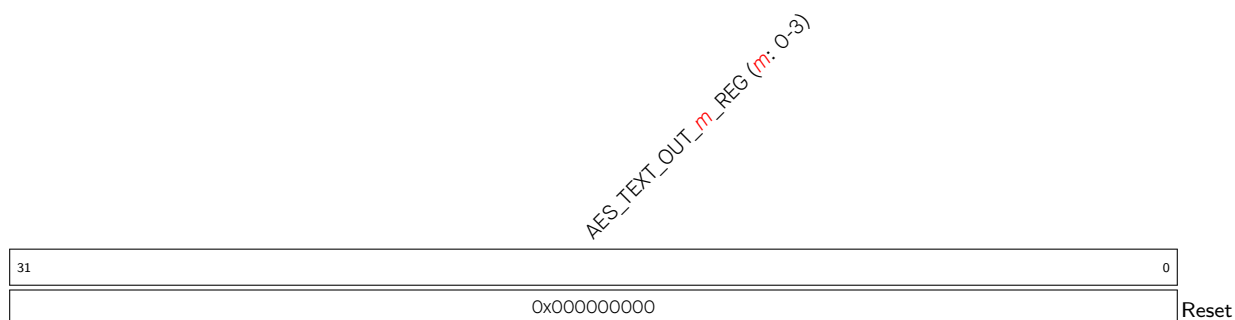
AES_KEY_ *n*_REG (*n*: 0-7) Represents AES key data. (R/W)

Register 16.2. AES_TEXT_IN_ *m*_REG (*m*: 0-3) (0x0020+4m*)**



AES_TEXT_IN_ *m*_REG (*m*: 0-3) Represents the source text data when the AES accelerator operates in the Typical AES working mode. (R/W)

Register 16.3. AES_TEXT_OUT_ *m*_REG (*m*: 0-3) (0x0030+4m*)**



AES_TEXT_OUT_ *m*_REG (*m*: 0-3) Represents the result text data when the AES accelerator operates in the Typical AES working mode. (RO)

Register 16.4. AES_MODE_REG (0x0040)

(reserved)																															AES_MODE	
31																														3	2	0
0x00000000																															0	Reset

AES_MODE Configures the key length and encryption/decryption of the AES accelerator.

- 0: AES-128 encryption
 - 1: Reserved
 - 2: AES-256 encryption
 - 3: Reserved
 - 4: AES-128 decryption
 - 5: Reserved
 - 6: AES-256 decryption
 - 7: Reserved
- (R/W)

Register 16.5. AES_TRIGGER_REG (0x0048)

(reserved)																															AES_TRIGGER																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

AES_TRIGGER Configures whether to start AES operation.

- 0: No effect
 - 1: Start
- (WT)

Register 16.6. AES_DMA_ENABLE_REG (0x0090)

(reserved)																															AES_DMA_ENABLE	
31																															1	0
0x00000000																															0	Reset

AES_DMA_ENABLE Configures the working mode of the AES accelerator.

- 0: Typical AES
 - 1: DMA-AES
- (R/W)

Register 16.7. AES_BLOCK_MODE_REG (0x0094)

(reserved)																AES_BLOCK_MODE			
31																3	2	0	
0x00000000																0	Reset		

AES_BLOCK_MODE Configures the block cipher mode of the AES accelerator operating under the DMA-AES working mode.

- 0: ECB (Electronic Code Block)
 - 1: CBC (Cipher Block Chaining)
 - 2: OFB (Output FeedBack)
 - 3: CTR (Counter)
 - 4: CFB8 (8-bit Cipher FeedBack)
 - 5: CFB128 (128-bit Cipher FeedBack)
 - 6: Reserved
 - 7: Reserved
- (R/W)

Register 16.8. AES_BLOCK_NUM_REG (0x0098)

AES_BLOCK_NUM																
31																0
0x00000000																Reset

AES_BLOCK_NUM Represents the Block Number of plaintext or ciphertext when the AES accelerator operates under the DMA-AES working mode. For details, see Section 16.6.4. (R/W)

Register 16.9. AES_INC_SEL_REG (0x009C)

(reserved)																AES_INC_SEL				
31																1	0			
0x00000000																0				Reset

AES_INC_SEL Configures the Standard Incrementing Function for CTR block operation.

- 0: INC₃₂
 - 1: INC₁₂₈
- (R/W)

Register 16.10. AES_DMA_EXIT_REG (0x00B8)

(reserved)																												AES_DMA_EXIT	
31																												1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

AES_DMA_EXIT Configures whether to exit AES operation.

0: No effect

1: Exit

Only valid for DMA-AES operation. (WT)

Register 16.11. AES_PSEUDO_REG (0x00D0)

(reserved)																								AES_PSEUDO_RNG_CNT				AES_PSEUDO_INC		AES_PSEUDO_BASE		AES_PSEUDO_EN	
31										10										9		7	6	5	4	1	0						
0										0										7		2	2	0	Reset								

Reset

AES_PSEUDO_EN Configures whether to enable the pseudo-round function of AES.

0: Disabled

1: Enabled

(R/W)

AES_PSEUDO_BASE Configures the basic number of pseudo-rounds. (R/W)

AES_PSEUDO_INC Configures the random incremental number of pseudo-rounds. (R/W)

AES_PSEUDO_RNG_CNT Configures the frequency of pseudo-key updates in the pseudo-round function. (R/W)

Register 16.12. AES_STATE_REG (0x004C)

(reserved)		AES_STATE	
31	2	1	0
0x00000000		0x0	Reset

AES_STATE Represents the working status of the AES accelerator.

In Typical AES working mode:

- 0: IDLE
- 1: WORK
- 2: No effect
- 3: No effect

In DMA-AES working mode:

- 0: IDLE
- 1: WORK
- 2: DONE
- 3: No effect

(RO)

Register 16.13. AES_INT_CLEAR_REG (0x00AC)

(reserved)		AES_INT_CLEAR	
31	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0	Reset

AES_INT_CLEAR Write 1 to clear the AES interrupt. (WT)

Register 16.14. AES_INT_ENA_REG (0x00B0)

(reserved)		AES_INT_ENA	
31	1	0	
0x00000000		0	Reset

AES_INT_ENA Write 1 to enable the AES interrupt. (R/W)

Register 16.15. AES_DATE_REG (0x00B4)

(reserved)				AES_DATE																											
31		28	27																												0
0	0	0	0	0x2312070																											Reset

AES_DATE Version control register. (R/W)

Chapter 17

ECC Accelerator (ECC)

17.1 Overview

Elliptic Curve Cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves. ECC uses smaller keys compared to RSA cryptography while providing equivalent security.

ESP32-C5's ECC accelerator can complete various calculations based on different elliptic curves, thus accelerating the ECC algorithm and ECC-derived algorithms such as ECDSA.

17.2 Feature List

ESP32-C5's ECC accelerator has the following features:

- Three different elliptic curves, namely, P-192, P-256, and P-384 defined in [FIPS 186-5](#)
- 11 working modes
- High-security mode
- Interrupt upon completion of calculation

17.3 ECC Basics

To better illustrate the functionality of the ECC accelerator, the basic knowledge and terminology used in this chapter are introduced in this section.

17.3.1 Elliptic Curve and Points on the Curves

The ECC algorithm is based on elliptic curves over prime fields, which can be represented as:

$$y^2 = x^3 + ax + b \bmod p$$

where,

- p is a prime number,
- a and b are two non-negative integers smaller than p ,
- and (x, y) is a point on the curve satisfying the representation.

17.3.2 Affine Coordinates and Jacobian Coordinates

An elliptic curve can be represented as below:

- In affine coordinates:

$$y^2 = x^3 + ax + b \bmod p$$

- In Jacobian coordinates:

$$Y^2 = X^3 + aXZ^4 + bZ^6 \bmod p$$

To convert affine coordinates (x, y) to/from Jacobian coordinates (X, Y, Z) :

- From Jacobian to Affine coordinates:

$$x = X/Z^2 \bmod p$$

$$y = Y/Z^3 \bmod p$$

- From Affine to Jacobian coordinates:

$$X = x$$

$$Y = y$$

$$Z = 1$$

17.3.3 Memory Blocks

ECC's memory blocks store the input and output data of the ECC operation.

Table 17.3-1. ECC Accelerator Memory Blocks

Memory Block ¹	Size (byte)	Starting Address ²	Ending Address ²	Access
ECC_MULT_Mem_k	48	0x100	0x12F	R/W
ECC_MULT_Mem_Px	48	0x130	0x15F	R/W
ECC_MULT_Mem_Py	48	0x160	0x18F	R/W
ECC_MULT_Mem_Qx	48	0x190	0x1BF	R/W
ECC_MULT_Mem_Qy	48	0x1C0	0x1EF	R/W
ECC_MULT_Mem_Qz	48	0x1F0	0x21F	R/W

¹ The memory blocks store different types of data in different working modes. Refer to Section 17.4.2 for more details.

² Address offset related to the ECC accelerator base address is provided in Table 4.3-2 in Chapter 4 *System and Memory*.

17.3.4 Data and Data Block

ESP32-C5's ECC can operate on 192-bit, 256-bit, or 384-bit data, depending on the elliptic curves.

For example, the 256-bit data $D[255 : 0]$ can be divided into eight 32-bit data blocks

$D[n][31 : 0] (n = 0, 1, \dots, 7)$. Data blocks with smaller indexes correspond to lower binary bits. To be specific:

$$D[255 : 0] = D[7][31 : 0], D[6][31 : 0], D[5][31 : 0], D[4][31 : 0], D[3][31 : 0], D[2][31 : 0], D[1][31 : 0], D[0][31 : 0]$$

17.3.5 Writing Data

Writing data means writing data to an ECC memory block and using this data as the input to the ECC algorithm. To be specific, writing data to an ECC memory block means writing $D[n][31 : 0]$ ($n = 0, 1, \dots, 7$) to the “starting address of this ECC memory block + $4 \times n$ ” sequentially:

- write $D[0]$ to “starting address”
- write $D[1]$ to “starting address + 4”
- ...
- write $D[11]$ to “starting address + 44”

Note:

When storing data, write only the data block of the required length. Do not append a 0 to the most significant bit. For example, for 192-bit data, store only the 192 bits without appending 0.

17.3.6 Reading Data

Reading data means reading data from the starting address of an ECC memory block and using this data as the output from the ECC algorithm. To be specific, reading data from an ECC memory block means reading $D[n][31 : 0]$ ($n = 0, 1, \dots, 11$) from the “starting address of this ECC memory block + $4 \times n$ ” successively:

- read $D[0]$ from “starting address”
- read $D[1]$ from “starting address + 4”
- ...
- read $D[11]$ from “starting address + 44”

Note:

When reading data, only the data block of the required length needs to be read. For example, to read 192-bit data, simply read the lower 192 bits (i.e., 6 data blocks).

17.3.7 Standard Calculation and Jacobian Calculation

ESP32-C5's ECC performs Affine Point Calculation (including Affine Point Verification, Affine Point Add, and Affine Point Multiplication) using the affine coordinates and Jacobian Calculation (including Jacobian Point Verification, Jacobian Point Add, and Jacobian Point Multiplication) using the Jacobian coordinates.

17.4 Function Description

17.4.1 Curve Mode

ESP32-C5's ECC supports acceleration based on three curve modes, each corresponding to an elliptic curve. By configuring the `ECC_MULT_CURVE_MODE` field, users can select the desired key size. For details, see

Table 17.4-1 below.

Table 17.4-1. ECC Accelerator Curve Mode Selection

ECC_MULT_CURVE_MODE	Elliptic Curves*
0	FIPS P-192
1	FIPS P-256
2	FIPS P-384

* See definition of FIPS P-192, P-256, and P-384 defined in [FIPS 186-5](#).

17.4.2 Working Modes

ESP32-C5's ECC accelerator supports 11 working modes based on two elliptic curves described in the above section. By configuring the `ECC_MULT_WORK_MODE` field, users can select the desired working mode. For details, see Table 17.4-2.

Table 17.4-2. Working Modes of ECC Accelerator

ECC_MULT_WORK_MODE	Working Modes
0	Affine Point Multi
1	Reserved
2	Affine Point Verif
3	Affine Point Verif + Multi
4	Jacobian Point Multi
5	Point Add
6	Jacobian Point Verif
7	Affine Point Verif + Jacobian Point Multi
8	Mod Add
9	Mod Sub
10	Mod Multi
11	Mod Div

Note:

Note that the calculation of Jacobian Point Multi mode is about 10% faster than that of the Affine Point Multi mode.

Detailed descriptions about different working modes are provided in the following sections.

17.4.2.1 Affine Point Multiplication (Affine Point Multi)

Affine Point Multiplication can be represented as:

$$Q = (Q_x, Q_y) = (J_x, J_y, J_z) = k \cdot (P_x, P_y)$$

where,

- (Q_x, Q_y) is the affine expression of point Q.

- (J_x, J_y, J_z) is the Jacobian expression of point Q.
- Input: P_x , P_y , and k are stored in [ECC_MULT_Mem_Px](#), [ECC_MULT_Mem_Py](#), and [ECC_MULT_Mem_k](#) respectively.
- Output: Q_x and Q_y are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#) respectively.

17.4.2.2 Affine Point Verification (Affine Point Verif)

Affine Point Verification can be used to verify if a point (P_x, P_y) is on a selected elliptic curve.

- Input: P_x and P_y are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#) respectively.
- Output: The verification result is stored in the [ECC_MULT_VERIFICATION_RESULT](#) bit.

17.4.2.3 Affine Point Verification + Affine Point Multiplication (Affine Point Verif + Multi)

In this mode, ECC first verifies if point (P_x, P_y) is on the selected elliptic curve. If so, the following multiplication is performed:

$$Q = (Q_x, Q_y) = (J_x, J_y, J_z) = k \cdot (P_x, P_y)$$

where,

- (Q_x, Q_y) is the affine expression of point Q.
- (J_x, J_y, J_z) is the Jacobian expression of point Q.
- Input: P_x , P_y , and k are stored in [ECC_MULT_Mem_Px](#), [ECC_MULT_Mem_Py](#), and [ECC_MULT_Mem_k](#) respectively.
- Output:
 - The verification result is stored in the [ECC_MULT_VERIFICATION_RESULT](#) bit.
 - Q_x and Q_y are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#) respectively.
 - J_x , J_y , and J_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#).

17.4.2.4 Jacobian Point Multiplication (Jacobian Point Multi)

Jacobian Point Multiplication can be represented as:

$$Q = (Q_x, Q_y, Q_z) = k \cdot (P_x, P_y, 1)$$

where,

- (Q_x, Q_y, Q_z) is the Jacobian expression of point Q.
- 1 in the point's Jacobian coordinates is automatically completed by hardware.
- Input: P_x , P_y , and k are stored in [ECC_MULT_Mem_Px](#), [ECC_MULT_Mem_Py](#), and [ECC_MULT_Mem_k](#) respectively.
- Output: Q_x , Q_y , and Q_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#) respectively.

17.4.2.5 Point Addition (Point Add)

Point Addition can be represented as:

$$R = (R_x, R_y) = (J_x, J_y, J_z) = (P_x, P_y, 1) + (Q_x, Q_y, Q_z)$$

where,

- (R_x, R_y) is the affine expression of point R.
- (J_x, J_y, J_z) is the Jacobian expression of point R.
- 1 in the point's Jacobian coordinates is automatically completed by hardware.
- Input:
 - P_x and P_y are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - Q_x , Q_y , and Q_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#).
- Output:
 - R_x and R_y are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - J_x , J_y and J_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#).

17.4.2.6 Jacobian Point Verification (Jacobian Point Verif)

Jacobian Point Verification can be used to verify if point (Q_x, Q_y, Q_z) is on a selected elliptic curve.

- (Q_x, Q_y, Q_z) is the Jacobian expression of point Q.
- Input: Q_x , Q_y , and Q_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#) respectively.
- Output: The verification result is stored in the [ECC_MULT_VERIFICATION_RESULT](#) bit.

17.4.2.7 Affine Point Verification + Jacobian Point Multiplication (Affine Point Verif + Jacobian Point Multi)

In this mode, ECC first verifies if point (P_x, P_y) is on the selected elliptic curve. If so, the following multiplication is performed:

$$Q = (Q_x, Q_y, Q_z) = k \cdot (P_x, P_y, 1)$$

where,

- (Q_x, Q_y, Q_z) is the Jacobian expression of point Q.
- 1 in the point's Jacobian coordinates is automatically completed by hardware.
- Input: P_x , P_y , and k are stored in [ECC_MULT_Mem_Px](#), [ECC_MULT_Mem_Py](#), and [ECC_MULT_Mem_k](#).
- Output:
 - The verification result is stored in the [ECC_MULT_VERIFICATION_RESULT](#) bit.
 - Q_x , Q_y , and Q_z are stored in [ECC_MULT_Mem_Qx](#), [ECC_MULT_Mem_Qy](#), and [ECC_MULT_Mem_Qz](#).

17.4.2.8 Mod Addition (Mod Add)

Mod Addition can be represented as:

$$R = A + B \bmod N$$

where,

- Input:
 - A and B are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - The value of N is related to the register fields below:
 - * [ECC_MULT_CURVE_MODE](#) to select the related curve.
 - * [ECC_MULT_MOD_BASE](#) to choose using mod base or order of the base point.
- Output: R is stored in [ECC_MULT_Mem_Px](#).

17.4.2.9 Mod Subtraction (Mod Sub)

Mod Subtraction can be represented as:

$$R = A - B \bmod N$$

where,

- Input:
 - A and B are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - The value of N is related to the register fields below:
 - * [ECC_MULT_CURVE_MODE](#) to select the related curve.
 - * [ECC_MULT_MOD_BASE](#) to choose using mod base or order of the base point.
- Output: R is stored in [ECC_MULT_Mem_Px](#).

17.4.2.10 Mod Multiplication (Mod Multi)

Mod Multiplication can be represented as:

$$R = A \cdot B \bmod N$$

where,

- Input:
 - A and B are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - The value of N is related to the register fields below:
 - * [ECC_MULT_CURVE_MODE](#) to select the related curve.
 - * [ECC_MULT_MOD_BASE](#) to choose using mod base or order of the base point.
- Output: R is stored in [ECC_MULT_Mem_Py](#).

17.4.2.11 Mod Division (Mod Div)

Mod Division can be represented as:

$$R = A \cdot B^{-1} \bmod N$$

where,

- Input:
 - A and B are stored in [ECC_MULT_Mem_Px](#) and [ECC_MULT_Mem_Py](#).
 - The value of N is related to the register fields below:
 - * [ECC_MULT_CURVE_MODE](#) to select the related curve.
 - * [ECC_MULT_MOD_BASE](#) to choose using mod base or order of the base point.
- Output: R is stored in [ECC_MULT_Mem_Py](#).

17.4.3 Enhancing Anti-Attack Performance

When performing different types of point multiplication calculations, i.e., mode 0, 3, 4, and 7, ESP32-C5's ECC accelerator offers an additional option to further enhance its anti-attack performance by ensuring that all point multiplication calculations take the same maximum runtime.

To enable the enhanced anti-attack performance, set [ECC_MULT_SECURITY_MODE](#) to 1. If this setting is enabled, each point multiplication calculation of the ECC accelerator will:

- consume the same amount of time;
- consume the same amount of power.

17.4.4 Clock

ECC uses two types of clocks:

- `clk_ecc`: function clock for ECC to operate
- `clk_apb_ecc`: bus clock used to configure ECC registers

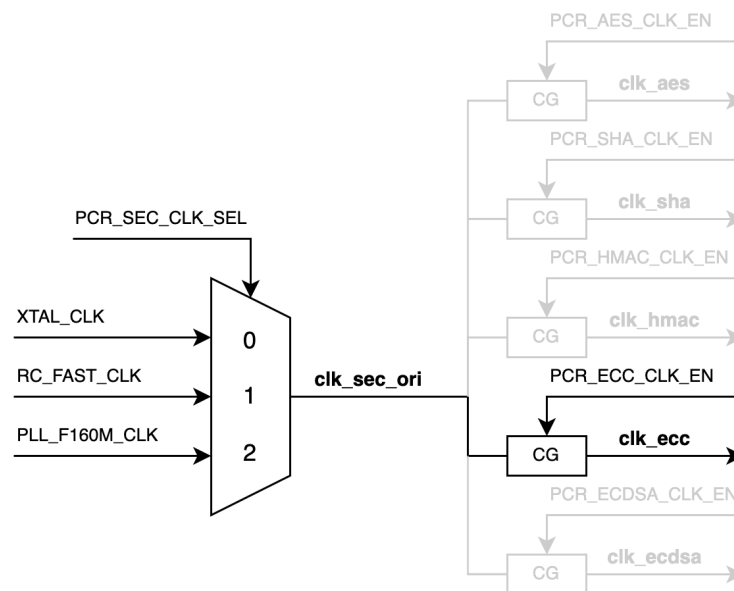


Figure 17.4-1. clk_ecc Diagram

As shown in Figure 17.4-1 *clk_ecc Diagram*, the secure function source clock clk_sec_ori can be sourced from:

- XTAL_CLK
- RC_FAST_CLK
- PLL_F160M_CLK

ECC's function clock clk_ecc is configured by the clock gating register [PCR_ECC_CLK_EN](#). When this register is set to 1, the clock gate is enabled, and clk_ecc is activated.

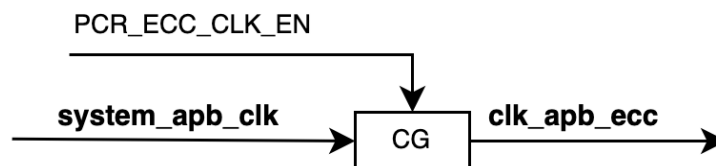


Figure 17.4-2. clk_apb_ecc Diagram

As shown in Figure 17.4-2 *clk_apb_ecc Diagram*, ECC's configuration clock clk_apb_ecc can be sourced from SYSTEM_APB_CLK.

ECC's configuration clock clk_apb_ecc is controlled by the clock gating register [PCR_ECC_CLK_EN](#). When this register is set to 1, the clock gate is enabled, and clk_apb_ecc is activated.

For more information about clocks, see Section 6 *Reset and Clock*.

17.4.5 Reset

The ECC module supports two types of resets:

- Full self-reset: Write 1 and then 0 to [PCR_ECC_RST_EN](#) to reset the entire ECC module.

- Cryptographic protection reset: Write 1 and then 0 to [PCR_ECDSA_RST_EN](#) to reset the entire ECC module.

Note:

Due to shared resources among cryptographic accelerators, the ECDSA module reuses resources from the ECC module during computation. To protect ECDSA operation data, resetting the ECDSA module also resets the ECC module.

17.5 Interrupts

ESP32-C5's ECC accelerator can generate the ECC_INTR interrupt signal that will be sent to the [Interrupt Matrix](#).

ECC_INTR has only one interrupt source to generate the ECC_INTR interrupt signal, i.e., ECC_MULT_CALC_DONE_INT, which is triggered on the completion of an ECC calculation.

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The ECC_MULT_CALC_DONE_INT interrupt source is configured by the following registers (refer to Section [17.7 Register Summary](#) for more information):

- [ECC_MULT_CALC_DONE_INT_RAW](#): Stores the raw interrupt status of ECC_MULT_CALC_DONE_INT.
- [ECC_MULT_CALC_DONE_INT_ST](#): Indicates the status of the ECC_MULT_CALC_DONE_INT interrupt. This bit is generated by enabling or disabling the [ECC_MULT_CALC_DONE_INT_RAW](#) bit via [ECC_MULT_CALC_DONE_INT_ENA](#).
- [ECC_MULT_CALC_DONE_INT_ENA](#): Enables or disables the ECC_MULT_CALC_DONE_INT interrupt.
- [ECC_MULT_CALC_DONE_INT_CLR](#): Set this bit to clear the ECC_MULT_CALC_DONE_INT interrupt status. By setting this bit to 1, the [ECC_MULT_CALC_DONE_INT_RAW](#) and [ECC_MULT_CALC_DONE_INT_ST](#) bits will be cleared.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter [3 Interrupt Matrix \[to be added later\]](#) > Section [3.1 Interrupt Terminology](#).

17.6 Programming Procedures

The programming procedure for configuring ECC is described below:

1. Configure the ECC clock and reset. Refer to Sections [17.4.4](#) and [17.4.5](#) for detailed information.
2. Select the key size and working mode as described in Section [17.4](#).
3. Enable the [ECC_MULT_CALC_DONE_INT](#) interrupt as described in Section [17.5](#).
4. Set the [ECC_MULT_START](#) field to start ECC calculation.
5. Wait for the [ECC_MULT_CALC_DONE_INT](#) interrupt, which indicates the completion of the ECC calculation.
6. Check the result as described in Section [17.4](#).

17.7 Register Summary

The addresses in this section are relative to ECC accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Interrupt Registers			
ECC_MULT_INT_RAW_REG	ECC raw interrupt status register	0x000C	R/SS/WTC
ECC_MULT_INT_ST_REG	ECC masked interrupt status register	0x0010	RO
ECC_MULT_INT_ENA_REG	ECC interrupt enable register	0x0014	R/W
ECC_MULT_INT_CLR_REG	ECC interrupt clear register	0x0018	WT
Configuration Register			
ECC_MULT_CONF_REG	ECC configuration register	0x001C	varies
Version Register			
ECC_MULT_DATE_REG	Version control register	0x00FC	R/W

17.8 Registers

The addresses in this section are relative to ECC accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 17.1. ECC_MULT_INT_RAW_REG (0x000C)

(reserved)																												ECC_MULT_CALC_DONE_INT_RAW	
31																												1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
																													Reset

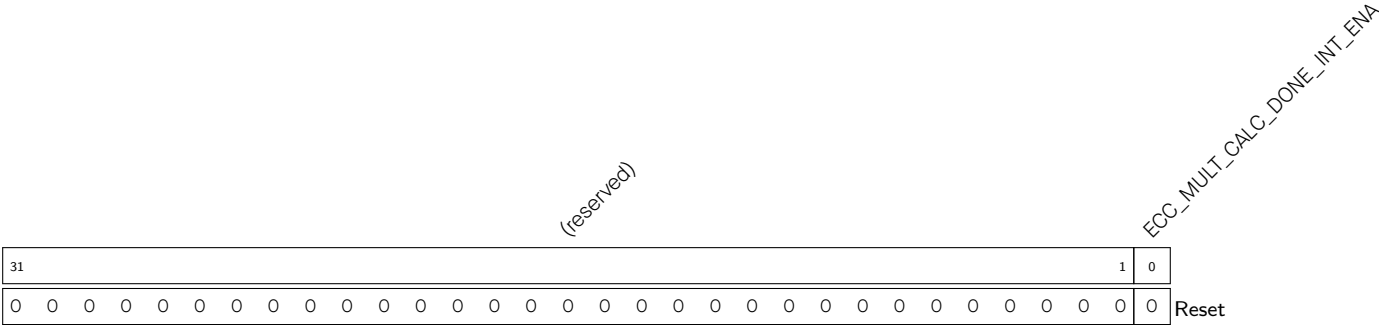
ECC_MULT_CALC_DONE_INT_RAW The raw interrupt status of the [ECC_MULT_CALC_DONE_INT](#) interrupt. (R/SS/WTC)

Register 17.2. ECC_MULT_INT_ST_REG (0x0010)

(reserved)																															ECC_MULT_CALC_DONE_INT_ST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

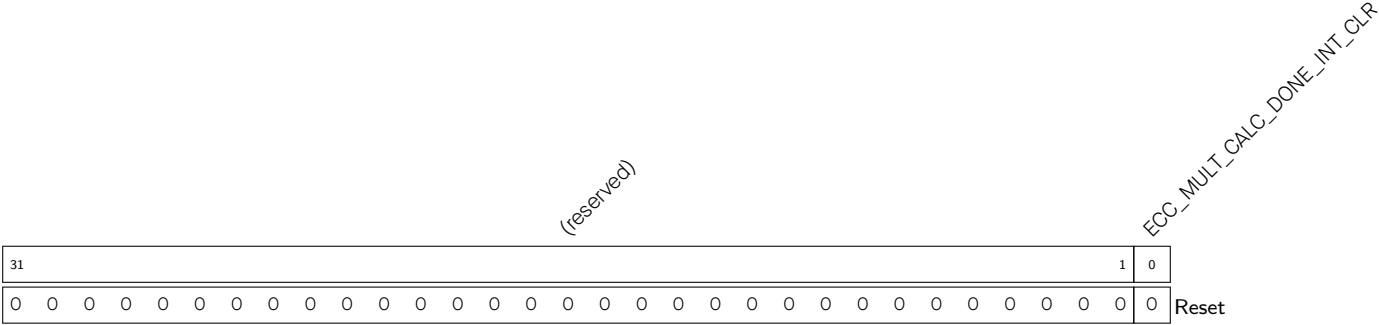
ECC_MULT_CALC_DONE_INT_ST The masked interrupt status of the [ECC_MULT_CALC_DONE_INT](#) interrupt. (RO)

Register 17.3. ECC_MULT_INT_ENA_REG (0x0014)



ECC_MULT_CALC_DONE_INT_ENA Write 1 to enable the ECC_MULT_CALC_DONE_INT interrupt.
(R/W)

Register 17.4. ECC_MULT_INT_CLR_REG (0x0018)



ECC_MULT_CALC_DONE_INT_CLR Write 1 to clear the ECC_MULT_CALC_DONE_INT interrupt. (WT)

Register 17.5. ECC_MULT_CONF_REG (0x001C)

ECC_MULT_MEM_CLOCK_GATE_FORCE_ON																												(reserved)																												ECC_MULT_SECURITY_MODE												ECC_MULT_WORK_MODE												ECC_MULT_MOD_BASE												ECC_MULT_CURVE_MODE												ECC_MULT_RESET												ECC_MULT_START											
31		30		29		28																										10		9		8				5		4		3		2		1		0																																																																													
0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		0		Reset																																																																																											

ECC_MULT_START Configures whether to start the calculation of the ECC accelerator. This bit will be self-cleared when the calculation is done.

0: No effect

1: Start the calculation of the ECC accelerator

(R/W/SC)

ECC_MULT_RESET Configures whether to reset the ECC accelerator.

0: No effect

1: Reset

(WT)

ECC_MULT_CURVE_MODE Configures the curve mode of the ECC accelerator.

0: P-192

1: P-256

2: P-384

3: Invalid

(R/W)

ECC_MULT_MOD_BASE Configures whether to choose using the mod base or order of base point in the mod operation. Only valid in working modes 8-11.

0: n (order of base point)

1: p (mod base of curve)

(R/W)

Continued on the next page...

Register 17.5. ECC_MULT_CONF_REG (0x001C)

Continued from the previous page...

ECC_MULT_WORK_MODE Configures the working mode of the ECC accelerator.

- 0: Affine Point Multi mode
 - 1: Reserved
 - 2: Affine Point Verif mode
 - 3: Affine Point Verif + Multi mode
 - 4: Jacobian Point Multi mode
 - 5: Point Add mode
 - 6: Jacobian Point Verif mode
 - 7: Affine Point Verif + Jacobian Point Multi mode
 - 8: Mod Add mode
 - 9: Mod Sub mode
 - 10: Mod Multi mode
 - 11: Mod Div mode
- (R/W)

ECC_MULT_SECURITY_MODE Configures whether to enable high-security mode when calculating point multiplication. Only valid in working modes 0, 3, 4, and 7.

- 0: Disable
 - 1: Enable
- (R/W)

ECC_MULT_VERIFICATION_RESULT Represents the verification result of the ECC accelerator, valid only when the calculation is done.

- 0: Verification failed
 - 1: Verification passed
- (R/SS)

ECC_MULT_CLK_EN Configures whether to force on register clock gate.

- 0: No effect
 - 1: Force on
- (R/W)

ECC_MULT_MEM_CLOCK_GATE_FORCE_ON Configures whether to force the ECC memory clock gate on.

- 0: No effect
 - 1: Force on
- (R/W)

Register 17.6. ECC_MULT_DATE_REG (0x00FC)

(reserved)				ECC_MULT_DATE																									
31	28	27	0																										
0	0	0	0	0x2408120																								Reset	

ECC_MULT_DATE Version control register. (R/W)

Chapter 18

HMAC Accelerator (HMAC)

18.1 Overview

The Hash-based Message Authentication Code (HMAC) module computes Message Authentication Codes (MACs) using hash algorithm SHA-256 and keys as described in RFC 2104. The 256-bit HMAC key is stored in an eFuse key block and can be set as read-protected, i. e., the key is not accessible from outside the HMAC accelerator.

18.2 Feature List

- Standard HMAC-SHA-256 algorithm
- HMAC-SHA-256 calculation based on key in eFuse,
 - whose result cannot be accessed by software in downstream mode for high security
 - whose result can be accessed by software in upstream mode
- Generates required keys for the Digital Signature Algorithm (DSA) peripheral in downstream mode
- Re-enables soft-disabled JTAG in downstream mode

18.3 Functional Description

The HMAC module operates in two modes: upstream mode and downstream mode. In upstream mode, users provide the HMAC message and read back the calculation result. In downstream mode, the HMAC module provides input to two possible other internal hardware modules: On the one hand, an HMAC can be used to enable JTAG after JTAG has been temporarily disabled before. On the other hand, an HMAC can be used as the decryption key for Digital Signature parameters stored in the memory for the DSA peripheral. Furthermore, the calculations happen internally and automatically in downstream mode, so that confidentiality of any key and derived key material is ensured, given correct configuration.

Note:

After the reset signal being released, the HMAC module will check whether the DSA key exists in the eFuse. If the key exists, the HMAC module will enter downstream digital signature algorithm mode and finish the DSA key calculation automatically. This process is automatically completed by hardware and does not require software participation. When the downstream operation after reset is completed, the HMAC will automatically return to the idle state.

18.3.1 Upstream Mode

To calculate the HMAC value in upstream mode, users should perform the following steps:

1. Initialize the HMAC module and enter upstream mode.
2. Write the correctly padded message to the HMAC, one block at a time.
3. Read back the result from HMAC.

For details of this process, please see Section [18.5](#).

Note:

Common use cases for the upstream mode are challenge-response protocols supporting HMAC-SHA-256. Assume the two entities in the challenge-response protocol are A and B respectively, and the data message they expect to exchange is M. The general authentication process of this protocol is as follows:

- A calculates a unique random number M .
- A sends M to B.
- B calculates the HMAC (through M and KEY) and sends the result to A.
- A calculates the HMAC (through M and KEY) internally.
- A compares the two results. If the results are the same, then the identity of B is authenticated.

18.3.2 Downstream Mode - JTAG Enable Feature

JTAG debugging can be disabled by eFuse in a way which allows later re-enabling using the HMAC module. (For more details, please see Chapter [2 eFuse Controller \(eFuse\) \[to be added later\]](#).) The HMAC module will expect the user to supply the HMAC result for one of the eFuse keys. The HMAC module will check whether the supplied HMAC matches the one calculated from the chosen key. If both HMACs are the same, JTAG will be enabled until the user calls the HMAC module to clear the results and consequently disable JTAG again.

To re-enable JTAG, users should perform the following steps:

1. Enable the HMAC module by initializing clock and reset signals of HMAC, and enter downstream JTAG enable mode by configuring [HMAC_SET_PARA_PURPOSE_REG](#). Then, wait for the calculation to complete. Please see Section [18.5](#) for more details.
2. Write 1 to the [HMAC_SOFT_JTAG_CTRL_REG](#) register to enter JTAG re-enable mode.
3. Write the 256-bit HMAC value to register [HMAC_WR_JTAG_REG](#). This value is obtained by performing a local HMAC calculation from the 32-byte 0x00 using SHA-256 and the key that has been written to the eFuse. It needs to be written 8 times and 32-bit each time in big-endian word order.
4. If the HMAC result calculated from the key in the eFuse matches the value that users wrote in step 3, then JTAG is re-enabled. Otherwise, JTAG remains disabled.
5. After writing 1 to [HMAC_SET_INVALIDATE_JTAG_REG](#) or resetting the chip, JTAG will be disabled. If users want to re-enable JTAG again, they need to repeat the above steps again.

18.3.3 Downstream Mode - Digital Signature Algorithm and Key Derivation Feature

The Digital Signature Algorithm (DSA) module encrypts its parameters using the AES-CBC algorithm. The HMAC module is used as a Key Derivation Function (KDF) to derive the AES key to decrypt these parameters (parameter decryption key).

Before starting the DSA module, users need to obtain the parameter decryption key for the DSA module through HMAC calculation. For more information, please see Chapter [21 Digital Signature Algorithm \(DSA\)](#). After the chip is powered on, the HMAC module will check whether the key required to calculate the parameter decryption key has been burned in the eFuse block. If the key has been burned, HMAC module will automatically enter the downstream digital signature algorithm mode and complete the HMAC calculation based on the chosen key.

18.4 HMAC eFuse Configuration

Each HMAC key burned into an eFuse block has a key purpose, specifying for which functionality the key can be used. The HMAC module will not accept a key with a non-matching purpose for any functionality. The HMAC module provides three different functionalities: re-enabling JTAG, DSA KDF in downstream mode, and pure HMAC calculation in upstream mode. For each functionality, there exists a corresponding key purpose, listed in Table [18.4-1](#). Additionally, another purpose specifies a key which may be used for re-enabling JTAG as well as for serving as DSA KDF.

Before enabling HMAC to do calculations, user should make sure the key to be used has been burned in eFuse by reading the registers EFUSE_KEY_PURPOSE_x (We have a total of 6 keys in eFuse, so the value of x is 0 ~ 5. Among which, EFUSE_KEY_PURPOSE_0 ~ EFUSE_KEY_PURPOSE_1 belong to the register EFUSE_RD_REPEAT_DATA1_REG, and EFUSE_KEY_PURPOSE_2 ~ EFUSE_KEY_PURPOSE_5 belong to the register EFUSE_RD_REPEAT_DATA2_REG) from [2 eFuse Controller \(eFuse\) \[to be added later\]](#). Take upstream mode as an example, if there is no EFUSE_KEY_PURPOSE_HMAC_UP in EFUSE_KEY_PURPOSE_0 ~ 5, it means there is no key in eFuse that can be used for the HMAC upstream mode. Users can burn a key to eFuse as follows:

1. Prepare a secret 256-bit HMAC key and burn the key to an empty eFuse block y . As there are 6 blocks for storing a key in eFuse and the numbers of those blocks range from 4 to 9, the value of y is 4 ~ 9. Hence, when talking about key0, it means eFuse block4. Then, program the purpose to EFUSE_KEY_PURPOSE_ $(y - 4)$. Take upstream mode as an example: after programming the key, the user should program EFUSE_KEY_PURPOSE_HMAC_UP (corresponding value is 8) to EFUSE_KEY_PURPOSE_ $(y - 4)$. Please see Chapter [2 eFuse Controller \(eFuse\) \[to be added later\]](#) on how to program eFuse keys.
2. If needed, configure this eFuse key block to be read protected, so that users cannot read its value. A copy of this key should be kept by any party who needs to verify this device.

Please note that the key whose purpose is EFUSE_KEY_PURPOSE_HMAC_DOWN_ALL can be used for both re-enabling JTAG or DSA.

Table 18.4-1. HMAC Purposes and Configuration Value

Purpose(<i>m</i>)	Mode	Value	Description
JTAG Re-enable	Downstream	6	EFUSE_KEY_PURPOSE_HMAC_DOWN_JTAG
DSA KDF	Downstream	7	EFUSE_KEY_PURPOSE_HMAC_DOWN_DIGITAL_SIGNATURE
HMAC Calculation	Upstream	8	EFUSE_KEY_PURPOSE_HMAC_UP
Both JTAG Re-enable and DSA KDF	Downstream	5	EFUSE_KEY_PURPOSE_HMAC_DOWN_ALL

Select eFuse Key Blocks and HMAC Purposes

The eFuse controller provides six key blocks, i.e., KEY0 ~ 5. To select a particular KEY n for an HMAC calculation, write the key number n to register [HMAC_SET_PARA_KEY_REG](#).

Write a correct purpose to register [HMAC_SET_PARA_PURPOSE_REG](#) (see Section 18.5). Note that the purpose of the key has also been programmed to eFuse memory. Only when the configured HMAC purpose matches the defined purpose of KEY n , the HMAC module will execute the configured calculation. Otherwise, it will return a matching error and stop the current calculation.

For example, suppose a user selects KEY3 for HMAC calculation, and the value programmed to EFUSE_KEY_PURPOSE_3 is 6 (EFUSE_KEY_PURPOSE_HMAC_DOWN_JTAG). Based on Table 18.4-1, KEY3 can be used to re-enable JTAG. If the value written to register [HMAC_SET_PARA_PURPOSE_REG](#) is also 6, then the HMAC module will start the process to re-enable JTAG.

Select the Key from the Key Manager

Users can deploy the HMAC key in the key manager. To select this key from the key manager, write the key number 7 to register [HMAC_SET_PARA_KEY_REG](#). When users choose the HMAC key from the key manager, the HMAC purpose will automatically be "HMAC Calculation", and any configuration to register [HMAC_SET_PARA_PURPOSE_REG](#) takes no effect.

18.5 HMAC Process (Detailed)

The process for users to call HMAC in ESP32-C5 is as follows:

18.5.1 Enable HMAC module

1. Set the peripheral clock bits for HMAC and SHA peripherals in register HP_SYS_CLKRST_REG_CRYPTO_HMAC_CLK_EN, and clear the corresponding peripheral reset bits in register HP_SYS_CLKRST_REG_RST_EN_HMAC. For information on those registers, please see Chapter 6 [Reset and Clock](#).
2. Write 1 to register [HMAC_SET_START_REG](#).

18.5.2 Configure HMAC keys and key purposes

1. Write the key purpose m to register [HMAC_SET_PARA_PURPOSE_REG](#). The possible key purpose values are shown in Table 18.4-1. For more information, please refer to Section 18.4.
2. Select KEY n in eFuse memory as the key by writing n (ranges from 0 to 5, and 7) to register

[HMAC_SET_PARA_KEY_REG](#). For more information, please refer to Section 18.4.

3. Write 1 to register [HMAC_SET_PARA_FINISH_REG](#) to complete the configuration.
4. Read register [HMAC_QUERY_ERROR_REG](#). If its value is 1, it means the purpose of the selected block does not match the configured key purpose and the calculation will not proceed. If its value is 0, it means the purpose of the selected block matches the configured key purpose, and then the calculation can proceed.
5. When the value of [HMAC_SET_PARA_PURPOSE_REG](#) is not 8, it means the HMAC module is in downstream mode, proceed with 18.5.3. When the value is 8, it means the HMAC module is in upstream mode, proceed with 18.5.4.

18.5.3 Downstream mode process

1. Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
2. To clear the result and make further usage of the dependent hardware (JTAG or DSA) impossible, write 1 to either register [HMAC_SET_INVALIDATE_JTAG_REG](#) to clear the result generated by the JTAG key; or to register [HMAC_SET_INVALIDATE_DS_REG](#) to clear the result generated by DSA key. Afterwards, the HMAC process needs to be restarted to re-enable any of the dependent peripherals.

18.5.4 Upstream mode process

1. Write message in upstream mode:
 - (a) Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
 - (b) Apply SHA padding to message as described in Section 18.6.1, and get message block: Block_*n* (*n* ≥ 0)
 - (c) Perform one of the following operations according to the block number *n*:
 - If there is only one message block in total which has included all padding bits:
 - i. Write the 512-bit Block_*n* to register [HMAC_WR_MESSAGE_n_REG](#) (*n*: 0-15). Write 1 to register [HMAC_SET_MESSAGE_ONE_REG](#), to trigger the processing of this message block.
 - ii. Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
 - iii. Write 1 to register [HMAC_ONE_BLOCK_REG](#), and finish message transmission.
 - If Block_*n* is the last padded block:
 - i. Write the 512-bit Block_0 to register [HMAC_WR_MESSAGE_n_REG](#) (*n*: 0-15). Write 1 to register [HMAC_SET_MESSAGE_ONE_REG](#), to trigger the processing of this message block.
 - ii. Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
 - iii. Message transmission finished automatically.
 - If Block_*n* is the second last padded block:
 - i. Write the 512-bit Block_*n* to register [HMAC_WR_MESSAGE_n_REG](#) (*n*: 0-15). Write 1 to register [HMAC_SET_MESSAGE_ONE_REG](#), to trigger the processing of this message block.
 - ii. Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.

- iii. Write 1 to register [HMAC_SET_MESSAGE_PAD_REG](#), and continue message transmission (jump back to step(c)).
- If Block_*n* is neither the last nor the second last message block:
 - i. Write the 512-bit Block_*n* to register [HMAC_WR_MESSAGE_n_REG](#) (*n*: 0-15). Write 1 to register [HMAC_SET_MESSAGE_ONE_REG](#), to trigger the processing of this message block.
 - ii. Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
 - iii. Write 1 to register [HMAC_SET_MESSAGE_ING_REG](#), and continue message transmission (jump back to step(c)).
- 2. Read hash result in upstream mode:
 - (a) Poll Status register [HMAC_QUERY_BUSY_REG](#) until it reads 0.
 - (b) Read hash result from register [HMAC_RD_RESULT_n_REG](#) (*n*: 0-7).
 - (c) Write 1 to register [HMAC_SET_RESULT_FINISH_REG](#) to finish calculation. The result will be cleared at the same time.
 - (d) Upstream mode operation is completed.

Note:

The SHA accelerator can be called directly, or used internally by the DSA module and the HMAC module. However, they can not share the hardware resources simultaneously. Therefore, the SHA module must not be called neither by the CPU nor by the DSA module when the HMAC module is in use.

18.6 HMAC Algorithm Details

18.6.1 Padding Bits

The HMAC module uses SHA-256 as hash algorithm. If the input message is not a multiple of 512 bits, the user must apply a SHA-256 padding algorithm in software. The SHA-256 padding algorithm is the same as described in Section *Padding the Message* of [FIPS PUB 180-4](#). In downstream mode, users do not need to input any message or apply padding. The HMAC module uses a default 32-byte pattern of 0x00 for re-enabling JTAG and a 32-byte pattern of 0xff for deriving the AES key for the DSA module.

For the convenience of reading, here we will briefly describe the process of message padding. As shown in Figure [18.6-1](#), suppose the length of the unpadded message is *m* bits. Padding steps are as follows:

1. Append one bit of value “1” to the end of the unpadded message.
2. Append *k* bits of value “0”, where *k* is the smallest non-negative number which satisfies $m + 1 + k \equiv 448 \pmod{512}$.
3. Append a 64-bit integer value as a binary block. This block consists of the length of the unpadded message as a big-endian binary integer value *m*.

In upstream mode, if the length of the unpadded message is a multiple of 512 bits, users can configure hardware to apply SHA padding by writing 1 to [HMAC_SET_MESSAGE_END_REG](#) or do padding work themselves by writing 1 to [HMAC_SET_MESSAGE_PAD_REG](#). If the length is not a multiple of 512 bits, SHA

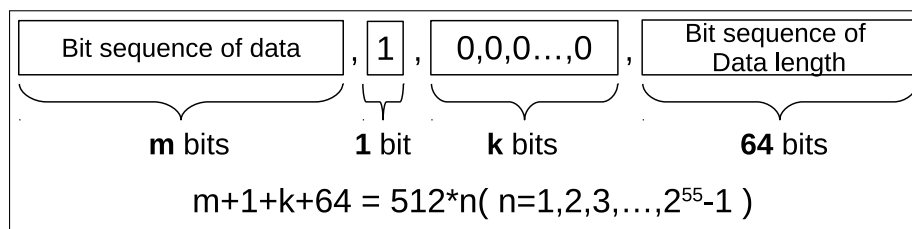


Figure 18.6-1. HMAC SHA-256 Padding Diagram

padding must be manually applied by the user. After the user prepared the padding data, they should complete the subsequent configuration according to the Section [18.5](#).

18.6.2 HMAC Algorithm Structure

The structure of the implemented algorithm in the HMAC module is shown in Figure [18.6-2](#). This is the standard HMAC algorithm as described in RFC 2104.

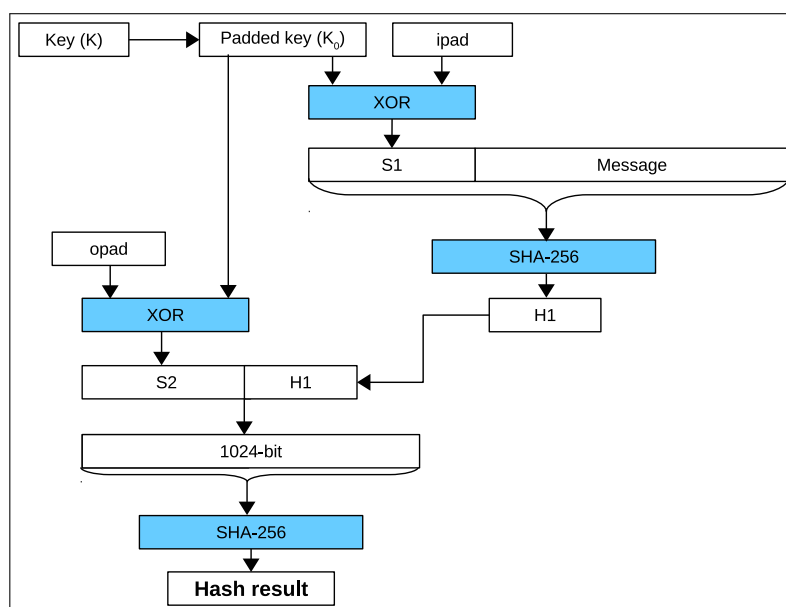


Figure 18.6-2. HMAC Structure Schematic Diagram

In Figure [18.6-2](#):

1. ipad is a 512-bit message block composed of 64 bytes of 0x36.
2. opad is a 512-bit message block composed of 64 bytes of 0x5c.

The HMAC module appends a 256-bit 0 sequence after the bit sequence of the 256-bit key K in order to get a 512-bit K_0 . Then, the HMAC module XORs K_0 with ipad to get the 512-bit S1. Afterwards, the HMAC module appends the input message (multiple of 512 bits) after the 512-bit S1, and exercises the SHA-256 algorithm to get the 256-bit H1.

The HMAC module appends the 256-bit SHA-256 hash result H1 to the 512-bit S2 value, which is calculated using the XOR operation of K_0 and opad. A 768-bit sequence will be generated. Then, the HMAC module uses the SHA padding algorithm described in Section [18.6.1](#) to pad the 768-bit sequence to a 1024-bit sequence, and applies the SHA-256 algorithm to get the final hash result (256-bit).

18.6.3 Register Summary

The addresses in this section are relative to HMAC Accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

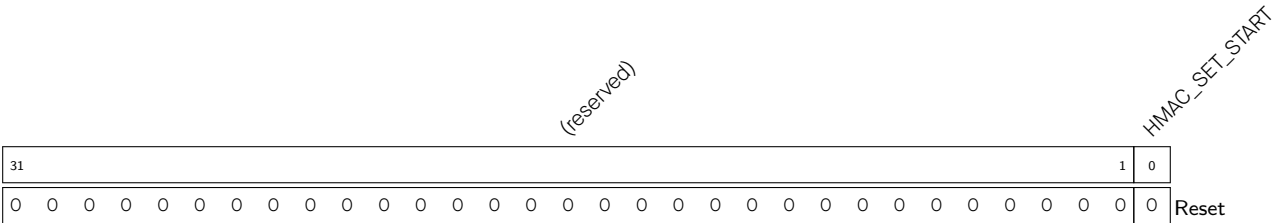
Name	Description	Address	Access
Control/Status Registers			
HMAC_SET_START_REG	HMAC start control register	0x0040	WS
HMAC_SET_PARA_FINISH_REG	HMAC configuration completion register	0x004C	WS
HMAC_SET_MESSAGE_ONE_REG	HMAC message control register	0x0050	WS
HMAC_SET_MESSAGE_ING_REG	HMAC message continue register	0x0054	WS
HMAC_SET_MESSAGE_END_REG	HMAC message end register	0x0058	WS
HMAC_SET_RESULT_FINISH_REG	HMAC result reading finish register	0x005C	WS
HMAC_SET_INVALIDATE_JTAG_REG	Invalidate JTAG result register	0x0060	WS
HMAC_SET_INVALIDATE_DS_REG	Invalidate digital signature result register	0x0064	WS
HMAC_QUERY_ERROR_REG	Stores matching results between keys generated by users and corresponding purposes	0x0068	RO
HMAC_QUERY_BUSY_REG	Busy state of HMAC module	0x006C	RO
HMAC_SET_MESSAGE_PAD_REG	Software padding register	0x00F0	WO
HMAC_ONE_BLOCK_REG	One block message register	0x00F4	WS
Configuration Registers			
HMAC_SET_PARA_PURPOSE_REG	HMAC parameter configuration register	0x0044	WO
HMAC_SET_PARA_KEY_REG	HMAC parameters configuration register	0x0048	WO
HMAC_WR_JTAG_REG	Re-enable JTAG register 1	0x00FC	WO
Configuration Register			
HMAC_SOFT_JTAG_CTRL_REG	Jtag register 0.	0x00F8	WS
Version Register			
HMAC_DATE_REG	Version control register	0x01FC	R/W

18.6.4 Registers

The addresses in this section are relative to HMAC Accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 18.1. HMAC_SET_START_REG (0x0040)

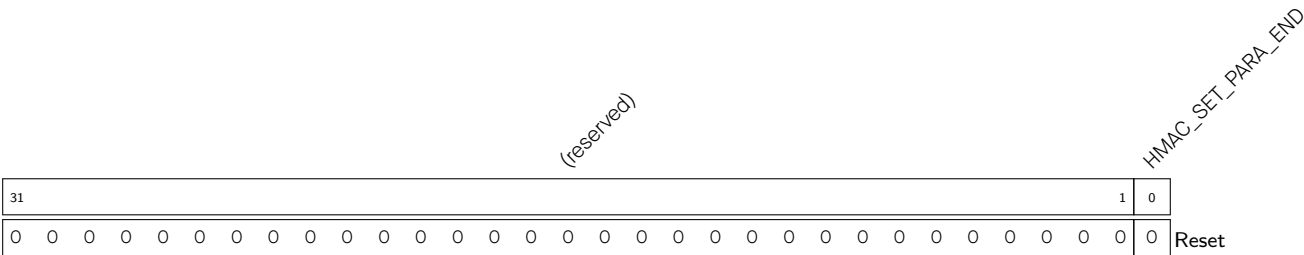


HMAC_SET_START Configures whether or not to enable HMAC.

- 0: Disable HMAC
- 1: Enable HMAC

(WS)

Register 18.2. HMAC_SET_PARA_FINISH_REG (0x004C)

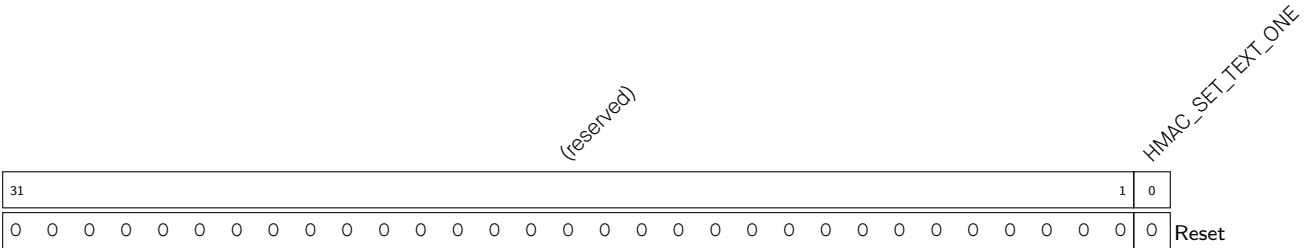


HMAC_SET_PARA_END Configures whether to finish HMAC configuration.

- 0: No effect
- 1: Finish configuration

(WS)

Register 18.3. HMAC_SET_MESSAGE_ONE_REG (0x0050)



HMAC_SET_TEXT_ONE Calls SHA to calculate one message block. (WS)

Register 18.4. HMAC_SET_MESSAGE_ING_REG (0x0054)

(reserved)																																HMAC_SET_TEXT_ING																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

HMAC_SET_TEXT_ING Configures whether or not there are unprocessed message blocks.

0: No unprocessed message block

1: There are still some message blocks to be processed.

(WS)

Register 18.5. HMAC_SET_MESSAGE_END_REG (0x0058)

(reserved)																																HMAC_SET_TEXT_END																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

HMAC_SET_TEXT_END Configures whether to start hardware padding.

0: No effect

1: Start hardware padding

(WS)

Register 18.6. HMAC_SET_RESULT_FINISH_REG (0x005C)

(reserved)																															HMAC_SET_RESULT_END				
31																															1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

HMAC_SET_RESULT_END Configures whether to exit upstream mode and clear calculation results.

0: Not exit

1: Exit upstream mode and clear calculation results.

(WS)

Register 18.7. HMAC_SET_INVALIDATE_JTAG_REG (0x0060)

Diagram illustrating the structure of the HMAC_SET_INVALIDATE_JTAG register. The register is 32 bits wide. Bit 31 is labeled "reserved". Bit 0 is labeled "Reset".

HMAC_SET_INVALIDATE_JTAG Configures whether or not to clear calculation results when re-enabling JTAG in downstream mode.

0: Not clear

1: Clear calculation results

(WS)

Register 18.8. HMAC_SET_INVALIDATE_DS_REG (0x0064)

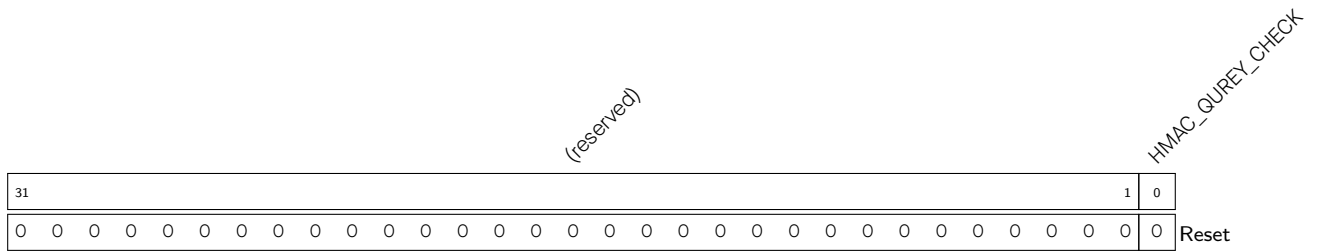
[illegible]

HMAC_SET_INVALIDATE_DS Configures whether or not to clear calculation results of the DSA module in downstream mode.

0: Not clear

1: Clear calculation results

(WS)

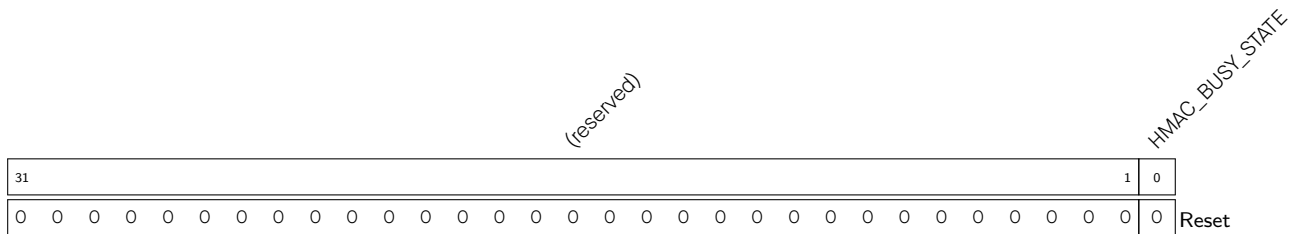
Register 18.9. HMAC_QUERY_ERROR_REG (0x0068)

HMAC_QUERY_CHECK Represents whether or not an HMAC key matches the purpose.

0: Match

1: Error

(RO)

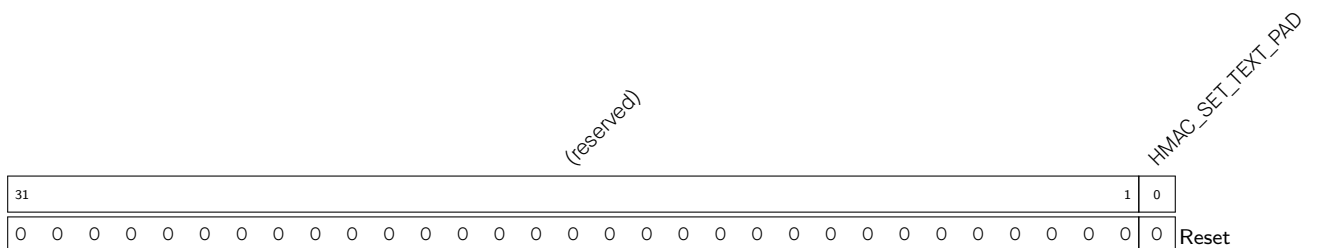
Register 18.10. HMAC_QUERY_BUSY_REG (0x006C)

HMAC_BUSY_STATE Represents whether or not HMAC is in a busy state. Before configuring HMAC, please make sure HMAC is in an IDLE state.

0: Idle

1: HMAC is still working on the calculation

(RO)

Register 18.11. HMAC_SET_MESSAGE_PAD_REG (0x00F0)

HMAC_SET_TEXT_PAD Configures whether or not the padding is applied by software.

0: Not applied by software

1: Applied by software

(WO)

Register 18.12. HMAC_ONE_BLOCK_REG (0x00F4)

(reserved)																												HMAC_SET_ONE_BLOCK	
31																												1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

HMAC_SET_ONE_BLOCK Write 1 to indicate there is only one block which already contains padding bits and there is no need for padding. (WS)

Register 18.13. HMAC_SET_PARA_PURPOSE_REG (0x0044)

(reserved)																												HMAC_PURPOSE_SET											
31																												4				3				0			
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0				Reset							

Reset

HMAC_PURPOSE_SET Configures the HMAC purpose, refer to the Table [HMAC Key Purpose](#). (WO)

Register 18.14. HMAC_SET_PARA_KEY_REG (0x0048)

(reserved)																												HMAC_KEY_SET					
31																												3		2		0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0		Reset			

Reset

HMAC_KEY_SET Configures HMAC key. There are six eFuse keys with index 0 verb+ + 5 and one key from the key manager with index 7. Write the index of the selected key to this field. (WO)

Register 18.15. HMAC_WR_JTAG_REG (0x00FC)

HMAC_WR_JTAG																															
31																															0
0																															
Reset																															

HMAC_WR_JTAG Writes the comparing input used for re-enabling JTAG. (WO)

Register 18.16. HMAC_SOFT_JTAG_CTRL_REG (0x00F8)

(reserved)																																HMAC_SOFT_JTAG_CTRL															
31																															1	0															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset													

HMAC_SOFT_JTAG_CTRL Configures whether or not to enable JTAG authentication mode.

- 0: Disable
- 1: Enable

(WS)

Register 18.17. HMAC_DATE_REG (0x01FC)

(reserved)			HMAC_DATE																											
31	30	29																												0
0	0	0x20230901																												Reset

HMAC_DATE HMAC date information/ HMAC version information. (R/W)

Chapter 19

RSA Accelerator (RSA)

19.1 Introduction

The RSA accelerator provides hardware support for high-precision computation used in various RSA asymmetric cipher algorithms, significantly reducing their run time and reducing their software complexity. Compared with RSA algorithms implemented solely in software, this hardware accelerator can speed up RSA algorithms significantly. The RSA accelerator also supports operands of different lengths, which provides more flexibility during the computation.

19.2 Features

The following functionality is supported:

- Large-number modular exponentiation with two acceleration options
- Large-number modular multiplication
- Large-number multiplication
- Operands of different lengths
- Interrupt on completion of computation

19.3 Functional Description

The RSA accelerator is activated by setting the [PCR_RSA_CLK_EN](#) bit and clearing the [PCR_RSA_RST_EN](#) bit in the [PCR_RSA_CONF_REG](#) register. Additionally, users also need to clear [PCR_DS_RST_EN](#) and [PCR_ECDSA_RST_EN](#) bits to reset [Digital Signature Algorithm \(DSA\)](#) and [Elliptic Curve Digital Signature Algorithm \(ECDSA\)](#).

The RSA accelerator is only available after the [RSA-related memories](#) are initialized. The content of the [RSA_QUERY_CLEAN_REG](#) register is 0 during initialization and will become 1 after the initialization is done. Therefore, wait until [RSA_QUERY_CLEAN_REG](#) becomes 1 before using the RSA accelerator.

The [RSA_INT_ENA_REG](#) register is used to control the interrupt triggered on completion of computation. Write 1 or 0 to this field to enable or disable the interrupt. By default, the interrupt function of the RSA accelerator is enabled.

Notice:

ESP32-C5's [Digital Signature Algorithm \(DSA\)](#) module also calls the RSA accelerator when working. Therefore, users cannot access the RSA accelerator when the [Digital Signature Algorithm \(DSA\)](#) module is working.

19.3.1 Large-Number Modular Exponentiation

Large-number modular exponentiation performs $Z = X^Y \bmod M$. The computation is based on Montgomery multiplication. Therefore, aside from the X , Y , and M arguments, two additional ones are needed — $\bar{r}$ and M' , which need to be calculated in advance by software.

The RSA accelerator supports operands of length $N = 32 \times x$, where $x \in \{1, 2, 3, \dots, 96\}$. The bit lengths of arguments Z , X , Y , M , and $\bar{r}$ can be arbitrary N , but all numbers in a calculation must be of the same length. The bit length of M' must be 32.

To represent the numbers used as operands, let us define a base- b positional notation, as follows:

$$b = 2^{32}$$

Using this notation, each number is represented by a sequence of base- b digits:

$$n = \frac{N}{32}$$

$$Z = (Z_{n-1}Z_{n-2} \cdots Z_0)_b$$

$$X = (X_{n-1}X_{n-2} \cdots X_0)_b$$

$$Y = (Y_{n-1}Y_{n-2} \cdots Y_0)_b$$

$$M = (M_{n-1}M_{n-2} \cdots M_0)_b$$

$$\bar{r} = (\bar{r}_{n-1}\bar{r}_{n-2} \cdots \bar{r}_0)_b$$

Each of the values in $Z_{n-1} \cdots Z_0$, $X_{n-1} \cdots X_0$, $Y_{n-1} \cdots Y_0$, $M_{n-1} \cdots M_0$, $\bar{r}_{n-1} \cdots \bar{r}_0$ represents one base- b digit (a 32-bit word).

Z_{n-1} , X_{n-1} , Y_{n-1} , M_{n-1} and $\bar{r}_{n-1}$ are the most significant bits of Z , X , Y , M , and $\bar{r}$, while Z_0 , X_0 , Y_0 , M_0 and $\bar{r}_0$ are the least significant bits.

If we define $R = b^n$, the additional argument $\bar{r}$ can be calculated as $\bar{r} = R^2 \bmod M$.

Also, argument M' can be calculated using the formula below:

$$M' = -M^{-1} \bmod b$$

where, M^{-1} is the [modular multiplicative inverse](#) of M , and it can be calculated with the extended binary GCD algorithm.

Large-number modular exponentiation on the ESP32-C5 can be implemented as follows:

1. Write 1 or 0 to the [RSA_INT_ENA](#) field of the register [RSA_INT_ENA_REG](#) to enable or disable the interrupt function.
2. Configure relevant registers:
 - (a) Write $(\frac{N}{32} - 1)$ to the [RSA_MODE_REG](#) register.
 - (b) Write M' to the [RSA_M_PRIME_REG](#) register.
 - (c) Configure registers related to the acceleration options, which are described later in Section [19.3.4](#).
3. Write X_i , Y_i , M_i and $\bar{r}_i$ for $i \in \{0, 1, \dots, n-1\}$ to memory blocks [RSA_X_MEM](#), [RSA_Y_MEM](#), [RSA_M_MEM](#) and [RSA_Z_MEM](#). The capacity of each memory block is 96 words. Each word of each

memory block can store one base- b digit. The memory blocks use the little endian format for storage, i.e., the least significant digit of each number is in the lowest address.

Users need to write data to each memory block only according to the length of the number; data beyond this length is ignored.

4. Write 1 to the [RSA_SET_START_MODEXP](#) field of the register [RSA_SET_START_MODEXP_REG](#) to start computation.
5. Wait for the completion of computation, which happens when the content of [RSA_QUERY_IDLE](#) becomes 1 or the RSA interrupt occurs.
6. Read the result Z_i for $i \in \{0, 1, \dots, n-1\}$ from [RSA_Z_MEM](#).
7. Write 1 to [RSA_CLEAR_INTERRUPT](#) to clear the interrupt, if you have the interrupt enabled.

After the computation, the [RSA_MODE_REG](#) register, memory blocks [RSA_Y_MEM](#) and [RSA_M_MEM](#), as well as the [RSA_M_PRIME_REG](#) remain unchanged. However, X_i in [RSA_X_MEM](#) and $\bar{r}_i$ in [RSA_Z_MEM](#) computation are overwritten, and only these overwritten memory blocks need to be re-initialized before starting another computation.

19.3.2 Large-Number Modular Multiplication

Large-number modular multiplication performs $Z = X \times Y \bmod M$. This computation is based on Montgomery multiplication. Therefore, similar to the large-number modular exponentiation, two additional arguments are needed – $\bar{r}$ and M' , which need to be calculated in advance by software.

The RSA accelerator supports large-number modular multiplication with operands of 96 different lengths.

The computation can be executed as follows:

1. Write 1 or 0 to the [RSA_INT_ENA_REG](#) register to enable or disable the interrupt function.
2. Configure relevant registers:
 - (a) Write $(\frac{N}{32} - 1)$ to the [RSA_MODE_REG](#) register.
 - (b) Write M' to the [RSA_M_PRIME_REG](#) register.
3. Write X_i , Y_i , M_i , and $\bar{r}_i$ for $i \in \{0, 1, \dots, n-1\}$ to memory blocks [RSA_X_MEM](#), [RSA_Y_MEM](#), [RSA_M_MEM](#), and [RSA_Z_MEM](#), respectively. The capacity of each memory block is 96 words. Each word of each memory block can store one base- b digit. The memory blocks use the little endian format for storage, i.e., the least significant digit of each number is in the lowest address.

Users need to write data to each memory block only according to the length of the number; data beyond this length are ignored.
4. Write 1 to the [RSA_SET_START_MODMULT](#) field.
5. Wait for the completion of computation, which happens when the content of [RSA_QUERY_IDLE](#) becomes 1 or the RSA interrupt occurs.
6. Read the result Z_i for $i \in \{0, 1, \dots, n-1\}$ from [RSA_Z_MEM](#).
7. Write 1 to [RSA_CLEAR_INTERRUPT](#) to clear the interrupt, if you have the interrupt enabled.

After the computation, the length of operands in [RSA_MODE_REG](#), the X_i in memory [RSA_X_MEM](#), the Y_i in memory [RSA_Y_MEM](#), the M_i in memory [RSA_M_MEM](#), and the M' in memory [RSA_M_PRIME_REG](#) remain unchanged. However, the $\bar{r}_i$ in memory [RSA_Z_MEM](#) has already been overwritten, and only this overwritten memory block needs to be re-initialized before starting another computation.

19.3.3 Large-Number Multiplication

Large-number multiplication performs $Z = X \times Y$. The length of result Z is twice that of operand X and operand Y . Therefore, the RSA accelerator only supports large-number multiplication with operand length $N = 32 \times x$, where $x \in \{1, 2, 3, \dots, 48\}$. The length $\hat{N}$ of result Z is $2 \times N$.

The computation can be executed as follows:

1. Write 1 or 0 to the [RSA_INT_ENA_REG](#) register to enable or disable the interrupt function.
2. Write $(\frac{\hat{N}}{32} - 1)$, i.e., $(\frac{N}{16} - 1)$ to the [RSA_MODE_REG](#) register.
3. Write X_i and Y_i for $i \in \{0, 1, \dots, n - 1\}$ to memory blocks [RSA_X_MEM](#) and [RSA_Z_MEM](#). Each word of each memory block can store one base- b digit. The memory blocks use the little endian format for storage, i.e., the least significant digit of each number is in the lowest address. n is $\frac{N}{32}$.

Write X_i for $i \in \{0, 1, \dots, n - 1\}$ to the address of the i words of the [RSA_X_MEM](#) memory block. Note that Y_i for $i \in \{0, 1, \dots, n - 1\}$ will not be written to the address of the i words of the [RSA_Z_MEM](#) register, but the address of the $n + i$ words, i.e., the base address of the [RSA_Z_MEM](#) memory plus the address offset $4 \times (n + i)$.

Users need to write data to each memory block only according to the length of the number; data beyond this length is ignored.

4. Write 1 to the [RSA_SET_START_MULT](#) register.
5. Wait for the completion of computation, which happens when the content of [RSA_QUERY_IDLE](#) becomes 1 or the RSA interrupt occurs.
6. Read the result Z_i for $i \in \{0, 1, \dots, \hat{n} - 1\}$ from the [RSA_Z_MEM](#) register. $\hat{n}$ is $2 \times n$.
7. Write 1 to [RSA_CLEAR_INTERRUPT](#) to clear the interrupt, if you have the interrupt enabled.

After the computation, the length of operands in [RSA_MODE_REG](#) and the X_i in memory [RSA_X_MEM](#) remain unchanged. However, the Y_i in memory [RSA_Z_MEM](#) has already been overwritten, and only this overwritten memory block needs to be re-initialized before starting another computation.

19.3.4 Options for Additional Acceleration

The ESP32-C5 RSA accelerator also provides [SEARCH](#) and [CONSTANT_TIME](#) options that can be configured to further accelerate the large-number modular exponentiation. By default, both options are configured as no additional acceleration.

Users can choose to use one or two of these options to further accelerate the computation. Note that, even when none of these two options is configured, using the hardware RSA accelerator is still much faster than implementing the RSA algorithm in software.

To be more specific, when neither of these two options are configured for additional acceleration, the time required to calculate $Z = X^Y \bmod M$ is solely determined by the lengths of operands. When either or both of

these two options are configured for additional acceleration, the time required is also correlated with the 0/1 distribution of Y .

To better illustrate how these two options work, first assume Y is represented in binaries as

$$Y = (\tilde{Y}_{N-1}\tilde{Y}_{N-2}\cdots\tilde{Y}_{t+1}\tilde{Y}_t\tilde{Y}_{t-1}\cdots\tilde{Y}_0)_2$$

where,

- N is the length of Y ,
- $\tilde{Y}_t$ is 1,
- $\tilde{Y}_{N-1}, \tilde{Y}_{N-2}, \dots, \tilde{Y}_{t+1}$ are all equal to 0,
- and $\tilde{Y}_{t-1}, \tilde{Y}_{t-2}, \dots, \tilde{Y}_0$ are either 0 or 1 but exactly m bits should be equal to 0 and $t - m$ bits 1, i.e., the Hamming weight of $\tilde{Y}_{t-1}\tilde{Y}_{t-2}, \dots, \tilde{Y}_0$ is $t - m$.

When either of these two options is configured for additional acceleration:

- SEARCH Option (Configuring [RSA_SEARCH_ENABLE](#) to 1 for additional acceleration)
 - The accelerator ignores the bit positions of $\tilde{Y}_i$, where $i > \alpha$. Search position α is set by configuring the [RSA_SEARCH_POS_REG](#) register. Set α to a number smaller than $N-1$, which otherwise leads to the same result as if this option is not used for additional acceleration. The best acceleration performance can be achieved by setting α to t , in which case all the 0 bits in $\tilde{Y}_{N-1}, \tilde{Y}_{N-2}, \dots, \tilde{Y}_{t+1}$ are ignored during the calculation. Note that if you set α to be less than t , then the result of the modular exponentiation $Z = X^Y \bmod M$ will be incorrect.
 - Note that this option compromises the security because it ignores some bits, which essentially shortens the key length, thus should not be enabled for applications with high security requirement.
- CONSTANT_TIME Option (Configuring [RSA_CONSTANT_TIME_REG](#) to 0 for additional acceleration)
 - The accelerator speeds up the calculation by simplifying the calculation concerning the 0 bits of Y . Therefore, the higher the proportion of bits 0 against bits 1, the better is the acceleration performance.
 - Note that this option also compromises the security because its time cost correlates with the 0/1 distribution of the key, which may be used in a Side Channel Attack (SCA), thus should not be enabled for applications with high security requirement.

Below is an example to demonstrate the performance of the RSA accelerator under different combinations of [SEARCH](#) and [CONSTANT_TIME](#) configuration.

In this example:

- We perform $Z = X^Y \bmod M$
- $N = 3072$
- $Y = 65537$
- X and M are taken at random
- When the SEARCH option is enabled, α , i.e., the position register [RSA_SEARCH_POS_REG](#), is set to 16.

Table 19.3-1 below demonstrates the time cost in clock cycles under different combinations of [SEARCH](#) and [CONSTANT_TIME](#) configuration when performing $Z = X^Y \bmod M$ as described above.

Table 19.3-1. Acceleration Performance

SEARCH Option	CONSTANT_TIME Option	Time Cost (clock cycle)
No acceleration	No acceleration	174.7×10^6
Acceleration	No acceleration	1.023×10^6
No acceleration	Acceleration	0.546×10^6
Acceleration	Acceleration	0.540×10^6

As shown in Table 19.3-1:

- The time cost is the biggest when none of these two options is configured for additional acceleration.
- The time cost is the smallest when both of these two options are configured for additional acceleration.
- The time cost can be dramatically reduced when either or both option(s) are configured for additional acceleration.

19.4 Interrupts

ESP32-C5's RSA accelerator can generate the following interrupt signal that will be sent to the [Interrupt Matrix](#).

- RSA_INTR

There is one internal interrupt source from the RSA accelerator that can generate the above interrupt signal. The interrupt source from the RSA accelerator is listed with its trigger condition and the resulting interrupt signal in Table 19.4-1.

Table 19.4-1. RSA's Internal Interrupt Source

Internal Interrupt Source	Trigger Condition	Interrupt Signal
RSA_CALC_DONE_INT	Completion of an RSA calculation	RSA_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

19.5 Memory Summary

The addresses in this section are relative to the RSA accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Size (byte)	Starting Address	Ending Address	Access
RSA_M_MEM	Represents M	384	0x0000	0x017F	R/W
RSA_Z_MEM	Represents Z	384	0x0200	0x037F	R/W
RSA_Y_MEM	Represents Y	384	0x0400	0x057F	R/W
RSA_X_MEM	Represents X	384	0x0600	0x077F	R/W

19.6 Register Summary

The addresses in this section are relative to the RSA accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Control/Configuration Registers			
RSA_M_PRIME_REG	Configures M'	0x0800	R/W
RSA_MODE_REG	Configures RSA length	0x0804	R/W
RSA_SET_START_MODEXP_REG	Starts modular exponentiation	0x080C	WT
RSA_SET_START_MODMULT_REG	Starts modular multiplication	0x0810	WT
RSA_SET_START_MULT_REG	Starts multiplication	0x0814	WT
RSA_QUERY_IDLE_REG	Represents the RSA status	0x0818	RO
RSA_CONSTANT_TIME_REG	Configures the CONSTANT_TIME option	0x0820	R/W
RSA_SEARCH_ENABLE_REG	Configures the SEARCH option	0x0824	R/W
RSA_SEARCH_POS_REG	Configures the SEARCH position	0x0828	R/W
Status Register			
RSA_QUERY_CLEAN_REG	RSA initialization status	0x0808	RO
Interrupt Registers			
RSA_INT_CLR_REG	Clears RSA interrupt	0x081C	WT
RSA_INT_ENA_REG	Enables the RSA interrupt	0x082C	R/W
Version Control Register			
RSA_DATE_REG	Version control register	0x0830	R/W

19.7 Registers

The addresses in this section are relative to the RSA accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 19.1. RSA_M_PRIME_REG (0x0800)

RSA_M_PRIME																														
31																													0	
0x000000																														Reset

RSA_M_PRIME Configures M' .
(R/W)

Register 19.2. RSA_MODE_REG (0x0804)

(reserved)																												RSA_MODE																				
31																												7	6										0									
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0										Reset										

RSA_MODE Configures the RSA length.
(R/W)

Register 19.3. RSA_SET_START_MODEXP_REG (0x080C)

(reserved)																																RSA_SET_START_MODEXP															
31																															1	0															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset												

RSA_SET_START_MODEXP Configures whether or not to start the modular exponentiation.
0: No effect
1: Start
(WT)

Register 19.4. RSA_SET_START_MODMULT_REG (0x0810)

[illegible]

RSA_SET_START_MODMULT Configures whether or not to start the modular multiplication.

0: No effect

1: Start

(WT)

Register 19.5. RSA_SET_START_MULT_REG (0x0814)

[illegible]

RSA_SET_START_MULT Configures whether or not to start the multiplication.

0: No effect

1: Start

(WT)

Register 19.6. RSA_QUERY_IDLE_REG (0x0818)

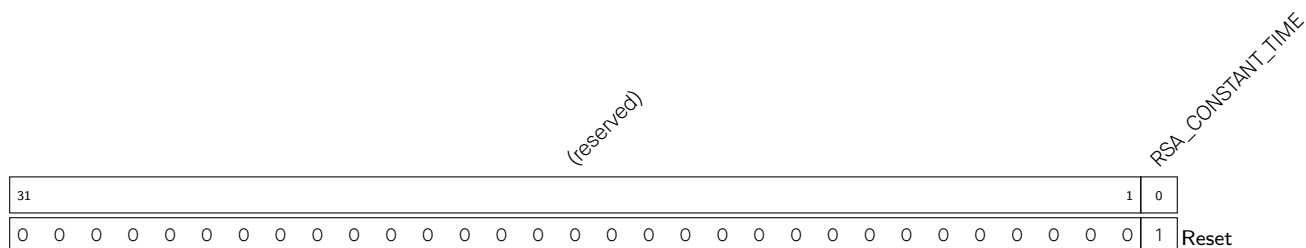
[illegible]

RSA_QUERY_IDLE Represents the RSA status.

0: Busy

1: Idle

(RO)

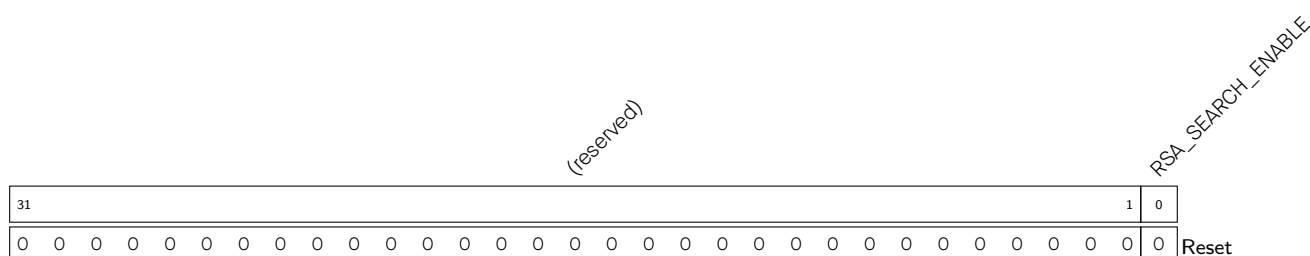
Register 19.7. RSA_CONSTANT_TIME_REG (0x0820)

RSA_CONSTANT_TIME Configures the CONSTANT_TIME option.

0: Acceleration

1: No acceleration (default)

(R/W)

Register 19.8. RSA_SEARCH_ENABLE_REG (0x0824)

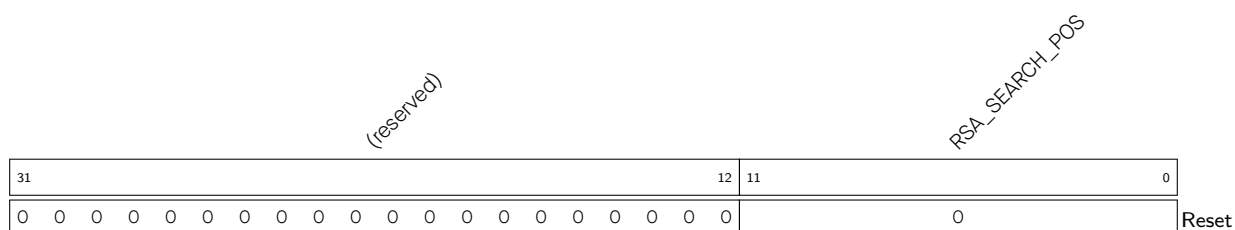
RSA_SEARCH_ENABLE Configures the SEARCH option.

0: No acceleration (default)

1: Acceleration

This option should be used together with [RSA_SEARCH_POS_REG](#).

(R/W)

Register 19.9. RSA_SEARCH_POS_REG (0x0828)

RSA_SEARCH_POS Configures the starting address to start search.

This field should be used together with [RSA_SEARCH_ENABLE_REG](#). The field is only valid when [RSA_SEARCH_ENABLE](#) is 1.

(R/W)

Register 19.10. RSA_QUERY_CLEAN_REG (0x0808)

(reserved)																																RSA_QUERY_CLEAN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

RSA_QUERY_CLEAN Represents whether or not the RSA memory completes initialization.

0: Not complete

1: Completed

(RO)

Register 19.11. RSA_INT_CLR_REG (0x081C)

(reserved)																															RSA_CLEAR_INTERRUPT																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RSA_CLEAR_INTERRUPT Write 1 to clear the RSA interrupt.

(WT)

Register 19.12. RSA_INT_ENA_REG (0x082C)

(reserved)																														RSA_INT_ENA	
31																														1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RSA_INT_ENA Write 1 to enable the RSA interrupt.

(R/W)

Register 19.13. RSA_DATE_REG (0x0830)

(reserved)		RSA_DATE	
31	30	29	0
0	0	0x20200618	
			Reset

RSA_DATE Version control register.
 (R/W)

Chapter 20

SHA Accelerator (SHA)

20.1 Introduction

ESP32-C5 integrates an SHA accelerator, which is a hardware device that speeds up the SHA algorithm significantly, compared to a SHA algorithm implemented solely in software. The SHA accelerator integrated in ESP32-C5 has two working modes, which are [Typical SHA](#) and [DMA-SHA](#).

20.2 Features

The following functionality is supported:

- The following hash algorithms introduced in [FIPS PUB 180-4 Spec](#)
 - SHA-1
 - SHA-224
 - SHA-256
- Two working modes
 - Typical SHA
 - DMA-SHA
- Interleaved function when working in Typical SHA working mode
- Interrupt function when working in DMA-SHA working mode

20.3 Working Modes

The SHA accelerator integrated in ESP32-C5 has two working modes.

- [Typical SHA Working Mode](#): all the data is written and read via CPU directly.
- [DMA-SHA Working Mode](#): all the data is read via DMA. That is, users can configure the DMA controller to read all the data needed for hash operation, thus releasing CPU for completing other tasks.

The SHA accelerator is activated by setting the [PCR_SHA_CLK_EN](#) bit and clearing the [PCR_SHA_RST_EN](#) bit in the [PCR_SHA_CONF_REG](#) register. Additionally, users also need to clear [PCR_DS_RST_EN](#), [PCR_HMAC_RST_EN](#), and [PCR_ECDSA_RST_EN](#) bits to reset [Digital Signature Algorithm \(DSA\)](#), [HMAC Accelerator \(HMAC\)](#), and [Elliptic Curve Digital Signature Algorithm \(ECDSA\)](#).

Users can start the SHA accelerator with different working modes by configuring registers [SHA_START_REG](#) and [SHA_DMA_START_REG](#). For details, please see Table [20.3-1](#).

Table 20.3-1. SHA Accelerator Working Mode

Working Mode	Configuration Method
Typical SHA	Set SHA_START_REG to 1
DMA-SHA	Set SHA_DMA_START_REG to 1

Users can choose hash algorithms by configuring the [SHA_MODE_REG](#) register. For details, please see Table 20.3-2.

Table 20.3-2. SHA Hash Algorithm Selection

Hash Algorithm	SHA_MODE_REG Configuration
SHA-1	0
SHA-224	1
SHA-256	2

Notice:

ESP32-C5's [Digital Signature Algorithm \(DSA\)](#) and [HMAC Accelerator \(HMAC\)](#) modules also call the SHA accelerator when working. Therefore, users cannot access the SHA accelerator when these modules are working.

20.4 Function Description

The SHA accelerator generates the message digest via two steps: [Preprocessing](#) and [Hash operation](#).

20.4.1 Preprocessing

Preprocessing consists of three steps: [padding the message](#), [parsing the message into message blocks](#) and [setting the initial hash value](#).

20.4.1.1 Padding the Message

The SHA accelerator can only process message blocks of 512 bits. Thus, all the messages should be padded to a multiple of 512 bits before the hash operation.

Suppose that the length of the message M is m bits. Then M shall be padded as introduced below:

1. First, append the bit "1" to the end of the message;
2. Second, append k bits of zeros, where k is the smallest, non-negative solution to the equation $m + 1 + k \equiv 448 \pmod{512}$;
3. Last, append the 64-bit block of value equal to the number m expressed using a binary representation.

For more details, please refer to [FIPS PUB 180-4 Spec](#) > Section "Padding the Message".

20.4.1.2 Parsing the Message

The message and its padding must be parsed into N 512-bit blocks, $M^{(1)}, M^{(2)}, \dots, M^{(N)}$. Since the 512 bits of the input block may be expressed as sixteen 32-bit words, the first 32 bits of message block i are denoted

$M_0^{(i)}$, the next 32 bits are $M_1^{(i)}$, and so on up to $M_{15}^{(i)}$.

During the task, all the message blocks are written into the `SHA_M_n_REG`: $M_0^{(i)}$ is stored in `SHA_M_0_REG`, $M_1^{(i)}$ stored in `SHA_M_1_REG`, ..., and $M_{15}^{(i)}$ stored in `SHA_M_15_REG`.

Note:

For more information about “message block”, please refer to [FIPS PUB 180-4 Spec](#) > Section “Glossary of Terms and Acronyms”.

20.4.1.3 Setting the Initial Hash Value

Before hash operation begins for any secure hash algorithms, the initial Hash value $H^{(0)}$ must be set based on different algorithms. However, the SHA accelerator uses the initial Hash values (constant C) stored in the hardware for hash tasks.

20.4.2 Hash Operation

After the preprocessing, the ESP32-C5 SHA accelerator starts to hash a message M and generates message digest of different lengths, depending on different hash algorithms. As described above, the ESP32-C5 SHA accelerator supports two working modes, which are [Typical SHA](#) and [DMA-SHA](#). The operation process for the SHA accelerator under two working modes is described in the following subsections.

20.4.2.1 Typical SHA Mode Process

Usually, the SHA accelerator will process all blocks of a message and produce a message digest before starting the computation of the next message digest.

However, ESP32-C5 SHA also supports optional “interleaved” message digest calculation in Typical SHA mode, which means before SHA completes all blocks of the current message, users are given a chance to insert new computation of another message digest upon the completion of each individual block of the current message.

Specifically, users can read out the message digest from registers `SHA_H_n_REG` after completing part of a message digest calculation, and use the SHA accelerator for a different calculation. After the different calculation completes, users can restore the previous message digest to registers `SHA_H_n_REG`, and resume the accelerator with the previously paused calculation.

Typical SHA Process

1. Select a hash algorithm.
 - Configure the `SHA_MODE_REG` register based on Table 20.3-2.
2. Process the current message block.
 - Write the message block in registers `SHA_M_n_REG`.
3. Start the SHA accelerator¹.
 - If this is the first time to execute this step, set the `SHA_START_REG` register to 1 to start the SHA accelerator. In this case, the accelerator uses the initial hash value stored in hardware for a given algorithm configured in Step 1 to start the calculation;

- If this is not the first time to execute this step², set the [SHA_CONTINUE_REG](#) register to 1 to start the SHA accelerator. In this case, the accelerator uses the hash value stored in the [SHA_H_n_REG](#) register to start the calculation.
4. Check the progress of the current message block.
 - Poll register [SHA_BUSY_REG](#) until the content of this register becomes 0, indicating the accelerator has completed the calculation for the current message block and now is in the “idle” status³.
 5. Decide if you have more message blocks to process:
 - If yes, please go back to Step 2.
 - Otherwise, please continue.
 6. Obtain the message digest.
 - Read the message digest from registers [SHA_H_n_REG](#).

Note:

1. In this step, the software can also write the next message block (to be processed) in registers [SHA_M_n_REG](#), if any, while the hardware starts SHA calculation, to save time.
2. You are resuming the SHA accelerator with the previously paused calculation.
3. Here you can decide if you want to insert other calculations. If yes, please go to the [process for interleaved calculations](#) for details.

As mentioned above, ESP32-C5 SHA accelerator supports “interleaving” calculation under the **Typical SHA working mode**.

The process to implement interleaved calculation is described below.

1. Prepare to hand the SHA accelerator over for an interleaved calculation by storing the following data of the previous calculation.
 - The selected hash algorithm configured in the [SHA_MODE_REG](#) register.
 - The message digest stored in registers [SHA_H_n_REG](#).
2. Perform the interleaved calculation. For the detailed process of the interleaved calculation, please refer to [Typical SHA process](#) or [DMA-SHA process](#), depending on the working mode of your interleaved calculation.
3. Prepare to hand the SHA accelerator back to the previously paused calculation by restoring the following data of the previous calculation.
 - Write the previously stored hash algorithm back to register [SHA_MODE_REG](#).
 - Write the previously stored message digest back to registers [SHA_H_n_REG](#).
4. Write the next message block from the previous paused calculation in registers [SHA_M_n_REG](#), and set the [SHA_CONTINUE_REG](#) register to 1 to restart the SHA accelerator with the previously paused calculation.

20.4.2.2 DMA-SHA Mode Process

ESP32-C5 SHA accelerator does not support “interleaving” message digest calculation at the level of individual message blocks when using DMA, which means you cannot insert new calculation before a complete DMA-SHA process (of one or more message blocks) completes. In this case, users who need interleaved operation are recommended to divide the message blocks and perform several DMA-SHA calculations, instead of trying to compute all the messages in one go.

Single DMA-SHA calculation supports up to 63 data blocks.

In contrast to the Typical SHA working mode, when the SHA accelerator is working under the DMA-SHA mode, all data read are completed via DMA. Therefore, users are required to configure the DMA controller following the description in Chapter 3 [GDMA Controller \(GDMA\)](#).

DMA-SHA process

1. Select a hash algorithm.
 - Select a hash algorithm by configuring the [SHA_MODE_REG](#) register. For details, please refer to Table 20.3-2.
2. Configure the [SHA_INT_ENA_REG](#) register to enable or disable interrupt (Set 1 to enable).
3. Configure the number of message blocks.
 - Write the number of message blocks M to the [SHA_DMA_BLOCK_NUM_REG](#) register.
4. Start the DMA-SHA calculation.
 - If the current DMA-SHA calculation follows a previous calculation, firstly write the message digest from the previous calculation to registers [SHA_H_n_REG](#), then write 1 to register [SHA_DMA_CONTINUE_REG](#) to start SHA accelerator;
 - Otherwise, write 1 to register [SHA_DMA_START_REG](#) to start the accelerator.
5. Wait till the completion of the DMA-SHA calculation, which happens when:
 - The content of [SHA_BUSY_REG](#) register becomes 0, or
 - An SHA interrupt occurs. In this case, please clear interrupt by writing 1 to the [SHA_INT_CLEAR_REG](#) register.
6. Obtain the message digest:
 - Read the message digest from registers [SHA_H_n_REG](#).

20.4.3 Message Digest

After the hash task completes, the SHA accelerator writes the message digest from the task to registers [SHA_H_n_REG](#) (n : 0~7). The lengths of the generated message digest are different depending on different hash algorithms. For details, see Table 20.4-1 below:

Table 20.4-1. The Storage and Length of Message Digest from Different Algorithms

Hash Algorithm	Length of Message Digest (in bits)	Storage ¹
SHA-1	160	SHA_H_0_REG ~ SHA_H_4_REG
SHA-224	224	SHA_H_0_REG ~ SHA_H_6_REG
SHA-256	256	SHA_H_0_REG ~ SHA_H_7_REG

¹ The message digest is stored in registers from most significant bits to the least significant bits, with the first word stored in register [SHA_H_0_REG](#) and the second word stored in register [SHA_H_1_REG](#)... For details, please see subsection [20.4.1.2](#).

20.4.4 Interrupt

ESP32-C5's SHA accelerator can generate the following interrupt signal(s) that will be sent to the [Interrupt Matrix](#).

- SHA_INTR

There is an internal interrupt source from SHA that can generate the above interrupt signal(s). The interrupt source from SHA is listed with their trigger conditions and the resulted interrupt signal(s) in Table [20.4-2](#).

Table 20.4-2. SHA's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
SHA_CALC_DONE_INT	Completion of an message digest calculation in DMA-SHA mode	SHA_INTR

Note:

- For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter [3 Interrupt Matrix \[to be added later\]](#) > Section [3.1 Interrupt Terminology](#).
- Different from the standard interrupt register group, the interrupt register group of SHA only contains the INT_ENA ([SHA_INT_ENA_REG](#)) and INT_CLR ([SHA_INT_CLEAR_REG](#)) fields, and does not support users to read the INT_RAW and INT_ST fields.

20.5 Register Summary

The addresses in this section are relative to the SHA accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

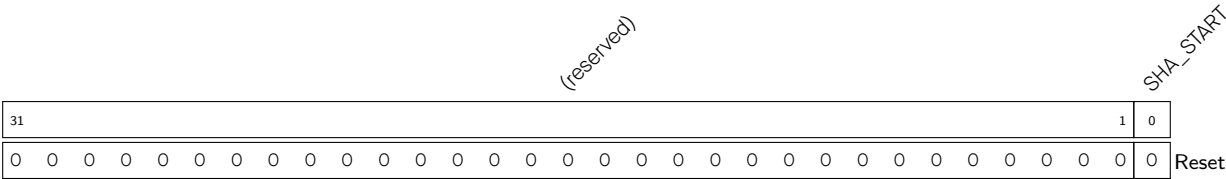
Name	Description	Address	Access
Control/Configuration Registers			
SHA_MODE_REG	Configures SHA algorithm	0x0000	R/W
SHA_CONTINUE_REG	Continues SHA operation (only effective in Typical SHA mode)	0x0014	WO
SHA_DMA_START_REG	Starts the SHA accelerator for DMA-SHA operation	0x001C	WO
SHA_START_REG	Starts the SHA accelerator for Typical SHA operation	0x0010	WO
SHA_DMA_CONTINUE_REG	Continues SHA operation (only effective in DMA-SHA mode)	0x0020	WO
SHA_DMA_BLOCK_NUM_REG	Block number register (only effective in DMA-SHA mode)	0x000C	R/W
Status Registers			
SHA_BUSY_REG	Represents if SHA Accelerator is busy or not	0x0018	RO
Interrupt Registers			
SHA_INT_CLEAR_REG	DMA-SHA interrupt clear register	0x0024	WO
SHA_INT_ENA_REG	DMA-SHA interrupt enable register	0x0028	R/W
Data Registers			
SHA_H_0_REG	Hash value	0x0040	R/W
SHA_H_1_REG	Hash value	0x0044	R/W
SHA_H_2_REG	Hash value	0x0048	R/W
SHA_H_3_REG	Hash value	0x004C	R/W
SHA_H_4_REG	Hash value	0x0050	R/W
SHA_H_5_REG	Hash value	0x0054	R/W
SHA_H_6_REG	Hash value	0x0058	R/W
SHA_H_7_REG	Hash value	0x005C	R/W
SHA_M_0_REG	Message	0x0080	R/W
SHA_M_1_REG	Message	0x0084	R/W
SHA_M_2_REG	Message	0x0088	R/W
SHA_M_3_REG	Message	0x008C	R/W
SHA_M_4_REG	Message	0x0090	R/W
SHA_M_5_REG	Message	0x0094	R/W
SHA_M_6_REG	Message	0x0098	R/W
SHA_M_7_REG	Message	0x009C	R/W
SHA_M_8_REG	Message	0x00A0	R/W
SHA_M_9_REG	Message	0x00A4	R/W
SHA_M_10_REG	Message	0x00A8	R/W
SHA_M_11_REG	Message	0x00AC	R/W
SHA_M_12_REG	Message	0x00B0	R/W
SHA_M_13_REG	Message	0x00B4	R/W
SHA_M_14_REG	Message	0x00B8	R/W
SHA_M_15_REG	Message	0x00BC	R/W

Name	Description	Address	Access
Version Register			
SHA_DATE_REG	Version control register	0x002C	R/W

20.6 Registers

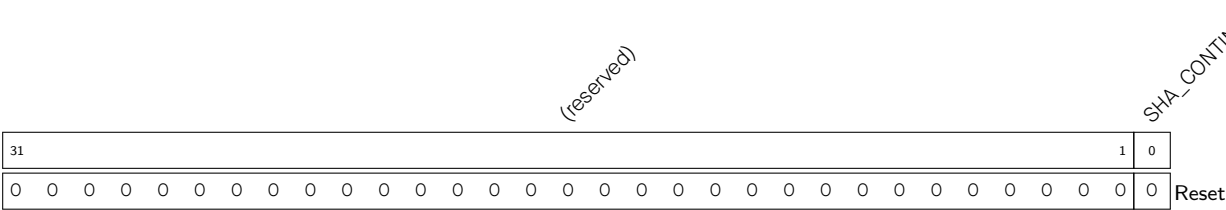
The addresses in this section are relative to the SHA accelerator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 20.1. SHA_START_REG (0x0010)



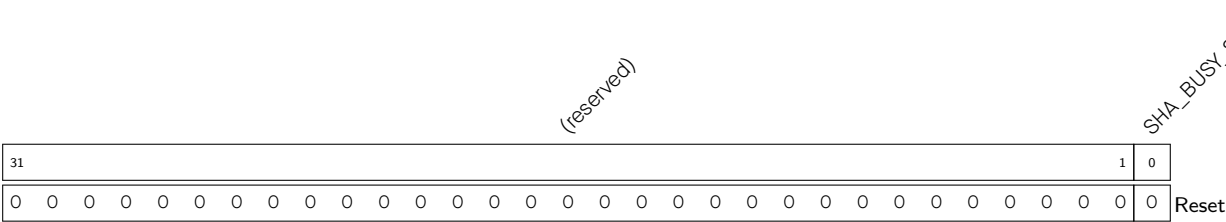
SHA_START Write 1 to start Typical SHA calculation. (WO)

Register 20.2. SHA_CONTINUE_REG (0x0014)



SHA_CONTINUE Write 1 to continue Typical SHA calculation. (WO)

Register 20.3. SHA_BUSY_REG (0x0018)



SHA_BUSY_STATE Represents the states of SHA accelerator.

- 0: idle
- 1: busy

(RO)

Register 20.4. SHA_DMA_START_REG (0x001C)

(reserved)																															SHA_DMA_START																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SHA_DMA_START Write 1 to start DMA-SHA calculation. (WO)

Register 20.5. SHA_DMA_CONTINUE_REG (0x0020)

(reserved)																															SHA_DMA_CONTINUE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SHA_DMA_CONTINUE Write 1 to continue DMA-SHA calculation. (WO)

Register 20.6. SHA_INT_CLEAR_REG (0x0024)

(reserved)																															SHA_CLEAR_INTERRUPT																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SHA_CLEAR_INTERRUPT Write 1 to clear DMA-SHA interrupt. (WO)

Register 20.7. SHA_INT_ENA_REG (0x0028)

(reserved)																															SHA_INTERRUPT_ENA				
31																															1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SHA_INTERRUPT_ENA Write 1 to enable DMA-SHA interrupt. (R/W)

(reserved)

SHA_DATE

SHA_DATE Version control register. (R/W)

(reserved)

SHA_MODE

0: SHA-1
1: SHA-224
2: SHA-256
3 ~ 7: invalid value
(R/W)

(reserved)

SHA_DMA_BLOCK_NUM

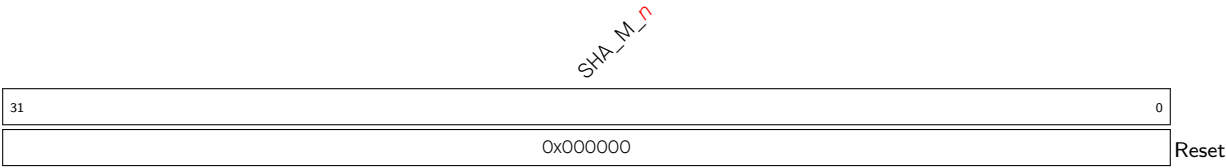
SHA_DMA_BLOCK_NUM Configures the DMA-SHA block number. (R/W)

SHA_H_n

0x000000

SHA_H_*n* Represents the *n*th 32-bit piece of the Hash value. (R/W)

Register 20.12. SHA_M_ *n* _REG (*n*: 0-15) (0x0080+4**n*)



SHA_M_ *n* Represents the *n*th 32-bit piece of the message. (R/W)

Chapter 21

Digital Signature Algorithm (DSA)

21.1 Overview

The Digital Signature Algorithm (DSA) is used to verify the authenticity and integrity of a message using a cryptographic algorithm. This can be used to validate a device's identity to a server or to check the integrity of a message.

ESP32-C5 includes a Digital Signature Algorithm (DSA) module providing hardware acceleration of messages' signatures based on RSA. HMAC is used as the key derivation function (KDF) to output the *DSA_KEY* key using a key stored in eFuse or deployed by the Key Manager as the input key. Subsequently, the DSA module uses *DSA_KEY* to decrypt the pre-encrypted parameters and calculate the signature. The whole process happens in hardware so that neither the decryption key for the RSA parameters nor the input key for the HMAC key derivation function can be seen by users while calculating the signature.

21.2 Features

- RSA digital signatures with key length up to 3072 bits
- Encrypted private key data, only decryptable by the DSA module
- SHA-256 digest to protect private key data against tampering by an attacker

21.3 Functional Description

21.3.1 Overview

The DSA peripheral calculates RSA signatures as $Z = X^Y \bmod M$, where Z is the signature, X is the input message, and Y and M are the RSA private key parameters.

Private key parameters are stored in flash as ciphertext. They are decrypted using a key (*DSA_KEY*) which can:

- Only be calculated by the DSA peripheral via the HMAC peripheral. The required inputs (*HMAC_KEY*) to generate the key (*DSA_KEY*) are only stored in eFuse and can only be accessed by the HMAC peripheral. That is to say, the DSA peripheral hardware can decrypt the private key, and the private key in plaintext is never accessed by the software. For more detailed information about eFuse and HMAC peripherals, please refer to [Chapter 2 eFuse Controller \(eFuse\) \[to be added later\]](#) and [Chapter 18 HMAC Accelerator \(HMAC\)](#).
- Deploy from Key Manager module. The *DSA_KEY* must be deployed by the Key Manager in a secured way.

The input message X will be sent directly to the DSA peripheral by the software each time a signature is needed. After the RSA signature operation, the signature Z is read back by the software.

For better understanding, we define some symbols and functions here, which are only applicable to this chapter:

- 1^s : A bit string consisting of s bits with the value of “1”.
- $[x]_s$: A bit string of s bits, in which s is an integer multiple of 8 bits. If x is a number ($x < 2^s$), it is represented in little-endian byte order in the bit string. x may be a variable such as $[Y]_{4096}$ or a hexadecimal constant such as $[0x0C]_8$. If necessary, the value $[x]_t$ can be right-padded with $(s - t)$ number of zeros to reach s bits in length, and finally get $[x]_s$. For example, $[0x05]_8 = 00000101$, $[0x05]_{16} = 0000010100000000$, $[0x0005]_{16} = 0000000000000101$, $[0x13]_8 = 00010011$, $[0x13]_{16} = 0001001100000000$, $[0x0013]_{16} = 0000000000010011$.
- $||$: A bit string concatenation operator for joining multiple-bit strings into a longer bit string.

21.3.2 Private Key Operands

Private key operands Y (private key exponent) and M (key modulus) are generated by the user. They have a particular RSA key length (up to 3072 bits). Two additional private key operands are needed: $\bar{r}$ and M' . These two operands are derived from Y and M .

Operands Y , M , $\bar{r}$, and M' are encrypted by the user along with an authentication digest and stored as a single ciphertext C . C is input to the DSA peripheral in this encrypted format, decrypted by the hardware, and then used for RSA signature calculation. A detailed description of how to generate C is provided in Section 21.3.3.

The DSA peripheral supports RSA signature calculation $Z = X^Y \bmod M$, in which the length of operands should be $N = 32 \times x$ where $x \in \{1, 2, 3, \dots, 96\}$. The bit lengths of arguments Z , X , Y , M , and $\bar{r}$ should be an arbitrary value in N , and all of them in a calculation must be of the same length, while the bit length of M' should always be 32. For more detailed information about RSA calculation, please refer to Section 19.3.1 *Large-Number Modular Exponentiation* in Chapter 19 *RSA Accelerator (RSA)*.

21.3.3 Software Prerequisites

If you want to use the DSA module, the software needs a series of preparations, as shown in Figure 21.3-1. The left side lists preparations required by the software before the hardware starts the RSA signature calculation, while the right side lists the hardware workflow during the entire calculation procedure.

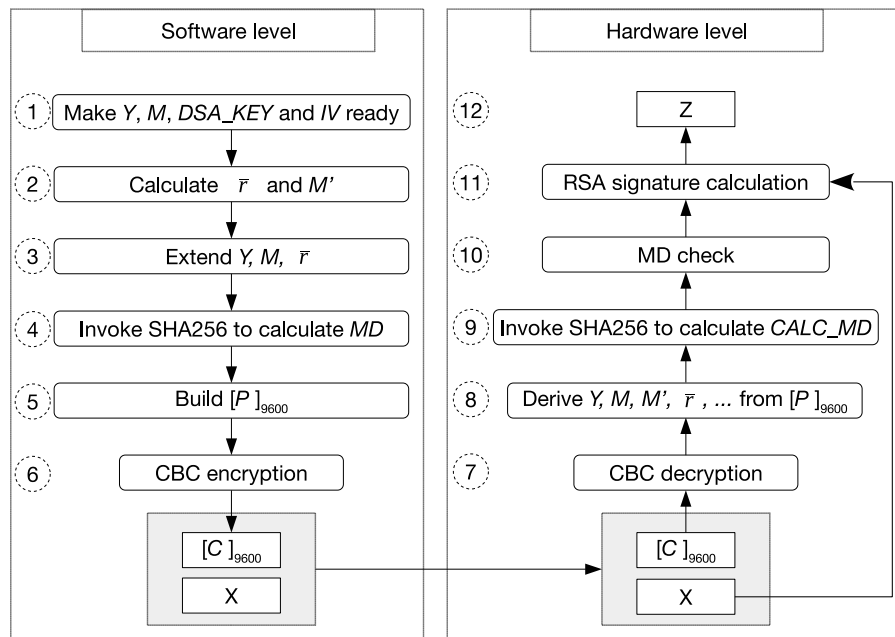


Figure 21.3-1. Software Preparations and Hardware Working Process

Note:

1. The software preparation (left side in Figure 21.3-1) is a one-time operation before any signature is calculated, while the hardware calculation (right side in Figure 21.3-1) repeats for every signature calculation.
2. Software preparation requires configuring the clock reset. For more information, please refer to Chapter 6 [Reset and Clock](#).

Users need to follow the steps shown in the left part of Figure 21.3-1 to calculate C . Detailed instructions are as follows:

- **Step 1:** Prepare operands Y and M whose lengths should meet the requirements in Section 21.3.2. Define $[L]_{32} = \frac{N}{32} - 1$ (i.e., for RSA 3072, $[L]_{32} = [0x60-1]_{32}$). Prepare $[DSA\_KEY]_{256}$:

- when calculated by the DSA peripheral via the HMAC peripheral: $DSA\_KEY = \text{HMAC-SHA256}([HMAC\_KEY]_{256}, 1^{256})$.
- when from Key Manager module: just use the deployed DSA_KEY

Generate a random $[IV]_{128}$ which should meet the requirements of the AES-CBC block encryption algorithm. For more information on AES, please refer to Chapter 16 [AES Accelerator \(AES\)](#).

- **Step 2:** Calculate $\bar{r}$ and M' based on M .
- **Step 3:** Extend Y , M , and $\bar{r}$ in order to get $[Y]_{3072}$, $[M]_{3072}$, and $[\bar{r}]_{3072}$, respectively. This step is only required for Y , M , and $\bar{r}$ whose length are less than 3072 bits, since their largest length are 3072 bits.
- **Step 4:** Calculate MD authentication code using the SHA-256:
 $[MD]_{256} = \text{SHA256}([Y]_{3072} || [M]_{3072} || [\bar{r}]_{3072} || [M']_{32} || [L]_{32} || [IV]_{128})$
- **Step 5:** Build $[P]_{9600} = ([Y]_{3072} || [M]_{3072} || [\bar{r}]_{3072} || [Box]_{384})$, where $[Box]_{384} = ([MD]_{256} || [M']_{32} || [L]_{32} || [\beta]_{64})$ and $[\beta]_{64}$ is a PKCS#7 padding value, i.e., a $[0x0808080808080808]_{64}$ string composed of 8 bytes (0x80). The purpose of $[\beta]_{64}$ is to make the bit length of P a multiple of 128.

- **Step 6:** Calculate $C = [C]_{9600} = \text{AES-CBC-ENC}([P]_{9600}, [DSA\_KEY]_{256}, [IV]_{128})$, where C is the ciphertext with a length of 1200 bytes. C can also be calculated as $C = [C]_{9600} = ([\hat{Y}]_{3072} || [\hat{M}]_{3072} || [\hat{r}]_{3072} || [\hat{Box}]_{384})$, where $[\hat{Y}]_{3072}$, $[\hat{M}]_{3072}$, $[\hat{r}]_{3072}$, $[\hat{Box}]_{384}$ are the four sub-parameters of C , and correspond to the ciphertext of $[Y]_{3072}$, $[M]_{3072}$, $[r]_{3072}$, $[Box]_{384}$ respectively.

21.3.4 DSA Operation at the Hardware Level

The hardware operation is triggered each time a digital signature needs to be calculated. The inputs are the pre-generated private key ciphertext C , a unique message X , and IV .

The DSA operation at the hardware level can be divided into the following three stages:

1. Decryption: Step 7 and 8 in Figure 21.3-1

The decryption process is the inverse of Step 6 in Figure 21.3-1. The DSA module will call the AES accelerator to decrypt C in CBC block mode and get the resulting plaintext. The decryption process can be represented by $P = \text{AES-CBC-DEC}(C, DSA\_KEY, IV)$, where IV (i.e., $[IV]_{128}$) is defined by the user. $[DSA\_KEY]_{256}$ is

- when calculated by the DSA peripheral via the HMAC peripheral: provided by the HMAC module, derived from $HMAC\_KEY$ stored in eFuse, which is not readable by users.
- when from Key Manager module: DSA_KEY deployed by Key Manager, which is not readable by users.

With P , the DSA module can derive $[Y]_{3072}$, $[M]_{3072}$, $[r]_{3072}$, $[M']_{32}$, $[L]_{32}$, MD authentication code, and the padding value $[\beta]_{64}$. This process is the inverse of Step 5.

2. Check: Step 9 and 10 in Figure 21.3-1

The DSA module will perform two checks: MD check and padding check. The padding check is not shown in Figure 21.3-1, as it happens at the same time as the MD check.

- MD check: The DSA module calls SHA-256 to calculate the hash value $[CALC\_MD]_{256}$ ($[CALC\_MD]_{256}$ is calculated the same way and with same parameters as $[MD]_{256}$, see step 4). Then, $[CALC\_MD]_{256}$ is compared against the MD authentication code $[MD]_{256}$ from step 4. Only when the two match does the MD check pass.
- Padding check: The DSA module checks if $[\beta]_{64}$ complies with the aforementioned PKCS#7 format. Only when $[\beta]_{64}$ complies with the format does the padding check pass.

The DSA module will only perform subsequent operations if MD check passes. If the padding check fails, a warning is generated, but it does not affect the subsequent operations.

3. Calculation: Step 11 and 12 in Figure 21.3-1

The DSA module treats X (input by the user) and Y , M , $\bar{r}$ (decrypted in step 8) as big numbers. With M' , all operands to perform $X^Y \bmod M$ are in place. The operand length is defined by L only. The DSA module will calculate the signed result Z by calling RSA to perform $Z = X^Y \bmod M$.

21.3.5 DSA Operation at the Software Level

The software steps below should be followed each time a digital signature needs to be calculated. The inputs are the pre-generated private key ciphertext C , a unique message X , and IV . These software steps trigger

the hardware steps described in Section 21.3.4.

We assume that the software has called the HMAC peripheral and the HMAC peripheral has calculated DSA_KEY based on $HMAC\_KEY$.

1. **Prerequisites:** Prepare operands C , X , IV according to Section 21.3.3.
2. **Activate the DSA peripheral:** Write 1 to [DS_SET_START_REG](#).
3. **Check if DSA_KEY is ready** depending on from where the DSA_KEY comes from:
 - Write 1 to [DS_KEY_SOURCE_REG](#) to select the DSA_KEY deployed by the Key Manager. Verify that DSA_KEY has been successfully deployed by checking the register field `KEYMNG_KEY_DS_VLD`. When its value is 1, it indicates that DSA_KEY has been successfully deployed and is ready.
 - Write 0 to [DS_KEY_SOURCE_REG](#) to select the DSA_KEY from HMAC. Poll [DS_QUERY_BUSY_REG](#) until the software reads 0.

If the software does not read 0 in [DS_QUERY_BUSY_REG](#) after approximately 1 ms, it indicates a problem with HMAC initialization. In such a case, the software can read register [DS_QUERY_KEY_WRONG_REG](#) to get more information:

- If the software reads 0 in [DS_QUERY_KEY_WRONG_REG](#), it indicates that the HMAC peripheral has not been called.
 - If the software reads any value from 1 to 15 in [DS_QUERY_KEY_WRONG_REG](#), it indicates that HMAC was called, but the DSA module did not successfully get the DSA_KEY value from the HMAC peripheral. This may indicate that the HMAC operation has been interrupted due to a software concurrency problem.
4. **Write IV to memory block [DS_IV_MEM](#):** Write the content in the IV block to the memory block [DS_IV_MEM](#), which is 16 bytes. For more information on the IV block, please refer to Chapter 16 [AES Accelerator \(AES\)](#).
 5. **Write X to memory block [DS_X_MEM](#):** Write X_i ($i \in \{0, 1, \dots, n-1\}$), where $n = \frac{N}{32}$, to memory block [DS_X_MEM](#) whose capacity is 96 words. Each word can store one base- b digit. The memory block uses the little endian format for storage, i.e., the least significant digit of the operand is in the lowest address. Words in [DS_X_MEM](#) block after the configured length of X (N bits, as described in Section 21.3.2), are ignored.
 6. **Write C to corresponding memory blocks:** Write the four sub-parameters of C to corresponding memory blocks:
 - Write $\widehat{Y}_i$ ($i \in \{0, 1, \dots, 95\}$) to [DS_Y_MEM](#).
 - Write $\widehat{M}_i$ ($i \in \{0, 1, \dots, 95\}$) to [DS_M_MEM](#).
 - Write $\widehat{r}_i$ ($i \in \{0, 1, \dots, 95\}$) to [DS_RB_MEM](#).
 - write $\widehat{Box}_i$ ($i \in \{0, 1, \dots, 11\}$) to [DS_BOX_MEM](#).

The capacity of [DS_Y_MEM](#), [DS_M_MEM](#), and [DS_RB_MEM](#) is 96 words, whereas the capacity of [DS_BOX_MEM](#) is only 12 words. Each word can store one base- b digit. The memory blocks use the little endian format for storage, i.e., the least significant digit of the operand is in the lowest address.

7. **Start DSA operation:** Write 1 to register [DS_SET_CONTINUE_REG](#).

8. **Wait for the operation to be completed:** Poll register [DS_QUERY_BUSY_REG](#) until the software reads 0.
9. **Query check result:** Read register [DS_QUERY_CHECK_REG](#) and conduct subsequent operations as illustrated below based on the return value:
 - If the value is 0, it indicates that both the padding check and MD check pass. Users can continue to get the signed result Z .
 - If the value is 1, it indicates that the padding check passes but MD check fails. The signed result Z is invalid. The operation will resume directly from Step [11](#).
 - If the value is 2, it indicates that the padding check fails but the MD check passes. Users can continue to get the signed result Z . But please note that the data does not comply with the aforementioned PKCS#7 padding format, which may not be what you want.
 - If the value is 3, it indicates that both the padding check and MD check fail. In this case, some fatal errors have occurred and the signed result Z is invalid. The operation will resume directly from Step [11](#).
10. **Read the signed result:** Read the signed result Z_i ($i \in \{0, 1, \dots, n - 1\}$), where $n = \frac{N}{32}$, from memory block [DS_Z_MEM](#). The memory block stores Z in little-endian byte order.
11. **Exit the operation:** Write 1 to [DS_SET_FINISH_REG](#), and then poll [DS_QUERY_BUSY_REG](#) until the software reads 0.

After the operation, all the input/output registers and memory blocks are cleared.

21.4 Memory Summary

The addresses in this section are relative to the Digital Signature Algorithm base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Name	Description	Size (byte)	Starting Address	Ending Address	Access
DS_Y_MEM	Memory block Y	512	0x0000	0x01FF	R/W
DS_M_MEM	Memory block M	512	0x0200	0x03FF	R/W
DS_RB_MEM	Memory block $\bar{r}$	512	0x0400	0x05FF	R/W
DS_BOX_MEM	Memory block Box	48	0x0600	0x062F	R/W
DS_IV_MEM	Memory block IV	16	0x0630	0x063F	R/W
DS_X_MEM	Memory block X	512	0x0800	0x09FF	R/W
DS_Z_MEM	Memory block Box Z	512	0x0A00	0x0BFF	R/W

21.5 Register Summary

The addresses in this section are relative to Digital Signature Algorithm base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

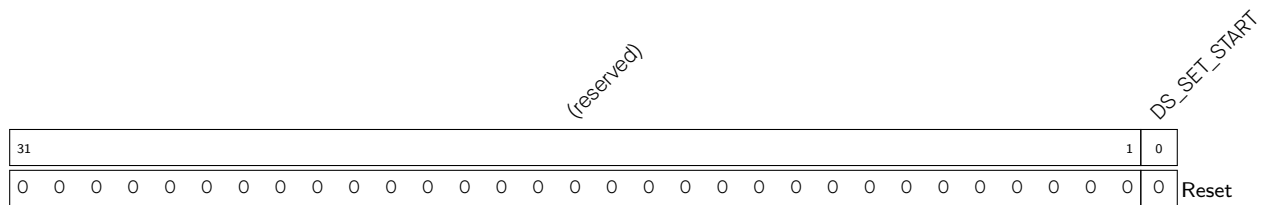
The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Control/Status registers			
DS_SET_START_REG	Activates the DSA module	0x0E00	WT
DS_SET_CONTINUE_REG	Continues DSA operation	0x0E04	WT
DS_SET_FINISH_REG	Ends DSA operation	0x0E08	WT
DS_QUERY_BUSY_REG	Status of the DSA module	0x0E0C	RO
DS_QUERY_KEY_WRONG_REG	Checks the reason why <i>DS_KEY</i> is not ready	0x0E10	RO
DS_QUERY_CHECK_REG	Queries DSA check result	0x0E14	RO
Configuration registers			
DS_KEY_SOURCE_REG	Configures DSA key source	0x0E18	R/W
version control register			
DS_DATE_REG	Version control register	0x0E20	R/W

21.6 Registers

The addresses in this section are relative to Digital Signature Algorithm base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 21.1. DS_SET_START_REG (0x0E00)



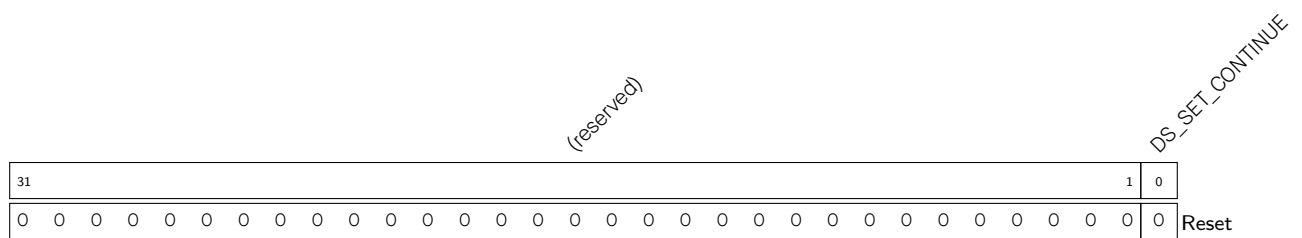
DS_SET_START Configures whether or not to activate the DSA peripheral.

0: Invalid

1: Activate the DSA peripheral

(WT)

Register 21.2. DS_SET_CONTINUE_REG (0x0E04)



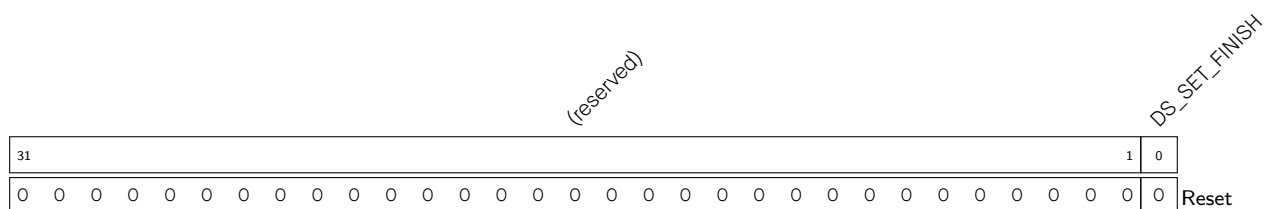
DS_SET_CONTINUE Configures whether or not to continue the DSA operation.

0: No effect

1: Continue the DSA operation

(WT)

Register 21.3. DS_SET_FINISH_REG (0x0E08)



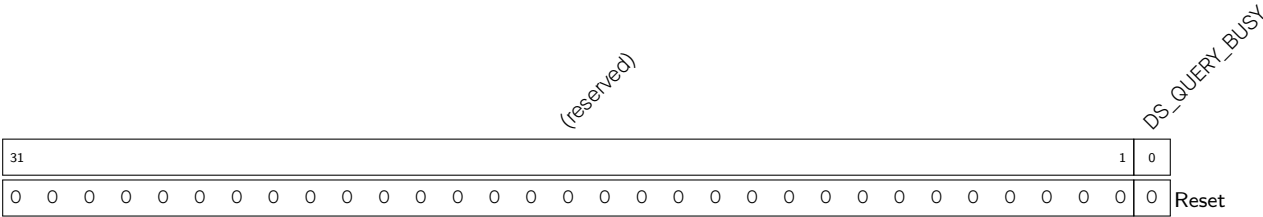
DS_SET_FINISH Configures whether or not to end the DSA operation.

0: No effect

1: End the DSA operation

(WT)

Register 21.4. DS_QUERY_BUSY_REG (0x0E0C)

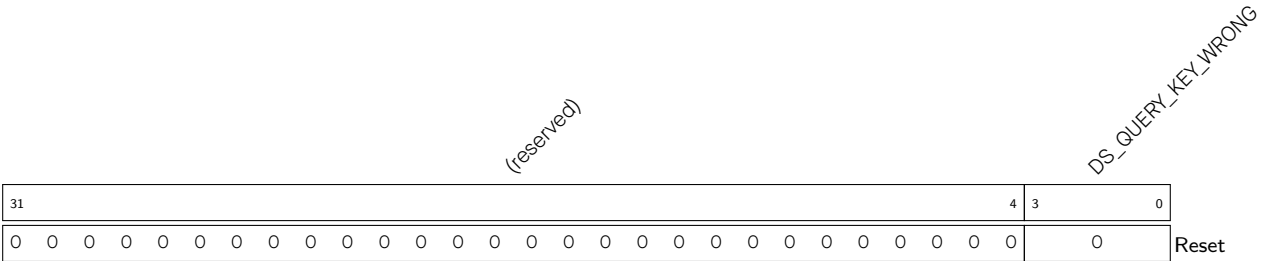


DS_QUERY_BUSY Represents whether or not the DSA module is idle.

- 0: The DSA module is idle
- 1: The DSA module is busy

(RO)

Register 21.5. DS_QUERY_KEY_WRONG_REG (0x0E10)

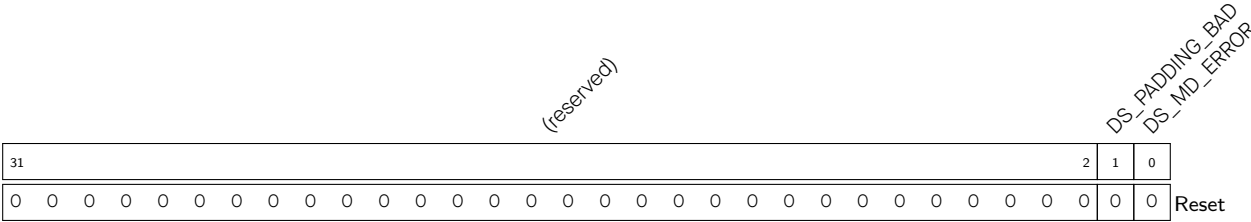


DS_QUERY_KEY_WRONG Represents the specific problem with HMAC initialization.

- 0: HMAC is not called
- 1-15: HMAC was activated, but the DSA peripheral did not successfully receive the *DSA_KEY* from the HMAC peripheral.
- Larger than 15: invalid

(RO)

Register 21.6. DS_QUERY_CHECK_REG (0x0E14)



DS_MD_ERROR Represents whether or not the MD check passes.

0: The MD check passes

1: The MD check fails

(RO)

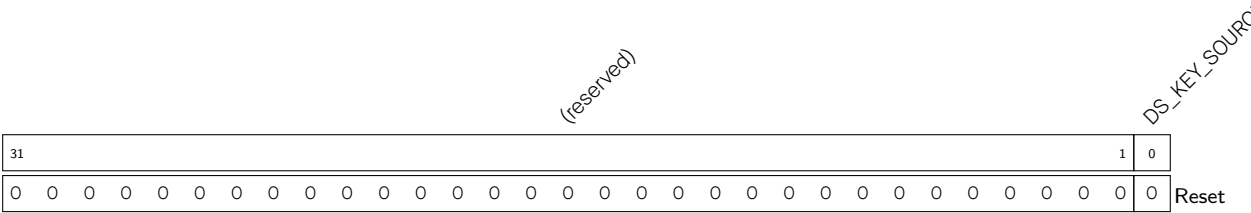
DS_PADDING_BAD Represents whether or not the padding check passes.

0: The padding check passes

1: The padding check fails

(RO)

Register 21.7. DS_KEY_SOURCE_REG (0x0E18)



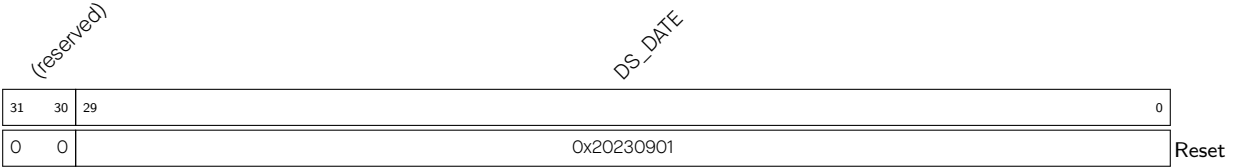
DS_KEY_SOURCE Represents the source of the *DSA_KEY*.

0: HMAC

1: Key Manager

(R/W)

Register 21.8. DS_DATE_REG (0x0E20)



DS_DATE Version control register (R/W)

Chapter 22

Elliptic Curve Digital Signature Algorithm (ECDSA)

22.1 Introduction

In cryptography, the Elliptic Curve Digital Signature Algorithm (ECDSA) offers a variant of the Digital Signature Algorithm (DSA) which uses elliptic-curve cryptography.

ESP32-C5's ECDSA accelerator provides a secure and efficient environment for computing ECDSA signatures. It offers fast computations while ensuring the confidentiality of the signing process to prevent information leakage. This makes it a valuable tool for applications that require high-speed cryptographic operations with strong security guarantees. By using the ECDSA accelerator, users can be confident that their data is protected without sacrificing performance.

22.2 Features

ESP32-C5's ECDSA accelerator supports:

- Digital signature generation and verification
- Three different elliptic curves, namely, P-192, P-256, and P-384 defined in [FIPS 186-5](#)
- Multiple hash algorithms for message hash in the ECDSA operation, including SHA-224, SHA-256, SHA-384, SHA-512, SHA-512-224, and SHA-512-256, defined in [FIPS PUB 180-4 Spec](#)
- Dynamic access permission in different operation statuses to ensure information security

22.3 ECDSA Basics

22.3.1 Domain Parameters

ECDSA uses parameters that define the elliptic curve over a finite field, as well as the generator point and the order of the base point. These parameters are usually referred to as domain parameters, and they are required for key generation, signature generation, and signature verification.

The domain parameters used in ECDSA consist of the followings:

- The elliptic curve domain parameters, which include:
 - The prime modulus p , which specifies the size of the finite field over which the elliptic curve is defined.
 - The base point G on the curve, which can be used to generate public keys.

- The order n of the base point, which is the number of points on the curve that can be generated by repeatedly adding G to itself.
- The parameters for the hash function, which include:
 - The hash algorithm used to generate a fixed-length hash value from the message being signed.
 - The length of the hash value, which determines the size of the signature.

Together, these parameters define the ECDSA domain.

22.3.2 Key Generation

The process of key generation in ECDSA is described below:

1. Select an elliptic curve domain by defining:
 - the prime modulus p
 - the coefficients a and b
 - the base point G and its order n
2. Generate a private key:
 - A private key d is a randomly chosen integer between 1 and $n-1$. It is essential to use a secure [Random Number Generator \(RNG\)](#) to ensure that the private key is truly random and cannot be predicted or reproduced.
 - The private key is used for signature generation. This key must be kept secret and secure.
 - Once a private key is chosen, it should be burned into the eFuse in ESP32-C5. (Please refer to Chapter [2 eFuse Controller \(eFuse\) \[to be added later\]](#)).
3. Compute the public key:
 - The public key Q is calculated as $Q = dG$.
4. Export the public key:
 - The public key Q is typically represented as a pair of coordinates (Q_x, Q_y) that can be shared with others.

22.3.3 Signature Generation

The ECDSA signature generation process is described below:

1. Select a message m to be signed.
2. Calculate the hash of the message e : e is equal to $\text{HASH}(m)$, where HASH is a cryptographic hash function, such as SHA-256.
3. Compute the digest of the message hash z : Let z be the L_n leftmost bits of e , where L_n is the bit length of the base point order n .
4. Select a random number k , which is chosen between 1 and $n-1$, where n is the order of the base point on the elliptic curve.
5. Compute the signature: The signature is calculated as follows:

- (a) Compute the point $(x, y) = kG$
 - (b) Calculate $r = x \bmod n$. If r is equal to zero, return to the previous step and select a new random value of k .
 - (c) Calculate $s = k^{-1} * (z + d * r) \bmod n$. If s is equal to zero, return to step 4 and select a new random value of k .
 - (d) The signature is the pair (r, s) .
6. Send the message m and signature (r, s) to the recipient.

22.3.4 Signature Verification

The recipient can then use the public key associated with the private key used for signature generation to verify the signature. The verification process involves checking that the signature was generated using the correct private key and that the signature is valid for the given message.

The ECDSA signature verification process is described below:

1. Obtain public key Q and receive the message m and signature (r, s) .
2. Calculate the hash of the message e : e is equal to $\text{HASH}(m)$, where HASH is a cryptographic hash function, such as SHA-256.
3. Compute the digest of the message z : Let z be the L_n leftmost bits of e , where L_n is the bit length of the base point order n .
4. Verify the signature: The signature is verified as follows:
 - (a) Verify that r and s are integers between 1 and $n-1$, where n is the order of the base point on the elliptic curve. If either r or s is outside of this range, the signature is invalid.
 - (b) Calculate $u_1 = z * s^{-1} \bmod n$ and $u_2 = r * s^{-1} \bmod n$.
 - (c) Calculate the point $(x_1, y_1) = u_1 * G + u_2 * Q$, where G is the base point on the elliptic curve, and Q is the public key associated with the private key used for signature generation.
 - (d) Verify that $r = x_1 \bmod n$. If r is not equal to $x_1 \bmod n$, the signature is invalid.
5. Accept or reject the signature: If the signature is valid, the recipient can be confident that the message was not tampered with and that it came from the expected sender, thus can accept the message as authentic. Otherwise, the recipient rejects the message as invalid.

22.4 Functional Description

This section describes the details of ESP32-C5's ECDSA accelerator.

22.4.1 ECDSA Working Modes

The ECDSA accelerator integrated in the ESP32-C5 has three working modes, which are Signature Generation, Signature Verification and Public Key Export modes.

Users can select the working mode for the ECDSA accelerator by configuring `ECDSA_WORK_MODE` according to Table 22.4-1 below.

Table 22.4-1. ECDSA Working Mode

ECDSA_WORK_MODE	Working Mode
0	Signature Verification
1	Signature Generation
2	Public Key Export

Users can select the elliptic curves used by configuring [ECDSA_ECC_CURVE](#) according to Table 22.4-2.

Table 22.4-2. ECDSA Elliptic Curves Selection

ECDSA_ECC_CURVE	Elliptic Curve
0	P-192
1	P-256
2	P-384

Users can select the SHA algorithms for message hash by configuring [ECDSA_SHA_MODE](#) according to Table 22.4-3.

Table 22.4-3. ECDSA SHA Algorithm

ECDSA_SHA_MODE	SHA Algorithm
1	SHA-224
2	SHA-256
3	SHA-384
4	SHA-512
5	SHA-512-224
6	SHA-512-256
Others	Invalid

Additionally, users can check the working status of the ECDSA accelerator by inquiring the [ECDSA_STATE_REG](#) register and comparing the return value against the Table 22.4-4 below.

Table 22.4-4. ECDSA Working Status

ECDSA_STATE_REG	Status	Description
0	IDLE	Idle or completed operation. Corresponding to IDLE stage.
1	LOAD	Waiting for users to load information into ECDSA. Corresponding to LOAD stage.
2	GAIN	Waiting for users to gain information from ECDSA. Corresponding to GAIN stage.
3	BUSY	In the middle of a hardware operation. Corresponding to PREP, PROC, and POST stage.

22.4.2 Data and Data Block

ESP32-C5's ECDSA accelerator can operate on data of 192, 256, 384, or 512 bits. For example, the data ($D[255 : 0]$) can be divided into 32-bit data blocks.

Take 256-bit long data as an example, $D[n][31 : 0]$ ($n = 0, 1, \dots, 7$). Data blocks with the smaller serial number correspond to the lower binary bits. To be specific:

$$D[255 : 0] = D[7][31 : 0], D[6][31 : 0], D[5][31 : 0], D[4][31 : 0], D[3][31 : 0], D[2][31 : 0], D[1][31 : 0], D[0][31 : 0]$$

22.4.2.1 Writing Data

Writing data means writing data to an ECDSA memory block and using this data as the input to the ECDSA algorithm. To be specific, writing data to an ECDSA memory block means writing $D[n][31 : 0]$ to the “starting address of this ECDSA memory block + $4 \times n$ ”. For a 256-bit long data:

- write $D[0]$ to “starting address”
- write $D[1]$ to “starting address + 4”
- ...
- write $D[7]$ to “starting address + 28”

Note:

When storing data, write only the data block of the required length. Do not append a 0 to the most significant bit. For example, for 192-bit data, store only the 192 bits without appending 0.

22.4.2.2 Reading Data

Reading data means reading data from the starting address of an ECDSA memory block and using this data as the output from the ECDSA algorithm. To be specific, reading data from an ECDSA memory block means reading $D[n][31 : 0]$ from the “starting address of this ECDSA memory block + $4 \times n$ ”. For a 256-bit long data:

- read $D[0]$ from “starting address”
- read $D[1]$ from “starting address + 4”
- ...
- read $D[7]$ from “starting address + 28”

Note:

When reading data, only the data block of the required length needs to be read. For example, to read 192-bit data, simply read the lower 192 bits (i.e., 6 data blocks).

22.4.2.3 Padding the Message

The SHA accelerator can only process message blocks of 512 bits. Thus, all the messages should be padded to a multiple of 512 bits before the hash operation.

Suppose that the length of the message M is L_M bits. Then M shall be padded as introduced below:

1. First, append the bit “1” to the end of the message;
2. Second, append L_A bits of zeros, where L_A is the smallest, non-negative solution to the equation $L_M + 1 + L_A \equiv 448 \bmod 512$;
3. Last, append the 64-bit block of value equal to the number L_M expressed using a binary representation.

For more details, please refer to [FIPS PUB 180-4 Spec](#) > Section “Padding the Message”.

22.4.2.4 Parsing the Message

The message and its padding must be parsed into N 512-bit message blocks: $M^{(1)}, M^{(2)}, \dots, M^{(N)}$.

Note:

1. For more details about “parsing the message”, please refer to [FIPS PUB 180-4 Spec](#) > Section “Parsing the Message”.
2. For more information on “message block”, please refer to [FIPS PUB 180-4 Spec](#) > Section “Glossary of Terms and Acronyms”.

22.4.3 Security Features

To ensure the security of the ECDSA operation process, the ECDSA accelerator implements a variety of security functions.

22.4.3.1 High Anti-Attack Performance

ESP32-C5’s ECDSA accelerator leverages ECC’s enhanced anti-attack performance (refer to Section [17.4.3 Enhancing Anti-Attack Performance](#)) every time it performs an operation. This means that each time a signature is generated and verified, ECDSA consumes:

- the same amount of time;
- the same amount of power.

This provides ESP32-C5’s ECDSA accelerator strong anti-attack performance.

22.4.3.2 Dynamic Access Permission

ESP32-C5’s ECDSA accelerator has implemented a dynamic access permission mechanism to prevent any possibility of key theft by tampering with the configuration or accessing the data during the operation.

By implementing this dynamic access permission mechanism, the accesses for ECDSA registers are designed to vary in different statuses. For example, [ECDSA_CONF_REG](#) is only available for reading and writing when the accelerator is in the IDLE status. In this way, the configuration information is protected from reading or writing when the accelerator is in other statuses, such as LOAD, GAIN, and BUSY. For details about all ECDSA working statuses, please refer to Table [22.4-4](#).

For detailed information on the dynamic access permission of each ECDSA register, please refer to Section [Register Summary](#).

22.4.3.3 Hardware Occupation

During the ECDSA operation, the following hardware modules will be occupied by ESP32-C5's ECDSA accelerator:

- SHA Accelerator
- ECC Accelerator

Among them, the SHA accelerator will be released when [ECDSA_SHA_RELEASE_INT](#) is triggered, while the ECC accelerator will be occupied during the whole ECDSA operation.

Note:

Hardware occupation is a mechanism to protect multiplexed modules and storage space. When a module is hardware occupied, the user will fail to:

- read or write data to the module's registers or memories.
- disable the module clock.
- reset the module.

After hardware occupation is finished, the occupied module will be automatically reset. In addition, when the user performs a software reset to the master module, all the occupied modules will be reset at the same time.

22.5 Programming Procedures

22.5.1 ECDSA Process

The overall ECDSA process consists of the following six stages.

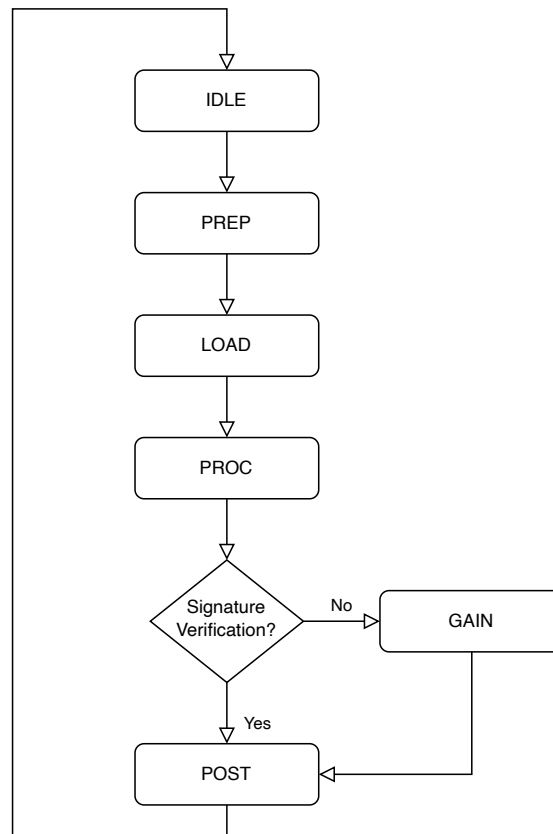


Figure 22.5-1. ECDSA Process

The detailed programming procedures of each stage are described in the following sections.

22.5.1.1 IDLE Stage

In the IDLE stage:

1. Configure the static parameters including eFuse bits:
 - (a) ECDSA_KEY: The value of private key d in ECDSA. To correctly configure the key value in eFuse, users need to write the key value in KEY n ($n = 0-5$), and set the corresponding EFUSE_KEY_PURPOSE_ n as ECDSA_KEY. Please refer to Chapter 2 *eFuse Controller (eFuse) [to be added later]* for more detailed configuration steps.
2. Configure ECDSA_CONF_REG, including the following fields:
 - (a) ECDSA_WORK_MODE: Select the working mode of the ECDSA accelerator.
 - (b) ECDSA_ECC_CURVE: Select the elliptic curve of the ECDSA accelerator.
 - (c) ECDSA_SOFTWARE_SET_Z: Configure whether to use direct input z .
 - (d) ECDSA_DETERMINISTIC_K: Configure whether to use deterministic k .
3. Configure the register field ECDSA_START to enter the PREP stage.

Note:

For more details about Deterministic ECDSA, please refer to [RFC6979](#).

22.5.1.2 PREP Stage

In the PREP stage, [ECDSA_STATE_REG](#) is BUSY, and the ECDSA accelerator performs preparation.

Wait till the PREP stage to end by polling [ECDSA_BUSY](#) until it is not BUSY. Then the ECDSA accelerator will automatically enter the LOAD stage.

22.5.1.3 LOAD Stage

In the LOAD stage:

1. Provide input z into the ECDSA accelerator using one of the following options:
 - Direct input z : write z to [ECDSA_MEM_Z](#).
 - ECDSA SHA interface: generate z from the message. For details, please refer to Section [22.5.1.7](#).
2. According to the selected [ECDSA_WORK_MODE](#), configure as follows:
 - Signature Verification:
 - (a) write signature (r, s) to [ECDSA_MEM_R](#) and [ECDSA_MEM_S](#).
 - (b) write public key (Qx, Qy) to [ECDSA_MEM_Qx](#) and [ECDSA_MEM_Qy](#).
 - Signature Generation:
 - (a) no other configuration is required.
 - Public Key Export:
 - (a) no other configuration is required.
3. Write 1 to [ECDSA_LOAD_DONE](#), indicating that the configuration is done. Then the accelerator will automatically enter the PROC stage.

22.5.1.4 PROC Stage

In the PROC stage, [ECDSA_STATE_REG](#) is BUSY, and the ECDSA accelerator performs ECDSA operation based on the selected working mode.

Wait till the PROC stage to end by polling [ECDSA_BUSY](#) until it is not BUSY. Then the ECDSA accelerator will automatically enter the either the GAIN stage or the POST stage depending on the selected working mode:

- Signature generation: GAIN stage
- Signature verification: POST stage

22.5.1.5 GAIN Stage

When the Signature Generation mode or the Public Key Export mode is selected, the ECDSA accelerator enters the GAIN stage after PROC stage:

1. Read data from the ECDSA memory:
 - read signature (r, s) from [ECDSA_MEM_R](#) and [ECDSA_MEM_S](#) only in the Signature Generation mode.
 - read public key (Q_x, Q_y) from [ECDSA_MEM_Qx](#) and [ECDSA_MEM_Qy](#) both in the Signature Generation mode and the Public Key Export mode.
2. Write 1 to [ECDSA_GET_DONE](#), indicating that the GAIN stage is done. Then the accelerator will automatically enter the POST stage.

22.5.1.6 POST Stage

In the POST stage, [ECDSA_STATE_REG](#) is BUSY, and the ECDSA accelerator performs some wrap-up work of ECDSA operation.

Wait till the POST stage to end by polling [ECDSA_BUSY](#) until it is not BUSY. Then the ECDSA accelerator will automatically return to the IDLE stage.

22.5.1.7 ECDSA SHA Interface

ESP32-C5's ECDSA accelerator can automatically execute hash operation and generates z based on a direct input message.

For message hash, ECDSA accelerator supports SHA algorithms SHA-224 (only valid when P-192 is selected as the elliptic curve) and SHA-256.

To use the ECDSA SHA interface, complete the following steps:

1. Pad the message by following the steps described in Section [22.4.2.3](#).
2. Parse the message and its padding into message blocks. See details in Section [22.4.2.4](#).
3. Process the current message block.
 - Write the current message block into [ECDSA_MEM_M](#).
4. Start the ECDSA SHA interface¹.
 - If this is the first time to execute this step, write 1 to [ECDSA_SHA_START](#) to start the ECDSA SHA interface;
 - If this is not the first time to execute this step², write 1 to [ECDSA_SHA_CONTINUE](#) to continue the operation.
5. Check the progress of the current message block processing by polling.
 - Poll register [ECDSA_SHA_BUSY](#) until it's IDLE, indicating that the interface has completed the operation for the current message block.
6. Process the next message block:
 - If there is message block to process, go back to Step [3](#).
 - If there is no more message blocks, exit.

Note:

1. In this step, the software can also write the next message block to be processed, if any, in register `ECDSA_MEM_M`, while the interface starts SHA operation, to save time.
2. You are resuming the ECDSA SHA interface with the previously paused operation.

22.5.2 Clock

ECDSA uses two types of clocks:

- `clk_ecdsa`: function clock for ECDSA to operate
- `clk_apb_ecdsa`: bus clock used to configure ECDSA registers

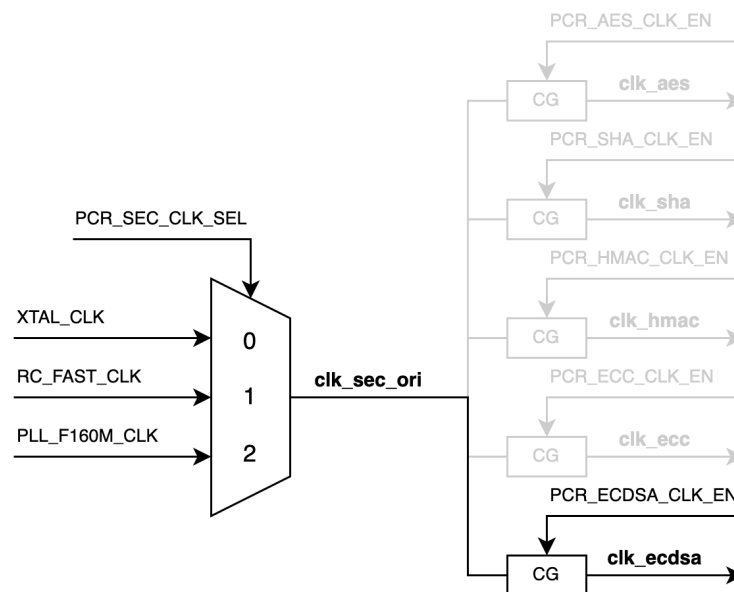


Figure 22.5-2. `clk_ecdsa` Diagram

As shown in Figure 22.5-2 *clk_ecdsa Diagram*, the secure function source clock `clk_sec_ori` can be sourced from:

- `XTAL_CLK`
- `RC_FAST_CLK`
- `PLL_F160M_CLK`

ECDSA's function clock `clk_ecdsa` is configured by the clock gating register `PCR_ECDSA_CLK_EN`. When this register is set to 1, the clock gate is enabled, and `clk_ecdsa` is activated.

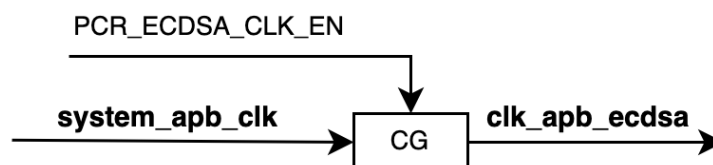


Figure 22.5-3. `clk_apb_ecdsa` Diagram

As shown in Figure [22.5-3 clk_apb_ecdsa Diagram](#), ECDSA's configuration clock clk_apb_ecdsa can be sourced from SYSTEM_APB_CLK.

ECDSA's configuration clock clk_apb_ecdsa is controlled by the clock gating register [PCR_ECDSA_CLK_EN](#). When this register is set to 1, the clock gate is enabled, and clk_apb_ecdsa is activated.

For more information about clocks, see Section [6 Reset and Clock](#).

22.5.3 Reset

The ECDSA module supports full reset. Write 1 and then 0 to [PCR_ECDSA_RST_EN](#) to reset the entire ECDSA module.

22.6 Interrupts

ESP32-C5's ECDSA accelerator can generate the interrupt signal ECDSA_INTR and send it to [Interrupt Matrix](#).

The ECDSA accelerator has four interrupt sources that can generate the ECDSA_INTR interrupt signal as follows. Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#).

- ECDSA_PREP_DONE_INT: triggered on the completion of the PREP stage. This interrupt source is configured by the following registers:
 - [ECDSA_PREP_DONE_INT_RAW](#): stores the raw interrupt status of ECDSA_PREP_DONE_INT.
 - [ECDSA_PREP_DONE_INT_ST](#): indicates the status of the ECDSA_PREP_DONE_INT interrupt. This field is generated by enabling/disabling the [ECDSA_PREP_DONE_INT_RAW](#) field.
 - [ECDSA_PREP_DONE_INT_ENA](#): enables/disables the ECDSA_PREP_DONE_INT interrupt.
 - [ECDSA_PREP_DONE_INT_CLR](#): set this bit to clear the ECDSA_PREP_DONE_INT interrupt status. By setting this bit to 1, fields [ECDSA_PREP_DONE_INT_RAW](#) and [ECDSA_PREP_DONE_INT_ST](#) will be cleared.
- ECDSA_PROC_DONE_INT: triggered on the completion of the PROC stage. This interrupt source is configured by the following registers:
 - [ECDSA_PROC_DONE_INT_RAW](#): stores the raw interrupt status of ECDSA_PROC_DONE_INT.
 - [ECDSA_PROC_DONE_INT_ST](#): indicates the status of the ECDSA_PROC_DONE_INT interrupt. This field is generated by enabling/disabling the [ECDSA_PROC_DONE_INT_RAW](#) field.
 - [ECDSA_PROC_DONE_INT_ENA](#): enables/disables the ECDSA_PROC_DONE_INT interrupt.
 - [ECDSA_PROC_DONE_INT_CLR](#): set this bit to clear the ECDSA_PROC_DONE_INT interrupt status. By setting this bit to 1, fields [ECDSA_PROC_DONE_INT_RAW](#) and [ECDSA_PROC_DONE_INT_ST](#) will be cleared.
- ECDSA_POST_DONE_INT: triggered on the completion of the POST stage. This interrupt source is configured by the following registers:
 - [ECDSA_POST_DONE_INT_RAW](#): stores the raw interrupt status of ECDSA_POST_DONE_INT.

- [ECDSA_POST_DONE_INT_ST](#): indicates the status of the ECDSA_POST_DONE_INT interrupt. This field is generated by enabling/disabling the [ECDSA_POST_DONE_INT_RAW](#) field.
 - [ECDSA_POST_DONE_INT_ENA](#): enables/disables the ECDSA_POST_DONE_INT interrupt.
 - [ECDSA_POST_DONE_INT_CLR](#): set this bit to clear the ECDSA_POST_DONE_INT interrupt status. By setting this bit to 1, fields [ECDSA_POST_DONE_INT_RAW](#) and [ECDSA_POST_DONE_INT_ST](#) will be cleared.
- ECDSA_SHA_RELEASE_INT: triggered when SHA is released. This interrupt source is configured by the following registers:
 - [ECDSA_SHA_RELEASE_INT_RAW](#): stores the raw interrupt status of ECDSA_SHA_RELEASE_INT.
 - [ECDSA_SHA_RELEASE_INT_ST](#): indicates the status of the ECDSA_SHA_RELEASE_INT interrupt. This field is generated by enabling/disabling the [ECDSA_SHA_RELEASE_INT_RAW](#) field via [ECDSA_SHA_RELEASE_INT_ENA](#).
 - [ECDSA_SHA_RELEASE_INT_ENA](#): enables/disables the ECDSA_SHA_RELEASE_INT interrupt.
 - [ECDSA_SHA_RELEASE_INT_CLR](#): set this bit to clear the ECDSA_SHA_RELEASE_INT interrupt status. By setting this bit to 1, fields [ECDSA_SHA_RELEASE_INT_RAW](#) and [ECDSA_SHA_RELEASE_INT_ST](#) will be cleared.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

22.7 Memory Blocks

ECDSA's memory blocks store input data and output data of the ECDSA operation.

Table 22.7-1. ECDSA Memory Blocks

Memory	Size (byte)	Starting Address [*]	Ending Address [*]	Access
ECDSA_MEM_M	128	0x280	0x2FF	R/W
ECDSA_MEM_R	48	0x3E0	0x40F	R/W
ECDSA_MEM_S	48	0x410	0x43F	R/W
ECDSA_MEM_Z	48	0x440	0x46F	R/W
ECDSA_MEM_Qx	48	0x470	0x49F	R/W
ECDSA_MEM_Qy	48	0x4A0	0x4CF	R/W

^{*} Address offset related to the ECDSA accelerator base address is provided in Table 4.3-2 in Chapter 4 *System and Memory*.

22.8 Register Summary

The addresses in this section are relative to the ECDSA base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

To enhance security, ECDSA registers have different read and write permissions in different operating statuses. The following is the abbreviation and corresponding relationship of each state:

- PI: IDLE status, corresponding to IDLE stage.
- PL: LOAD status, corresponding to LOAD stage.
- PG: GAIN status, corresponding to GAIN stage.
- PB: BUSY status, corresponding to PREP, PROC, and POST stage.

Other abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access				
			PI	PL	PG	PB	
Data Memory	See Table 22.7-1.						
Configuration Registers							
ECDSA_CONF_REG	ECDSA configuration register	0x0004	R/W	N/A			
ECDSA_START_REG	ECDSA start register	0x001C	WT			N/A	
Clock and Reset Register							
ECDSA_CLK_REG	ECDSA clock gate register	0x0008	R/W	N/A			
Interrupt Registers							
ECDSA_INT_RAW_REG	ECDSA interrupt raw register	0x000C	RO/WTC/SS				
ECDSA_INT_ST_REG	ECDSA interrupt status register	0x0010	RO				
ECDSA_INT_ENA_REG	ECDSA interrupt enable register	0x0014	R/W				
ECDSA_INT_CLR_REG	ECDSA interrupt clear register	0x0018	WT				
Status Registers							
ECDSA_STATE_REG	ECDSA status register	0x0020	RO				
Result Register							
ECDSA_RESULT_REG	ECDSA result register	0x0024	RO/SS			N/A	
SHA Registers							
ECDSA_SHA_MODE_REG	ECDSA SHA-control register (Hash algorithm)	0x0200	N/A	R/W	N/A		
ECDSA_SHA_START_REG	ECDSA SHA-control register (operation)	0x0210	N/A	WT	N/A		
ECDSA_SHA_CONTINUE_REG	ECDSA SHA-control register (operation)	0x0214	N/A	WT	N/A		
ECDSA_SHA_BUSY_REG	ECDSA SHA-control status register	0x0218	N/A	RO	N/A		
Version Register							
ECDSA_DATE_REG	Version control register	0x00FC	R/W	N/A			

22.9 Registers

The addresses in this section are relative to the ECDSA base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 22.1. ECDSA_CONF_REG (0x0004)

(reserved)																												ECDSA_DETERMINISTIC_K				ECDSA_SOFTWARE_SET_Z				(reserved)				ECDSA_ECC_CURVE				ECDSA_WORK_MODE			
31																												7	6	5	4	3	2	1	0												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset														

ECDSA_WORK_MODE Configures the working mode of ECDSA accelerator.

- 0: Signature Verification Mode
- 1: Signature Generation Mode
- 2: Public Key Export Mode
- 3: Invalid

(R/W)

ECDSA_ECC_CURVE Configures the elliptic curve used.

- 0: P-192
- 1: P-256
- 2: P-384
- 3: Invalid

(R/W)

ECDSA_SOFTWARE_SET_Z Configures how the parameter *z* is set.

- 0: Generated from SHA result
- 1: Written by software

(R/W)

ECDSA_DETERMINISTIC_K Configures how the parameter *k* is set.

- 0: Automatically generated by TRNG
- 1: Generated by the deterministic derivation algorithm

(R/W)

Register 22.2. ECDSA_START_REG (0x001C)

(reserved)																																ECDSA_GET_DONE ECDSA_LOAD_DONE ECDSA_START																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECDSA_START Configures whether to start the ECDSA operation. This bit will be self-cleared after configuration.

0: No effect

1: Start the ECDSA operation

(WT)

ECDSA_LOAD_DONE Write 1 to generate a signal indicating the ECDSA accelerator's LOAD operation is done. This bit will be self-cleared after configuration. (WT)

ECDSA_GET_DONE Write 1 to generate a signal indicating the ECDSA accelerator's GAIN operation is done. This bit will be self-cleared after configuration. (WT)

Register 22.3. ECDSA_CLK_REG (0x0008)

(reserved)																															ECDSA_CLK_GATE_FORCE_ON				
31																															1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

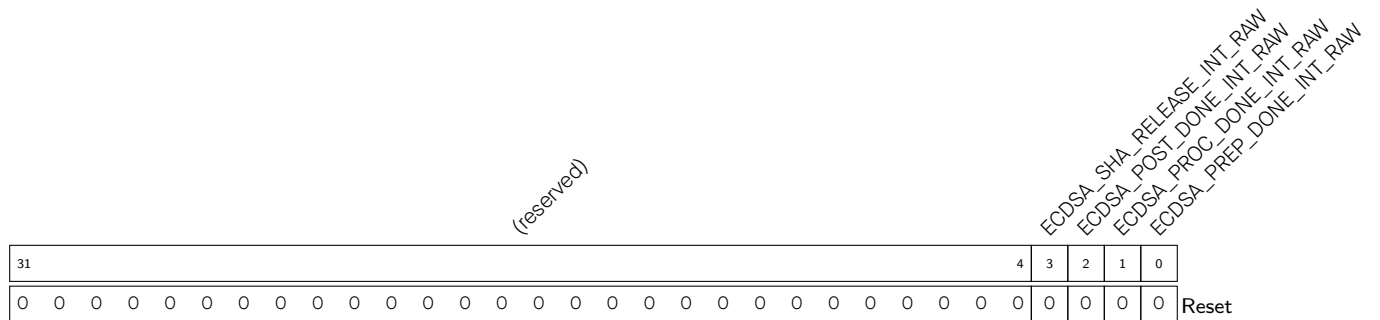
ECDSA_CLK_GATE_FORCE_ON Configures whether to force on ECDSA memory clock gate.

0: No effect

1: Force on

(R/W)

Register 22.4. ECDSA_INT_RAW_REG (0x000C)



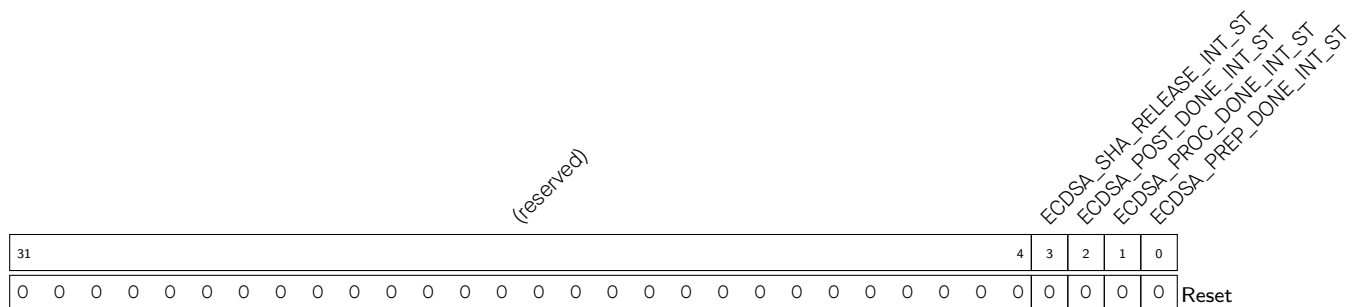
ECDSA_PREP_DONE_INT_RAW The raw interrupt status of the [ECDSA_PREP_DONE_INT](#) interrupt. (R/SS/WTC)

ECDSA_PROC_DONE_INT_RAW The raw interrupt status of the [ECDSA_PROC_DONE_INT](#) interrupt. (R/SS/WTC)

ECDSA_POST_DONE_INT_RAW The raw interrupt status of the [ECDSA_POST_DONE_INT](#) interrupt. (R/SS/WTC)

ECDSA_SHA_RELEASE_INT_RAW The raw interrupt status of the [ECDSA_SHA_RELEASE_INT](#) interrupt. (R/SS/WTC)

Register 22.5. ECDSA_INT_ST_REG (0x0010)



ECDSA_PREP_DONE_INT_ST The masked interrupt status of the [ECDSA_PREP_DONE_INT](#) interrupt. (RO)

ECDSA_PROC_DONE_INT_ST The masked interrupt status of the [ECDSA_PROC_DONE_INT](#) interrupt. (RO)

ECDSA_POST_DONE_INT_ST The masked interrupt status of the [ECDSA_POST_DONE_INT](#) interrupt. (RO)

ECDSA_SHA_RELEASE_INT_ST The masked interrupt status of the [ECDSA_SHA_RELEASE_INT](#) interrupt. (RO)

Register 22.6. ECDSA_INT_ENA_REG (0x0014)

(reserved)																											
31																					4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

ECDSA_PREP_DONE_INT_ENA Write 1 to enable the [ECDSA_PREP_DONE_INT](#) interrupt. (R/W)

ECDSA_PROC_DONE_INT_ENA Write 1 to enable the [ECDSA_PROC_DONE_INT](#) interrupt. (R/W)

ECDSA_POST_DONE_INT_ENA Write 1 to enable the [ECDSA_POST_DONE_INT](#) interrupt. (R/W)

ECDSA_SHA_RELEASE_INT_ENA Write 1 to enable the [ECDSA_SHA_RELEASE_INT](#) interrupt. (R/W)

Register 22.7. ECDSA_INT_CLR_REG (0x0018)

(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																												4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECDSA_PREP_DONE_INT_CLR Write 1 to clear the [ECDSA_PREP_DONE_INT](#) interrupt. (WT)

ECDSA_PROC_DONE_INT_CLR Write 1 to clear the [ECDSA_PROC_DONE_INT](#) interrupt. (WT)

ECDSA_POST_DONE_INT_CLR Write 1 to clear the [ECDSA_POST_DONE_INT](#) interrupt. (WT)

ECDSA_SHA_RELEASE_INT_CLR Write 1 to clear the [ECDSA_SHA_RELEASE_INT](#) interrupt. (WT)

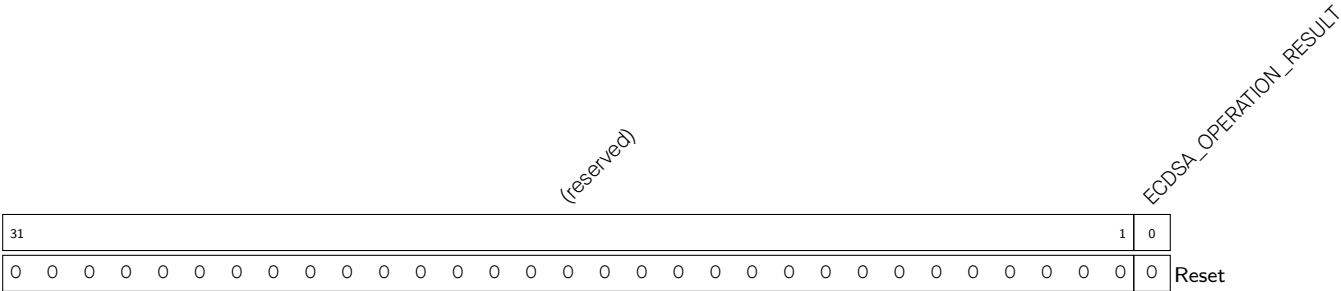
Register 22.8. ECDSA_STATE_REG (0x0020)



ECDSA_BUSY Represents the working status of the ECDSA accelerator.

- 0: IDLE
- 1: LOAD
- 2: GAIN
- 3: BUSY
- (RO)

Register 22.9. ECDSA_RESULT_REG (0x0024)

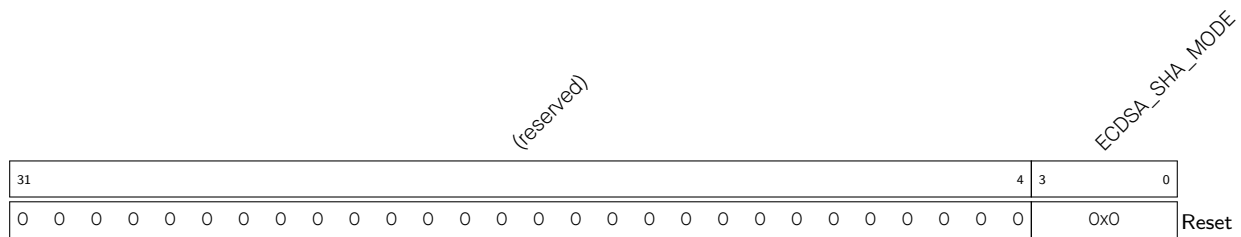


ECDSA_OPERATION_RESULT Indicates if the ECDSA operation is successful.

- 0: Not successful
- 1: Successful

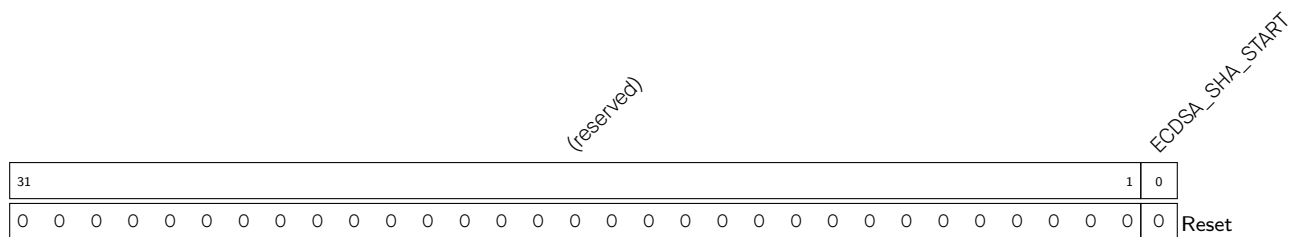
Only valid when the ECDSA operation is done.

(RO/SS)

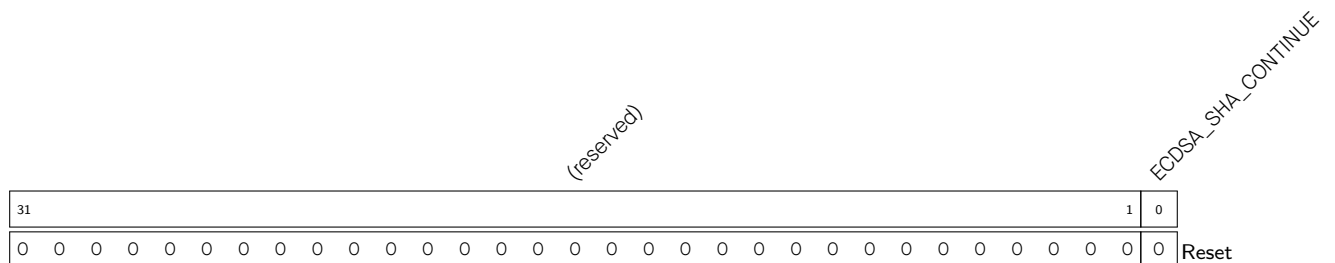
Register 22.10. ECDSA_SHA_MODE_REG (0x0200)

ECDSA_SHA_MODE Configures SHA algorithms for message hash.

- 1:SHA-224
- 2:SHA-256
- 3: SHA-384
- 4: SHA-512
- 5: SHA-512-224
- 6: SHA-512-256
- Others: invalid
- (R/W)

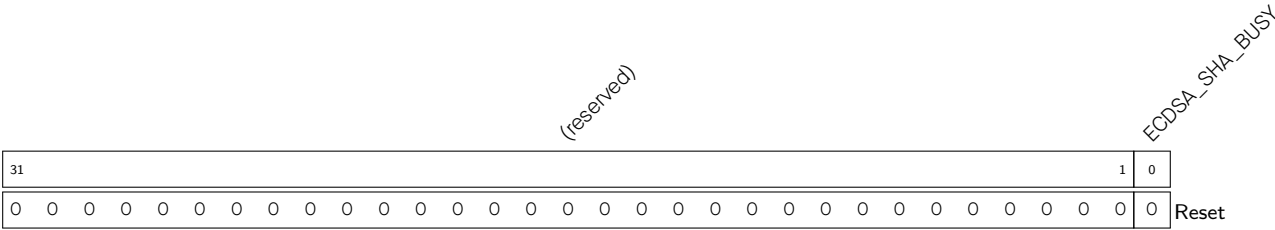
Register 22.11. ECDSA_SHA_START_REG (0x0210)

ECDSA_SHA_START Write 1 to start the first SHA operation in the ECDSA process. This bit will be self-cleared after configuration. (WT)

Register 22.12. ECDSA_SHA_CONTINUE_REG (0x0214)

ECDSA_SHA_CONTINUE Write 1 to start the latter SHA operation in the ECDSA process. This bit will be self-cleared after configuration. (WT)

Register 22.13. ECDSA_SHA_BUSY_REG (0x0218)

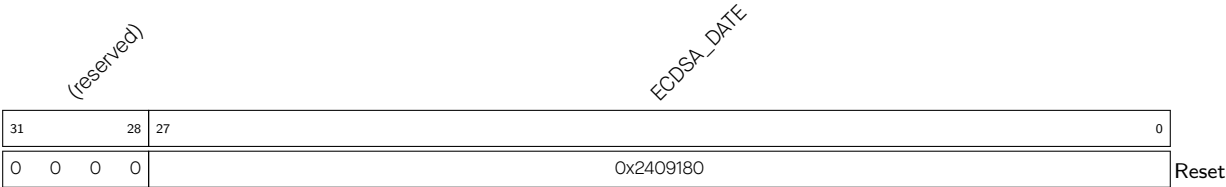


ECDSA_SHA_BUSY Represents the working status of the SHA accelerator in the ECDSA process.

- 0: IDLE
- 1: BUSY

(RO)

Register 22.14. ECDSA_DATE_REG (0x00FC)



ECDSA_DATE The ECDSA version control register. (R/W)

Chapter 23

External Memory Encryption and Decryption (XTS_AES)

23.1 Overview

The ESP32-C5 integrates an External Memory Encryption and Decryption module that complies with the XTS-AES standard algorithm specified in [IEEE Std 1619-2007](#), providing security for users' application code and data stored in the external memory (flash and RAM). Users can store proprietary firmware and sensitive data (e.g., credentials for gaining access to a private network) in the external flash, or store general data in the external RAM.

23.2 Features

- General XTS-AES algorithm, compliant with IEEE Std 1619-2007
- Software-based manual encryption
- High-speed auto encryption and decryption without software's participation
- Encryption and decryption functions jointly enabled/disabled by registers configuration, eFuse parameters, and boot mode
- Configurable Anti-DPA

23.3 Module Structure

The External Memory Encryption and Decryption module consists of three blocks, namely the Manual Encryption block, Auto Encryption block and Auto Decryption block. The module architecture is shown in Figure [23.3-1](#).

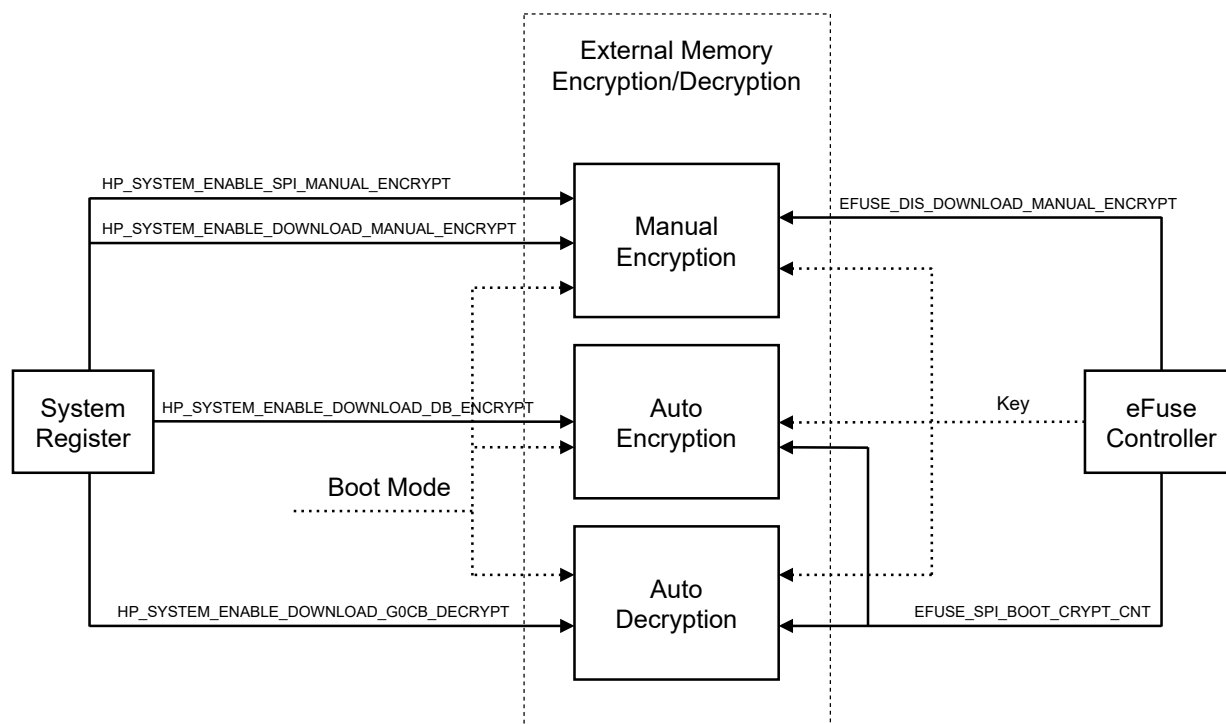


Figure 23.3-1. Architecture of the External Memory Encryption and Decryption

The Manual Encryption block can encrypt instructions and data, which will then be written to the external flash as ciphertext via SPI1.

When the CPU writes data to the external RAM through cache, the Auto Encryption block will automatically encrypt the data first, then the data will be written to the external RAM as ciphertext.

When the CPU reads from the external flash or external RAM through cache, the Auto Decryption block will automatically decrypt the ciphertext to retrieve instructions and data.

In the System Registers peripheral (see [5 System Registers \[to be added later\]](#)), the following three bits in register HP_SYSTEM_EXTERNAL_DEVICE_ENCRYPT_DECRYPT_CONTROL_REG are relevant to the external memory encryption and decryption:

- HP_SYSTEM_ENABLE_DOWNLOAD_MANUAL_ENCRYPT
- HP_SYSTEM_ENABLE_DOWNLOAD_GOCB_DECRYPT
- HP_SYSTEM_ENABLE_DOWNLOAD_DB_ENCRYPT
- HP_SYSTEM_ENABLE_SPI_MANUAL_ENCRYPT

The XTS_AES module also fetches two parameters from the peripheral eFuse Controller, which are: EFUSE_DIS_DOWNLOAD_MANUAL_ENCRYPT and EFUSE_SPI_BOOT_CRYPT_CNT. For detailed information, please see [Chapter 2 eFuse Controller \(eFuse\) \[to be added later\]](#).

23.4 Functional Description

23.4.1 XTS Algorithm

Both manual encryption and auto encryption/decryption use the XTS algorithm. During implementation, the XTS algorithm is characterized by a "data unit" of 1024 bits, defined in the Section *XTS-AES encryption procedure* of [XTS-AES Tweakable Block Cipher](#) Standard. For more information about the XTS-AES algorithm, please refer to [IEEE Std 1619-2007](#).

23.4.2 Key

The Manual Encryption block, Auto Encryption block, and Auto Decryption block share the same *Key* when implementing the XTS algorithm. The *Key* is provided by the eFuse hardware and cannot be accessed by software.

The *Key* is 256-bit long. The value of the *Key* is determined by the content in one eFuse block from BLOCK4 ~ BLOCK9. For easier description, we define:

- Block_A: the block whose key purpose is EFUSE_KEY_PURPOSE_XTS_AES_128_KEY (please refer to Table 2 *eFuse Controller (eFuse) [to be added later]* in Chapter 2 *eFuse Controller (eFuse) [to be added later]*). If Block_A exists, a 256-bit *Key_A* is stored in it.

There are two possibilities of how the *Key* is generated depending on whether Block_A exists or not, as shown in Table 23.4-1. In each case, the *Key* can be uniquely determined.

Table 23.4-1. *Key* Generated Based on *Key_A*

Block _A	<i>Key</i>	<i>Key</i> Length (bit)
Exists	<i>Key_A</i>	256
Does not exist	0 ²⁵⁶	256

Notes:

- "0²⁵⁶" indicates a bit string that consists of 256-bit zeros.
- Using 0²⁵⁶ as *Key* is not secure. It is recommended to configure a valid key.

For more information of key purposes, please refer to Table 2 *eFuse Controller (eFuse) [to be added later]* in Chapter 2 *eFuse Controller (eFuse) [to be added later]*.

23.4.3 Target Memory Space

The target memory space refers to a continuous address space in the external memory where the first encrypted ciphertext is stored. The target memory space can be uniquely determined by three relevant parameters: type, size, and base address, whose definitions are listed below.

- Type: the *type* of the target memory space, either external flash or external RAM. Value 0 indicates external flash, while 1 indicates external RAM.
- Size: the *size* of the target memory space, indicating the number of bytes encrypted in one encryption operation, which supports 16, 32, or 64 bytes.
- Base address: the *base_addr* of the target memory space. It is a 24-bit physical address, with range of 0x0000_0000 ~ 0x00FF_FFFF. It should be aligned to *size*, i.e., *base_addr*%*size* == 0.

For example, if there are 16 bytes of instruction data that need to be encrypted and written to address 0x130 ~ 0x13F in the external flash, then the target space is 0x130 ~ 0x13F, type is 0 (external flash), size is 16 (bytes), and the base address is 0x130.

The encryption of any length (must be multiples of 16 bytes) of plaintext instruction/data can be completed separately in multiple operations, and each operation has its individual target memory space and the relevant parameters.

For Auto Encryption/Decryption blocks, these parameters are automatically determined by hardware. For the Manual Encryption blocks, these parameters should be configured manually by users.

Note:

The “tweak” defined in Chapter 5.1 *Data units and tweaks* of [IEEE Std 1619-2007](#) is a 128-bit non-negative integer (*tweak*), which can be generated according to $tweak = type * 2^{30} + (base\_addr \& 0x3FFFFFF80)$. The lowest 7 bits and the highest 97 bits in *tweak* are always zero.

23.4.4 Data Writing

For Auto Encryption/Decryption blocks, data writing is automatically applied in hardware. For Manual Encryption blocks, data writing should be applied by users. The Manual Encryption block has a register block which consists of 16 registers, i.e., [XTS_AES_PLAIN_n_REG](#) (n : 0 ~ 15), that are dedicated to data writing and can store up to 512 bits of plaintext at a time.

Actually, the Manual Encryption block does not care where the plaintext comes from, but only where the ciphertext will be stored. Because of the strict correspondence between plaintext and ciphertext, in order to better describe how the plaintext is stored in the register block, we assume that the plaintext is stored in the target memory space in the first place and replaced by ciphertext after encryption. Therefore, the following description in this section no longer has the concept of “plaintext”, but uses “target memory space” instead.

How mapping between target memory space and registers works:

Assume a word in the target memory space is stored in *address*, define $offset = address \% 64$, $n = offset / 4$, then the word will be stored in register [XTS_AES_PLAIN_n_REG](#).

For example, when the *size* is 64, all registers in the register block will be used. The mapping between *offset* and registers now is shown in Table 23.4-2.

Table 23.4-2. Mapping Between Offsets and Registers

<i>offset</i>	Register	<i>offset</i>	Register
0x00	XTS_AES_PLAIN_0_REG	0x20	XTS_AES_PLAIN_8_REG
0x04	XTS_AES_PLAIN_1_REG	0x24	XTS_AES_PLAIN_9_REG
0x08	XTS_AES_PLAIN_2_REG	0x28	XTS_AES_PLAIN_10_REG
0x0C	XTS_AES_PLAIN_3_REG	0x2C	XTS_AES_PLAIN_11_REG
0x10	XTS_AES_PLAIN_4_REG	0x30	XTS_AES_PLAIN_12_REG
0x14	XTS_AES_PLAIN_5_REG	0x34	XTS_AES_PLAIN_13_REG
0x18	XTS_AES_PLAIN_6_REG	0x38	XTS_AES_PLAIN_14_REG
0x1C	XTS_AES_PLAIN_7_REG	0x3C	XTS_AES_PLAIN_15_REG

23.4.5 Manual Encryption Block

The Manual Encryption block is a peripheral module. It is equipped with registers and can be accessed by the CPU directly. Registers embedded in this block, the System Registers peripheral, eFuse parameters, and boot mode jointly configure and use this module.

The Manual Encryption block is operational only under certain conditions:

- In SPI Boot mode:

If bit `HP_SYSTEM_ENABLE_SPI_MANUAL_ENCRYPT` in register `HP_SYSTEM_EXTERNAL_DEVICE_ENCRYPT_DECRYPT_CONTROL_REG` is 1, the Manual Encryption block can be enabled. Otherwise, it is not operational.

- In Download Boot mode:

If bit `HP_SYSTEM_ENABLE_DOWNLOAD_MANUAL_ENCRYPT` in register `HP_SYSTEM_EXTERNAL_DEVICE_ENCRYPT_DECRYPT_CONTROL_REG` is 1 and the eFuse parameter `EFUSE_DIS_DOWNLOAD_MANUAL_ENCRYPT` is 0, the Manual Encryption block can be enabled. Otherwise, it is not operational.

Note:

Even though the CPU can skip cache and get the encrypted instruction/data directly by reading the external memory, users can by no means access *Key*.

23.4.6 Auto Encryption Block

The Auto Encryption block is not a conventional peripheral, so it does not have any registers and cannot be accessed by the CPU directly. The System Registers peripheral, eFuse parameters, and boot mode jointly configure and use this block.

The Auto Encryption block is operational only under certain conditions:

- In SPI Boot mode:

when `EFUSE_SPI_BOOT_CRYPT_CNT` (3 bits in total) is set to 1, 2, 4 or 7 (i.e., there is an odd number of 1s in its binary representation), then the Auto Encryption block can be enabled. Otherwise, it is not operational.

- In Joint Download Boot mode:

If bit `SYSTEM_ENABLE_DOWNLOAD_DB_ENCRYPT` in register `SYSTEM_EXTERNAL_DEVICE_ENCRYPT_DECRYPT_CONTROL_REG` is 1, the Auto Encryption block can be enabled. Otherwise, it is not operational.

Note:

- When the Auto Encryption block is enabled, it automatically encrypts data written by the CPU to external RAM. The encrypted data is stored in the external RAM, and this encryption process is fully hardware-based, requiring no software intervention. The operation is transparent to the cache, and the encryption key remains inaccessible to the user during the process.
- When the Auto Encryption block is disabled, it bypasses the CPU's access request to the cache and does not

process the data. As a result, data is written directly to external RAM in plaintext.

23.4.7 Auto Decryption Block

The Auto Decryption block is not a conventional peripheral, so it does not have any registers and cannot be accessed by the CPU directly. The System Registers peripheral, eFuse parameters, and boot mode jointly configure and use this block.

The Auto Decryption block is operational only under certain conditions:

- In SPI Boot mode

when EFUSE_SPI_BOOT_CRYPT_CNT (3 bits in total) is set to 1, 2, 4 or 7 (i.e., there is an odd number of 1s in its binary representation), then the Auto Decryption block can be enabled. Otherwise, it is not operational.

- In Download Boot mode

when bit HP_SYS_ENABLE_DOWNLOAD_GOCB_DECRYPT in register HP_SYS_EXTERNAL_DEVICE_ENCRYPT_DECRYPT_CONTROL_REG is 1, the Auto Decryption block can be enabled. Otherwise, it is not operational.

Note:

- When the Auto Decryption block is enabled, it automatically decrypts encrypted data or instructions read by the CPU from external memory via the cache. This decryption process is hardware-based and requires no software intervention, remaining transparent to the cache. The decryption key is not accessible to the software during this process.
- When the Auto Decryption block is disabled, it does not affect the contents stored in external memory, regardless of whether they are encrypted. Consequently, the CPU reads the original data stored in external memory via the cache.

23.5 Software Process

When the Manual Encryption block operates, software needs to be involved in the process. The steps are as follows:

1. Configure XTS_AES:

- Write 0 to register [XTS_AES_DESTINATION_REG](#).
- Set register [XTS_AES_PHYSICAL_ADDRESS_REG](#) to *base_addr*.
- Set register [XTS_AES_LINESIZE_REG](#) to $\frac{size}{32}$.

For definitions of *base_addr* and *size*, please refer to Section [23.4.3](#).

2. Write plaintext instructions/data to the registers block XTS_AES_PLAIN_*n*_REG (*n*: 0-15). For detailed information, please refer to Section [23.4.4](#).

Please write data to registers according to your actual needs, and the unused ones could be set to arbitrary values.

3. Wait for Manual Encryption block to be idle. Poll register [XTS_AES_STATE_REG](#) until it reads 0 which indicates the Manual Encryption block is idle.
4. Trigger manual encryption by writing 1 to register [XTS_AES_TRIGGER_REG](#).
5. Wait for the encryption process completion. Poll register [XTS_AES_STATE_REG](#) until it reads 2.
Step 1 to 5 are the steps of encrypting plaintext instructions/data with the Manual Encryption block using the Key.
6. Write 1 to register [XTS_AES_RELEASE_REG](#) to grant SPI1 the access to the encrypted ciphertext. After this, the value of register [XTS_AES_STATE_REG](#) will become 3.
7. Call SPI1 to write the ciphertext in the external flash (see Section [API Reference - Flash Encrypt in ESP-IDF Programming Guide](#)).
8. Write 1 to register [XTS_AES_DESTROY_REG](#) to destroy the ciphertext. After this, the value of register [XTS_AES_STATE_REG](#) will become 0.

Repeat the above steps according to the amount of plaintext instructions/data that need to be encrypted.

23.6 Anti-DPA

DPA (Differential Power Analysis) is a side-channel attack method in cryptography, through which an attacker can statistically analyze data collected from multiple encryption operations to calculate intermediate values in the encryption computation. ESP32-C5 XTS_AES supports two Anti-DPA methods to defend against external DPA attacks.

The XTS-AES algorithm can be divided into two steps, according to [IEEE Std 1619-2007](#):

- Step 1: Calculating Tweak value.
- Step 2: Calculating Cipher/Plain text. In this section, we define this step as calculating Data units.

ESP32-C5 allows users to enable anti-DPA function separately during the above-mentioned two steps to enhance security. See details in the following sections.

23.6.1 Clock Anti-DPA Function

Clock Anti-DPA function is the first method to defend against external DPA attacks. Different security levels can be configured through registers:

- First we define the below parameters for a better description:
 - `select_reg` = [XTS_AES_CRYPT_DPA_SELECT_REGISTER](#)
 - `reg_d_dpa_en` = [XTS_AES_CRYPT_CALC_D_DPA_EN](#)
 - `efuse_dpa_en` = [EFUSE_CRYPT_DPA_ENABLE](#)
 - `reg_anti_dpa_level` = [XTS_AES_CRYPT_SECURITY_LEVEL](#)
 - `efuse_anti_dpa_level` = 3
- Configure the security level of Anti-DPA for the XTS_AES module:

$$Anti\_DPA\_level = select\_reg ? (reg\_anti\_dpa\_level) : (efuse\_dpa\_en * efuse\_anti\_dpa\_level)$$

When *Anti_DPA_level* equals 0, Anti-DPA is disabled. The higher the value of *Anti_DPA_level* is, the stronger the Anti-DPA ability is.

- Configure whether to enable Anti-DPA when the XTS-AES algorithm is calculating Data units:

$$Anti\_DPA\_enabled\_in\_calc\_D = select\_reg ? reg\_d\_dpa\_en : efuse\_dpa\_en$$

If *Anti_DPA_level* is not 0, when *Anti_DPA_enabled_in_calc_D* equals to 1, Anti-DPA is enabled when XTS-AES algorithm is calculating Data units.

If *Anti_DPA_level* is not 0, Anti-DPA is always enabled when the XTS-AES algorithm is calculating Tweak value.

Note:

- Even if *efuse_dpa_en* is set to 1, you can still disable anti-DPA by configuring *select_reg* = 1 and *reg_anti_dpa_level* = 0.
- Configuring whether or not to enable Anti-DPA will have an impact on the external storage access bandwidth:
 - When Anti-DPA is enabled during the calculation of Data units, the read and write bandwidth will be reduced by more than 50% when the Anti-DPA level >= 4.
 - When Anti-DPA is disabled during the calculation of Data units, the read and write bandwidth will be reduced by more than 50% when the Anti-DPA level >= 6.

23.6.2 Pseudo-round Anti-DPA Function

The pseudo-round function of ESP32-C5's XTS-AES is the second method to achieve higher anti-attack performance. When the pseudo-round function is enabled, XTS-AES will randomly add pseudo-rounds before and after the original operation rounds. In a pseudo-round, XTS-AES will use the pseudo-key, which is automatically generated by the hardware, to perform a round of fake operations. The operations in the pseudo-round do not affect the original XTS-AES operation results.

The configuration related to pseudo-round is as follows:

Mode of pseudo-round function

Users can choose to enable pseudo-round function in different calculation steps of XTS-AES:

- 0: not enable pseudo-round function during calculation;
- 1: enable pseudo-round function when calculating Tweak value;
- 2: enable pseudo-round function when calculating Tweak value and some rounds of calculating Data units;
- 3: enable pseudo-round function when calculating Tweak value and all rounds of calculating Data units.

Total number of pseudo-rounds

The total number of pseudo-rounds that will be randomly inserted into each XTS-AES operation round is controlled by:

- *pseudo_base*: configures the basic round number of pseudo-round function.
- *pseudo_inc*: configures the random incremental round number of pseudo-round function.

The total number of pseudo-rounds will be randomly in the range

$$[pseudo\_base, pseudo\_base + (2^{pseudo\_inc} - 1)]$$

Random number update frequency

pseudo_rnd_cnt configures the frequency of random key updates in the pseudo-round function. The higher the configured value, the higher the update frequency. This value is usually recommended to be set to maximum (7).

Configuration

The parameters mentioned above can be configured through eFuse bits or XTS_AES related registers, depending on the value of EFUSE_XTS_DPA_PSEUDO_LEVEL:

- 0: The user can configure using XTS_AES related registers.
- 1, 2, or 3: User configuration is not allowed.

See Table [23.6-1](#) for details.

Table 23.6-1. Configuration of XTS_AES Pseudo-round Anti-DPA

EFUSE_XTS_DPA_PSEUDO_LEVEL	pseudo_mode	pseudo_base	pseudo_inc	pseudo_rnd_cnt
0	XTS_AES_PSEUDO_MODE	XTS_AES_PSEUDO_BASE	XTS_AES_PSEUDO_INC	XTS_AES_PSEUDO_RNG_CNT
1	1	4	2	7
2	2	4	2	7
3	3	4	2	7

23.7 Register Summary

The addresses in this section are relative to the External Memory Encryption and Decryption base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

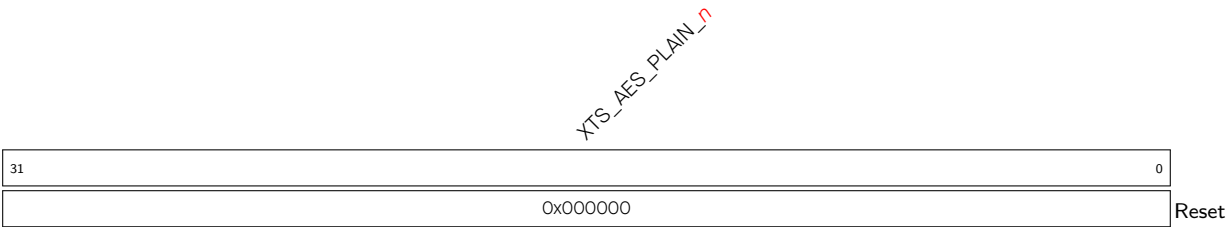
The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Plaintext Register Heap			
XTS_AES_PLAIN_0_REG	Plaintext register 0	0x0300	R/W
XTS_AES_PLAIN_1_REG	Plaintext register 1	0x0304	R/W
XTS_AES_PLAIN_2_REG	Plaintext register 2	0x0308	R/W
XTS_AES_PLAIN_3_REG	Plaintext register 3	0x030C	R/W
XTS_AES_PLAIN_4_REG	Plaintext register 4	0x0310	R/W
XTS_AES_PLAIN_5_REG	Plaintext register 5	0x0314	R/W
XTS_AES_PLAIN_6_REG	Plaintext register 6	0x0318	R/W
XTS_AES_PLAIN_7_REG	Plaintext register 7	0x031C	R/W
XTS_AES_PLAIN_8_REG	Plaintext register 8	0x0320	R/W
XTS_AES_PLAIN_9_REG	Plaintext register 9	0x0324	R/W
XTS_AES_PLAIN_10_REG	Plaintext register 10	0x0328	R/W
XTS_AES_PLAIN_11_REG	Plaintext register 11	0x032C	R/W
XTS_AES_PLAIN_12_REG	Plaintext register 12	0x0330	R/W
XTS_AES_PLAIN_13_REG	Plaintext register 13	0x0334	R/W
XTS_AES_PLAIN_14_REG	Plaintext register 14	0x0338	R/W
XTS_AES_PLAIN_15_REG	Plaintext register 15	0x033C	R/W
Configuration Registers			
XTS_AES_LINESIZE_REG	Configures the size of target memory space	0x0340	R/W
XTS_AES_DESTINATION_REG	Configures the type of the external memory	0x0344	R/W
XTS_AES_PHYSICAL_ADDRESS_REG	Stores the physical address of the external memory	0x0348	R/W
XTS_AES_DPA_CTRL_REG	Configures the Anti-DPA function	0x0388	R/W
XTS_AES_PSEUDO_ROUND_CONF_REG	XTS-AES pseudo-round function configuration register	0x038C	R/W
Control/status Registers			
XTS_AES_TRIGGER_REG	Activates AES algorithm	0x034C	WT
XTS_AES_RELEASE_REG	Releases control	0x0350	WT
XTS_AES_DESTROY_REG	Destroys control	0x0354	WT
XTS_AES_STATE_REG	Status register	0x0358	RO
Version control register			
XTS_AES_DATE_REG	Version control register	0x035C	R/W

23.8 Registers

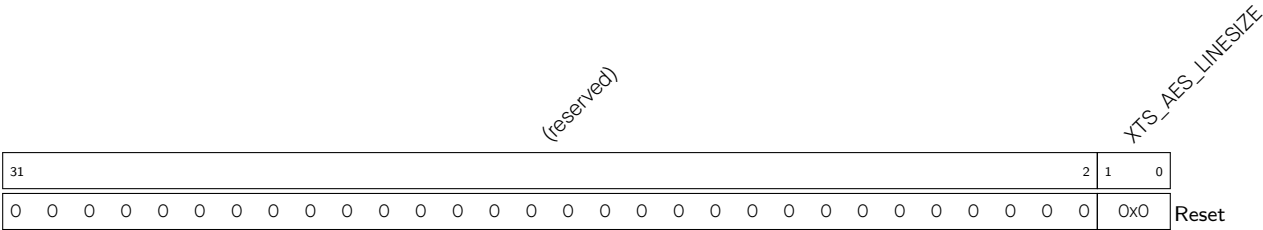
The addresses in this section are relative to the External Memory Encryption and Decryption base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 23.1. XTS_AES_PLAIN_ *n* _REG (*n*: 0-15) (0x0300+4**n*)



XTS_AES_PLAIN_ *n* Configures the *n*th 32-bit piece of plain text. (R/W)

Register 23.2. XTS_AES_LINESIZE_REG (0x0340)



XTS_AES_LINESIZE Configures the data size of one encryption operation.

- 0: 16 bytes
- 1: 32 bytes
- 2: 64 bytes
- 3: Invalid

(R/W)

Register 23.3. XTS_AES_DESTINATION_REG (0x0344)

[illegible]

XTS_AES_DESTINATION Configures the type of external memory for Manual Encryption. Currently, it must be set to 0, as the Manual Encryption block only supports flash encryption. Set this bit to 1 may cause an error.

0: flash

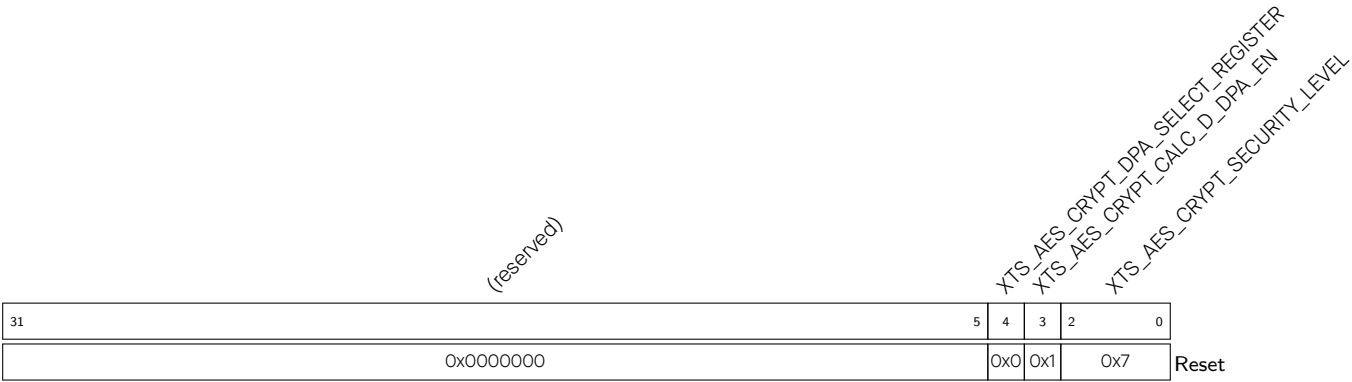
1: external RAM

(R/W)

Register 23.4. XTS_AES_PHYSICAL_ADDRESS_REG (0x0348)

XTS_AES_PHYSICAL_ADDRESS Configures the physical address which will be used in Manual Encryption. This value should be aligned with the byte number configured via the [XTS_AES_LINESIZE](#) parameter. (R/W)

Register 23.5. XTS_AES_DPA_CTRL_REG (0x0388)



XTS_AES_CRYPT_DPA_SELECT_REGISTER Configures whether the clock Anti-DPA function is controlled by eFuse or register.

0: The clock Anti-DPA function is configured by register.

1: The clock Anti-DPA function is configured by eFuse.

(R/W)

XTS_AES_CRYPT_CALC_D_DPA_EN Configures whether to enable the clock Anti-DPA in the XTS_AES algorithm.

0: Enable the clock Anti-DPA only when calculating Tweak value

1: Enable the clock Anti-DPA both when calculating Tweak value and Date units

Note that this field is only effective when [XTS_AES_CRYPT_SECURITY_LEVEL](#) is not 0.

(R/W)

XTS_AES_CRYPT_SECURITY_LEVEL Configures the security level of external memory encryption and decryption.

0: Disable the clock Anti-DPA function

1-7: The bigger the number is, the more secure the encryption and decryption are

(R/W)

Register 23.6. XTS_AES_PSEUDO_ROUND_CONF_REG (0x038C)

(reserved)																				XTS_AES_PSEUDO_INC		XTS_AES_PSEUDO_BASE		XTS_AES_PSEUDO_RNG_CNT		XTS_AES_PSEUDO_MODE			
31																			11	10	9	8	5		4	2		1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2		7		0		Reset	

XTS_AES_MODE_PSEUDO Set the calculation steps of XTS-AES to enable pseudo-round Anti-DPA function.

0: not enable the pseudo-round Anti-DPA function during calculation

1: enable the pseudo-round Anti-DPA function when calculating Tweak value

2: enable the pseudo-round Anti-DPA function when calculating Tweak value and some rounds of calculating Data units

3: enable the pseudo-round Anti-DPA function when calculating Tweak value and all rounds of calculating Data units

(R/W)

XTS_AES_PSEUDO_RNG_CNT Configures the update frequency of the pseudo-key in the pseudo-round Anti-DPA function. (R/W)

XTS_AES_PSEUDO_BASE Configures the basic round number of pseudo-round Anti-DPA function. (R/W)

XTS_AES_PSEUDO_INC Configures the incremental round number of pseudo-round Anti-DPA function. (R/W)

Register 23.7. XTS_AES_TRIGGER_REG (0x034C)

(reserved)																															XTS_AES_TRIGGER				
31																																	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

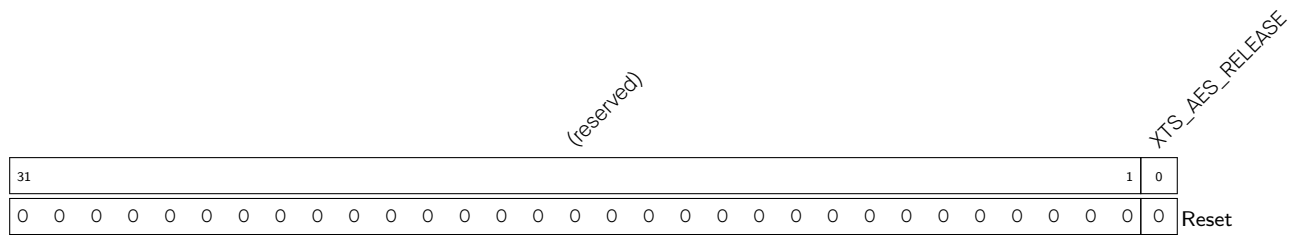
XTS_AES_TRIGGER Set this bit to trigger the process of manual encryption calculation.

0: Disable manual encryption

1: Enable manual encryption

This action should only be asserted when manual encryption status is 0. After this action, manual encryption status becomes 1. After calculation is done, manual encryption status becomes 2.

(WT)

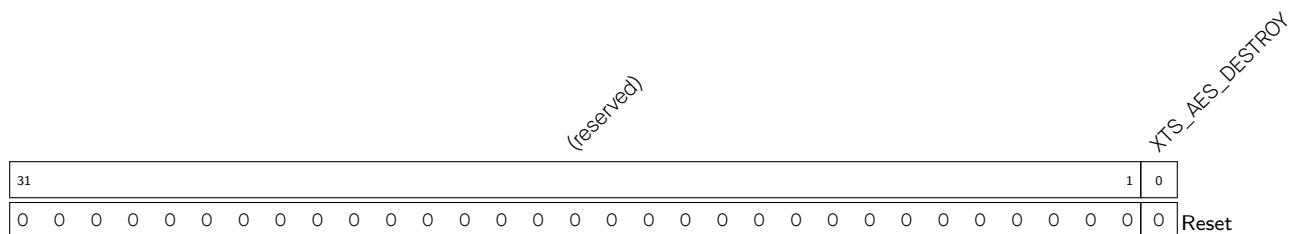
Register 23.8. XTS_AES_RELEASE_REG (0x0350)

XTS_AES_RELEASE Set this bit to release encrypted result to MSPI.

0: do not release

1: release

This action should only be asserted when manual encryption status is 2. After this action, manual encryption status will become 3. (WT)

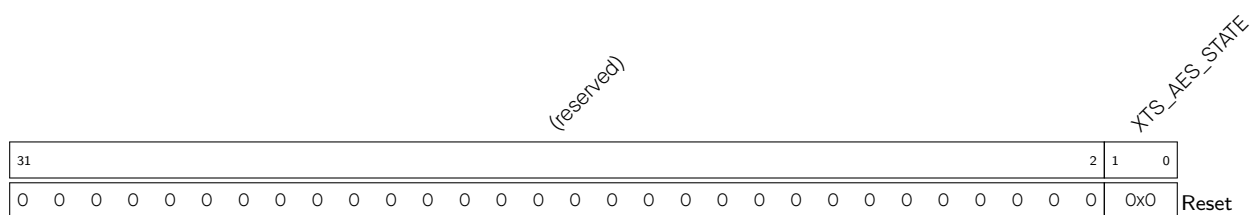
Register 23.9. XTS_AES_DESTROY_REG (0x0354)

XTS_AES_DESTROY Set this bit to destroy encrypted result.

0: No effect

1: Destroy encrypted result

This action should be asserted only when manual encryption status is 3. After this action, manual encryption status will become 0. (WT)

Register 23.10. XTS_AES_STATE_REG (0x0358)

XTS_AES_STATE Represents the status of the Manual Encryption block. 0 (XTS_AES_IDLE): Idle

1 (XTS_AES_BUSY): Busy with encryption

2 (XTS_AES_DONE): Encryption completed, but the encrypted result is not accessible to MSPI

3 (XTS_AES_RELEASE): Encrypted result is accessible to MSPI

(RO)

Register 23.11. XTS_AES_DATE_REG (0x035C)

(reserved)		XTS_AES_DATE	
31	30	29	0
0	0	0x20210907	
			Reset

XTS_AES_DATE Version control register. (R/W)

Chapter 24

Random Number Generator (RNG)

24.1 Introduction

The ESP32-C5 contains a true random number generator, which generates 32-bit random numbers that can be used for cryptographic operations, among other things.

24.2 Features

The random number generator in ESP32-C5 generates true random numbers, which means random numbers derived from a physical process, rather than by means of an algorithm. All generated random numbers have an equal probability of appearing within a specific range.

24.3 Functional Description

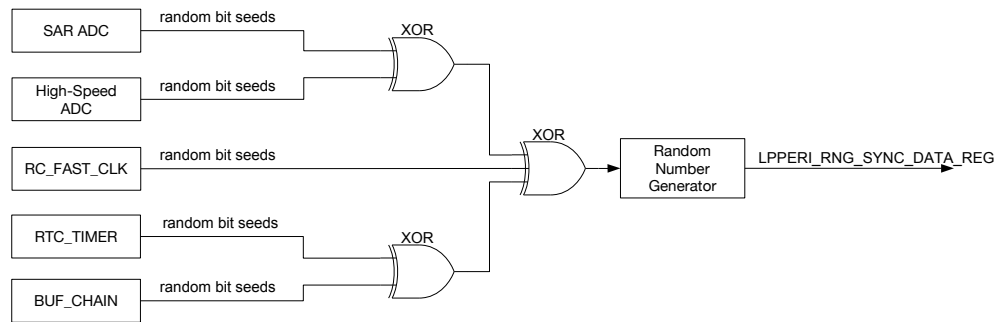
Every 32-bit value read from the [LPPERI_RNG_DATA_SYNC_REG](#) register of the random number generator is a true random number. These true random numbers are generated based on the **thermal noise** in the system, the **asynchronous clock mismatch**, and the **asynchronous counter**.

- **Thermal noise** comes from the high-speed ADC, or SAR ADC, or both. Whenever the high-speed ADC or SAR ADC is enabled, bit streams will be generated and fed into the random number generator through an XOR logic gate as random seeds.
- RC_FAST_CLK is an **asynchronous clock source** that can generate metastability in the circuit. This metastability can be used as a random number entropy to feed into the random number generator.
- **Asynchronous counters** count using an asynchronous clock source. The asynchronous clock source generates circuit metastability, which can be used as a random number seed to feed into the random number generator. There are two types of asynchronous counters:
 - On-chip [RTC Timer \(RTC_TIMER\)](#): The counter directly counts using the on-chip RTC timer. The count value is fed into the random number generator as a random number seed after applying XOR logic.
 - **Ring buffer (BUF_CHAIN)** as the noise source.

These two types of asynchronous counters can work independently or in combination.

Note:

For limitations of particular noise sources, please check [notes in Section Programming Procedures](#).

**Figure 24.3-1. Noise Source**

When there is noise coming from the SAR ADC, the random number generator is fed with a 2-bit entropy in one clock cycle of RC_FAST_CLK, which is generated from an internal RC oscillator (see Chapter 6 [Reset and Clock](#) for details). Thus, it is advisable to read the [LPPERI_RNG_DATA_SYNC_REG](#) register at a maximum rate of 1 MHz to obtain the maximum entropy.

When there is noise coming from the high-speed ADC, the random number generator is fed with a 2-bit entropy in one APB clock cycle, which is normally 80 MHz. Thus, it is advisable to read the [LPPERI_RNG_DATA_SYNC_REG](#) register at a maximum rate of 5 MHz to obtain the maximum entropy.

A data sample of 2 GB, which is read from the random number generator at a rate of 5 MHz with only the high-speed ADC being enabled, has been tested using the Dieharder Random Number Testsuite (version 3.31.1). The sample passed all tests.

24.4 Programming Procedure

To use the random number generator, set [LPPERI_RNG_CK_EN](#) to enable its clock. The clock is automatically enabled when [Elliptic Curve Digital Signature Algorithm \(ECDSA\)](#) is used, regardless of [LPPERI_RNG_CK_EN](#).

When using the random number generator, ensure the entropy source is enabled. Otherwise, pseudo-random numbers will be returned.

- SAR ADC can be enabled by using the DIG ADC controller. For details, please refer to Chapter 36 [ADC Controller](#).
- High-speed ADC is enabled automatically when the Wi-Fi or Bluetooth module is enabled.
- RC_FAST_CLK is always enabled
- Asynchronous counters
 - RTC timer (RTC_TIMER) is enabled by setting the [LPPERI_RTC_TIMER_EN](#) field.
 - Ring buffer (BUF_CHAIN) is enabled by setting the [LPPERI_RNG_SAMPLE_ENABLE](#) bit.

Note:

1. When the Wi-Fi module is enabled, the value read from the high-speed ADC can be saturated in some extreme

cases, which lowers the entropy. Thus, it is advisable to also enable the SAR ADC as the noise source for the random number generator for such cases.

2. Enabling RC_FAST_CLK and asynchronous counters increases the RNG entropy. However, to ensure maximum entropy, it's recommended to always enable the SAR ADC or high-speed ADC as well.

Read the [LPPERI_RNG_DATA_SYNC_REG](#) register multiple times until sufficient random numbers have been generated. Ensure the rate at which the register is read does not exceed the frequencies described in Section [24.3](#) above.

24.5 Register Summary

The addresses in this section are relative to Random Number Generator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
LPPERI_RNG_CFG_REG	RNG configuration register	0x0024	varies
LPPERI_RNG_DATA_SYNC_REG	RNG result synchronization register	0x0028	RO

24.6 Registers

The addresses in this section are relative to Random Number Generator base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 24.1. LPPERI_RNG_CFG_REG (0x0024)

(reserved)																(reserved)																LPPERI_RTC_TIMER_EN								(reserved)								LPPERI_RNG_SAMPLE_ENABLE																							
31								24								23								12								11		10		9								1		0																									
0x0								0								0								0								0								0								3		0								0								0		0		Reset	

- LPPERI_RNG_SAMPLE_ENABLE** Configures whether to enable BUF_CHAIN.
0: Disable
1: Enable
(R/W)
- LPPERI_RTC_TIMER_EN** Bit[0]: Configures whether to enable the RTC timer before CRC.
Bit[1]: Configures whether to enable the RTC timer after CRC.
0: Disable
1: Enable
(R/W)

Register 24.2. LPPERI_RNG_DATA_SYNC_REG (0x0028)

LPPERI_RND_SYNC_DATA																															
31																															0
0																															
Reset																															

- LPPERI_RND_SYNC_DATA** Represents the RNG synchronization result. (RO)

Part V

Connectivity Interface

This part addresses the connectivity aspects of the system, describing components related to various communication interfaces like I2C, I2S, SPI, UART, USB, and more. The part also covers interfaces to generate signals used in remote control, motor control, LED control, etc.

Chapter 25

UART Controller (UART)

25.1 Overview

A UART is a character-oriented data link for asynchronous communication between devices. Such communication does not add clock signals to the data sent. Therefore, in order to communicate successfully, the transmitter and the receiver must operate at the same baud rate with the same stop bit(s) and parity bit(s).

A UART data frame usually begins with one start bit, followed by data bits, one parity bit (optional), and one or more stop bits. UART controllers on ESP32-C5 support various lengths of data bits and stop bits. These controllers also support software and hardware flow control as well as GDMA for high-speed data transfer. This allows developers to use multiple UART ports at minimal software cost.

ESP32-C5 has three UART controllers, including two regular UARTs and one low-power (LP) UART. These UARTs are compatible with various UART devices, and support Infrared Data Association (IrDA) and RS485 communication. In this chapter, the two regular UART controllers are referred to as UART n , in which n denotes 0, 1. LP UART is the cut-down version of the regular UART, with a separate group of registers. For differences between UART and LP UART, please refer to Table 25.2-1.

25.2 Features

Table 25.2-1 lists the feature comparison between UART and LP UART:

Table 25.2-1. UART and LP UART Feature Comparison

UART Feature	LP UART Feature
Programmable baud rate up to 5 MBaud	
128 x 8 bit RAM respectively for the TX channel and RX channel of a UART controller	20 x 8-bit RAM, shared by the TX FIFO and RX FIFO of LP UART
Full-duplex asynchronous communication	
Data bits (5 to 8 bits)	
Stop bits (1, 1.5, or 2 bits)	
Parity bit	
Special character AT_CMD detection	
RS485 protocol	—
IrDA protocol	—

Cont'd on next page

Table 25.2-1 – cont'd from previous page

UART Feature	LP_UART Feature
High-speed data communication using GDMA	—
Receive timeout	
UART as wakeup source	
Software and hardware flow control	
Three prescalable clock sources: 1. XTAL_CLK 2. RC_FAST_CLK 3. PLL_F80M_CLK	Three prescalable clock sources 1. RC_FAST_CLK 2. XTAL_DIV_CLK 3. PLL_F8M_CLK

The following description mainly covers regular UART controllers.

25.3 UART Structure

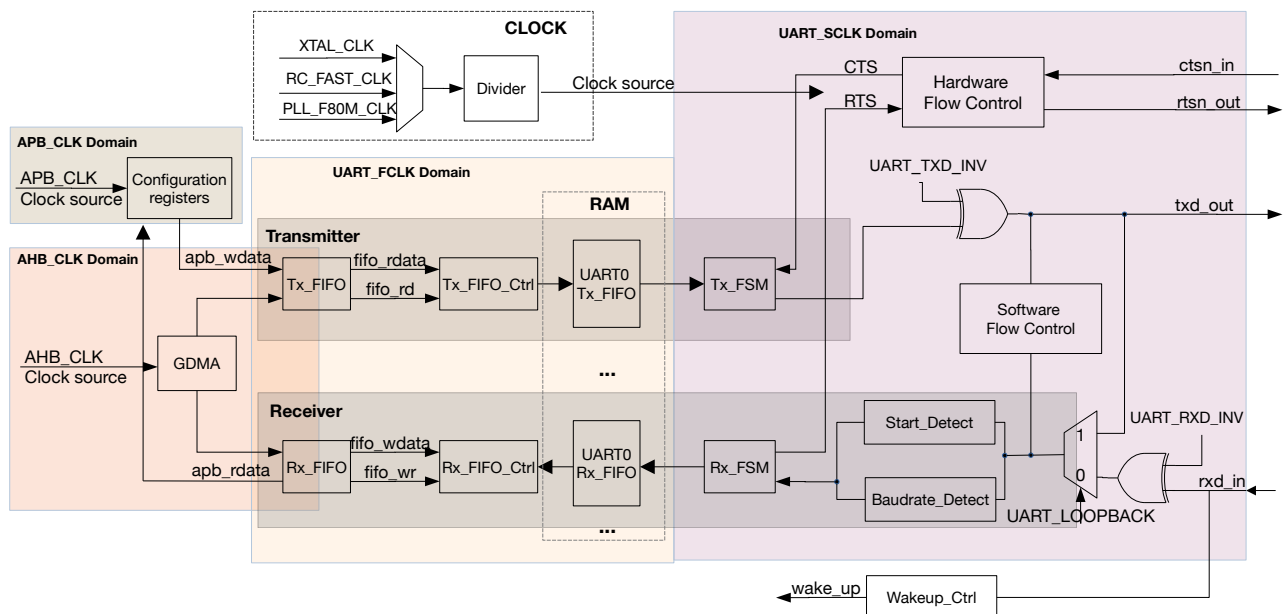


Figure 25.3-1. UART Structure

Figure 25.3-1 shows the basic structure of a UART controller. A UART controller works in four clock domains, namely APB_CLK, AHB_CLK, UART_SCLK, and UART_FCLK. APB_CLK and AHB_CLK are synchronized but with different frequencies (APB_CLK is derived from AHB_CLK by division), and likewise UART_SCLK and UART_FCLK are synchronized but with different frequencies (UART_SCLK is derived from UART_FCLK by division). UART_FCLK has three clock sources: PLL_F80M_CLK, RC_FAST_CLK, and crystal clock XTAL_CLK (for details, please refer to Chapter 6 *Reset and Clock*), which are selected by configuring `PCR_UARTn_SCLK_SEL`. The selected clock source is divided by a divider to generate UART_SCLK clock signals. The divisor is configured by `PCR_UARTn_SCLK_DIV_NUM` for the integral part, `PCR_UARTn_SCLK_DIV_A` for the denominator of the fractional part, and `PCR_UARTn_SCLK_DIV_B` for the numerator of the fractional part. The divisor ranges from 1 ~ 256. Only regular UART has such a divider; LP UART does not.

A UART controller can be broken down into two parts based on functions: a transmitter and a receiver.

The transmitter contains a TX FIFO (i.e., Tx_FIFO in Figure 25.3-1), which buffers data to be sent. Software can write data to Tx_FIFO via the APB bus, or move data to Tx_FIFO using GDMA. Tx_FIFO_Ctrl controls writing and reading Tx_FIFO. When Tx_FIFO is not empty, Tx_FSM reads data bits in the data frame via Tx_FIFO_Ctrl, and converts them into a bitstream. The levels of output bitstream signal txd_out can be inverted by configuring the [UART_TXD_INV](#) field.

The receiver contains an RX FIFO (i.e., Rx_FIFO in Figure 25.3-1), which buffers data to be processed. The input bitstream signal rxd_in is transferred to the UART controller, and its level can be inverted by configuring [UART_RXD_INV](#) field. Baudrate_Detect measures the baud rate of input bitstream signal rxd_in by detecting its minimum pulse width. Start_Detect detects the start bit in a data frame. If the start bit is detected, Rx_FSM stores data bits in the data frame into Rx_FIFO by Rx_FIFO_Ctrl. Software can read data from Rx_FIFO via the APB bus, or receive data using GDMA.

HW_Flow_Ctrl controls rxd_in and txd_out data flows by standard UART RTS and CTS flow control signals (rtsn_out and ctsn_in). SW_Flow_Ctrl controls data flows by adding special characters to outgoing data and detecting special characters in incoming data. When a UART controller is in Light-sleep mode (see Chapter 4 [Low-power Management \[to be added later\]](#) for more details), a wake_up signal can be generated in four ways and sent to PMU, which then wakes up the ESP32-C5 chip. For more information about wakeup, please refer to Section 25.4.8.

25.4 Functional Description

25.4.1 Clock and Reset

UART controllers are asynchronous. Their register configuration module works in the APB_CLK domain. TX FIFO and RX FIFO work across the AHB_CLK and UART_FCLK domains. The UART RAM control unit works in the UART_FCLK domain. The UART transmission and reception control module works in the UART_SCLK domain, i.e., UART Core's clock domain.

When the frequency of the UART_SCLK is higher than the frequency needed to generate the baud rate, the UART Core can be clocked at a lower frequency by the divider, in order to reduce power consumption. Usually, the UART Core's clock frequency is lower than the APB_CLK's frequency, and can be divided by the largest divisor when higher than the frequency needed to generate the baud rate. The frequency of the UART Core's clock can also be at most twice higher than the APB_CLK. The clock for the UART transmitter and the UART receiver can be controlled independently. To enable the clock for the UART transmitter, [UART_TX_SCLK_EN](#) shall be set; to enable the clock for the UART receiver, [UART_RX_SCLK_EN](#) shall be set.

To ensure that the configured register values are synchronized from APB_CLK domain to the UART_SCLK domain, please follow the procedures in Section 25.5.

To reset the whole UART, please:

- Enable the UART Core's clock by setting [PCR_UART_n_CLK_EN](#) to 1.
- Write 1 to [PCR_UART_n_RST_EN](#).
- Clear [PCR_UART_n_RST_EN](#) to 0.

25.4.2 UART FIFO

The transmitter and the receiver on the UART controller each use a 128 x 8-bit RAM, and access their respective RAM through a separate 4 x 8-bit asynchronous FIFO interface. The RAM and asynchronous FIFO interface for the transmitter and the receiver are independent and cannot be shared.

UART Tx_FIFO is reset by setting [UART_TXFIFO_RST](#). UART Rx_FIFO is reset by setting [UART_RXFIFO_RST](#).

Data to be sent is written to TX FIFO via the APB bus or using GDMA, read automatically, and converted from a frame into a bitstream by hardware Tx_FSM. Data received is converted from a bitstream into a frame by hardware Rx_FSM, written into RX FIFO, and then stored into RAM via the APB bus or using GDMA. The two UART controllers share one GDMA channel.

The empty signal threshold for Tx_FIFO is configured by setting [UART_TXFIFO_EMPTY_THRHD](#). When data stored in Tx_FIFO is less than [UART_TXFIFO_EMPTY_THRHD](#), a UART_TXFIFO_EMPTY_INT interrupt is generated. The full signal threshold for Rx_FIFO is configured by setting [UART_RXFIFO_FULL_THRHD](#). When data stored in Rx_FIFO is greater than or equal to [UART_RXFIFO_FULL_THRHD](#), a UART_RXFIFO_FULL_INT interrupt is generated. In addition, when Rx_FIFO receives more data than its capacity, a UART_RXFIFO_OVF_INT interrupt is generated.

UART_n can access FIFO via register [UART_FIFO_REG](#). You can put data into TX FIFO by writing [UART_RXFIFO_RD_BYTE](#), and get data in RX FIFO by reading [UART_RXFIFO_RD_BYTE](#).

25.4.3 Baud Rate Generation and Detection

25.4.3.1 Baud Rate Generation

Before a UART controller sends or receives data, the baud rate should be configured by setting corresponding registers. The baud rate generator of a UART controller functions by dividing the input clock source. It can divide the clock source by a fractional amount. The divisor is configured by [UART_CLKDIV_SYNC_REG](#): [UART_CLKDIV](#) for the integral part, and [UART_CLKDIV_FRAG](#) for the fractional part. When using the 80 MHz input clock, the UART controller supports a maximum baud rate of 5 MBaud.

The divisor of the baud rate divider is equal to

$$UART\_CLKDIV + \frac{UART\_CLKDIV\_FRAG}{16}$$

meaning that the final baud rate is equal to

$$\frac{INPUT\_FREQ}{UART\_CLKDIV + \frac{UART\_CLKDIV\_FRAG}{16}}$$

where INPUT_FREQ is the frequency of UART Core's source clock. For example, if [UART_CLKDIV](#) = 694 and [UART_CLKDIV_FRAG](#) = 7, then the divisor value is

$$694 + \frac{7}{16} = 694.4375$$

When [UART_CLKDIV_FRAG](#) is 0, the baud rate generator is an integer clock divider where an output pulse is generated every [UART_CLKDIV](#) input pulses.

When [UART_CLKDIV_FRAG](#) is not 0, the divider is fractional and the output baud rate clock pulses are not strictly uniform. As shown in Figure 25.4-1, for every 16 output pulses, the generator divides either

($\text{UART\_CLKDIV} + 1$) input pulses or UART_CLKDIV input pulses per output pulse. A total of UART_CLKDIV_FRAG output pulses are generated by dividing ($\text{UART\_CLKDIV} + 1$) input pulses, and the remaining ($16 - \text{UART\_CLKDIV\_FRAG}$) output pulses are generated by dividing UART_CLKDIV input pulses.

The output pulses are interleaved as shown in Figure 25.4-1 below, to make the output timing more uniform:

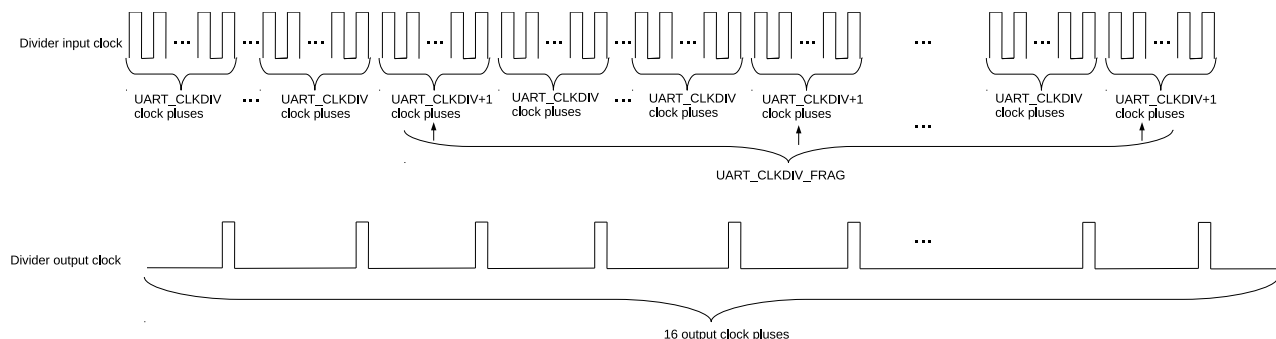


Figure 25.4-1. UART Controllers Division

To support IrDA (see Section 25.4.7 for details), the fractional clock divider for IrDA data transmission generates clock signals divided by $16 \times \text{UART\_CLKDIV\_SYNC\_REG}$. This divider works similarly as the one elaborated above: it takes $\text{UART\_CLKDIV}/16$ as the integer value and the lowest four bits of UART_CLKDIV as the fractional value.

25.4.3.2 Baud Rate Detection

Automatic baud rate detection (Autobaud) on UARTs is enabled by setting `UART_AUTOBAUD_EN`. The Baudrate_Detect module shown in Figure 25.4-2 filters any noise whose pulse width is shorter than `UART_GLITCH_FILT`.

Before communication starts, the transmitter could send random data to the receiver for baud rate detection. `UART_LOWPULSE_MIN_CNT` stores the minimum low pulse width, `UART_HIGHPULSE_MIN_CNT` stores the minimum high pulse width, `UART_POSEDGE_MIN_CNT` stores the minimum pulse width between two positive edges, and `UART_NEGEDGE_MIN_CNT` stores the minimum pulse width between two negative edges. These four fields are read by software to determine the transmitter's baud rate.

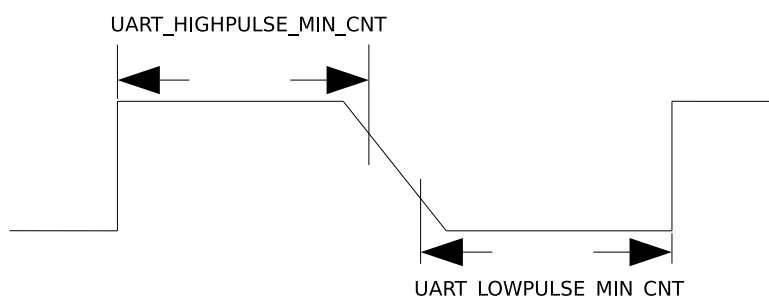


Figure 25.4-2. The Timing Diagram of Weak UART Signals Along Negative Edges

The baud rate can be determined in the following three ways:

1. Normally, to avoid sampling erroneous data along positive or negative edges in a metastable state, which results in the inaccuracy of `UART_LOWPULSE_MIN_CNT` or `UART_HIGHPULSE_MIN_CNT`, use a

weighted average of these two values to eliminate errors for 1-bit pulses. In this case, the baud rate is calculated as follows:

$$B_{\text{uart}} = \frac{f_{\text{clk}}}{(\text{UART\_LOWPULSE\_MIN\_CNT} + \text{UART\_HIGHPULSE\_MIN\_CNT} + 2)/2}$$

2. If UART signals are weak along negative edges as shown in Figure 25.4-2, which leads to an inaccurate average of `UART_LOWPULSE_MIN_CNT` and `UART_HIGHPULSE_MIN_CNT`, use `UART_POSEDGE_MIN_CNT` to determine the transmitter's baud rate as follows:

$$B_{\text{uart}} = \frac{f_{\text{clk}}}{(\text{UART_POSEDGE_MIN_CNT} + 1)/2}$$

3. If UART signals are weak along positive edges, use `UART_NEGEDGE_MIN_CNT` to determine the transmitter's baud rate as follows:

$$B_{\text{uart}} = \frac{f_{\text{clk}}}{(\text{UART_NEGEDGE_MIN_CNT} + 1)/2}$$

25.4.4 UART Data Frame

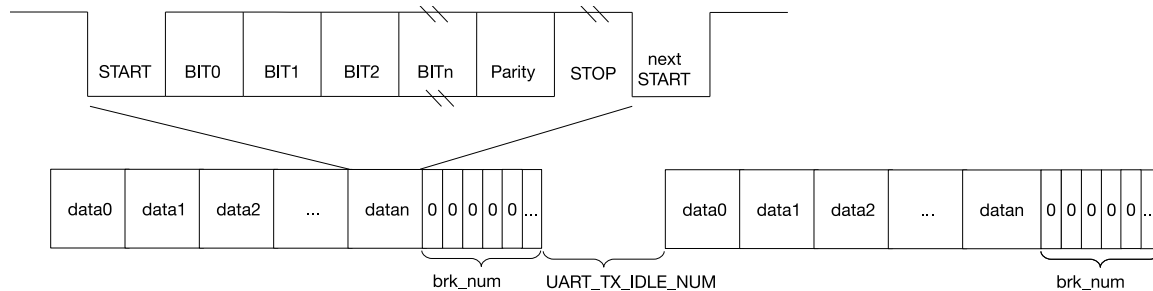


Figure 25.4-3. Structure of UART Data Frame

Figure 25.4-3 shows the basic structure of a data frame. A frame starts with one start bit, and ends with stop bits which can be 1, 1.5, or 2 bit-long, configured by `UART_STOP_BIT_NUM` (in RS485 mode turnaround delay may be added. See details in Section 25.4.6.2). The start bit is logical low, whereas stop bits are logical high.

The actual data length can be anywhere between 5 ~ 8 bit, configured by `UART_BIT_NUM`. When `UART_PARITY_EN` is set, a parity bit is added after data bits. `UART_PARITY` is used to choose even parity or odd parity. When the receiver detects a parity bit error in the data received, a `UART_PARITY_ERR_INT` interrupt is generated, and the data received will still be stored into RX FIFO. When the receiver detects a data frame error, a `UART_FRM_ERR_INT` interrupt is generated, and the data received by default is stored into RX FIFO.

If all data in Tx_FIFO has been sent, a `UART_TX_DONE_INT` interrupt is generated. After this, if the `UART_TXD_BRK` bit is set, then the transmitter will enter the break condition and send several NULL characters. After the NULL characters are sent, the TX data line is logical low. The number of NULL characters is configured by `UART_TX_BRK_NUM`. Once the transmitter has sent all NULL characters, a `UART_TX_BRK_DONE_INT` interrupt is generated. The minimum interval between data frames can be configured using `UART_TX_IDLE_NUM`. If the transmitter stays idle for `UART_TX_IDLE_NUM` or more time, a `UART_TX_BRK_IDLE_DONE_INT` interrupt is generated.

The receiver can also detect the Break conditions when the RX data line detects any low logical level for one NULL character transmission, and a UART_BRK_DET_INT interrupt will be triggered to detect when a break condition has been completed.

The receiver can detect the current bus state through the timeout interrupt UART_RXFIFO_TOUT_INT. The UART_RXFIFO_TOUT_INT interrupt will be triggered when the bus is in the idle state for more than [UART_RX_TOUT_THRHD](#) bit time on current baud rate after the receiver has received at least one byte. You can use this interrupt to detect whether all the data from the transmitter has been sent.

25.4.5 AT_CMD Character Structure

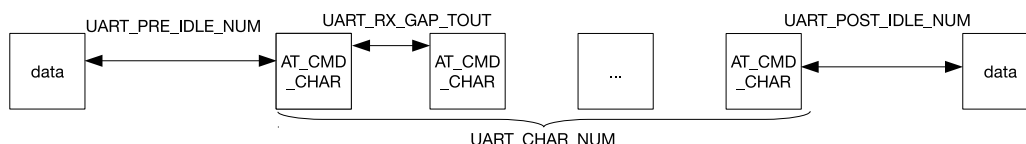


Figure 25.4-4. AT_CMD Character Structure

Figure 25.4-4 is the structure of a special character AT_CMD. If the receiver constantly receives AT_CMD_CHAR and the following conditions are met, a UART_AT_CMD_CHAR_DET_INT interrupt is generated.

- The interval between the first AT_CMD_CHAR and the last non-AT_CMD_CHAR character is at least [UART_PRE_IDLE_NUM](#) cycles.
- The interval between two AT_CMD_CHAR characters is less than [UART_RX_GAP_TOUT](#) in the unit of baud rate cycles.
- The number of AT_CMD_CHAR characters is equal to or greater than [UART_CHAR_NUM](#).
- The interval between the last AT_CMD_CHAR character and next non-AT_CMD_CHAR character is at least [UART_POST_IDLE_NUM](#) cycles.

Note: Given that the interval between AT_CMD_CHAR characters is less than [UART_RX_GAP_TOUT](#) in the unit of baud rate cycles, the APB_CLK frequency is suggested not to be lower than 8 MHz.

25.4.6 RS485

The two regular UART controllers support RS485 communication mode. In this mode differential signals are used to transmit data, so it can communicate over longer distances at higher bit rates than RS232. RS485 has two-wire half-duplex and four-wire full-duplex options. UART controllers support two-wire half-duplex transmission and bus snooping.

25.4.6.1 Driver Control

As shown in Figure 25.4-5, in a two-wire multidrop network, an external RS485 transceiver is needed for differential to single-ended conversion or the other way around. An RS485 transceiver contains a driver and a receiver. When a UART controller is not in transmitter mode, the connection to the differential line can be broken by disabling the driver. When DE is 1, the driver is enabled; when DE is 0, the driver is disabled.

The UART receiver converts differential signals to single-ended signals via an external receiver. RE is the enable control signal for the receiver. When RE is 0, the receiver is enabled; when RE is 1, the receiver is

disabled. If RE is configured as 0, the UART controller is allowed to snoop data on the bus, including the data sent by itself.

DE can be controlled by either software or hardware. To reduce the cost of software, in our design DE is controlled by hardware. As shown in Figure 25.4-5, DE is connected to dtrn_out of UART (please refer to Section 25.4.9.1 for more details).

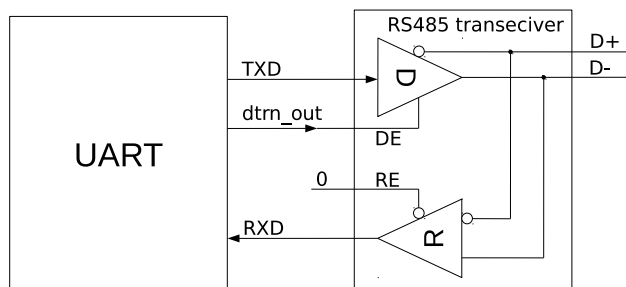


Figure 25.4-5. Driver Control Diagram in RS485 Mode

25.4.6.2 Turnaround Delay

By default, the UART controllers work in receiver mode. When a UART controller is switched from transmitter mode to receiver mode, the RS485 protocol requires a turnaround delay of one cycle after the stop bit. The UART transmitter supports adding a turnaround delay of one cycle before the start bit or after the stop bit. When [UART_DLO_EN](#) is set, a turnaround delay of one cycle is added before the start bit; when [UART_DL1_EN](#) is set, a turnaround delay of one cycle is added after the stop bit.

25.4.6.3 Bus Snooping

In a two-wire multidrop network, UART controllers support bus snooping if RE of the external RS485 transceiver is 0. By default, a UART controller is not allowed to transmit and receive data simultaneously. If [UART_RS485TX_RX_EN](#) is set and the external RS485 transceiver is configured as in Figure 25.4-5, a UART controller may receive data in transmitter mode and snoop the bus. If [UART_RS485RXBY_TX_EN](#) is set, a UART controller may transmit data in receiver mode.

The two UART controllers can snoop the data sent by themselves. In transmitter mode, when a UART controller monitors a collision between the data sent and the data received, a [UART_RS485_CLASH_INT](#) is generated; when a UART controller monitors a data frame error, a [UART_RS485_FRM_ERR_INT](#) interrupt is generated; when a UART controller monitors a parity bit error, a [UART_RS485_PARITY_ERR_INT](#) is generated.

25.4.7 IrDA

IrDA protocol consists of three layers, namely the physical layer, the link access protocol, and the link management protocol. The UART controllers implement IrDA's physical layer. In IrDA encoding, a UART controller supports data rates up to 115.2 kbit/s (SIR, or serial infrared mode). As shown in Figure 25.4-6, the IrDA encoder converts a non-return to zero code (NRZ) signal to a return to zero inverted code (RZI) signal and sends it to the external driver and infrared LED. This encoder uses modulated signals whose pulse width is 3/16 bits to indicate logic "0", and low levels to indicate logic "1". The IrDA decoder receives signals from the infrared receiver and converts them to NRZ signals. In most cases, the receiver is high when it is idle, and the parity bit of the encoder output is opposite to that of the decoder input. If a low pulse is detected, it indicates that a start bit has been received.

When IrDA function is enabled, one bit is divided into 16 clock cycles. If the bit to be sent is zero, then the 9th, 10th, and 11th clock cycle are high.

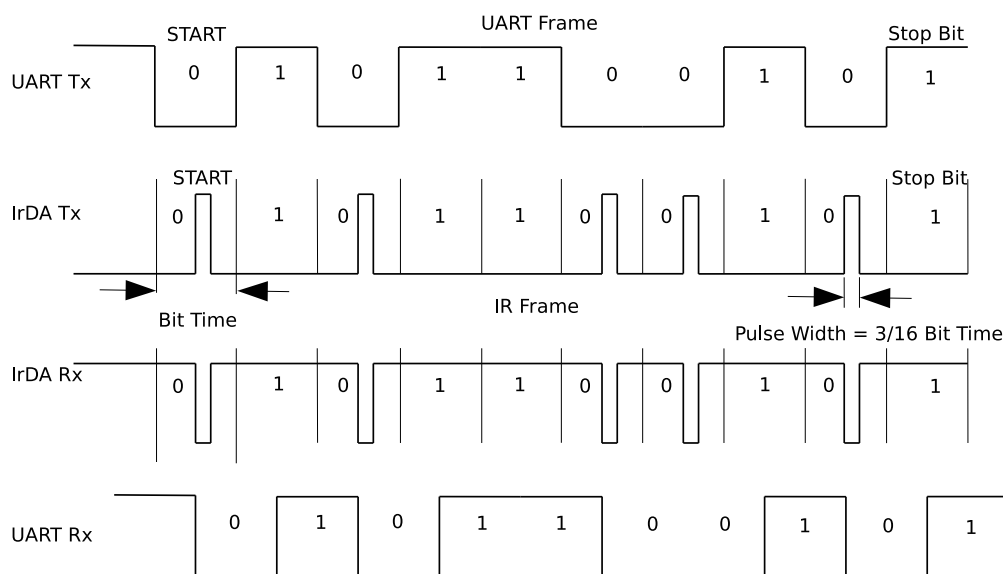


Figure 25.4-6. The Timing Diagram of Encoding and Decoding in SIR mode

The IrDA transceiver is half-duplex, meaning that it cannot send and receive data simultaneously. As shown in Figure 25.4-7, IrDA function is enabled by setting `UART_IRDA_EN`. When `UART_IRDA_TX_EN` is set to 1, the IrDA transceiver is enabled to send data and not allowed to receive data; when `UART_IRDA_TX_EN` is reset to 0, the IrDA transceiver is enabled to receive data and not allowed to send data.

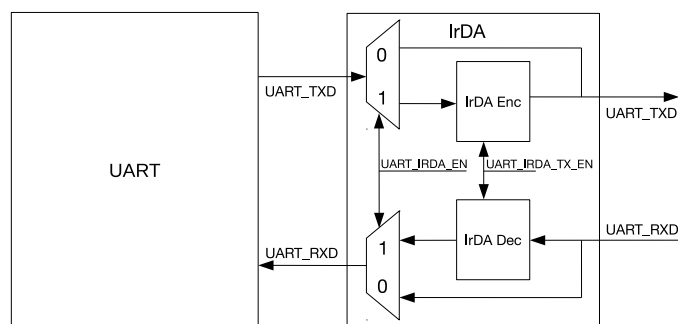


Figure 25.4-7. IrDA Encoding and Decoding Diagram

25.4.8 Wakeup

UART can be set as wakeup source. When a UART controller is in Light-sleep mode, a wake_up signal can be generated in four ways and be sent to the PMU module, which then wakes up ESP32-C5.

- `UART_WK_MODE_SEL = 0`: When all the clocks are disabled, the chip can be woken up by reverting RXD for multiple cycles until the number of positive edges is greater than or equal to `UART_ACTIVE_THRESHOLD + 3`.
- `UART_WK_MODE_SEL = 1`: UART Core keeps working, so the UART receiver can still receive data and store the received data in RX FIFO. When the number of data bytes in RX FIFO is greater than `UART_RX_WAKE_UP_THRHD`, the chip can be woken up from the Light-sleep mode.

- [UART_WK_MODE_SEL](#) = 2: When the UART receiver detects a start bit, the chip will be woken up.
- [UART_WK_MODE_SEL](#) = 3: When the UART receiver receives a specific character sequence, the chip will be woken up. The wakeup characters can be defined by configuring [UART_WK_CHAR0](#), [UART_WK_CHAR1](#), [UART_WK_CHAR2](#), [UART_WK_CHAR3](#), and [UART_WK_CHAR4](#). These characters can be formed into different character sequences by configuring [UART_CHAR_NUM](#) and [UART_WK_CHAR_MASK](#), as shown in Table 25.4-1. Once the sequence is detected, the chip will be woken up. For the last configuration in Table 25.4-1, UART will detect for CHAR0 ~ CHAR4 in order.

Table 25.4-1. UART_CHAR_WAKEUP Mode Configuration

UART_CHAR_NAME	UART_WP_CHAR_MASK	Character Sequence
1	0xF	CHAR4
2	0x7	CHAR3/CHAR4
3	0x3	CHAR2/CHAR3/CHAR4
4	0x1	CHAR1/CHAR2/CHAR3/CHAR4
5	0x0	CHAR0/CHAR1/CHAR2/CHAR3/CHAR4

After the chip is woken up by UART, it is necessary to clear the wake_up signal by transmitting data to UART in Active mode or resetting the whole UART, otherwise the number of rising edges required for the next wakeup will be reduced.

25.4.9 Flow Control

UART controllers have two ways to control data flow, namely hardware flow control and software flow control. Hardware flow control is achieved using output signal `rtsn_out` and input signal `ctsn_in`. Software flow control is achieved by inserting special characters in the data flow sent and detecting special characters in the data flow received.

25.4.9.1 Hardware Flow Control

Figure 25.4-8 shows the hardware flow control of a UART controller. Hardware flow control uses output signal `rtsn_out` and input signal `dsrn_in`. Figure 25.4-9 illustrates how these signals are connected between UART on ESP32-C5 (hereinafter referred to as IU0) and the external UART (hereinafter referred to as EU0).

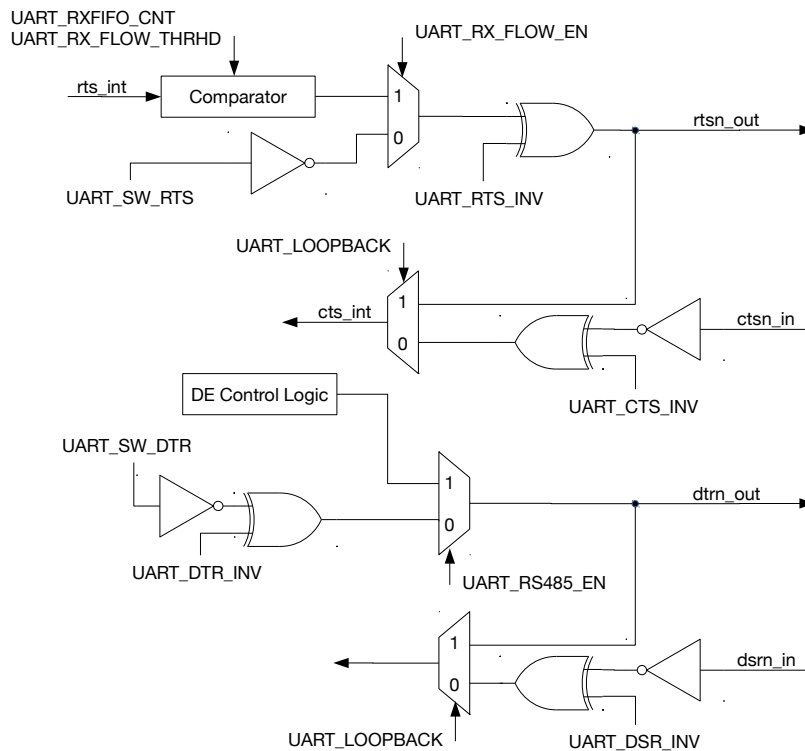


Figure 25.4-8. Hardware Flow Control Diagram

When rtsn_out of IUO is low, EUO is allowed to send data. When rtsn_out of IUO is high, EUO is notified to stop sending data until rtsn_out of IUO returns to low. The output signal rtsn_out can be controlled in two ways.

- Software control: Enter this mode by clearing `UART_RX_FLOW_EN` to 0. In this mode, the level of rtsn_out is changed by configuring `UART_SW_RTS`.
- Hardware control: Enter this mode by setting `UART_RX_FLOW_EN` to 1. In this mode, rtsn_out is pulled high when data in Rx_FIFO exceeds `UART_RX_FLOW_THRHD`.

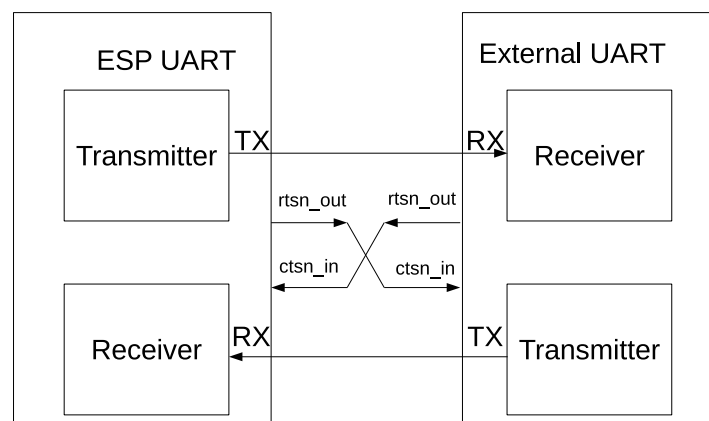


Figure 25.4-9. Connection between Hardware Flow Control Signals

When ctsn_in of IUO is low, IUO is allowed to send data; when ctsn_in is high, IUO is not allowed to send data. When IUO detects an edge change of ctsn_in, a `UART_CTS_CHG_INT` interrupt is generated.

If `dtrn_out` of IUO is high, it indicates that IUO is ready to transmit data. `dtrn_out` is generated by configuring the [UART_SW_DTR](#) field. When the IUO transmitter detects an edge change of `dsrn_in`, a `UART_DSR_CHG_INT` interrupt is generated. After this interrupt is detected, software can obtain the level of input signal `dsrn_in` by reading [UART_DSRN](#). If `dsrn_in` is high, it indicates that EUO is ready to transmit data.

In a two-wire RS485 multidrop network enabled by setting [UART_RS485_EN](#), `dtrn_out` is generated by hardware and used for transmit/receive turnaround. When data transmission starts, `dtrn_out` is pulled high and the external driver is enabled; when data transmission completes, `dtrn_out` is pulled low and the external driver is disabled. Please note that when there is a turnaround delay of one cycle added after the stop bit, `dtrn_out` is pulled low after the delay.

UART loopback test is enabled by setting [UART_LOOPBACK](#). In the test, UART output signal `txd_out` is connected to its input signal `rx_d_in`, `rtsn_out` is connected to `ctsn_in`, and `dtrn_out` is connected to `dsrn_out`. If the data sent matches the data received, it indicates that UART controllers are working properly.

25.4.9.2 Software Flow Control

Instead of CTS/RTS lines, software flow control uses XON/XOFF characters to start or stop data transmission. Such flow control is enabled by setting [UART_SW_FLOW_CON_EN](#) to 1.

When using software flow control, hardware automatically detects if there are XON/XOFF characters in the data flow received, and generate a `UART_SW_XOFF_INT` or a `UART_SW_XON_INT` interrupt accordingly. If an XOFF character is detected, the transmitter stops data transmission once the current byte has been transmitted; if an XON character is detected, the transmitter starts data transmission. In addition, software can force the transmitter to stop sending data by setting [UART_FORCE_XOFF](#), or to start sending data by setting [UART_FORCE_XON](#).

Software determines whether to insert flow control characters based on the remaining room in RX FIFO. When [UART_SEND_XOFF](#) is set, the transmitter sends an XOFF character configured by [UART_XOFF_CHAR](#) after the current byte in transmission; when [UART_SEND_XON](#) is set, the transmitter sends an XON character configured by [UART_XON_CHAR](#) after the current byte in transmission. If the RX FIFO of a UART controller stores more data than [UART_XOFF_THRESHOLD](#), [UART_SEND_XOFF](#) is set by hardware. As a result, the transmitter sends an XOFF character configured by [UART_XOFF_CHAR](#) after the current byte in transmission. If the RX FIFO of a UART controller stores less data than [UART_XON_THRESHOLD](#), [UART_SEND_XON](#) is set by hardware. As a result, the transmitter sends an XON character configured by [UART_XON_CHAR](#) after the current byte in transmission.

In full-duplex mode, when the UART receiver receives an XOFF character, the UART transmitter is not allowed to send any data including XOFF even if the UART receiver receives more data than its threshold. To avoid deadlocks in software flow control or overflow caused thereby, you can set [UART_XON_XOFF_STILL_SEND](#). In this way, the UART transmitter can still send an XOFF character when it is not allowed to send any data.

25.4.10 GDMA Mode

The two UART controllers on ESP32-C5 share one TX/RX GDMA (General Direct Memory Access) channel via UHCI (Universal Host Controller Interface). In GDMA mode, UART controllers support the decoding and encoding of HCI data packets. The [UHCI_UART_SEL](#) field determines which UART controller occupies the GDMA TX/RX channel.

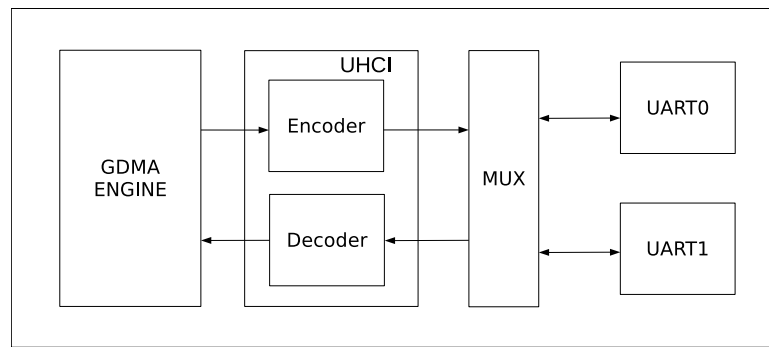


Figure 25.4-10. Data Transfer in GDMA Mode

Figure 25.4-10 shows how data is transferred using GDMA. Before GDMA receives data, software prepares an inlink. `GDMA_INLINK_ADDR_CH $n$` points to the first receive descriptor in the inlink. After `GDMA_INLINK_START_CH $n$` is set, UHCI sends data that UART has received to the decoder. The decoded data is then stored into the RAM pointed by the inlink under the control of GDMA.

Before GDMA sends data, software prepares an outlink and data to be sent. `GDMA_OUTLINK_ADDR_CH $n$` points to the first transmit descriptor in the outlink. After `GDMA_OUTLINK_START_CH $n$` is set, GDMA reads data from the RAM pointed by outlink. The data is then encoded by the encoder, and sent sequentially by the UART transmitter.

HCI data packets have separators at the beginning and the end, with data bits in the middle (separators + data bits + separators). The encoder inserts separators in front of and after data bits, and replaces data bits identical to separators with special characters. The decoder removes separators in front of and after data bits, and replaces special characters with separators. There can be more than one continuous separator at the beginning and the end of a data packet. The separator is configured by `UHCI_SEPER_CHAR`, 0xC0 by default. The special character is configured by `UHCI_ESC_SEQO_CHAR0` (0xDB by default) and `UHCI_ESC_SEQO_CHAR1` (0xDD by default). When all data has been sent, a `GDMA_OUT_TOTAL_EOF_CH $n$ _INT` interrupt is generated. When all data has been received, a `GDMA_IN_SUC_EOF_CH $n$ _INT` is generated.

25.4.11 Interrupts

ESP32-C5's UART n and UHCI can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- UART n _INTR
- UHCI_INTR

There are several internal interrupt sources from UART n and UHCI that can generate the above interrupt signals.

UART n interrupt sources are listed as follows:

- `UART_AT_CMD_CHAR_DET_INT`: Triggered when the receiver detects an AT_CMD character.
- `UART_RS485_CLASH_INT`: Triggered when a collision is detected between the transmitter and the receiver in RS485 mode.
- `UART_RS485_FRM_ERR_INT`: Triggered when an error is detected in the data frame sent by the transmitter in RS485 mode.

- `UART_RS485_PARITY_ERR_INT`: Triggered when an error is detected in the parity bit sent by the transmitter in RS485 mode.
- `UART_TX_DONE_INT`: Triggered when all data in the transmitter's TX FIFO has been sent.
- `UART_TX_BRK_IDLE_DONE_INT`: Triggered when the transmitter stays idle for the minimum interval (threshold) after sending the last data bit.
- `UART_TX_BRK_DONE_INT`: Triggered when the transmitter has sent all NULL characters after all data in TX FIFO had been sent.
- `UART_GLITCH_DET_INT`: Triggered when the receiver detects a glitch in the middle of the start bit.
- `UART_SW_XOFF_INT`: Triggered when [UART_SW_FLOW_CON_EN](#) is set and the receiver receives a XOFF character.
- `UART_SW_XON_INT`: Triggered when [UART_SW_FLOW_CON_EN](#) is set and the receiver receives a XON character.
- `UART_RXFIFO_TOUT_INT`: Triggered when the receiver has received at least one byte, and the bus remains idle for [UART_RX_TOUT_THRHD](#) bit time.
- `UART_BRK_DET_INT`: Triggered when the receiver detects a NULL character (i.e., logic 0 for one NULL character transmission) after stop bits.
- `UART_CTS_CHG_INT`: Triggered when the receiver detects an edge change of CTSn signals.
- `UART_DSR_CHG_INT`: Triggered when the receiver detects an edge change of DSRn signals.
- `UART_RXFIFO_OVF_INT`: Triggered when the amount of data received by the receiver exceeds the storage capacity of the FIFO.
- `UART_FRM_ERR_INT`: Triggered when the receiver detects a data frame error.
- `UART_PARITY_ERR_INT`: Triggered when the receiver detects a parity error.
- `UART_TXFIFO_EMPTY_INT`: Triggered when TX FIFO stores less data than what [UART_TXFIFO_EMPTY_THRHD](#) specifies.
- `UART_RXFIFO_FULL_INT`: Triggered when the receiver receives more data than what [UART_RXFIFO_FULL_THRHD](#) specifies.
- `UART_WAKEUP_INT`: Triggered when UART is woken up.

UHCI interrupt sources are listed as follows:

- `UHCI_APP_CTRL1_INT`: Triggered when software sets [UHCI_APP_CTRL1_INT_RAW](#).
- `UHCI_APP_CTRL0_INT`: Triggered when software sets [UHCI_APP_CTRL0_INT_RAW](#).
- `UHCI_OUTLINK_EOF_ERR_INT`: Triggered when an EOF error is detected in a transmit descriptor.
- `UHCI_SEND_A_REG_Q_INT`: Triggered when UHCI has sent a series of short packets using `always_send`.
- `UHCI_SEND_S_REG_Q_INT`: Triggered when UHCI has sent a series of short packets using `single_send`.
- `UHCI_TX_HUNG_INT`: Triggered when UHCI takes too long to read RAM using a GDMA transmit channel.
- `UHCI_RX_HUNG_INT`: Triggered when UHCI takes too long to receive data using a GDMA receive channel.
- `UHCI_TX_START_INT`: Triggered when GDMA detects a separator character.

- UHCI_RX_START_INT: Triggered when a separator character has been sent.

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [25.6 Register Summary](#).

25.5 Programming Procedures

25.5.1 Register Type

All UART registers are in the APB_CLK domain.

UART configuration registers can be classified into two groups. One group of registers are read in the APB_CLK or AHB_CLK domains, so once such registers are configured no extra operations are required. The other group of registers are read in the UART_SCLK domain, and therefore need to implement the clock domain crossing design. Once these registers are configured, the configured values need to be synchronized to the UART Core's clock domain by writing to [UART_REG_UPDATE](#). Once all values have been synchronized, [UART_REG_UPDATE](#) will be automatically cleared by hardware. After configuring registers that need synchronization, it is recommended to check whether [UART_REG_UPDATE](#) is 0. This is to ensure that register values configured before have already been synchronized.

To distinguish between these two groups of registers easily, all registers that implement the clock domain crossing design have the _SYNC suffix, and are put together in Section [25.6](#). Those without the _SYNC suffix in Section [25.6](#) are configuration registers that require no clock domain crossing.

25.5.2 Detailed Steps

Figure 25.5-1 illustrates the process to program UART controllers, namely initialize UART, configure registers, enable the UART transmitter or receiver, and finish data transmission.

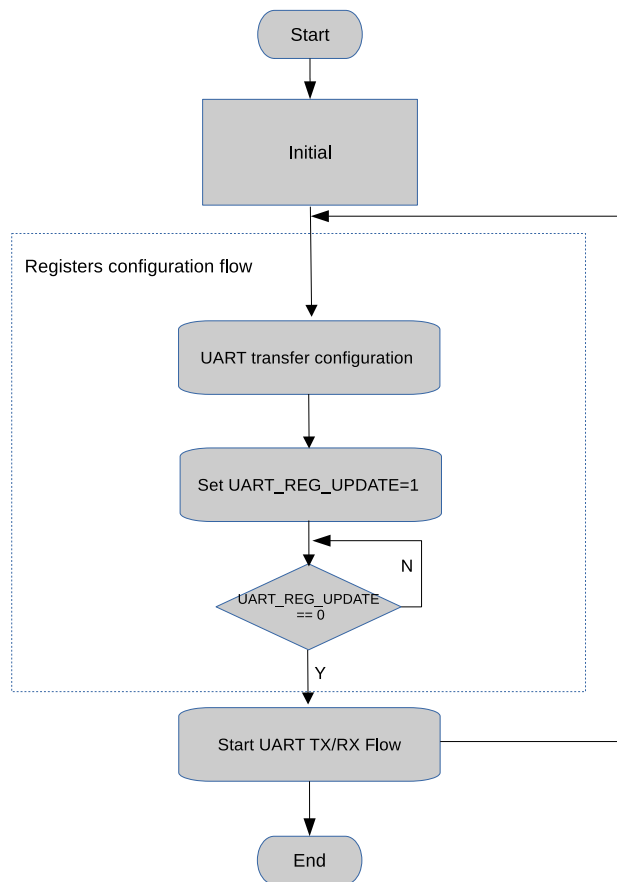


Figure 25.5-1. UART Programming Procedures

25.5.2.1 Initializing UART n

To initialize UART n :

- Write 1 to HP_SYS_CLKRST_RST_EN_UART n _APB.
- Clear HP_SYS_CLKRST_RST_EN_UART n _APB to 0.
- Write 1 to HP_SYS_CLKRST_RST_EN_UART n _CORE.
- Clear HP_SYS_CLKRST_RST_EN_UART n _CORE to 0.

25.5.2.2 Configuring UART n Communication

To configure UART n communication:

- Wait for `UART_REG_UPDATE` to become 0, which indicates the completion of the last synchronization.
- Select the clock source via HP_SYS_CLKRST_UART n _CLK_SRC_SEL.
- Configure divisor of the divider via HP_SYS_CLKRST_UART n _SCLK_DIV_NUM, HP_SYS_CLKRST_UART n _SCLK_DIV_DENOMINATOR, and

HP_SYS_CLKRST_UART n _SCLK_DIV_NUMERATOR.

- Configure the baud rate for transmission via [UART_CLKDIV](#) and [UART_CLKDIV_FRAG](#).
- Configure data length via [UART_BIT_NUM](#).
- Configure odd or even parity check via [UART_PARITY_EN](#) and [UART_PARITY](#).
- Optional steps depending on application ...
- Synchronize the configured values to the Core Clock domain by writing 1 to [UART_REG_UPDATE](#).

25.5.2.3 Enabling UART n

To enable UART n transmitter:

- Configure TX FIFO's empty threshold via [UART_TXFIFO_EMPTY_THRHD](#).
- Disable UART_TXFIFO_EMPTY_INT interrupt by clearing [UART_TXFIFO_EMPTY_INT_ENA](#).
- Write data to be sent to [UART_RXFIFO_RD_BYTE](#).
- Clear UART_TXFIFO_EMPTY_INT interrupt by setting [UART_TXFIFO_EMPTY_INT_CLR](#).
- Enable UART_TXFIFO_EMPTY_INT interrupt by setting [UART_TXFIFO_EMPTY_INT_ENA](#).
- Check [UART_TXFIFO_EMPTY_INT_ST](#) and wait for the completion of data transmission.

To enable UART n receiver:

- Configure RX FIFO's full threshold via [UART_RXFIFO_FULL_THRHD](#).
- Enable UART_RXFIFO_FULL_INT interrupt by setting [UART_RXFIFO_FULL_INT_ENA](#).
- Check [UART_RXFIFO_FULL_INT_ST](#) and wait until the RX FIFO is full.
- Read data from RX FIFO via [UART_RXFIFO_RD_BYTE](#), and obtain the number of bytes received in RX FIFO via [UART_RXFIFO_CNT](#).

25.6 Register Summary

25.6.1 UART Register Summary

The addresses in this section are relative to UART Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
FIFO Configuration Register			
UART_FIFO_REG	FIFO data register	0x0000	RO
UART_TOUT_CONF_SYNC_REG	UART threshold and allocation configuration	0x0064	R/W
UART Interrupt Register			
UART_INT_RAW_REG	Raw interrupt status	0x0004	R/WTC/SS
UART_INT_ST_REG	Masked interrupt status	0x0008	RO
UART_INT_ENA_REG	Interrupt enable bits	0x000C	R/W
UART_INT_CLR_REG	Interrupt clear bits	0x0010	WT
Configuration Register			
UART_CLKDIV_SYNC_REG	Clock divider configuration	0x0014	R/W
UART_RX_FILT_REG	RX filter configuration	0x0018	R/W
UART_CONFO_SYNC_REG	Configuration register 0	0x0020	R/W
UART_CONF1_REG	Configuration register 1	0x0024	R/W
UART_HWFC_CONF_SYNC_REG	Hardware flow control configuration	0x002C	R/W
UART_SLEEP_CONFO_REG	UART sleep configuration register 0	0x0030	R/W
UART_SLEEP_CONF1_REG	UART sleep configuration register 1	0x0034	R/W
UART_SLEEP_CONF2_REG	UART sleep configuration register 2	0x0038	R/W
UART_SWFC_CONFO_SYNC_REG	Software flow control character configuration	0x003C	varies
UART_SWFC_CONF1_REG	Software flow control character configuration	0x0040	R/W
UART_TXBRK_CONF_SYNC_REG	TX break character configuration	0x0044	R/W
UART_IDLE_CONF_SYNC_REG	Frame end idle time configuration	0x0048	R/W
UART_RS485_CONF_SYNC_REG	RS485 mode configuration	0x004C	R/W
UART_CLK_CONF_REG	UART core clock configuration	0x0088	R/W
UART_REG_UPDATE_REG	UART register configuration update	0x0098	R/W/SC
UART_ID_REG	UART ID register	0x009C	R/W
Status Register			
UART_STATUS_REG	UART status register	0x001C	RO
UART_MEM_TX_STATUS_REG	TX FIFO write and read offset address	0x0068	RO
UART_MEM_RX_STATUS_REG	Rx FIFO write and read offset address	0x006C	RO
UART_FSM_STATUS_REG	UART transmit and receive status	0x0070	RO
UART_AFIFO_STATUS_REG	UART asynchronous FIFO status	0x0090	RO
AT Escape Sequence Selection Configuration Register			
UART_AT_CMD_PRECNT_SYNC_REG	Pre-sequence timing configuration	0x0050	R/W
UART_AT_CMD_POSTCNT_SYNC_REG	Post-sequence timing configuration	0x0054	R/W
UART_AT_CMD_GAPTOUT_SYNC_REG	Timeout configuration	0x0058	R/W

Name	Description	Address	Access
UART_AT_CMD_CHAR_SYNC_REG	AT escape sequence detection configuration	0x005C	R/W
Autobaud Register			
UART_POSPULSE_REG	Autobaud high pulse register	0x0074	RO
UART_NEGPULSE_REG	Autobaud low pulse register	0x0078	RO
UART_LOWPULSE_REG	Autobaud minimum low pulse duration register	0x007C	RO
UART_HIGHPULSE_REG	Autobaud minimum high pulse duration register	0x0080	RO
UART_RXD_CNT_REG	Autobaud edge change count register	0x0084	RO
Version Register			
UART_DATE_REG	UART version control register	0x008C	R/W

25.6.2 LP UART Register Summary

The addresses in this section are relative to LP UART base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
FIFO Configuration Register			
LP_UART_FIFO_REG	FIFO data register	0x0000	RO
LP_UART_TOUT_CONF_SYNC_REG	LP UART threshold and allocation configuration	0x0064	R/W
LP UART Interrupt Register			
LP_UART_INT_RAW_REG	Raw interrupt status	0x0004	R/WTC/SS
LP_UART_INT_ST_REG	Masked interrupt status	0x0008	RO
LP_UART_INT_ENA_REG	Interrupt enable bits	0x000C	R/W
LP_UART_INT_CLR_REG	Interrupt clear bits	0x0010	WT
Configuration Register			
LP_UART_CLKDIV_SYNC_REG	Clock divider configuration	0x0014	R/W
LP_UART_RX_FILT_REG	RX filter configuration	0x0018	R/W
LP_UART_CONFO_SYNC_REG	Configuration register 0	0x0020	R/W
LP_UART_CONF1_REG	Configuration register 1	0x0024	R/W
LP_UART_HWFC_CONF_SYNC_REG	Hardware flow control configuration	0x002C	R/W
LP_UART_SLEEP_CONFO_REG	LP UART sleep configuration register 0	0x0030	R/W
LP_UART_SLEEP_CONF1_REG	LP UART sleep configuration register 1	0x0034	R/W
LP_UART_SLEEP_CONF2_REG	LP UART sleep configuration register 2	0x0038	R/W
LP_UART_SWFC_CONFO_SYNC_REG	Software flow control character configuration	0x003C	varies
LP_UART_SWFC_CONF1_REG	Software flow control character configuration	0x0040	R/W
LP_UART_TXBRK_CONF_SYNC_REG	TX break character configuration	0x0044	R/W
LP_UART_IDLE_CONF_SYNC_REG	Frame end idle time configuration	0x0048	R/W
LP_UART_DELAY_CONF_SYNC_REG	RS485 mode configuration	0x004C	R/W

Name	Description	Address	Access
LP_UART_CLK_CONF_REG	LP UART core clock configuration	0x0088	R/W
LP_UART_REG_UPDATE_REG	LP UART register configuration update register	0x0098	R/W/SC
LP_UART_ID_REG	LP UART ID register	0x009C	R/W
Status Register			
LP_UART_STATUS_REG	LP UART status register	0x001C	RO
LP_UART_MEM_TX_STATUS_REG	TX FIFO write and read offset address	0x0068	RO
LP_UART_MEM_RX_STATUS_REG	RX FIFO write and read offset address	0x006C	RO
LP_UART_FSM_STATUS_REG	LP UART transmit and receive status	0x0070	RO
LP_UART_AFIFO_STATUS_REG	LP UART asynchronous FIFO Status	0x0090	RO
AT Escape Sequence Selection Configuration Register			
LP_UART_AT_CMD_PRECNT_SYNC_REG	Pre-sequence timing configuration	0x0050	R/W
LP_UART_AT_CMD_POSTCNT_SYNC_REG	Post-sequence timing configuration	0x0054	R/W
LP_UART_AT_CMD_GAPTOU_SYNC_REG	Timeout configuration	0x0058	R/W
LP_UART_AT_CMD_CHAR_SYNC_REG	AT escape sequence detection configuration	0x005C	R/W
Version Register			
LP_UART_DATE_REG	LP UART version register	0x008C	R/W

25.6.3 UHCI Register Summary

The addresses in this section are relative to UHCI base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Register			
UHCI_CONFO_REG	UHCI configuration register	0x0000	R/W
UHCI_CONF1_REG	UHCI configuration register	0x0014	varies
UHCI_ESCAPE_CONF_REG	Escape character configuration	0x0020	R/W
UHCI_HUNG_CONF_REG	Timeout configuration	0x0024	R/W
UHCI_ACK_NUM_REG	UHCI ACK number configuration	0x0028	varies
UHCI_QUICK_SENT_REG	UHCI quick send configuration register	0x0030	varies
UHCI_REG_Q0_WORD0_REG	Q0 WORD0 quick send register	0x0034	R/W
UHCI_REG_Q0_WORD1_REG	Q0 WORD1 quick send register	0x0038	R/W
UHCI_REG_Q1_WORD0_REG	Q1 WORD0 quick send register	0x003C	R/W
UHCI_REG_Q1_WORD1_REG	Q1 WORD1 quick send register	0x0040	R/W
UHCI_REG_Q2_WORD0_REG	Q2 WORD0 quick send register	0x0044	R/W
UHCI_REG_Q2_WORD1_REG	Q2 WORD1 quick send register	0x0048	R/W
UHCI_REG_Q3_WORD0_REG	Q3 WORD0 quick send register	0x004C	R/W
UHCI_REG_Q3_WORD1_REG	Q3 WORD1 quick send register	0x0050	R/W
UHCI_REG_Q4_WORD0_REG	Q4 WORD0 quick send register	0x0054	R/W
UHCI_REG_Q4_WORD1_REG	Q4 WORD1 quick send register	0x0058	R/W

Name	Description	Address	Access
UHCI_REG_Q5_WORD0_REG	Q5 WORD0 quick send register	0x005C	R/W
UHCI_REG_Q5_WORD1_REG	Q5 WORD1 quick send register	0x0060	R/W
UHCI_REG_Q6_WORD0_REG	Q6 WORD0 quick send register	0x0064	R/W
UHCI_REG_Q6_WORD1_REG	Q6 WORD1 quick register	0x0068	R/W
UHCI_ESC_CONFO_REG	Escape sequence configuration register 0	0x006C	R/W
UHCI_ESC_CONF1_REG	Escape sequence configuration register 1	0x0070	R/W
UHCI_ESC_CONF2_REG	Escape sequence configuration register 2	0x0074	R/W
UHCI_ESC_CONF3_REG	Escape sequence configuration register 3	0x0078	R/W
UHCI_PKT_THRES_REG	Configuration register for packet length	0x007C	R/W
UHCI Interrupt Register			
UHCI_INT_RAW_REG	Raw interrupt status	0x0004	varies
UHCI_INT_ST_REG	Masked interrupt status	0x0008	RO
UHCI_INT_ENA_REG	Interrupt enable bits	0x000C	R/W
UHCI_INT_CLR_REG	Interrupt clear bits	0x0010	WT
UHCI Status Register			
UHCI_STATE0_REG	UHCI receive status	0x0018	RO
UHCI_STATE1_REG	UHCI transmit status	0x001C	RO
UHCI_RX_HEAD_REG	UHCI packet header register	0x002C	RO
Version Register			
UHCI_DATE_REG	UHCI version control register	0x0080	R/W

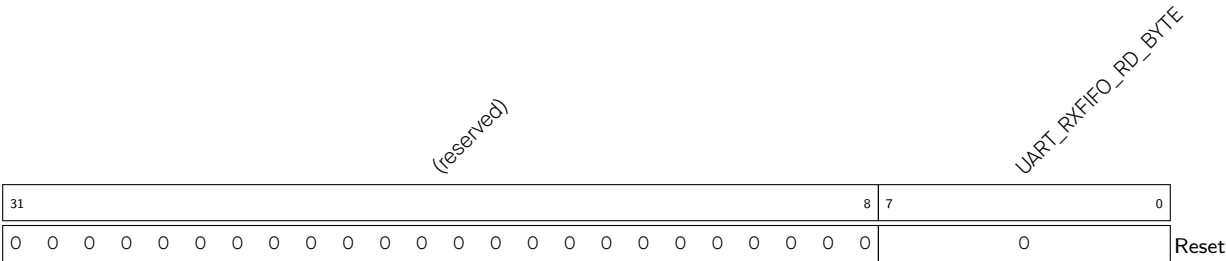
25.7 Registers

25.7.1 UART Registers

The addresses in this section are relative to UART Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

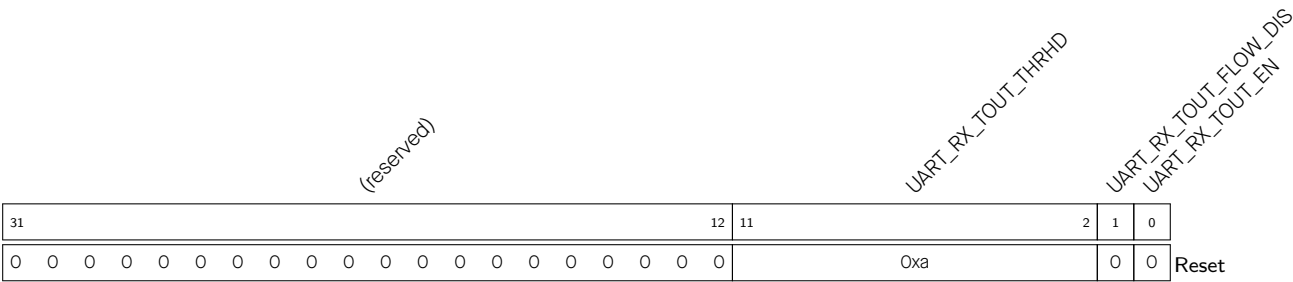
For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 25.1. UART_FIFO_REG (0x0000)



UART_RXFIFO_RD_BYTE Represents the data UART_n reads from FIFO.
Measurement unit: byte. (RO)

Register 25.2. UART_TOUT_CONF_SYNC_REG (0x0064)



- UART_RX_TOUT_EN** Configures whether or not to enable UART receiver’s timeout function.
0: Disable
1: Enable
(R/W)
- UART_RX_TOUT_FLOW_DIS** Configures whether or not to disable the idle status counter when hardware flow control is enabled.
0: Enable
1: Disable
(R/W)
- UART_RX_TOUT_THRHD** Configures the amount of time that the bus can remain idle before timeout.
Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.3. UART_INT_RAW_REG (0x0004)

(reserved)																UART_WAKEUP_INT_RAW UART_AT_CMD_CHAR_DET_INT_RAW UART_RS485_CLASH_INT_RAW UART_RS485_FRM_ERR_INT_RAW UART_RS485_PARITY_ERR_INT_RAW UART_TX_DONE_INT_RAW UART_TX_BRK_IDLE_DONE_INT_RAW UART_GLITCH_DET_INT_RAW UART_SW_XON_INT_RAW UART_SW_XOFF_INT_RAW UART_RXFIFO_TOUT_INT_RAW UART_CTS_CHG_INT_RAW UART_DSR_CHG_INT_RAW UART_FRM_ERR_INT_RAW UART_PARITY_ERR_INT_RAW UART_TXFIFO_EMPTY_INT_RAW UART_RXFIFO_FULL_INT_RAW																		
31												20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset				

UART_RXFIFO_FULL_INT_RAW The raw interrupt status of UART_RXFIFO_FULL_INT. (R/WTC/SS)

UART_TXFIFO_EMPTY_INT_RAW The raw interrupt status of UART_TXFIFO_EMPTY_INT. (R/WTC/SS)

UART_PARITY_ERR_INT_RAW The raw interrupt status of UART_PARITY_ERR_INT. (R/WTC/SS)

UART_FRM_ERR_INT_RAW The raw interrupt status of UART_FRM_ERR_INT. (R/WTC/SS)

UART_RXFIFO_OVF_INT_RAW The raw interrupt status of UART_RXFIFO_OVF_INT. (R/WTC/SS)

UART_DSR_CHG_INT_RAW The raw interrupt status of UART_DSR_CHG_INT. (R/WTC/SS)

UART_CTS_CHG_INT_RAW The raw interrupt status of UART_CTS_CHG_INT. (R/WTC/SS)

UART_BRK_DET_INT_RAW The raw interrupt status of UART_BRK_DET_INT. (R/WTC/SS)

UART_RXFIFO_TOUT_INT_RAW The raw interrupt status of UART_RXFIFO_TOUT_INT. (R/WTC/SS)

UART_SW_XON_INT_RAW The raw interrupt status of UART_SW_XON_INT. (R/WTC/SS)

UART_SW_XOFF_INT_RAW UART_SW_XOFF_INT. (R/WTC/SS)

UART_GLITCH_DET_INT_RAW The raw interrupt status of UART_GLITCH_DET_INT. (R/WTC/SS)

UART_TX_BRK_DONE_INT_RAW The raw interrupt status of UART_TX_BRK_DONE_INT. (R/WTC/SS)

UART_TX_BRK_IDLE_DONE_INT_RAW The raw interrupt status of UART_TX_BRK_IDLE_DONE_INT. (R/WTC/SS)

UART_TX_DONE_INT_RAW The raw interrupt status of UART_TX_DONE_INT. (R/WTC/SS)

UART_RS485_PARITY_ERR_INT_RAW The raw interrupt status of UART_RS485_PARITY_ERR_INT. (R/WTC/SS)

UART_RS485_FRM_ERR_INT_RAW The raw interrupt status of UART_RS485_FRM_ERR_INT. (R/WTC/SS)

Continued on the next page...

Register 25.3. UART_INT_RAW_REG (0x0004)

Continued from the previous page...

UART_RS485_CLASH_INT_RAW The raw interrupt status of UART_RS485_CLASH_INT. (R/WTC/SS)

UART_AT_CMD_CHAR_DET_INT_RAW The raw interrupt status of UART_AT_CMD_CHAR_DET_INT.
(R/WTC/SS)

UART_WAKEUP_INT_RAW The raw interrupt status of UART_WAKEUP_INT. (R/WTC/SS)

Register 25.4. UART_INT_ST_REG (0x0008)

(reserved)												UART_WAKEUP_INT_ST UART_AT_CMD_CHAR_DET_INT_ST UART_RS485_CLASH_INT_ST UART_RS485_FRM_ERR_INT_ST UART_RS485_PARITY_ERR_INT_ST UART_TX_DONE_INT_ST UART_TX_BRK_IDLE_DONE_INT_ST UART_GLITCH_DET_INT_ST UART_SW_XON_INT_ST UART_SW_XOFF_INT_ST UART_RXFIFO_TOUT_INT_ST UART_BRK_DET_INT_ST UART_CTS_CHG_INT_ST UART_DSR_CHG_INT_ST UART_RXFIFO_OVF_INT_ST UART_FRM_ERR_INT_ST UART_PARITY_ERR_INT_ST UART_TXFIFO_EMPTY_INT_ST UART_RXFIFO_FULL_INT_ST																					
31													20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset				

UART_RXFIFO_FULL_INT_ST The masked interrupt status of UART_RXFIFO_FULL_INT. (RO)

UART_TXFIFO_EMPTY_INT_ST The masked interrupt status of UART_TXFIFO_EMPTY_INT. (RO)

UART_PARITY_ERR_INT_ST The masked interrupt status of UART_PARITY_ERR_INT. (RO)

UART_FRM_ERR_INT_ST The masked interrupt status of UART_FRM_ERR_INT. (RO)

UART_RXFIFO_OVF_INT_ST The masked interrupt status of UART_RXFIFO_OVF_INT. (RO)

UART_DSR_CHG_INT_ST The masked interrupt status of UART_DSR_CHG_INT. (RO)

UART_CTS_CHG_INT_ST The masked interrupt status of UART_CTS_CHG_INT. (RO)

UART_BRK_DET_INT_ST The masked interrupt status of UART_BRK_DET_INT. (RO)

UART_RXFIFO_TOUT_INT_ST The masked interrupt status of UART_RXFIFO_TOUT_INT. (RO)

UART_SW_XON_INT_ST The masked interrupt status of UART_SW_XON_INT. (RO)

UART_SW_XOFF_INT_ST The masked interrupt status of UART_SW_XOFF_INT. (RO)

UART_GLITCH_DET_INT_ST The masked interrupt status of UART_GLITCH_DET_INT. (RO)

UART_TX_BRK_DONE_INT_ST The masked interrupt status of UART_TX_BRK_DONE_INT. (RO)

UART_TX_BRK_IDLE_DONE_INT_ST The masked interrupt status of UART_TX_BRK_IDLE_DONE_INT. (RO)

UART_TX_DONE_INT_ST The masked interrupt status of UART_TX_DONE_INT. (RO)

UART_RS485_PARITY_ERR_INT_ST The masked interrupt status of UART_RS485_PARITY_ERR_INT. (RO)

UART_RS485_FRM_ERR_INT_ST The masked interrupt status of UART_RS485_FRM_ERR_INT. (RO)

UART_RS485_CLASH_INT_ST The masked interrupt status of UART_RS485_CLASH_INT. (RO)

UART_AT_CMD_CHAR_DET_INT_ST The masked interrupt status of UART_AT_CMD_CHAR_DET_INT. (RO)

UART_WAKEUP_INT_ST The masked interrupt status of UART_WAKEUP_INT. (RO)

Register 25.5. UART_INT_ENA_REG (0x000C)

[illegible]

UART_RXFIFO_FULL_INT_ENA Write 1 to enable UART_RXFIFO_FULL_INT. (R/W)

UART_TXFIFO_EMPTY_INT_ENA Write 1 to enable UART_TXFIFO_EMPTY_INT. (R/W)

UART_PARITY_ERR_INT_ENA Write 1 to enable UART_PARITY_ERR_INT. (R/W)

UART_FRM_ERR_INT_ENA Write 1 to enable UART_FRM_ERR_INT. (R/W)

UART_RXFIFO_OVF_INT_ENA Write 1 to enable UART_RXFIFO_OVF_INT. (R/W)

UART_DSR_CHG_INT_ENA Write 1 to enable UART_DSR_CHG_INT. (R/W)

UART_CTS_CHG_INT_ENA Write 1 to enable UART_CTS_CHG_INT. (R/W)

UART_BRK_DET_INT_ENA Write 1 to enable UART_BRK_DET_INT. (R/W)

UART_RXFIFO_TOUT_INT_ENA Write 1 to enable UART_RXFIFO_TOUT_INT. (R/W)

UART_SW_XON_INT_ENA Write 1 to enable UART_SW_XON_INT.(R/W)

UART_SW_XOFF_INT_ENA Write 1 to enable UART_SW_XOFF_INT. (R/W)

UART_GLITCH_DET_INT_ENA Write 1 to enable UART_GLITCH_DET_INT. (R/W)

UART_TX_BRK_DONE_INT_ENA Write 1 to enable UART_TX_BRK_DONE_INT. (R/W)

UART_TX_BRK_IDLE_DONE_INT_ENA Write 1 to enable UART_TX_BRK_IDLE_DONE_INT. (R/W)

UART_TX_DONE_INT_ENA Write 1 to enable UART_TX_DONE_INT. (R/W)

UART_RS485_PARITY_ERR_INT_ENA Write 1 to enable UART_RS485_PARITY_ERR_INT. (R/W)

UART_RS485_FRM_ERR_INT_ENA Write 1 to enable UART_RS485_FRM_ERR_INT. (R/W)

UART_RS485_CLASH_INT_ENA Write 1 to enable UART_RS485_CLASH_INT. (R/W)

UART_AT_CMD_CHAR_DET_INT_ENA Write 1 to enable UART_AT_CMD_CHAR_DET_INT. (R/W)

UART_WAKEUP_INT_ENA Write 1 to enable UART_WAKEUP_INT. (R/W)

Register 25.6. UART_INT_CLR_REG (0x0010)

(reserved)																UART_WAKEUP_INT_CLR UART_AT_CMD_CHAR_DET_INT_CLR UART_RS485_CLASH_INT_CLR UART_RS485_FRM_ERR_INT_CLR UART_RS485_PARITY_ERR_INT_CLR UART_TX_DONE_INT_CLR UART_TX_BRK_IDLE_DONE_INT_CLR UART_GLITCH_DET_INT_CLR UART_SW_XOFF_INT_CLR UART_SW_XON_INT_CLR UART_RXFIFO_TOUT_INT_CLR UART_BRK_DET_INT_CLR UART_CTS_CHG_INT_CLR UART_DSR_CHG_INT_CLR UART_RXFIFO_OVF_INT_CLR UART_FRM_ERR_INT_CLR UART_PARITY_ERR_INT_CLR UART_TXFIFO_EMPTY_INT_CLR UART_RXFIFO_FULL_INT_CLR																
31												20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset		

UART_RXFIFO_FULL_INT_CLR Write 1 to clear UART_RXFIFO_FULL_INT. (WT)

UART_TXFIFO_EMPTY_INT_CLR Write 1 to clear UART_TXFIFO_EMPTY_INT. (WT)

UART_PARITY_ERR_INT_CLR Write 1 to clear UART_PARITY_ERR_INT. (WT)

UART_FRM_ERR_INT_CLR Write 1 to clear UART_FRM_ERR_INT. (WT)

UART_RXFIFO_OVF_INT_CLR Write 1 to clear UART_RXFIFO_OVF_INT. (WT)

UART_DSR_CHG_INT_CLR Write 1 to clear UART_DSR_CHG_INT. (WT)

UART_CTS_CHG_INT_CLR Write 1 to clear UART_CTS_CHG_INT. (WT)

UART_BRK_DET_INT_CLR Write 1 to clear UART_BRK_DET_INT. (WT)

UART_RXFIFO_TOUT_INT_CLR Write 1 to clear UART_RXFIFO_TOUT_INT. (WT)

UART_SW_XON_INT_CLR Write 1 to clear UART_SW_XON_INT. (WT)

UART_SW_XOFF_INT_CLR Write 1 to clear UART_SW_XOFF_INT. (WT)

UART_GLITCH_DET_INT_CLR Write 1 to clear UART_GLITCH_DET_INT. (WT)

UART_TX_BRK_DONE_INT_CLR Write 1 to clear UART_TX_BRK_DONE_INT. (WT)

UART_TX_BRK_IDLE_DONE_INT_CLR Write 1 to clear UART_TX_BRK_IDLE_DONE_INT. (WT)

UART_TX_DONE_INT_CLR Write 1 to clear UART_TX_DONE_INT. (WT)

UART_RS485_PARITY_ERR_INT_CLR Write 1 to clear UART_RS485_PARITY_ERR_INT. (WT)

UART_RS485_FRM_ERR_INT_CLR Write 1 to clear UART_RS485_FRM_ERR_INT. (WT)

UART_RS485_CLASH_INT_CLR Write 1 to clear UART_RS485_CLASH_INT. (WT)

UART_AT_CMD_CHAR_DET_INT_CLR Write 1 to clear UART_AT_CMD_CHAR_DET_INT. (WT)

UART_WAKEUP_INT_CLR Write 1 to clear UART_WAKEUP_INT. (WT)

Register 25.7. UART_CLKDIV_SYNC_REG (0x0014)

(reserved)								UART_CLKDIV_FRAG												(reserved)								UART_CLKDIV																																																																																			
31								24								23								20								19								12								11								0																																																							
0								0								0								0								0								0								0								0x0								0								0								0								0								0x2b6								Reset							

Reset

UART_CLKDIV Configures the integral part of the divisor for baud rate generation. (R/W)

UART_CLKDIV_FRAG Configures the fractional part of the divisor for baud rate generation. (R/W)

Register 25.8. UART_RX_FILT_REG (0x0018)

(reserved)																								UART_GLITCH_FILT_EN		UART_GLITCH_FILT																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
31																								9	8	7	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

Reset

UART_GLITCH_FILT Configures the width of a pulse to be filtered.

Measurement unit: UART Core's clock cycle.

Pulses whose width is lower than this value will be ignored. (R/W)

UART_GLITCH_FILT_EN Configures whether or not to enable RX signal filter.

0: Disable

1: Enable

(R/W)

Register 25.9. UART_CONFO_SYNC_REG (0x0020)

(reserved)								UART_TXFIFO_RST UART_RXFIFO_RST UART_SW_RTS UART_MEM_RTS UART_AUTOBAUD_EN UART_ERR_WDR_MASK UART_TXD_INV UART_RXD_INV UART_IRDA_EN UART_TX_FLOW_EN UART_LOOPBACK UART_IRDA_RX_INV UART_IRDA_TX_INV UART_IRDA_WCTL UART_IRDA_TX_EN UART_TXD_DPLX UART_STOP_BIT_NUM UART_BIT_NUM UART_PARITY_EN UART_PARITY																		
								31	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		3	0	0	Reset

UART_PARITY Configures the parity check mode.

0: Even parity

1: Odd parity

(R/W)

UART_PARITY_EN Configures whether or not to enable UART parity check.

0: Disable

1: Enable

(R/W)

UART_BIT_NUM Configures the number of data bits.

0: 5 bits

1: 6 bits

2: 7 bits

3: 8 bits

(R/W)

UART_STOP_BIT_NUM Configures the number of stop bits.

0: Invalid. No effect

1: 1 bit

2: 1.5 bits

3: 2 bits

(R/W)

UART_TXD_BRK Configures whether or not to send NULL characters when finishing data transmission.

0: Not send

1: Send

(R/W)

UART_IRDA_DPLX Configures whether or not to enable IrDA loopback test.

0: Disable

1: Enable

(R/W)

UART_IRDA_TX_EN Configures whether or not to enable the IrDA transmitter.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 25.9. UART_CONFO_SYNC_REG (0x0020)

Continued from the previous page...

UART_IRDA_WCTL Configures the 11th bit of the IrDA transmitter.

0: This bit is 0.

1: This bit is the same as the 10th bit.

(R/W)

UART_IRDA_TX_INV Configures whether or not to invert the level of the IrDA transmitter.

0: Not invert

1: Invert

(R/W)

UART_IRDA_RX_INV Configures whether or not to invert the level of the IrDA receiver.

0: Not invert

1: Invert

(R/W)

UART_LOOPBACK Configures whether or not to enable UART loopback test.

0: Disable

1: Enable

(R/W)

UART_TX_FLOW_EN Configures whether or not to enable flow control for the transmitter.

0: Disable

1: Enable

(R/W)

UART_IRDA_EN Configures whether or not to enable IrDA protocol.

0: Disable

1: Enable

(R/W)

UART_RXD_INV Configures whether or not to invert the level of UART RXD signal.

0: Not invert

1: Invert

(R/W)

UART_TXD_INV Configures whether or not to invert the level of UART TXD signal.

0: Not invert

1: Invert

(R/W)

UART_DIS_RX_DAT_OVF Configures whether or not to disable data overflow detection for the UART receiver.

0: Enable

1: Disable

(R/W)

Continued on the next page...

Register 25.9. UART_CONFO_SYNC_REG (0x0020)

Continued from the previous page...

UART_ERR_WR_MASK Configures whether or not to store the received data with errors into FIFO.

- 0: Store
 - 1: Not store
- (R/W)

UART_AUTOBAUD_EN Configures whether or not to enable baud rate detection.

- 0: Disable
 - 1: Enable
- (R/W)

UART_MEM_CLK_EN Configures whether or not to enable clock gating for UART memory.

- 0: Disable
 - 1: Enable
- (R/W)

UART_SW_RTS Configures the RTS signal used in software flow control.

- 0: The UART transmitter is allowed to send data.
 - 1: The UART transmitted is not allowed to send data.
- (R/W)

UART_RXFIFO_RST Configures whether or not to reset the UART RX FIFO.

- 0: Not reset
 - 1: Reset
- (R/W)

UART_TXFIFO_RST Configures whether or not to reset the UART TX FIFO.

- 0: Not reset
 - 1: Reset
- (R/W)

Register 25.10. UART_CONF1_REG (0x0024)

(reserved)										UART_CLK_EN UART_SW_DTR UART_DTR_INV UART_RTS_INV UART_DSR_INV UART_CTS_INV						UART_TXFIFO_EMPTY_THRHD								UART_RXFIFO_FULL_THRHD													
31											22	21	20	19	18	17	16	15									8	7									0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x60								0x60								Reset					

UART_RXFIFO_FULL_THRHD Configures the threshold for RX FIFO being full.
Measurement unit: byte. (R/W)

UART_TXFIFO_EMPTY_THRHD Configures the threshold for TX FIFO being empty.
Measurement unit: byte. (R/W)

UART_CTS_INV Configures whether or not to invert the level of UART CTS signal.
0: Not invert
1: Invert
(R/W)

UART_DSR_INV Configures whether or not to invert the level of UART DSR signal.
0: Not invert
1: Invert
(R/W)

UART_RTS_INV Configures whether or not to invert the level of UART RTS signal.
0: Not invert
1: Invert
(R/W)

UART_DTR_INV Configures whether or not to invert the level of UART DTR signal.
0: Not invert
1: Invert
(R/W)

UART_SW_DTR Configures the DTR signal used in software flow control.
0: Data to be transmitted is not ready.
1: Data to be transmitted is ready.
(R/W)

UART_CLK_EN Configures clock gating.
0: Support clock only when the application writes registers.
1: Always force the clock on for registers.
(R/W)

Register 25.11. UART_HWFC_CONF_SYNC_REG (0x002C)

(reserved)																												UART_RX_FLOW_EN		UART_RX_FLOW_THRHD		
31																												9	8	7	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0	0x0		Reset	

Reset

UART_RX_FLOW_THRHD Configures the maximum number of data bytes that can be received during hardware flow control.

Measurement unit: byte. (R/W)

UART_RX_FLOW_EN Configures whether or not to enable the UART receiver.

0: Disable

1: Enable

(R/W)

Register 25.12. UART_SLEEP_CONFO_REG (0x0030)

UART_WK_CHAR4								UART_WK_CHAR3								UART_WK_CHAR2								UART_WK_CHAR1							
31									24	23									16	15									8	7	0
0x0								0x0								0x0								0x0							

Reset

UART_WK_CHAR1 Configures wakeup character 1. (R/W)

UART_WK_CHAR2 Configures wakeup character 2. (R/W)

UART_WK_CHAR3 Configures wakeup character 3. (R/W)

UART_WK_CHAR4 Configures wakeup character 4. (R/W)

Register 25.13. UART_SLEEP_CONF1_REG (0x0034)

(reserved)																												UART_WK_CHAR0									
31																												8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0x0								Reset	

Reset

UART_WK_CHAR0 Configures wakeup character 0. (R/W)

Register 25.14. UART_SLEEP_CONF2_REG (0x0038)

(reserved)				UART_WK_MODE_SEL				UART_WK_CHAR_MASK				UART_WK_CHAR_NUM				UART_RX_WAKE_UP_THRHD				UART_ACTIVE_THRESHOLD			
31	28	27	26	25	21	20	18	17	10	9	0												
0	0	0	0	0	0x0				0x5				1				0xf0				Reset		

Reset

UART_ACTIVE_THRESHOLD Configures the number of RXD edge changes to wake up the chip in wakeup mode 0. (R/W)

UART_RX_WAKE_UP_THRHD Configures the number of received data bytes to wake up the chip in wakeup mode 1. (R/W)

UART_WK_CHAR_NUM Configures the number of wakeup characters. (R/W)

UART_WK_CHAR_MASK Configures whether or not to mask wakeup characters.

0: Not mask

1: Mask

(R/W)

UART_WK_MODE_SEL Configures which wakeup mode to select. See Section [25.4.8 Wakeup](#) for the explanation of each mode.

0: Mode 0

1: Mode 1

2: Mode 2

3: Mode 3

(R/W)

Register 25.15. UART_SWFC_CONFO_SYNC_REG (0x003C)

(reserved)																UART_SEND_XOFF				UART_SEND_XON				UART_FORCE_XOFF				UART_FORCE_XON				UART_XONOFF_DEL				UART_SW_FLOW_CON_EN				UART_XON_XOFF_STILL_SEND				UART_XOFF_CHAR				UART_XON_CHAR			
31								23	22	21	20	19	18	17	16	15								8	7								0																		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x13							0x11							Reset																					

UART_XON_CHAR Configures the XON character for flow control. (R/W)

UART_XOFF_CHAR Configures the XOFF character for flow control. (R/W)

UART_XON_XOFF_STILL_SEND Configures whether the UART transmitter can send XON or XOFF characters when it is disabled.

0: Cannot send

1: Can send

(R/W)

UART_SW_FLOW_CON_EN Configures whether or not to enable software flow control.

0: Disable

1: Enable

(R/W)

UART_XONOFF_DEL Configures whether or not to remove flow control characters from the received data.

0: Not move

1: Move

(R/W)

UART_FORCE_XON Configures whether the transmitter continues to sending data.

0: Not send

1: Send

(R/W)

UART_FORCE_XOFF Configures whether or not to stop the transmitter from sending data.

0: Not stop

1: Stop

(R/W)

UART_SEND_XON Configures whether or not to send XON characters.

0: Not send

1: Send

(R/W/SS/SC)

UART_SEND_XOFF Configures whether or not to send XOFF characters.

0: Not send

1: Send

(R/W/SS/SC)

Register 25.16. UART_SWFC_CONF1_REG (0x0040)

(reserved)																UART_XOFF_THRESHOLD								UART_XON_THRESHOLD										
31																16	15								8	7								0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0xe0								0x0								Reset		

UART_XON_THRESHOLD Configures the threshold for data in RX FIFO to send XON characters in software flow control.

Measurement unit: byte. (R/W)

UART_XOFF_THRESHOLD Configures the threshold for data in RX FIFO to send XOFF characters in software flow control.

Measurement unit: byte. (R/W)

Register 25.17. UART_TXBRK_CONF_SYNC_REG (0x0044)

(reserved)																								UART_TX_BRK_NUM									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								Oxa								Reset	

UART_TX_BRK_NUM Configures the number of NULL characters to be sent after finishing data transmission.

Valid only when UART_TXD_BRK is 1. (R/W)

Register 25.18. UART_IDLE_CONF_SYNC_REG (0x0048)

(reserved)												UART_TX_IDLE_NUM										UART_RX_IDLE_THRHD																																																	
31												20												19												10												9												0											
0 0 0 0 0 0 0 0 0 0 0 0												0x100												0x100												Reset																																			

UART_RX_IDLE_THRHD Configures the threshold to generate a frame end signal when the receiver takes more time to receive one data byte data.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

UART_TX_IDLE_NUM Configures the interval between two data transfers.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.19. UART_RS485_CONF_SYNC_REG (0x004C)

(reserved)																						UART_RS485_TX_DLY_NUM								UART_RS485_RX_DLY_NUM								UART_RS485RXBY_TX_EN								UART_RS485TX_RX_EN								UART_DL1_EN								UART_DLO_EN								UART_RS485_EN							
31										10										9		6		5		4		3		2		1		0		Reset																																									
0										0										0		0		0		0		0		0		0		0		0																																									

UART_RS485_EN Configures whether or not to enable RS485 mode.

0: Disable

1: Enable

(R/W)

UART_DLO_EN Configures whether or not to add a turnaround delay of 1 bit before the start bit.

0: Not add

1: Add

(R/W)

UART_DL1_EN Configures whether or not to add a turnaround delay of 1 bit after the stop bit.

0: Not add

1: Add

(R/W)

UART_RS485TX_RX_EN Configures whether or not to enable the receiver for data reception when the transmitter is transmitting data in RS485 mode.

0: Disable

1: Enable

(R/W)

UART_RS485RXBY_TX_EN Configures whether to enable the RS485 transmitter for data transmission when the RS485 receiver is busy.

0: Disable

1: Enable

(R/W)

UART_RS485_RX_DLY_NUM Configures the delay of internal data signals in the receiver.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

UART_RS485_TX_DLY_NUM Configures the delay of internal data signals in the transmitter.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.20. UART_CLK_CONF_REG (0x0088)

[illegible]

UART_TX_SCLK_EN Configures whether or not to enable UART TX clock.

0: Disable

1: Enable

(R/W)

UART_RX_SCLK_EN Configures whether or not to enable UART RX clock.

0: Disable

1: Enable

(R/W)

UART_TX_RST_CORE Write 1 and then write 0 to reset UART TX. (R/W)

UART_RX_RST_CORE Write 1 and then write 0 to reset UART RX. (R/W)

Register 25.21. UART_STATUS_REG (0x001C)

[illegible]

UART_RXFIFO_CNT Represents the number of valid data bytes in RX FIFO. (RO)

UART_DSRN Represents the level of the internal UART DSR signal. (RO)

UART_CTSN Represents the level of the internal UART CTS signal. (RO)

UART_RXD Represents the level of the internal UART RXD signal. (RO)

UART_TXFIFO_CNT Represents the number of valid data bytes in RX FIFO. (RO)

UART_DTRN Represents the level of the internal UART DTR signal. (RO)

UART_RTSN Represents the level of the internal UART RTS signal. (RO)

UART_TXD Represents the level of the internal UART TXD signal. (RO)

Register 25.22. UART_MEM_TX_STATUS_REG (0x0068)

(reserved)																UART_TX_SRAM_RADDR								(reserved)								UART_TX_SRAM_WADDR																																																															
31																17																16								9								8								7								0																															
0																0																0								0								0								0								0x0								0								0x0								Reset							

UART_TX_SRAM_WADDR Represents the offset address to write TX FIFO. (RO)

UART_TX_SRAM_RADDR Represents the offset address to read TX FIFO. (RO)

Register 25.23. UART_MEM_RX_STATUS_REG (0x006C)

(reserved)																UART_RX_SRAM_WADDR								(reserved)								UART_RX_SRAM_RADDR																																							
31																17																16								9								8								7								0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x80																0								0x80								Reset																							

UART_RX_SRAM_RADDR Represents the offset address to read RX FIFO. (RO)

UART_RX_SRAM_WADDR Represents the offset address to write RX FIFO. (RO)

Register 25.24. UART_FSM_STATUS_REG (0x0070)

(reserved)																								UART_ST_UTX_OUT				UART_ST_URX_OUT																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31																								8	7	4				3	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UART_ST_URX_OUT Represents the status of the receiver. (RO)

UART_ST_UTX_OUT Represents the status of the transmitter. (RO)

Register 25.25. UART_AFIFO_STATUS_REG (0x0090)

(reserved)																UART_RX_AFIFO_EMPTY UART_RX_AFIFO_FULL UART_TX_AFIFO_EMPTY UART_TX_AFIFO_FULL			
31													4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
																0	1	0	0

Reset

UART_TX_AFIFO_FULL Represents whether or not the APB TX asynchronous FIFO is full.

0: Not full

1: Full

(RO)

UART_TX_AFIFO_EMPTY Represents whether or not the APB TX asynchronous FIFO is empty.

0: Not empty

1: Empty

(RO)

UART_RX_AFIFO_FULL Represents whether or not the APB RX asynchronous FIFO is full.

0: Not full

1: Full

(RO)

UART_RX_AFIFO_EMPTY Represents whether or not the APB RX asynchronous FIFO is empty.

0: Not empty

1: Empty

(RO)

Register 25.26. UART_AT_CMD_PRECNT_SYNC_REG (0x0050)

(reserved)																UART_PRE_IDLE_NUM																															
31																15																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x901																Reset															

Reset

UART_PRE_IDLE_NUM Configures the idle time before the receiver receives the first AT_CMD.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.27. UART_AT_CMD_POSTCNT_SYNC_REG (0x0054)

(reserved)																UART_POST_IDLE_NUM																															
31																15																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x901																Reset															

UART_POST_IDLE_NUM Configures the interval between the last AT_CMD and subsequent data.
Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.28. UART_AT_CMD_GAPTOUT_SYNC_REG (0x0058)

[illegible]

UART_RX_GAP_TOUT Configures the interval between two AT_CMD characters.
Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.29. UART_AT_CMD_CHAR_SYNC_REG (0x005C)

(reserved)																UART_CHAR_NUM								UART_AT_CMD_CHAR															
31																15								7								0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x3								0x2b								Reset							

UART_AT_CMD_CHAR Configures the AT_CMD character. (R/W)

UART_CHAR_NUM Configures the number of continuous AT_CMD characters a receiver can receive.
(R/W)

Register 25.30. UART_POSPULSE_REG (0x0074)

Diagram illustrating the structure of the UART_CR register. The register is 32 bits wide, divided into two 16-bit halves. The top half (bits 31-16) is labeled "(reserved)". The bottom half (bits 15-0) is labeled "UART_POSEDGE_MIN_CNT" and contains the value "0xfff". A "Reset" label is at the bottom right.

UART_POSEDGE_MIN_CNT Represents the minimal input clock counter value between two positive edges. It is used for baud rate detection. (RO)

Register 25.31. UART_NEGPULSE_REG (0x0078)

(reserved)																UART_NEGEDGE_MIN_CNT															
31																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0fff															
																Reset															

UART_NEGEDGE_MIN_CNT Represents the minimal input clock counter value between two negative edges. It is used for baud rate detection. (RO)

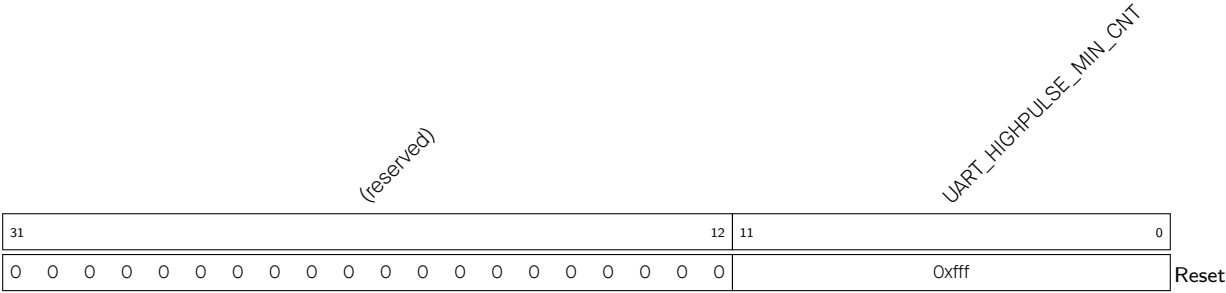
Register 25.32. UART_LOWPULSE_REG (0x007C)

(reserved)																UART_LOWPULSE_MIN_CNT															
31																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0fff															
																Reset															

UART_LOWPULSE_MIN_CNT Represents the minimum duration time of a low-level pulse. It is used for baud rate detection.

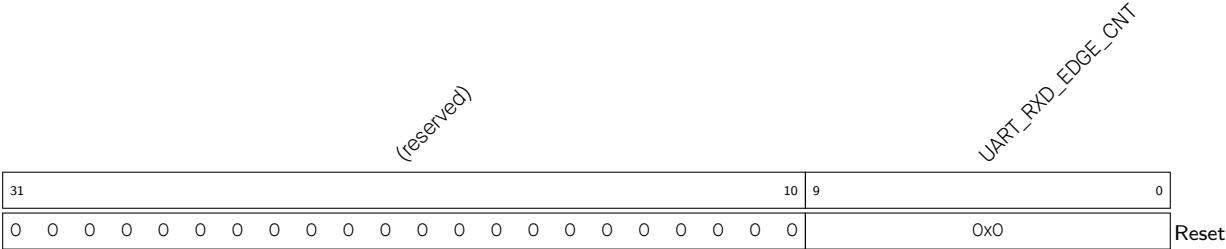
Measurement unit: APB_CLK clock cycle. (RO)

Register 25.33. UART_HIGHPULSE_REG (0x0080)



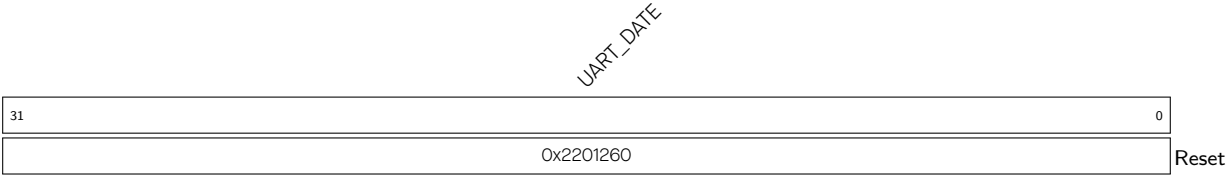
UART_HIGHPULSE_MIN_CNT Represents the maximum duration time for a high-level pulse. It is used for baud rate detection.
Measurement unit: APB_CLK clock cycle. (RO)

Register 25.34. UART_RXD_CNT_REG (0x0084)



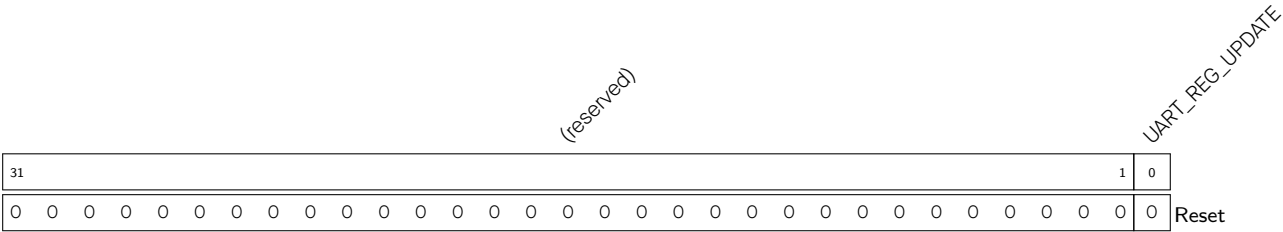
UART_RXD_EDGE_CNT Represents the number of RXD edge changes. It is used for baud rate detection. (RO)

Register 25.35. UART_DATE_REG (0x008C)



UART_DATE Version control register. (R/W)

Register 25.36. UART_REG_UPDATE_REG (0x0098)



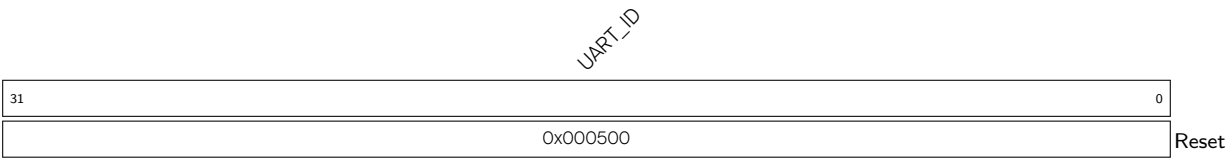
UART_REG_UPDATE Configures whether or not to synchronize registers.

0: Not synchronize

1: Synchronize

(R/W/SC)

Register 25.37. UART_ID_REG (0x009C)



UART_ID Configures the UART ID. (R/W)

Register 25.40. LP_UART_INT_RAW_REG (0x0004)

(reserved)																LP_UART_WAKEUP_INT_RAW		LP_UART_AT_CMD_CHAR_DET_INT_RAW		(reserved)																LP_UART_TX_DONE_INT_RAW		LP_UART_TX_BRK_IDLE_DONE_INT_RAW		LP_UART_TX_BRK_DONE_INT_RAW		LP_UART_GLITCH_DET_INT_RAW		LP_UART_SW_XOFF_INT_RAW		LP_UART_SW_XON_INT_RAW		LP_UART_RXFIFO_TOUT_INT_RAW		LP_UART_BRK_DET_INT_RAW		LP_UART_CTS_CHG_INT_RAW		LP_UART_DSR_CHG_INT_RAW		LP_UART_RXFIFO_OVF_INT_RAW		LP_UART_FRM_ERR_INT_RAW		LP_UART_PARTY_ERR_INT_RAW		LP_UART_TXFIFO_EMPTY_INT_RAW		LP_UART_RXFIFO_FULL_INT_RAW																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
31																20	19	18	17	15		14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_UART_RXFIFO_FULL_INT_RAW The raw interrupt status of LP_UART_RXFIFO_FULL_INT. (R/WTC/SS)

LP_UART_TXFIFO_EMPTY_INT_RAW The raw interrupt status of LP_UART_TXFIFO_EMPTY_INT. (R/WTC/SS)

LP_UART_PARITY_ERR_INT_RAW The raw interrupt status of LP_UART_PARITY_ERR_INT. (R/WTC/SS)

LP_UART_FRM_ERR_INT_RAW The raw interrupt status of LP_UART_FRM_ERR_INT. (R/WTC/SS)

LP_UART_RXFIFO_OVF_INT_RAW The raw interrupt status of LP_UART_RXFIFO_OVF_INT. (R/WTC/SS)

LP_UART_DSR_CHG_INT_RAW The raw interrupt status of LP_UART_DSR_CHG_INT. (R/WTC/SS)

LP_UART_CTS_CHG_INT_RAW The raw interrupt status of LP_UART_CTS_CHG_INT. (R/WTC/SS)

LP_UART_BRK_DET_INT_RAW The raw interrupt status of LP_UART_BRK_DET_INT. (R/WTC/SS)

LP_UART_RXFIFO_TOUT_INT_RAW The raw interrupt status of LP_UART_RXFIFO_TOUT_INT. (R/WTC/SS)

LP_UART_SW_XON_INT_RAW The raw interrupt status of LP_UART_SW_XON_INT. (R/WTC/SS)

LP_UART_SW_XOFF_INT_RAW LP_UART_SW_XOFF_INT. (R/WTC/SS)

LP_UART_GLITCH_DET_INT_RAW The raw interrupt status of LP_UART_GLITCH_DET_INT. (R/WTC/SS)

LP_UART_TX_BRK_DONE_INT_RAW The raw interrupt status of LP_UART_TX_BRK_DONE_INT. (R/WTC/SS)

LP_UART_TX_BRK_IDLE_DONE_INT_RAW The raw interrupt status of LP_UART_TX_BRK_IDLE_DONE_INT. (R/WTC/SS)

Continued on the next page...

Register 25.40. LP_UART_INT_RAW_REG (0x0004)

Continued from the previous page...

LP_UART_TX_DONE_INT_RAW The raw interrupt status of LP_UART_TX_DONE_INT. (R/WTC/SS)

LP_UART_AT_CMD_CHAR_DET_INT_RAW The raw interrupt status of LP_UART_AT_CMD_CHAR_DET_INT. (R/WTC/SS)

LP_UART_WAKEUP_INT_RAW The raw interrupt status of LP_UART_WAKEUP_INT. (R/WTC/SS)

Register 25.41. LP_UART_INT_ST_REG (0x0008)

(reserved)																LP_UART_WAKEUP_INT_ST LP_UART_AT_CMD_CHAR_DET_INT_ST (reserved)																LP_UART_TX_DONE_INT_ST LP_UART_TX_BRK_IDLE_DONE_INT_ST LP_UART_TX_BRK_GLITCH_DET_INT_ST LP_UART_SW_XON_INT_ST LP_UART_SW_XOFF_INT_ST LP_UART_BRK_DET_INT_ST LP_UART_CTS_CHG_INT_ST LP_UART_DSR_CHG_INT_ST LP_UART_RXFIFO_OVF_INT_ST LP_UART_FRM_ERR_INT_ST LP_UART_PARITY_ERR_INT_ST LP_UART_TXFIFO_EMPTY_INT_ST LP_UART_RXFIFO_FULL_INT_ST															
31													20	19	18	17	15		14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset																			

LP_UART_RXFIFO_FULL_INT_ST The masked interrupt status of LP_UART_RXFIFO_FULL_INT. (RO)

LP_UART_TXFIFO_EMPTY_INT_ST The masked interrupt status of LP_UART_TXFIFO_EMPTY_INT. (RO)

LP_UART_PARITY_ERR_INT_ST The masked interrupt status of LP_UART_PARITY_ERR_INT. (RO)

LP_UART_FRM_ERR_INT_ST The masked interrupt status of LP_UART_FRM_ERR_INT. (RO)

LP_UART_RXFIFO_OVF_INT_ST The masked interrupt status of LP_UART_RXFIFO_OVF_INT. (RO)

LP_UART_DSR_CHG_INT_ST The masked interrupt status of LP_UART_DSR_CHG_INT. (RO)

LP_UART_CTS_CHG_INT_ST The masked interrupt status of LP_UART_CTS_CHG_INT. (RO)

LP_UART_BRK_DET_INT_ST The masked interrupt status of LP_UART_BRK_DET_INT. (RO)

LP_UART_RXFIFO_TOUT_INT_ST The masked interrupt status of LP_UART_RXFIFO_TOUT_INT. (RO)

LP_UART_SW_XON_INT_ST The masked interrupt status of LP_UART_SW_XON_INT. (RO)

LP_UART_SW_XOFF_INT_ST The masked interrupt status of LP_UART_SW_XOFF_INT. (RO)

LP_UART_GLITCH_DET_INT_ST The masked interrupt status of LP_UART_GLITCH_DET_INT. (RO)

LP_UART_TX_BRK_DONE_INT_ST The masked interrupt status of LP_UART_TX_BRK_DONE_INT. (RO)

LP_UART_TX_BRK_IDLE_DONE_INT_ST The masked interrupt status of LP_UART_TX_BRK_IDLE_DONE_INT. (RO)

LP_UART_TX_DONE_INT_ST The masked interrupt status of LP_UART_TX_DONE_INT. (RO)

LP_UART_AT_CMD_CHAR_DET_INT_ST The masked interrupt status of LP_UART_AT_CMD_CHAR_DET_INT. (RO)

LP_UART_WAKEUP_INT_ST The masked interrupt status of LP_UART_WAKEUP_INT. (RO)

Register 25.42. LP_UART_INT_ENA_REG (0x000C)

(reserved)																LP_UART_WAKEUP_INT_ENA LP_UART_AT_CMD_CHAR_DET_INT_ENA (reserved)																LP_UART_TX_DONE_INT_ENA LP_UART_TX_BRK_IDLE_DONE_INT_ENA LP_UART_TX_BRK_DONE_INT_ENA LP_UART_GLITCH_DET_INT_ENA LP_UART_SW_XON_INT_ENA LP_UART_SW_XOFF_INT_ENA LP_UART_RXFIFO_TOUT_INT_ENA LP_UART_BRK_DET_INT_ENA LP_UART_CTS_CHG_INT_ENA LP_UART_DSR_CHG_INT_ENA LP_UART_RXFIFO_OVF_INT_ENA LP_UART_FRM_ERR_INT_ENA LP_UART_PARITY_ERR_INT_ENA LP_UART_TXFIFO_EMPTY_INT_ENA LP_UART_RXFIFO_FULL_INT_ENA																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
31													20	19	18	17	15		14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

LP_UART_RXFIFO_FULL_INT_ENA Write 1 to enable LP_UART_RXFIFO_FULL_INT. (R/W)**LP_UART_TXFIFO_EMPTY_INT_ENA** Write 1 to enable LP_UART_TXFIFO_EMPTY_INT. (R/W)**LP_UART_PARITY_ERR_INT_ENA** Write 1 to enable LP_UART_PARITY_ERR_INT. (R/W)**LP_UART_FRM_ERR_INT_ENA** Write 1 to enable LP_UART_FRM_ERR_INT. (R/W)**LP_UART_RXFIFO_OVF_INT_ENA** Write 1 to enable LP_UART_RXFIFO_OVF_INT. (R/W)**LP_UART_DSR_CHG_INT_ENA** Write 1 to enable LP_UART_DSR_CHG_INT. (R/W)**LP_UART_CTS_CHG_INT_ENA** Write 1 to enable LP_UART_CTS_CHG_INT. (R/W)**LP_UART_BRK_DET_INT_ENA** Write 1 to enable LP_UART_BRK_DET_INT. (R/W)**LP_UART_RXFIFO_TOUT_INT_ENA** Write 1 to enable LP_UART_RXFIFO_TOUT_INT. (R/W)**LP_UART_SW_XON_INT_ENA** Write 1 to enable LP_UART_SW_XON_INT. (R/W)**LP_UART_SW_XOFF_INT_ENA** Write 1 to enable LP_UART_SW_XOFF_INT. (R/W)**LP_UART_GLITCH_DET_INT_ENA** Write 1 to enable LP_UART_GLITCH_DET_INT. (R/W)**LP_UART_TX_BRK_DONE_INT_ENA** Write 1 to enable LP_UART_TX_BRK_DONE_INT. (R/W)**LP_UART_TX_BRK_IDLE_DONE_INT_ENA** Write 1 to enable LP_UART_TX_BRK_IDLE_DONE_INT. (R/W)**LP_UART_TX_DONE_INT_ENA** Write 1 to enable LP_UART_TX_DONE_INT. (R/W)**LP_UART_AT_CMD_CHAR_DET_INT_ENA** Write 1 to enable LP_UART_AT_CMD_CHAR_DET_INT. (R/W)**LP_UART_WAKEUP_INT_ENA** Write 1 to enable LP_UART_WAKEUP_INT. (R/W)

Register 25.43. LP_UART_INT_CLR_REG (0x0010)

(reserved)																LP_UART_WAKEUP_INT_CLR LP_UART_AT_CMD_CHAR_DET_INT_CLR (reserved) LP_UART_TX_DONE_INT_CLR LP_UART_TX_BRK_IDLE_DONE_INT_CLR LP_UART_TX_BRK_DONE_INT_CLR LP_UART_GLITCH_DET_INT_CLR LP_UART_SW_XOFF_INT_CLR LP_UART_SW_XON_INT_CLR LP_UART_RXFIFO_TOUT_INT_CLR LP_UART_BRK_DET_INT_CLR LP_UART_CTS_CHG_INT_CLR LP_UART_DSR_CHG_INT_CLR LP_UART_RXFIFO_OVF_INT_CLR LP_UART_FRM_ERR_INT_CLR LP_UART_PARITY_ERR_INT_CLR LP_UART_TXFIFO_EMPTY_INT_CLR LP_UART_RXFIFO_FULL_INT_CLR																		
31													20	19	18	17	15		14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						

LP_UART_RXFIFO_FULL_INT_CLR Write 1 to clear LP_UART_RXFIFO_FULL_INT. (WT)

LP_UART_TXFIFO_EMPTY_INT_CLR Write 1 to clear LP_UART_TXFIFO_EMPTY_INT. (WT)

LP_UART_PARITY_ERR_INT_CLR Write 1 to clear LP_UART_PARITY_ERR_INT. (WT)

LP_UART_FRM_ERR_INT_CLR Write 1 to clear LP_UART_FRM_ERR_INT. (WT)

LP_UART_RXFIFO_OVF_INT_CLR Write 1 to clear LP_UART_RXFIFO_OVF_INT. (WT)

LP_UART_DSR_CHG_INT_CLR Write 1 to clear LP_UART_DSR_CHG_INT. (WT)

LP_UART_CTS_CHG_INT_CLR Write 1 to clear LP_UART_CTS_CHG_INT. (WT)

LP_UART_BRK_DET_INT_CLR Write 1 to clear LP_UART_BRK_DET_INT. (WT)

LP_UART_RXFIFO_TOUT_INT_CLR Write 1 to clear LP_UART_RXFIFO_TOUT_INT. (WT)

LP_UART_SW_XON_INT_CLR Write 1 to clear LP_UART_SW_XON_INT. (WT)

LP_UART_SW_XOFF_INT_CLR Write 1 to clear LP_UART_SW_XOFF_INT. (WT)

LP_UART_GLITCH_DET_INT_CLR Write 1 to clear LP_UART_GLITCH_DET_INT. (WT)

LP_UART_TX_BRK_DONE_INT_CLR Write 1 to clear LP_UART_TX_BRK_DONE_INT. (WT)

LP_UART_TX_BRK_IDLE_DONE_INT_CLR Write 1 to clear LP_UART_TX_BRK_IDLE_DONE_INT. (WT)

LP_UART_TX_DONE_INT_CLR Write 1 to clear LP_UART_TX_DONE_INT. (WT)

LP_UART_AT_CMD_CHAR_DET_INT_CLR Write 1 to clear LP_UART_AT_CMD_CHAR_DET_INT. (WT)

LP_UART_WAKEUP_INT_CLR Write 1 to clear LP_UART_WAKEUP_INT. (WT)

Register 25.44. LP_UART_CLKDIV_SYNC_REG (0x0014)

(reserved)								LP_UART_CLKDIV_FRAG												(reserved)								LP_UART_CLKDIV											
31								24	23					20	19							12	11															0	
0	0	0	0	0	0	0	0	0x0				0	0	0	0	0	0	0	0	0x2b6																Reset			

LP_UART_CLKDIV Configures the integral part of the divisor for baud rate generation. (R/W)

LP_UART_CLKDIV_FRAG Configures the fractional part of the divisor for baud rate generation. (R/W)

Register 25.45. LP_UART_RX_FILT_REG (0x0018)

(reserved)																								LP_UART_GLITCH_FILT_EN		LP_UART_GLITCH_FILT																
31																								9	8	7	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0	0x8																Reset	

LP_UART_GLITCH_FILT Configures the width of a pulse to be filtered.

Measurement unit: UART Core's clock cycle.

Pulses whose width is lower than this value will be ignored. (R/W)

LP_UART_GLITCH_FILT_EN Configures whether or not to enable RX signal filter.

0: Disable

1: Enable

(R/W)

Register 25.46. LP_UART_CONFO_SYNC_REG (0x0020)

(reserved)								LP_UART_TXFIFO_RST LP_UART_RXFIFO_RST LP_UART_SW_RST LP_UART_MEM_CLK_EN (reserved) LP_UART_ERR_WD_MASK LP_UART_DS_RX_DAT_OVF LP_UART_TXD_INV (reserved) LP_UART_RXD_INV LP_UART_TX_FLOW_EN LP_UART_LOOPBACK											(reserved) LP_UART_TXD_BRK LP_UART_STOP_BIT_NUM LP_UART_BIT_NUM LP_UART_PARITY_EN LP_UART_PARITY																
31								24	23	22	21	20	19	18	17	16	15	14	13	12	11							7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						0	0	1	3	0	0	Reset	

LP_UART_PARITY Configures the parity check mode.

0: Even parity

1: Odd parity

(R/W)

LP_UART_PARITY_EN Configures whether or not to enable LP UART parity check.

0: Disable

1: Enable

(R/W)

LP_UART_BIT_NUM Configures the number of data bits.

0: 5 bits

1: 6 bits

2: 7 bits

3: 8 bits

(R/W)

LP_UART_STOP_BIT_NUM Configures the number of stop bits.

0: Invalid. No effect

1: 1 bit

2: 1.5 bits

3: 2 bits

(R/W)

LP_UART_TXD_BRK Configures whether or not to send NULL characters when finishing data transmission.

0: Not send

1: Send

(R/W)

LP_UART_LOOPBACK Configures whether or not to enable LP UART loopback test.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 25.46. LP_UART_CONFO_SYNC_REG (0x0020)

Continued from the previous page...

LP_UART_TX_FLOW_EN Configures whether or not to enable flow control for the transmitter.

0: Disable

1: Enable

(R/W)

LP_UART_RXD_INV Configures whether or not to invert the level of LP UART RXD signal.

0: Not invert

1: Invert

(R/W)

LP_UART_TXD_INV Configures whether or not to invert the level of LP UART TXD signal.

0: Not invert

1: Invert

(R/W)

LP_UART_DIS_RX_DAT_OVF Configures whether or not to disable data overflow detection for the LP UART receiver.

0: Enable

1: Disable

(R/W)

LP_UART_ERR_WR_MASK Configures whether or not to store the received data with errors into FIFO.

0: Store

1: Not store

(R/W)

LP_UART_MEM_CLK_EN Configures whether or not to enable clock gating for LP UART memory.

0: Disable

1: Enable

(R/W)

LP_UART_SW_RTS Configures the RTS signal used in software flow control.

0: The LP UART transmitter is allowed to send data.

1: The LP UART transmitted is not allowed to send data.

(R/W)

LP_UART_RXFIFO_RST Configures whether or not to reset the LP UART RX FIFO.

0: Not reset

1: Reset

(R/W)

LP_UART_TXFIFO_RST Configures whether or not to reset the LP UART TX FIFO.

0: Not reset

1: Reset

(R/W)

Register 25.47. LP_UART_CONF1_REG (0x0024)

(reserved)												LP_UART_CLK_EN				LP_UART_SW_DTR				LP_UART_DTR_INV				LP_UART_RTS_INV				LP_UART_DSR_INV				LP_UART_CTS_INV				LP_UART_TXFIFO_EMPTY_THRHD				(reserved)				LP_UART_RXFIFO_FULL_THRHD				(reserved)			
31												22	21	20	19	18	17	16	15	11				10	8		7	3				2	0																		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0xc				0				0	0	0xc				0				0	0	Reset															

LP_UART_RXFIFO_FULL_THRHD Configures the threshold for RX FIFO being full.

Measurement unit: byte. (R/W)

LP_UART_TXFIFO_EMPTY_THRHD Configures the threshold for TX FIFO being empty.

Measurement unit: byte. (R/W)

LP_UART_CTS_INV Configures whether or not to invert the level of LP UART CTS signal.

0: Not invert

1: Invert

(R/W)

LP_UART_DSR_INV Configures whether or not to invert the level of LP UART DSR signal.

0: Not invert

1: Invert

(R/W)

LP_UART_RTS_INV Configures whether or not to invert the level of LP UART RTS signal.

0: Not invert

1: Invert

(R/W)

LP_UART_DTR_INV Configures whether or not to invert the level of LP UART DTR signal.

0: Not invert

1: Invert

(R/W)

LP_UART_SW_DTR Configures the DTR signal used in software flow control.

0: Data to be transmitted is not ready.

1: Data to be transmitted is ready.

(R/W)

LP_UART_CLK_EN Configures clock gating.

0: Support clock only when the application writes registers.

1: Always force the clock on for registers.

(R/W)

Register 25.48. LP_UART_HWFC_CONF_SYNC_REG (0x002C)

(reserved)																								LP_UART_RX_FLOW_EN		LP_UART_RX_FLOW_THRHD		(reserved)			
31																								9	8	7	3		2	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0	0x0		0 0 0		Reset		

LP_UART_RX_FLOW_THRHD Configures the maximum number of data bytes that can be received during hardware flow control.

Measurement unit: byte. (R/W)

LP_UART_RX_FLOW_EN Configures whether or not to enable the LP UART receiver.

0: Disable

1: Enable

(R/W)

Register 25.49. LP_UART_SLEEP_CONFO_REG (0x0030)

LP_UART_WK_CHAR4								LP_UART_WK_CHAR3								LP_UART_WK_CHAR2								LP_UART_WK_CHAR1																																															
31																	24	23																	16	15																	8	7																	0
0x0								0x0								0x0								0x0								Reset																																							

LP_UART_WK_CHAR1 Configures wakeup character 1. (R/W)

LP_UART_WK_CHAR2 Configures wakeup character 2. (R/W)

LP_UART_WK_CHAR3 Configures wakeup character 3. (R/W)

LP_UART_WK_CHAR4 Configures wakeup character 4. (R/W)

Register 25.50. LP_UART_SLEEP_CONF1_REG (0x0034)

(reserved)																								LP_UART_WK_CHAR0									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0x0								Reset	

LP_UART_WK_CHAR0 Configures wakeup character 0. (R/W)

Register 25.51. LP_UART_SLEEP_CONF2_REG (0x0038)

(reserved)				LP_UART_WK_MODE_SEL				LP_UART_WK_CHAR_MASK				LP_UART_WK_CHAR_NUM				LP_UART_RX_WAKE_UP_THRHD				(reserved)				LP_UART_ACTIVE_THRESHOLD							
31		28		27	26	25		21		20	18		17	13		12	10		9	0											
0		0		0		0		0x0		0x5		1		0		0		0		0xf0										Reset	

Reset

LP_UART_ACTIVE_THRESHOLD Configures the number of RXD edge changes to wake up the chip in wakeup mode 0. (R/W)

LP_UART_RX_WAKE_UP_THRHD Configures the number of received data bytes to wake up the chip in wakeup mode 1. (R/W)

LP_UART_WK_CHAR_NUM Configures the number of wakeup characters. (R/W)

LP_UART_WK_CHAR_MASK Configures whether or not to mask wakeup characters.

0: Not mask

1: Mask

(R/W)

LP_UART_WK_MODE_SEL Configures which wakeup mode to select.

0: Mode 0

1: Mode 1

2: Mode 2

3: Mode 3

(R/W)

Register 25.52. LP_UART_SWFC_CONFO_SYNC_REG (0x003C)

(reserved)										LP_UART_SEND_XOFF										LP_UART_SEND_XON										LP_UART_FORCE_XOFF										LP_UART_FORCE_XON										LP_UART_SW_FLOW_CON_EN										LP_UART_XON_XOFF_STILL_SEND										LP_UART_XOFF_CHAR										LP_UART_XON_CHAR									
31									23	22	21	20	19	18	17	16	15									8	7									0																																																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x13								0x11								Reset																																																										

LP_UART_XON_CHAR Configures the XON character for flow control. (R/W)

LP_UART_XOFF_CHAR Configures the XOFF character for flow control. (R/W)

LP_UART_XON_XOFF_STILL_SEND Configures whether the LP UART transmitter can send XON or XOFF characters when it is disabled.

0: Cannot send

1: Can send

(R/W)

LP_UART_SW_FLOW_CON_EN Configures whether or not to enable software flow control.

0: Disable

1: Enable

(R/W)

LP_UART_XONOFF_DEL Configures whether or not to remove flow control characters from the received data.

0: Not move

1: Move

(R/W)

LP_UART_FORCE_XON Configures whether the transmitter continues to sending data.

0: Not send

1: Send

(R/W)

LP_UART_FORCE_XOFF Configures whether or not to stop the transmitter from sending data.

0: Not stop

1: Stop

(R/W)

LP_UART_SEND_XON Configures whether or not to send XON characters.

0: Not send

1: Send

(R/W/SS/SC)

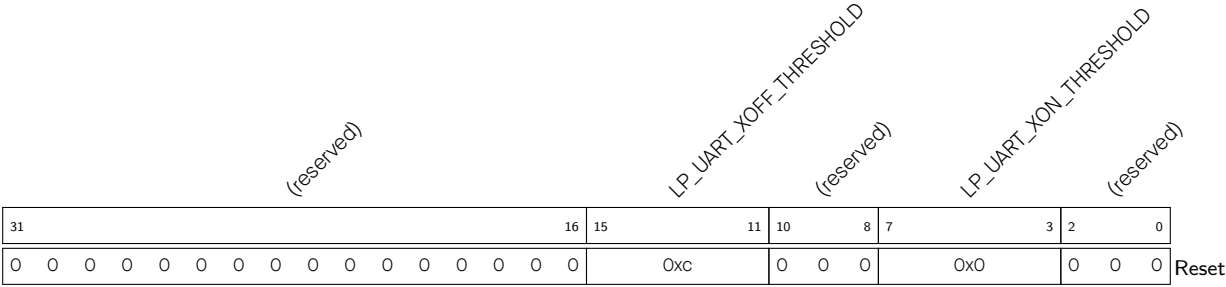
LP_UART_SEND_XOFF Configures whether or not to send XOFF characters.

0: Not send

1: Send

(R/W/SS/SC)

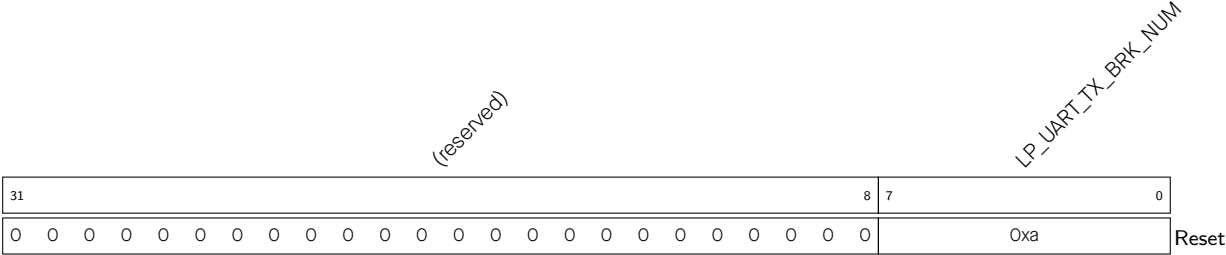
Register 25.53. LP_UART_SWFC_CONF1_REG (0x0040)



LP_UART_XON_THRESHOLD Configures the threshold for data in RX FIFO to send XON characters in software flow control.
Measurement unit: byte. (R/W)

LP_UART_XOFF_THRESHOLD Configures the threshold for data in RX FIFO to send XOFF characters in software flow control.
Measurement unit: byte. (R/W)

Register 25.54. LP_UART_TXBRK_CONF_SYNC_REG (0x0044)



LP_UART_TX_BRK_NUM Configures the number of NULL characters to be sent after finishing data transmission.
Valid only when LP_UART_TXD_BRK is 1. (R/W)

Register 25.55. LP_UART_IDLE_CONF_SYNC_REG (0x0048)

(reserved)												LP_UART_TX_IDLE_NUM												LP_UART_RX_IDLE_THRHD																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31												20												19												10												9												0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0												0											

LP_UART_RX_IDLE_THRHD Configures the threshold to generate a frame end signal when the receiver takes more time to receive one data byte data.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

LP_UART_TX_IDLE_NUM Configures the interval between two data transfers.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.56. LP_UART_DELAY_CONF_SYNC_REG (0x004C)

(reserved)																												LP_UART_DL1_EN LP_UART_DLO_EN (reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
31																												3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LP_UART_DLO_EN Configures whether to add a turnaround delay of 1 bit before the start bit.

0: Not add

1: Add

(R/W)

LP_UART_DL1_EN Configures whether to add a turnaround delay of 1 bit after the stop bit.

0: Not add

1: Add

(R/W)

Register 25.57. LP_UART_CLK_CONF_REG (0x0088)

(reserved)																(reserved)															
31				28		27		26		25		24		23		0															
0				0		0		0		1		1		0																Reset	

LP_UART_TX_SCLK_EN Configures whether or not to enable LP UART TX clock.

0: Disable

1: Enable

(R/W)

LP_UART_RX_SCLK_EN Configures whether or not to enable LP UART RX clock.

0: Disable

1: Enable

(R/W)

LP_UART_TX_RST_CORE Write 1 and then write 0 to reset LP UART TX. (R/W)

LP_UART_RX_RST_CORE Write 1 and then write 0 to reset LP UART RX. (R/W)

Register 25.58. LP_UART_STATUS_REG (0x001C)

[illegible]

LP_UART_RXFIFO_CNT Represents the number of valid data bytes in RX FIFO. (RO)

LP_UART_DSRN Represents the level of the internal LP UART DSR signal. (RO)

LP_UART_CTSN Represents the level of the internal LP UART CTS signal. (RO)

LP_UART_RXD Represents the level of the internal LP UART RXD signal. (RO)

LP_UART_TXFIFO_CNT Represents the number of valid data bytes in RX FIFO. (RO)

LP_UART_DTRN Represents the level of the internal LP UART DTR signal. (RO)

LP_UART_RTSN Represents the level of the internal LP UART RTS signal. (RO)

LP_UART_TXD Represents the level of the internal LP UART TXD signal. (RO)

(reserved)

LP_UART_TX_SRAM_RADDR

(reserved)

LP_UART_TX_SRAM_WADDR

(reserved)

LP_UART_TX_SRAM_RADDR Represents the offset address to read TX FIFO. (RO)

(reserved)

LP_UART_RX_SRAM_WADDR

(reserved)

LP_UART_RX_SRAM_RADDR

(reserved)

LP_UART_RX_SRAM_WADDR Represents the offset address to write RX FIFO. (RO)

(reserved)

LP_UART_ST_UTX_OUT

LP_UART_ST_URX_OUT

LP_UART_ST_UTX_OUT Represents the status of the transmitter. (RO)

Register 25.62. LP_UART_AFIFO_STATUS_REG (0x0090)

(reserved)

31 4 3 2 1 0

LP_UART_RX_AFIFO_EMPTY
LP_UART_RX_AFIFO_FULL
LP_UART_TX_AFIFO_EMPTY
LP_UART_TX_AFIFO_FULL

Reset

LP_UART_TX_AFIFO_FULL Represents whether or not the APB TX asynchronous FIFO is full.

0: Not full

1: Full

(RO)

LP_UART_TX_AFIFO_EMPTY Represents whether or not the APB TX asynchronous FIFO is empty.

0: Not empty

1: Empty

(RO)

LP_UART_RX_AFIFO_FULL Represents whether or not the APB RX asynchronous FIFO is full.

0: Not full

1: Full

(RO)

LP_UART_RX_AFIFO_EMPTY Represents whether or not the APB RX asynchronous FIFO is empty.

0: Not empty

1: Empty

(RO)

Register 25.63. LP_UART_AT_CMD_PRECNT_SYNC_REG (0x0050)

31	16	15	0
(reserved)		LP_UART_PRE_IDLE_NUM	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0x901	

Reset

LP_UART_PRE_IDLE_NUM Configures the idle time before the receiver receives the first AT_CMD.

Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.64. LP_UART_AT_CMD_POSTCNT_SYNC_REG (0x0054)

(reserved)																LP_UART_POST_IDLE_NUM																															
31																15																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x901																Reset															

LP_UART_POST_IDLE_NUM Configures the interval between the last AT_CMD and subsequent data.
Measurement unit: bit time (the time to transmit 1 bit). (R/W)

Register 25.65. LP_UART_AT_CMD_GAPTOUT_SYNC_REG (0x0058)

(reserved)																LP_UART_RX_GAP_TOUT																															
31																15																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																11																Reset															

LP_UART_RX_GAP_TOUT Configures the interval between two AT_CMD characters.
Measurement unit: bit time (the time to transmit 1 bit). (R/W)

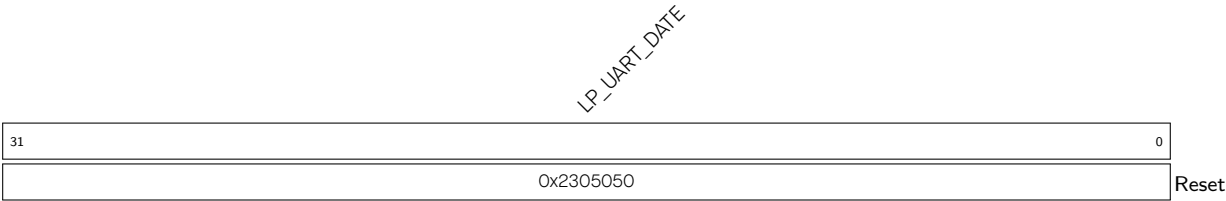
Register 25.66. LP_UART_AT_CMD_CHAR_SYNC_REG (0x005C)

(reserved)																LP_UART_CHAR_NUM								LP_UART_AT_CMD_CHAR										
31																16	15								8	7								0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x3								0x2b									Reset	

LP_UART_AT_CMD_CHAR Configures the AT_CMD character. (R/W)

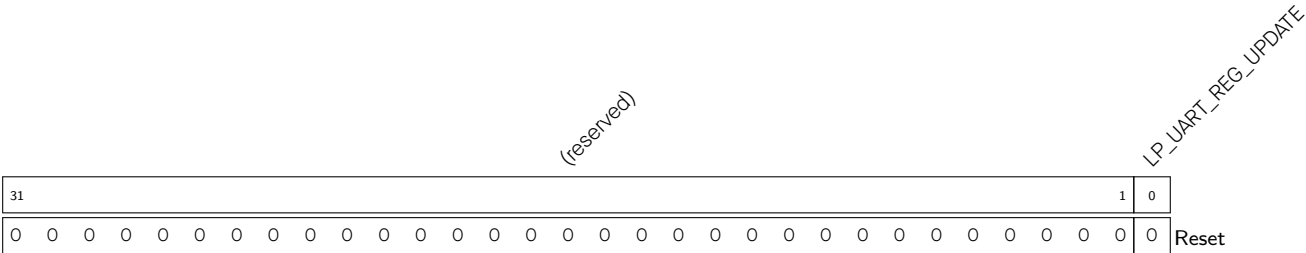
LP_UART_CHAR_NUM Configures the number of continuous AT_CMD characters a receiver can receive. (R/W)

Register 25.67. LP_UART_DATE_REG (0x008C)



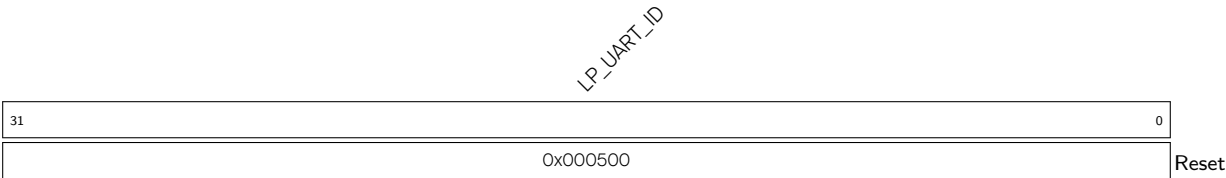
LP_UART_DATE Version control register. (R/W)

Register 25.68. LP_UART_REG_UPDATE_REG (0x0098)



LP_UART_REG_UPDATE Configures whether or not to synchronize registers.
0: Not synchronize
1: Synchronize
(R/W/SC)

Register 25.69. LP_UART_ID_REG (0x009C)



LP_UART_ID Configures the LP UART ID. (R/W)

25.7.3 UHCI Registers

The addresses in this section are relative to UHCI base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 25.70. UHCI_CONFO_REG (0x0000)

(reserved)																				UHCI_UART_RX_BRK_EOF_EN										UHCI_UART_CLK_EN										UHCI_ENCODE_CRC_EN										UHCI_LEN_EOF_EN										UHCI_UART_IDLE_EOF_EN										UHCI_CRC_REC_EN										UHCI_SEPER_EN										UHCI_UART_SEL										UHCI_RX_RST										UHCI_TX_RST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
31																				13										12	11	10	9	8	7	6	5	4	2										1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

UHCI_TX_RST Write 1 and then write 0 to reset the decoder state machine. (R/W)

UHCI_RX_RST Write 1 and then write 0 to reset the encoder state machine. (R/W)

UHCI_UART_SEL Select one UART from UART 0 ~ 1 to connect with UHCI.

0: Select UART0

1: Select UART1

2 ~ 7: No effect, as no UART will connect with UHCI

(R/W)

UHCI_UART1_CE Configures whether or not to connect UHCI with UART1.

0: Not connect

1: Connect

(R/W)

UHCI_SEPER_EN Configures whether or not to separate the data frame with a special character.

0: Not separate

1: Separate

(R/W)

UHCI_HEAD_EN Configures whether or not to encode the data packet with a formatting header.

0: Not use formatting header

1: Use formatting header

(R/W)

Continued on the next page...

Register 25.70. UHCI_CONFO_REG (0x0000)

Continued from the previous page...

UHCI_CRC_REC_EN Configures whether or not to enable the reception of the 16-bit CRC.

0: Disable

1: Enable

(R/W)

UHCI_UART_IDLE_EOF_EN Configures whether or not to stop receiving data when UART is idle.

0: Not stop

1: Stop

(R/W)

UHCI_LEN_EOF_EN Configures when the UHCI decoder stops receiving data.

0: Stops after receiving 0xC0

1: Stops when the number of received data bytes reach the specified value. When UHCI_HEAD_EN is 1, the specified value is the data length indicated by the UHCI packet header; when UHCI_HEAD_EN is 0, the specified value is the configured value.

(R/W)

UHCI_ENCODE_CRC_EN Configures whether or not to enable data integrity check by appending a 16 bit CCITT-CRC to the end of the data.

0: Disable

1: Enable

(R/W)

UHCI_CLK_EN Configures clock gating.

0: Support clock only when the application writes registers.

1: Always force the clock on for registers.

(R/W)

UHCI_UART_RX_BRK_EOF_EN Configures whether or not to stop UHCI from receiving data after UART has received a NULL frame.

0: Not stop

1: Stop

(R/W)

Register 25.71. UHCI_CONF1_REG (0x0014)

(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31																												9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

UHCI_CHECK_SUM_EN Configures whether or not to enable header checksum validation when UHCI receives a data packet.

0: Disable

1: Enable

(R/W)

UHCI_CHECK_SEQ_EN Configures whether or not to enable the sequence number check when UHCI receives a data packet.

0: Disable

1: Enable

(R/W)

UHCI_CRC_DISABLE Configures whether or not to enable CRC calculation.

0: Disable

1: Enable

Valid only when the Data Integrity Check Present bit in UHCI packet is 1.

(R/W)

UHCI_SAVE_HEAD Configures whether or not to save the packet header when UHCI receives a data packet.

0: Not save

1: Save

(R/W)

UHCI_TX_CHECK_SUM_RE Configures whether or not to encode the data packet with a checksum.

0: Not use checksum

1: Use checksum

(R/W)

UHCI_TX_ACK_NUM_RE Configures whether or not to encode the data packet with an acknowledgment when a reliable packet is to be transmitted.

0: Not use acknowledgement

1: Use acknowledgement

(R/W)

Continued on the next page...

Register 25.71. UHCI_CONF1_REG (0x0014)

Continued from the previous page...

UHCI_WAIT_SW_START Configures whether or not to put the UHCI encoder state machine to ST_SW_WAIT state.

0: No

1: Yes

(R/W)

UHCI_SW_START Configures whether or not to send data packets when the encoder state machine is in ST_SW_WAIT state.

0: Not send

1: Send

(R/W/SC)

Register 25.72. UHCI_ESCAPE_CONF_REG (0x0020)

[illegible]

UHCI_TX_CO_ESC_EN Configures whether or not to decode character 0xC0 when DMA receives data.

0: Not decode

1: Decode

(R/W)

UHCI_TX_DB_ESC_EN Configures whether or not to decode character 0xDB when DMA receives data.

0: Not decode

1: Decode

(R/W)

UHCI_TX_11_ESC_EN Configures whether or not to decode flow control character 0x11 when DMA receives data.

0: Not decode

1: Decode

(R/W)

UHCI_TX_13_ESC_EN	Configures whether or not to decode flow control character 0x13 when DMA receives data.

0: Not decode

1: Decode

(R/W)

UHCI_RX_CO_ESC_EN Configures whether or not to replace 0xC0 by special characters when DMA sends data.

0: Not replace

1: Replace

(R/W)

UHCI_RX_DB_ESC_EN Configures whether or not to replace 0xDB by special characters when DMA sends data.

0: Not replace

1: Replace

(R/W)

Continued on the next page...

Register 25.72. UHCI_ESCAPE_CONF_REG (0x0020)

Continued from the previous page...

UHCI_RX_11_ESC_EN Configures whether or not to replace flow control character 0x11 by special characters when DMA sends data.

0: Not replace

1: Replace

(R/W)

UHCI_RX_13_ESC_EN Configures whether or not to replace flow control character 0x13 by special characters when DMA sends data.

0: Not replace

1: Replace

(R/W)

Register 25.73. UHCI_HUNG_CONF_REG (0x0024)

(reserved)								UHCI_RXFIFO_TIMEOUT_ENA				UHCI_RXFIFO_TIMEOUT_SHIFT				UHCI_RXFIFO_TIMEOUT				UHCI_TXFIFO_TIMEOUT_ENA				UHCI_TXFIFO_TIMEOUT_SHIFT				UHCI_TXFIFO_TIMEOUT			
31								24	23	22	20	19	12	11	10	8	7	0													
0 0 0 0 0 0 0 0								1	0	0x10				1	0	0x10								Reset							

Reset

UHCI_TXFIFO_TIMEOUT Configures the timeout value for DMA data reception.

Measurement unit: ms. (R/W)

UHCI_TXFIFO_TIMEOUT_SHIFT Configures the upper limit of the timeout counter for TX FIFO. (R/W)

UHCI_TXFIFO_TIMEOUT_ENA Configures whether or not to enable the data reception timeout for TX FIFO.

0: Disable

1: Enable

(R/W)

UHCI_RXFIFO_TIMEOUT Configures the timeout value for DMA to read data from RAM.

Measurement unit: ms. (R/W)

UHCI_RXFIFO_TIMEOUT_SHIFT Configures the upper limit of the timeout counter for RX FIFO. (R/W)

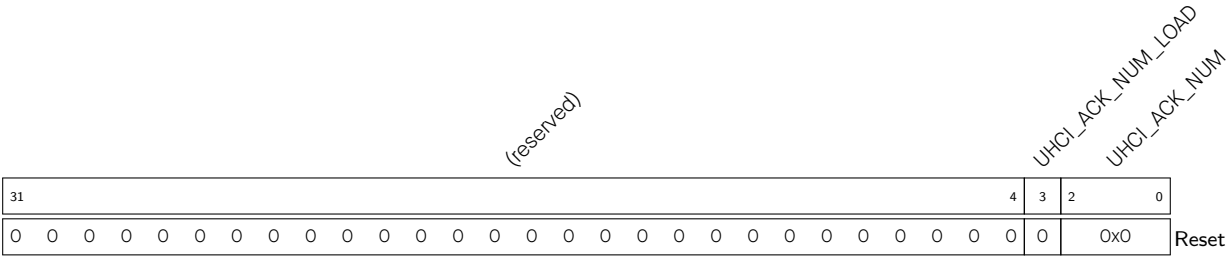
UHCI_RXFIFO_TIMEOUT_ENA Configures whether or not to enable the DMA data transmission timeout.

0: Disable

1: Enable

(R/W)

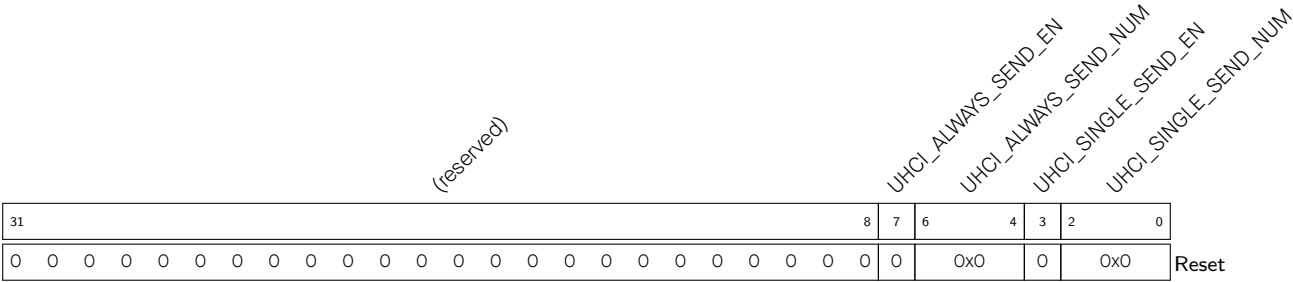
Register 25.74. UHCI_ACK_NUM_REG (0x0028)



UHCI_ACK_NUM Configures the number of acknowledgements used in software flow control.
(R/W)

UHCI_ACK_NUM_LOAD Configures whether or not load acknowledgements.
0: Not load
1: Load
(WT)

Register 25.75. UHCI_QUICK_SENT_REG (0x0030)



UHCI_SINGLE_SEND_NUM Configures the source of data to be transmitted in single_send mode.

- 0: Q0 register
- 1: Q1 register
- 2: Q2 register
- 3: Q3 register
- 4: Q4 register
- 5: Q5 register
- 6: Q6 register
- 7: Invalid. No effect
- (R/W)

UHCI_SINGLE_SEND_EN Configures whether or not to enable single_send mode.

- 0: Disable
- 1: Enable
- (R/W/SC)

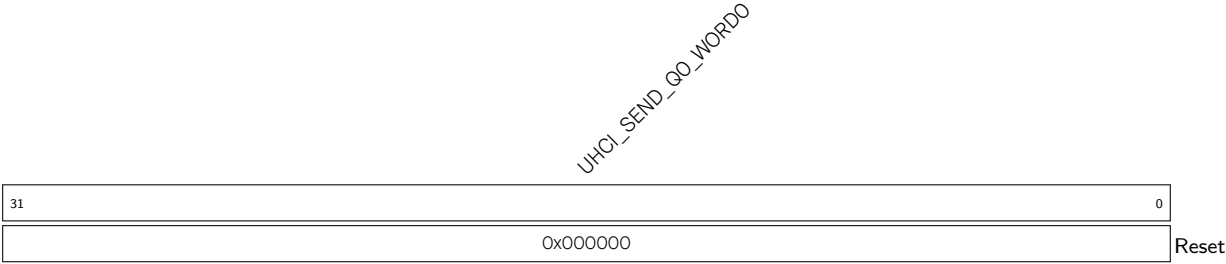
UHCI_ALWAYS_SEND_NUM Configures the source of data to be transmitted in always_send mode.

- 0: Q0 register
- 1: Q1 register
- 2: Q2 register
- 3: Q3 register
- 4: Q4 register
- 5: Q5 register
- 6: Q6 register
- 7: Invalid. No effect
- (R/W)

UHCI_ALWAYS_SEND_EN Configures whether or not to enable always_send mode.

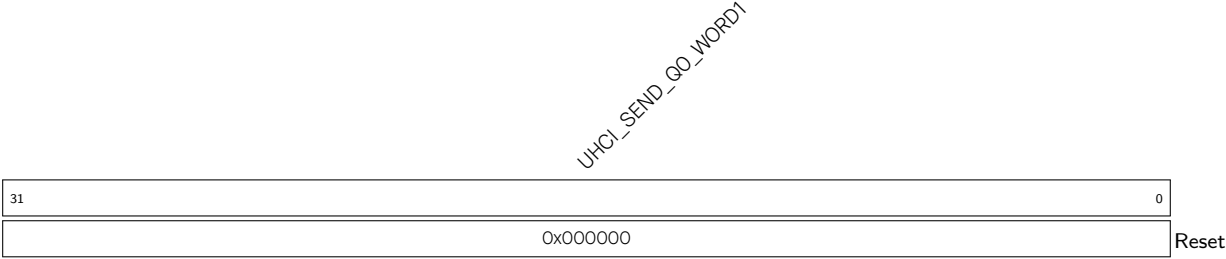
- 0: Disable
- 1: Enable
- (R/W)

Register 25.76. UHCI_REG_Q0_WORD0_REG (0x0034)



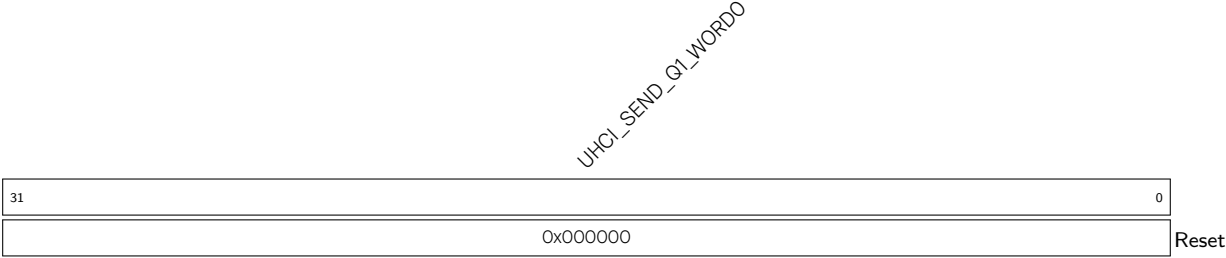
UHCI_SEND_Q0_WORD0 Data to be transmitted in Q0 register. (R/W)

Register 25.77. UHCI_REG_Q0_WORD1_REG (0x0038)

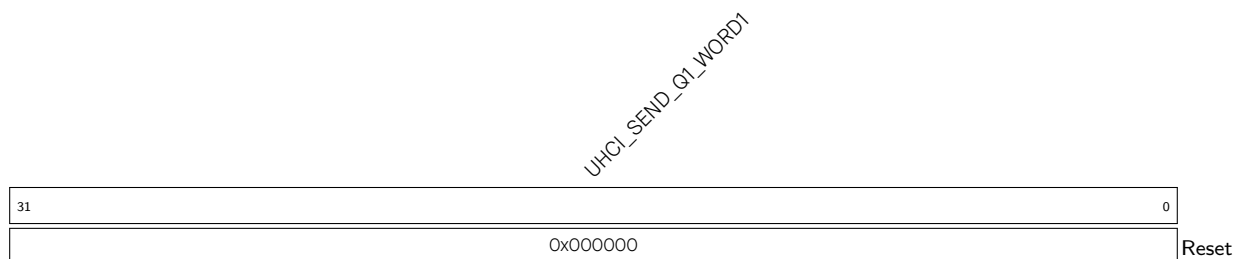


UHCI_SEND_Q0_WORD1 Data to be transmitted in Q0 register. (R/W)

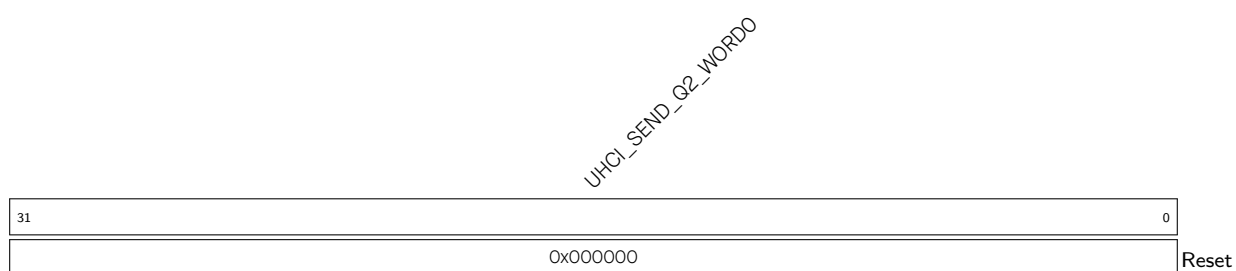
Register 25.78. UHCI_REG_Q1_WORD0_REG (0x003C)



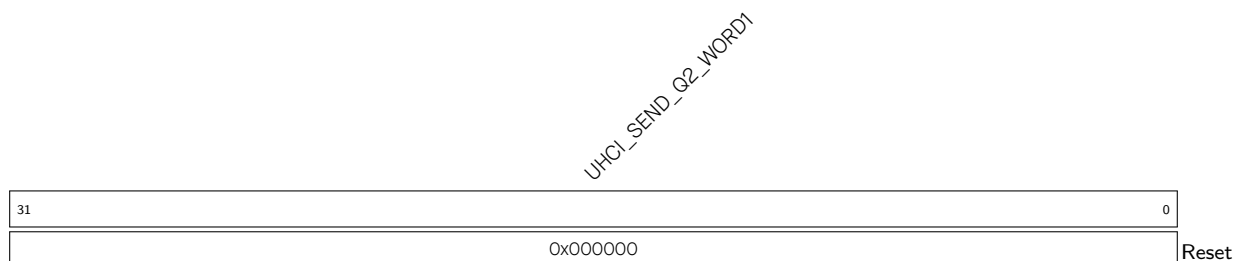
UHCI_SEND_Q1_WORD0 Data to be transmitted in Q1 register. (R/W)

Register 25.79. UHCI_REG_Q1_WORD1_REG (0x0040)

UHCI_SEND_Q1_WORD1 Data to be transmitted in Q1 register. (R/W)

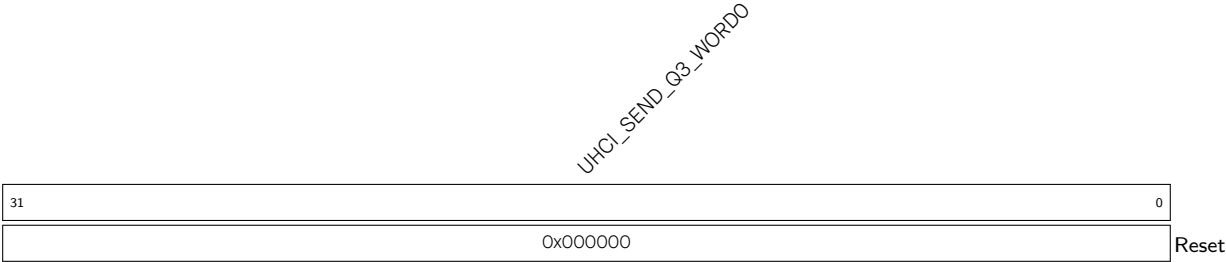
Register 25.80. UHCI_REG_Q2_WORD0_REG (0x0044)

UHCI_SEND_Q2_WORD0 Data to be transmitted in Q2 register. (R/W)

Register 25.81. UHCI_REG_Q2_WORD1_REG (0x0048)

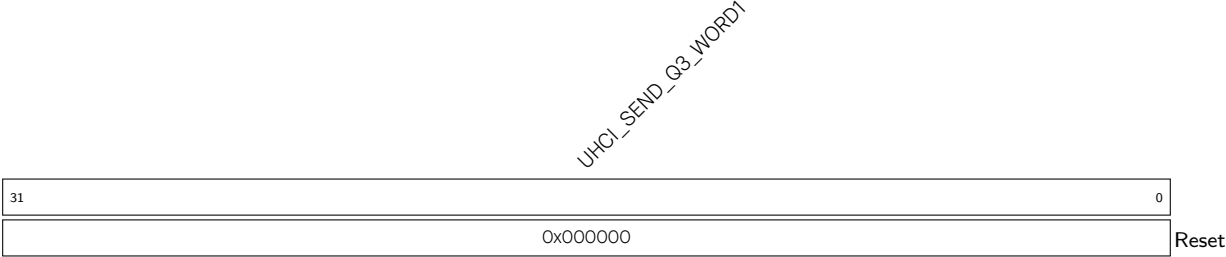
UHCI_SEND_Q2_WORD1 Data to be transmitted in Q2 register. (R/W)

Register 25.82. UHCI_REG_Q3_WORD0_REG (0x004C)



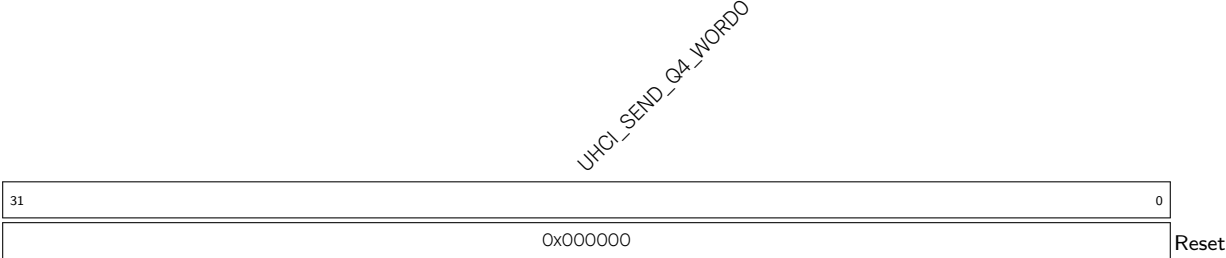
UHCI_SEND_Q3_WORD0 Data to be transmitted in Q3 register. (R/W)

Register 25.83. UHCI_REG_Q3_WORD1_REG (0x0050)



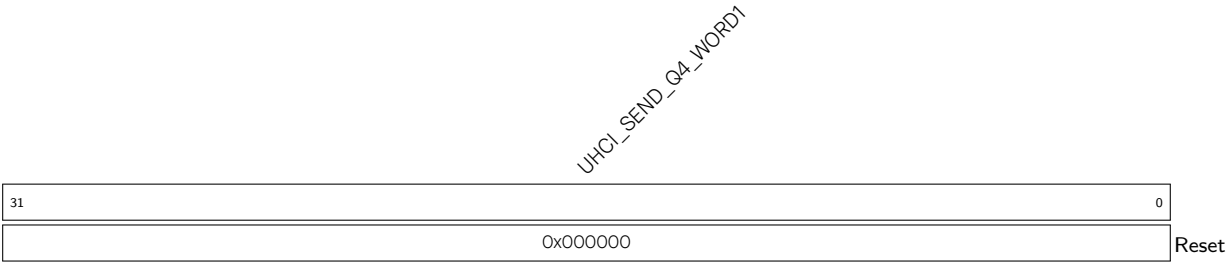
UHCI_SEND_Q3_WORD1 Data to be transmitted in Q3 register. (R/W)

Register 25.84. UHCI_REG_Q4_WORD0_REG (0x0054)



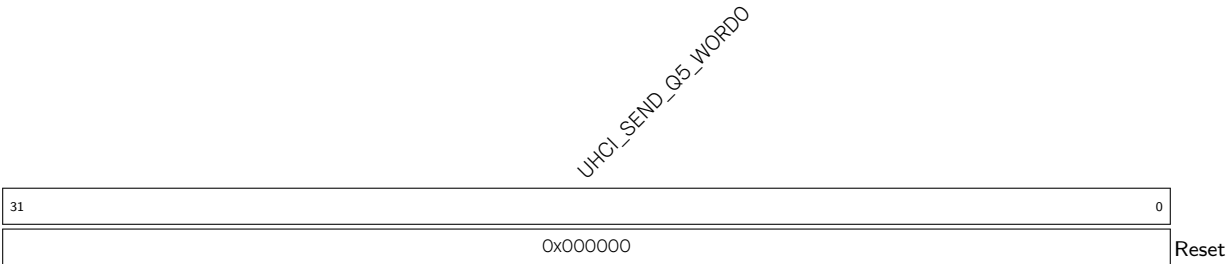
UHCI_SEND_Q4_WORD0 Data to be transmitted in Q4 register. (R/W)

Register 25.85. UHCI_REG_Q4_WORD1_REG (0x0058)



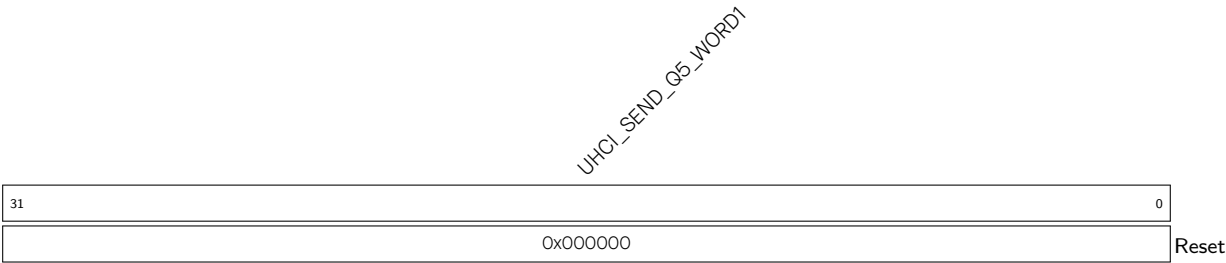
UHCI_SEND_Q4_WORD1 Data to be transmitted in Q4 register. (R/W)

Register 25.86. UHCI_REG_Q5_WORD0_REG (0x005C)



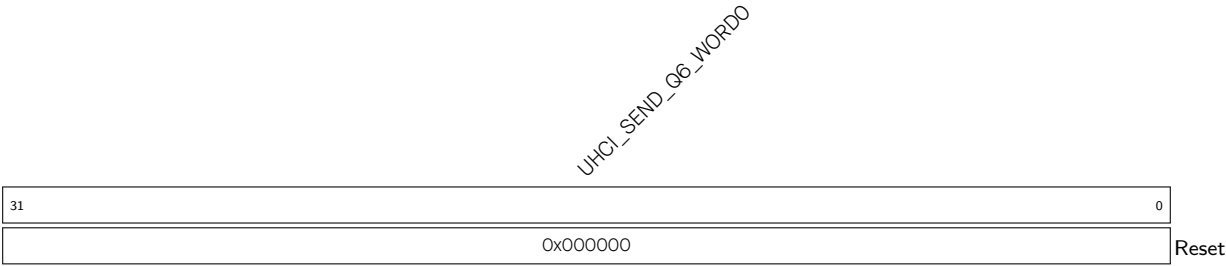
UHCI_SEND_Q5_WORD0 Data to be transmitted in Q5 register. (R/W)

Register 25.87. UHCI_REG_Q5_WORD1_REG (0x0060)



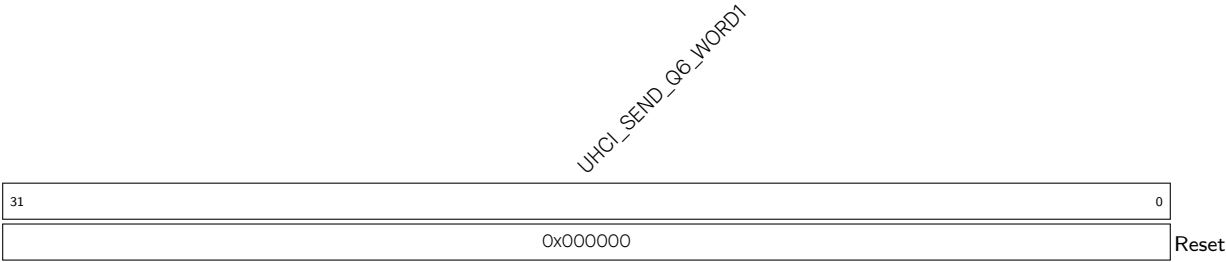
UHCI_SEND_Q5_WORD1 Data to be transmitted in Q5 register. (R/W)

Register 25.88. UHCI_REG_Q6_WORD0_REG (0x0064)



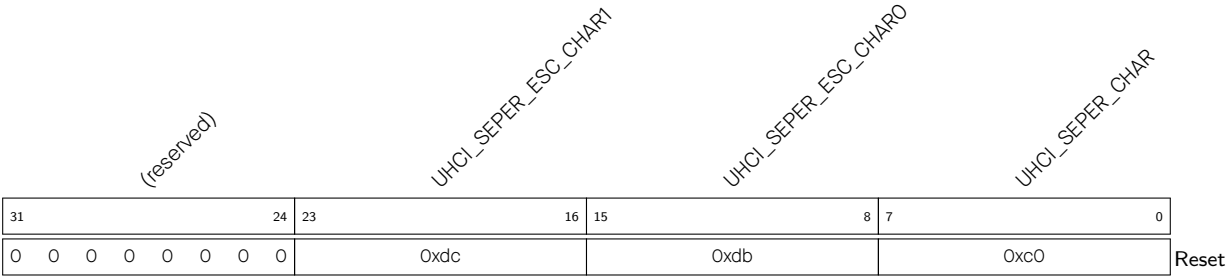
UHCI_SEND_Q6_WORD0 Data to be transmitted in Q6 register. (R/W)

Register 25.89. UHCI_REG_Q6_WORD1_REG (0x0068)



UHCI_SEND_Q6_WORD1 Data to be transmitted in Q6 register. (R/W)

Register 25.90. UHCI_ESC_CONFO_REG (0x006C)



UHCI_SEPER_CHAR Configures separators to encode data packets. The default value is 0xC0. (R/W)

UHCI_SEPER_ESC_CHAR0 Configures the first character of SLIP escape sequence. The default value is 0xDB. (R/W)

UHCI_SEPER_ESC_CHAR1 Configures the second character of SLIP escape sequence. The default value is 0xDC. (R/W)

Register 25.91. UHCI_ESC_CONF1_REG (0x0070)

(reserved)								UHCI_ESC_SEQ0_CHAR1								UHCI_ESC_SEQ0_CHAR0								UHCI_ESC_SEQ0																								
31								24								16								15								8								7								0
0	0	0	0	0	0	0	0	0xdd								0xdb								0xdb								Reset																

UHCI_ESC_SEQ0 Configures the character that needs to be encoded. The default value is 0xDB used as the first character of SLIP escape sequence. (R/W)

UHCI_ESC_SEQ0_CHAR0 Configures the first character of SLIP escape sequence. The default value is 0xDB. (R/W)

UHCI_ESC_SEQ0_CHAR1 Configures the second character of SLIP escape sequence. The default value is 0xDD. (R/W)

Register 25.92. UHCI_ESC_CONF2_REG (0x0074)

(reserved)								UHCI_ESC_SEQ1_CHAR1								UHCI_ESC_SEQ1_CHAR0								UHCI_ESC_SEQ1								
3124								2316								158								70								
00000000								0xde								0xdb								0x11								Reset

UHCI_ESC_SEQ1 Configures a character that need to be encoded. The default value is 0x11 used as a flow control character. (R/W)

UHCI_ESC_SEQ1_CHAR0 Configures the first character of SLIP escape sequence. The default value is 0xDB. (R/W)

UHCI_ESC_SEQ1_CHAR1 Configures the second character of SLIP escape sequence. The default value is 0xDE. (R/W)

Register 25.93. UHCI_ESC_CONF3_REG (0x0078)

(reserved)								UHCI_ESC_SEQ2_CHAR1								UHCI_ESC_SEQ2_CHAR0								UHCI_ESC_SEQ2												
31								24						16						15						8				7						0
0	0	0	0	0	0	0	0	0xdf					0xdb					0x13					Reset													

- UHCI_ESC_SEQ2 Configures the character that needs to be decoded. The default value is 0x13 used as a flow control character. (R/W)
- UHCI_ESC_SEQ2_CHAR0 Configures the first character of SLIP escape sequence. The default value is 0xDB. (R/W)
- UHCI_ESC_SEQ2_CHAR1 Configures the second character of SLIP escape sequence. The default value is 0xDF. (R/W)

Register 25.94. UHCI_PKT_THRES_REG (0x007C)

(reserved)																UHCL_PKT_THRS																																															
31																13																12																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x80																Reset																															

- UHCI_PKT_THRS Configures the maximum value of the packet length.
Measurement unit: byte.
Valid only when UHCI_HEAD_EN is 0. (R/W)

Register 25.95. UHCI_INT_RAW_REG (0x0004)

(reserved)																								UHCI_APP_CTRL1_INT_RAW UHCI_APP_CTRL0_INT_RAW UHCI_OUT_EOF_INT_RAW UHCI_SEND_A_REG_Q_INT_RAW UHCI_SEND_S_REG_Q_INT_RAW UHCI_TX_HUNG_INT_RAW UHCI_RX_HUNG_INT_RAW UHCI_TX_START_INT_RAW UHCI_RX_START_INT_RAW																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
31																								9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UHCI_RX_START_INT_RAW The raw interrupt status of UHCI_RX_START_INT. (R/WTC/SS)

UHCI_TX_START_INT_RAW The raw interrupt status of UHCI_TX_START_INT. (R/WTC/SS)

UHCI_RX_HUNG_INT_RAW The raw interrupt status of UHCI_RX_HUNG_INT. (R/WTC/SS)

UHCI_TX_HUNG_INT_RAW The raw interrupt status of UHCI_TX_HUNG_INT. (R/WTC/SS)

UHCI_SEND_S_REG_Q_INT_RAW The raw interrupt status of UHCI_SEND_S_REG_Q_INT. (R/WTC/SS)

UHCI_SEND_A_REG_Q_INT_RAW The raw interrupt status of UHCI_SEND_A_REG_Q_INT. (R/WTC/SS)

UHCI_OUT_EOF_INT_RAW The raw interrupt status of UHCI_OUT_EOF_INT. (R/WTC/SS)

UHCI_APP_CTRL0_INT_RAW The raw interrupt status of UHCI_APP_CTRL0_INT. (R/W)

UHCI_APP_CTRL1_INT_RAW The raw interrupt status of UHCI_APP_CTRL1_INT. (R/W)

Register 25.96. UHCI_INT_ST_REG (0x0008)

(reserved)																																UHCI_APP_CTRL1_INT_ST UHCI_APP_CTRL0_INT_ST UHCI_OUTLINK_EOF_ERR_INT_ST UHCI_SEND_A_REG_Q_INT_ST UHCI_SEND_S_REG_Q_INT_ST UHCI_TX_HUNG_INT_ST UHCI_RX_HUNG_INT_ST UHCI_TX_START_INT_ST UHCI_RX_START_INT_ST										
31																								9		8	7	6	5	4	3	2	1	0								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

UHCI_RX_START_INT_ST The masked interrupt status of UHCI_RX_START_INT. (RO)

UHCI_TX_START_INT_ST The masked interrupt status of UHCI_TX_START_INT. (RO)

UHCI_RX_HUNG_INT_ST The masked interrupt status of UHCI_RX_HUNG_INT. (RO)

UHCI_TX_HUNG_INT_ST The masked interrupt status of UHCI_TX_HUNG_INT. (RO)

UHCI_SEND_S_REG_Q_INT_ST The masked interrupt status of UHCI_SEND_S_REG_Q_INT. (RO)

UHCI_SEND_A_REG_Q_INT_ST The masked interrupt status of UHCI_SEND_A_REG_Q_INT. (RO)

UHCI_OUTLINK_EOF_ERR_INT_ST The masked interrupt status of UHCI_OUTLINK_EOF_ERR_INT.
(RO)

UHCI_APP_CTRL0_INT_ST The masked interrupt status of UHCI_APP_CTRL0_INT. (RO)

UHCI_APP_CTRL1_INT_ST The masked interrupt status of UHCI_APP_CTRL1_INT. (RO)

Register 25.97. UHCI_INT_ENA_REG (0x000C)

(reserved)																												UHCI_APP_CTRL1_INT_ENA UHCI_APP_CTRL0_INT_ENA UHCI_OUTLINK_EOF_ERR_INT_ENA UHCI_SEND_A_REG_Q_INT_ENA UHCI_SEND_S_REG_Q_INT_ENA UHCI_TX_HUNG_INT_ENA UHCI_RX_HUNG_INT_ENA UHCI_TX_START_INT_ENA UHCI_RX_START_INT_ENA										
31																												9	8	7	6	5	4	3	2	1	0	Reset
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0	0	0	0	0	0	0	0	0	0	

UHCI_RX_START_INT_ENA Write 1 to enable UHCI_RX_START_INT. (R/W)

UHCI_TX_START_INT_ENA Write 1 to enable UHCI_TX_START_INT. (R/W)

UHCI_RX_HUNG_INT_ENA Write 1 to enable UHCI_RX_HUNG_INT. (R/W)

UHCI_TX_HUNG_INT_ENA Write 1 to enable UHCI_TX_HUNG_INT. (R/W)

UHCI_SEND_S_REG_Q_INT_ENA Write 1 to enable UHCI_SEND_S_REG_Q_INT. (R/W)

UHCI_SEND_A_REG_Q_INT_ENA Write 1 to enable UHCI_SEND_A_REG_Q_INT. (R/W)

UHCI_OUTLINK_EOF_ERR_INT_ENA Write 1 to enable UHCI_OUTLINK_EOF_ERR_INT. (R/W)

UHCI_APP_CTRL0_INT_ENA Write 1 to enable UHCI_APP_CTRL0_INT. (R/W)

UHCI_APP_CTRL1_INT_ENA Write 1 to enable UHCI_APP_CTRL1_INT. (R/W)

Register 25.98. UHCI_INT_CLR_REG (0x0010)

(reserved)																UHCI_APP_CTRL1_INT_CLR			
																UHCI_APP_CTRL0_INT_CLR			
																UHCI_OUTLINK_EOF_ERR_INT_CLR			
																UHCI_SEND_A_REG_Q_INT_CLR			
																UHCI_SEND_S_REG_Q_INT_CLR			
																UHCI_TX_HUNG_INT_CLR			
																UHCI_RX_HUNG_INT_CLR			
																UHCI_TX_START_INT_CLR			
																UHCI_RX_START_INT_CLR			
31									9	8	7	6	5	4	3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

UHCI_RX_START_INT_CLR Write 1 to clear UHCI_RX_START_INT. (WT)

UHCI_TX_START_INT_CLR Write 1 to clear UHCI_TX_START_INT. (WT)

UHCI_RX_HUNG_INT_CLR Write 1 to clear UHCI_RX_HUNG_INT. (WT)

UHCI_TX_HUNG_INT_CLR Write 1 to clear UHCI_TX_HUNG_INT. (WT)

UHCI_SEND_S_REG_Q_INT_CLR Write 1 to clear UHCI_SEND_S_REG_Q_INT. (WT)

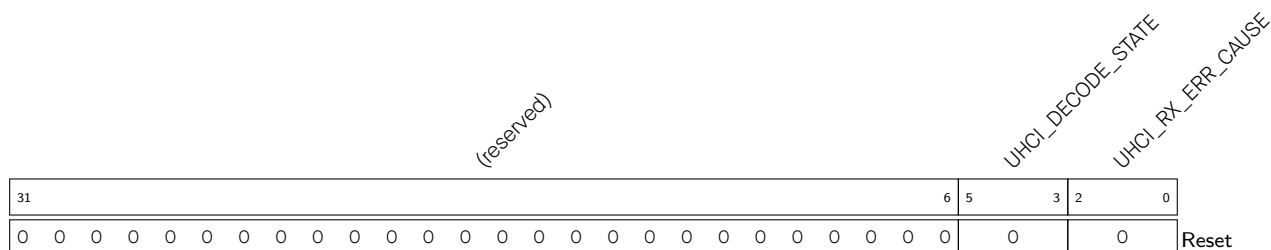
UHCI_SEND_A_REG_Q_INT_CLR Write 1 to clear UHCI_SEND_A_REG_Q_INT. (WT)

UHCI_OUTLINK_EOF_ERR_INT_CLR Write 1 to clear UHCI_OUTLINK_EOF_ERR_INT. (WT)

UHCI_APP_CTRL0_INT_CLR Write 1 to clear UHCI_APP_CTRL0_INT. (WT)

UHCI_APP_CTRL1_INT_CLR Write 1 to clear UHCI_APP_CTRL1_INT. (WT)

Register 25.99. UHCI_STATE0_REG (0x0018)

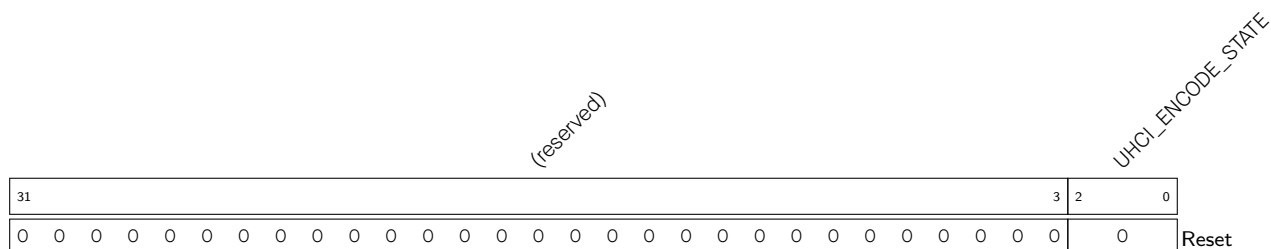


UHCI_RX_ERR_CAUSE Represents the error type when DMA has received a packet with error.

- 0: Invalid. No effect
1: Checksum error in the HCI packet
2: Sequence number error in the HCI packet
3: CRC bit error in the HCI packet
4: 0xC0 is found but the received HCI packet is not complete 5: 0xC0 is not found when the HCI packet has been received
6: CRC check error
7: Invalid. No effect
(RO)

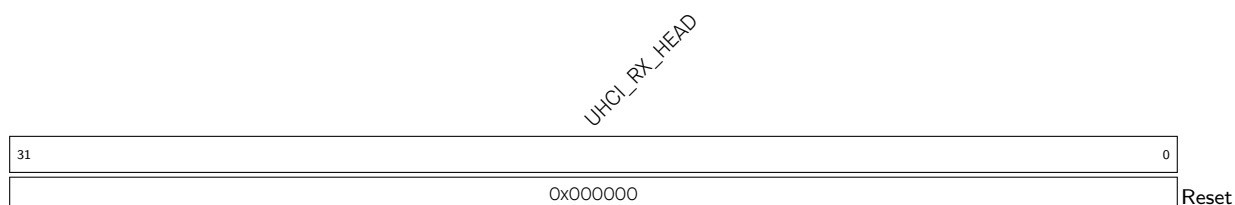
UHCI_DECODE_STATE Represents the UHCI decoder status. (RO)

Register 25.100. UHCI_STATE1_REG (0x001C)



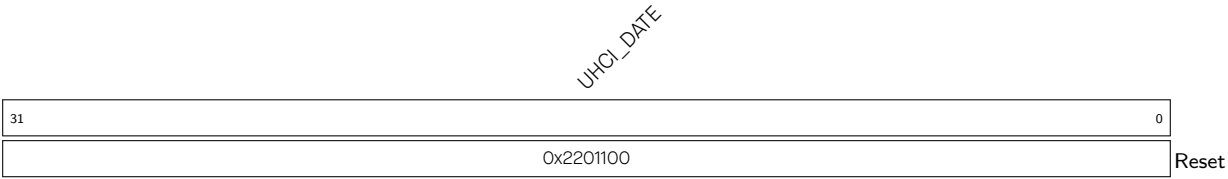
UHCI_ENCODE_STATE Represents the UHCI encoder status. (RO)

Register 25.101. UHCI_RX_HEAD_REG (0x002C)



UHCI_RX_HEAD Represents the header of the current received packet. (RO)

Register 25.102. UHCI_DATE_REG (0x0080)



UHCI_DATE Version control register. (R/W)

Chapter 26

I2C Controller (I2C)

The I2C (Inter-Integrated Circuit) bus allows ESP32-C5 to communicate with multiple external devices. These external devices can share one I2C bus.

ESP32-C5 has two I2C controllers: one in the main system and one in the low-power system. The I2C controller in the main system can act as a master or a slave (referred to as I2C below), while the one in the low-power system can only act as a master (referred to as LP_I2C below), which can still work when the main system sleeps.

26.1 Overview

The I2C bus has two lines, namely a serial data line (SDA) and a serial clock line (SCL). Both SDA and SCL lines are open-drain. The I2C bus can be connected to a single or multiple master devices and a single or multiple slave devices. However, only one master device can access a slave at a time via the bus.

The master initiates communication by generating a START condition: pulling the SDA line low while SCL is high. Then it issues nine clock pulses via SCL. The first eight pulses are used to transmit a 7-bit address followed by a read/write ($R/\overline{W}$) bit. If the address of an I2C slave matches the 7-bit address transmitted, this matching slave can respond by pulling SDA low on the ninth clock pulse. The master and the slave can send or receive data according to the $R/\overline{W}$ bit. Whether to terminate the data transfer or not is determined by the logic level of the acknowledge (ACK) bit. During data transfer, SDA changes only when SCL is low. Once the communication has finished, the master sends a STOP condition: pulling SDA up while SCL is high. If a master both reads and writes data in one transfer, then it should send a RSTART condition, a slave address, and a $R/\overline{W}$ bit before changing its operation. The RSTART condition is used to change the transfer direction and the mode of the devices (master mode or slave mode).

26.2 Feature List

The I2C controller of ESP32-C5 has the following features:

- Master mode and slave mode
- Communication between multiple masters and slaves
- Standard mode (100 Kbit/s)
- Fast mode (400 Kbit/s)
- 7-bit addressing and 10-bit addressing
- Continuous data transfer achieved by pulling SCL low in slave mode
- Programmable digital noise filtering

- Dual address mode, which uses slave address and slave memory or register address

26.3 Architecture Overview

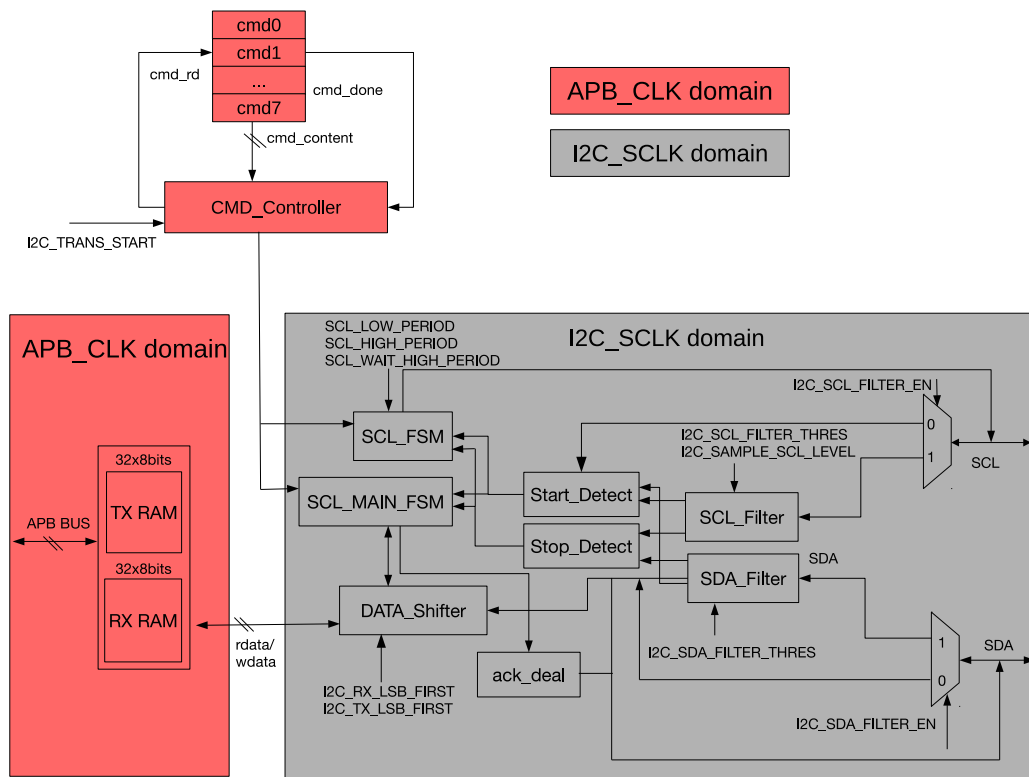


Figure 26.3-1. I2C Master Architecture

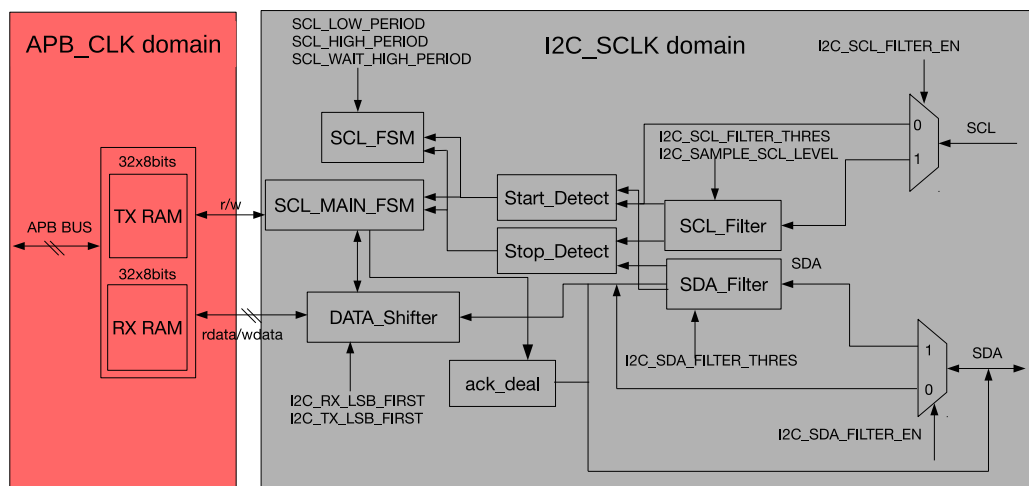


Figure 26.3-2. I2C Slave Architecture

The I2C controller runs either in master mode or slave mode, which is determined by `I2C_MS_MODE`. Figure 26.3-1 shows the architecture of a master, while Figure 26.3-2 shows that of a slave. The I2C controller has the following main parts:

- Transmit and receive memory (TX/RX RAM): stores data to be transmitted and data received respectively.

- Command controller (CMD_Controller): generates RSTART, STOP, WRITE, READ, and END commands
- SCL clock controller (SCL_FSM): generates the timing sequence conforming to the I2C protocol. Figure 26.3-3 and Table 26.3-1 are the timing diagram and corresponding parameters of the I2C protocol.
- SDA data controller (SCL_MAIN_FSM): controls the execution of I2C commands and the data sequence of the SDA line. It also controls the ACK_deal module to generate the ACK bit and detect the level of the ACK bit on the SDA line.
- Serial/parallel data converter (DATA_Shifter): shift data between serial and parallel form
- Filter for SCL (SCL_Filter): remove noises on SCL input signals
- Filter for SDA (SDA_Filter): remove noises on SDA input signals
- ACK bit controller (ack_deal): generate the ACK bit and detect the level of the ACK bit on the SDA line under the control of SCL_MAIN_FSM.

Besides, the I2C controller also has a clock module that generates I2C clocks, and a synchronization module that synchronizes the APB bus and the I2C controller.

The clock module is used to select clock sources, turn on and off clocks, and divide clocks. The synchronization module synchronizes signal transfer between different clock domains.

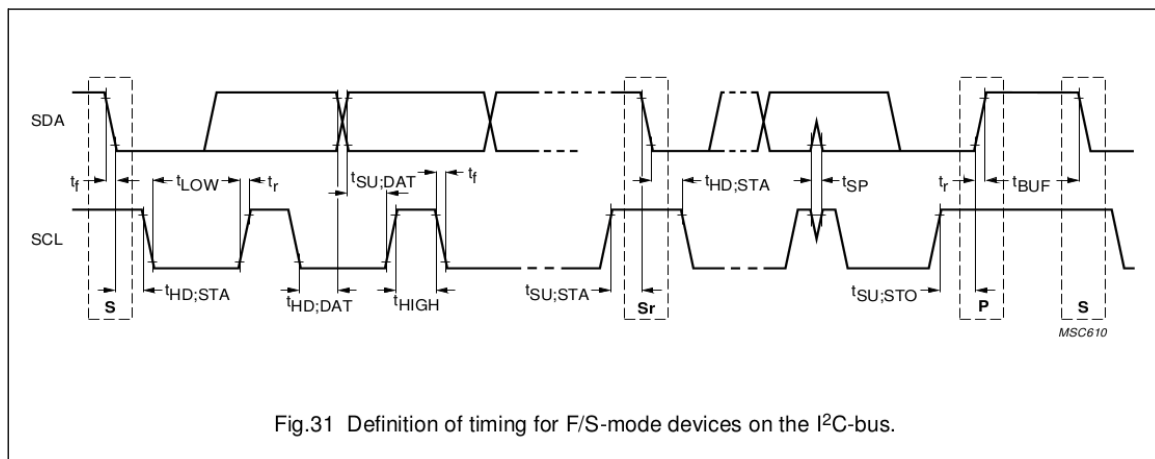


Fig.31 Definition of timing for F/S-mode devices on the I²C-bus.

Figure 26.3-3. I2C Protocol Timing (Cited from Fig.31 in [The I2C-bus specification](#) Version 2.1)

PARAMETER	SYMBOL	STANDARD-MODE		FAST-MODE		UNIT
		MIN.	MAX.	MIN.	MAX.	
SCL clock frequency	f _{SCL}	0	100	0	400	kHz
Hold time (repeated) START condition. After this period, the first clock pulse is generated	t _{HD;STA}	4.0	—	0.6	—	μs
LOW period of the SCL clock	t _{LOW}	4.7	—	1.3	—	μs
HIGH period of the SCL clock	t _{HIGH}	4.0	—	0.6	—	μs
Set-up time for a repeated START condition	t _{SU;STA}	4.7	—	0.6	—	μs
Data hold time: for CBUS compatible masters (see NOTE, Section 10.1.3) for I ² C-bus devices	t _{HD;DAT}	5.0	—	—	—	μs
		0 ⁽²⁾	3.45 ⁽³⁾	0 ⁽²⁾	0.9 ⁽³⁾	μs
Data set-up time	t _{SU;DAT}	250	—	100 ⁽⁴⁾	—	ns
Rise time of both SDA and SCL signals	t _r	—	1000	20 + 0.1C _b ⁽⁵⁾	300	ns
Fall time of both SDA and SCL signals	t _f	—	300	20 + 0.1C _b ⁽⁵⁾	300	ns
Set-up time for STOP condition	t _{SU;STO}	4.0	—	0.6	—	μs
Bus free time between a STOP and START condition	t _{BUF}	4.7	—	1.3	—	μs

Table 26.3-1. I2C Timing Parameters (Cited from Table 5 in [The I2C-bus specification](#) Version 2.1)

26.4 Functional Description

As mentioned above, one or more masters and one or more slaves can be connected on the I2C bus. The following sections describe the operations of the ESP32-C5 I2C controllers. Note that operations may differ between the I2C controllers in ESP32-C5 and other masters or slaves on the bus. Please refer to datasheets of individual I2C devices for specific information.

26.4.1 Clock Configuration

Registers, TX RAM, and RX RAM are configured and accessed in the APB_CLK clock domain. The main logic of the I2C controller, including SCL_FSM, SCL_MAIN_FSM, SCL_FILTER, SDA_FILTER, and DATA_SHIFTER, are in the I2C_SCLK clock domain.

You can configure the clock source for I2C_SCLK of I2C in the main system (HP) to XTAL_CLK or RC_FAST_CLK via [PCR_I2C_SCLK_SEL](#). For the clock source for I2C_SCLK of LP_I2C in the low-power system (LP), you can configure it to CLK_XTALD2 or CLK_ROOT_FAST via [LP_CLKRST_LP_I2C_CLK_SEL](#).

The steps to configure the clock source for I2C are as follows:

- Enable the clock source for I2C_SCLK of I2C by configuring [PCR_I2C_SCLK_EN](#) to 1.
- When [PCR_I2C_SCLK_SEL](#) is 0, the clock source is XTAL_CLK.
- When [PCR_I2C_SCLK_SEL](#) is 1, the clock source is RC_FAST_CLK.

The steps to configure the clock source for LP_I2C are as follows:

- Enable the clock source for I2C_SCLK of LP_I2C by configuring [LPPERI_LP_EXT_I2C_CK_EN](#) to 1.
- When [LP_CLKRST_LP_I2C_CLK_SEL](#) is 0, the clock source is CLK_ROOT_FAST.
- When [LP_CLKRST_LP_I2C_CLK_SEL](#) is 1, the clock source is CLK_XTALD2.

The clock source then passes through a fractional divider to generate I2C_SCLK of I2C according to the following formula:

$$I2C\_SCLK\_DIV\_NUM + 1 + \frac{I2C\_SCLK\_DIV\_A}{I2C\_SCLK\_DIV\_B}$$

In the formula, I2C_SCLK_DIV_NUM represents the integer part of the divisor, I2C_SCLK_DIV_A represents the numerator of the fractional part of the divisor, and I2C_SCLK_DIV_B represents the denominator of the fractional part of the divisor. Limited by timing parameters, the derived clock I2C_SCLK should operate at a frequency 20 times larger than SCL's frequency.

For I2C:

- Configure I2C_SCLK_DIV_NUM via [PCR_I2C_SCLK_DIV_NUM](#).
- Configure I2C_SCLK_DIV_A via [PCR_I2C_SCLK_DIV_A](#).
- Configure I2C_SCLK_DIV_B via [PCR_I2C_SCLK_DIV_B](#).

26.4.2 SCL and SDA Noise Filtering

SCL_Filter and SDA_Filter modules are identical and are used to filter signal noise on SCL and SDA, respectively. These filters can be enabled or disabled by configuring [I2C_SCL_FILTER_EN](#) and [I2C_SDA_FILTER_EN](#).

Take SCL_Filter as an example. When enabled, SCL_Filter samples input signals on the SCL line continuously. These input signals are valid only if they remain unchanged for consecutive [I2C_SCL_FILTER_THRES](#) I2C_SCLK clock cycles. Given that only valid input signals can pass through the filter, SCL_Filter can remove glitches whose pulse width is shorter than [I2C_SCL_FILTER_THRES](#) I2C_SCLK clock cycles, while SDA_Filter can remove glitches whose pulse width is shorter than [I2C_SDA_FILTER_THRES](#) I2C_SCLK clock cycles.

26.4.3 SCL Clock Stretching

The I2C controller operating in slave mode (i.e., as a slave) can perform clock stretching by holding the SCL line low. This action suspends data transmission, providing more time to process data. This function is enabled by setting the [I2C_SLAVE_SCL_STRETCH_EN](#) bit. The time period for releasing the SCL line from stretching is configured by setting the [I2C_STRETCH_PROTECT_NUM](#) field to avoid timing sequence errors.

The slave can activate clock stretching by pulling the SCL line low in response to one of four specific events.

1. Address match: The address of the slave matches the address sent by the master via the SDA line, and the $R/\overline{W}$ bit is 1.
2. RAM being full: RX RAM of the slave is full. Note that when the slave receives less than the FIFO depth, which is 32 bytes in ESP32-C5 I2C, it is not necessary to enable clock stretching; when the slave receives FIFO depth bytes or more, you may interrupt data transmission to wrapped around RAM via the FIFO threshold, or enable clock stretching for more time to process data. When clock stretching is enabled, [I2C_RX_FULL_ACK_LEVEL](#) must be cleared, otherwise there will be unpredictable consequences.

3. RAM being empty: The slave is sending data, but its TX RAM is empty.
4. Sending an ACK: If [I2C_SLAVE_BYTE_ACK_CTL_EN](#) is set, the slave pulls SCL low when sending an ACK bit. At this stage, software validates data and configures [I2C_SLAVE_BYTE_ACK_LVL](#) to control the level of the ACK bit. Note that when RX RAM of the slave is full, the level of the ACK bit to be sent is determined by [I2C_RX_FULL_ACK_LEVEL](#), instead of [I2C_SLAVE_BYTE_ACK_LVL](#). In this case, [I2C_RX_FULL_ACK_LEVEL](#) should also be cleared to ensure proper functioning of clock stretching.

When clock stretching occurs, the cause of stretching can be read from the [I2C_STRETCH_CAUSE](#) bit. Clock stretching can be disabled by setting the [I2C_SLAVE_SCL_STRETCH_CLR](#) bit.

26.4.4 Generating SCL Pulses in Idle State

Usually, when the I2C bus is idle, the SCL line is held high. The I2C controller in ESP32-C5 can be programmed to generate SCL pulses in an idle state. This function only works when the I2C controller is configured as master. If the [I2C_SCL_RST_SLV_EN](#) bit is set, hardware will send [I2C_SCL_RST_SLV_NUM](#) SCL pulses, and then automatically clear the [I2C_SCL_RST_SLV_EN](#) bit.

26.4.5 Synchronization

I2C registers are configured in the APB_CLK domain, whereas the I2C controller is configured in the asynchronous I2C_SCLK domain. Therefore, before being used by the I2C controller, register values should be synchronized by first writing configuration registers and then writing 1 to [I2C_CONF_UPGATE](#). Registers that need synchronization are listed in Table [26.4-1](#).

Table 26.4-1. I2C Synchronous Registers

Register	Field	Address
I2C_CTR_REG	I2C_SLV_TX_AUTO_START_EN	0x0004
	I2C_ADDR_10BIT_RW_CHECK_EN	
	I2C_ADDR_BROADCASTING_EN	
	I2C_SDA_FORCE_OUT	
	I2C_SCL_FORCE_OUT	
	I2C_SAMPLE_SCL_LEVEL	
	I2C_RX_FULL_ACK_LEVEL	
	I2C_MS_MODE	
	I2C_TX_LSB_FIRST	
	I2C_RX_LSB_FIRST	
	I2C_ARBITRATION_EN	
I2C_TO_REG	I2C_TIME_OUT_EN	0x000C
	I2C_TIME_OUT_VALUE	
I2C_SLAVE_ADDR_REG	I2C_ADDR_10BIT_EN	0x0010
	I2C_SLAVE_ADDR	
I2C_FIFO_CONF_REG	I2C_FIFO_ADDR_CFG_EN	0x0018
I2C_SCL_SP_CONF_REG	I2C_SDA_PD_EN	0x0080
	I2C_SCL_PD_EN	
	I2C_SCL_RST_SLV_NUM	
	I2C_SCL_RST_SLV_EN	
I2C_SCL_STRETCH_CONF_REG	I2C_SLAVE_BYTE_ACK_CTL_EN	0x0084
	I2C_SLAVE_BYTE_ACK_LVL	
	I2C_SLAVE_SCL_STRETCH_EN	
	I2C_STRETCH_PROTECT_NUM	
I2C_SCL_LOW_PERIOD_REG	I2C_SCL_LOW_PERIOD	0x0000
I2C_SCL_HIGH_PERIOD_REG	I2C_WAIT_HIGH_PERIOD	0x0038
	I2C_HIGH_PERIOD	
I2C_SDA_HOLD_REG	I2C_SDA_HOLD_TIME	0x0030
I2C_SDA_SAMPLE_REG	I2C_SDA_SAMPLE_TIME	0x0034
I2C_SCL_START_HOLD_REG	I2C_SCL_START_HOLD_TIME	0x0040
I2C_SCL_RSTART_SETUP_REG	I2C_SCL_RSTART_SETUP_TIME	0x0044
I2C_SCL_STOP_HOLD_REG	I2C_SCL_STOP_HOLD_TIME	0x0048
I2C_SCL_STOP_SETUP_REG	I2C_SCL_STOP_SETUP_TIME	0x004C
I2C_SCL_ST_TIME_OUT_REG	I2C_SCL_ST_TO_I2C	0x0078
I2C_SCL_MAIN_ST_TIME_OUT_REG	I2C_SCL_MAIN_ST_TO_I2C	0x007C
I2C_FILTER_CFG_REG	I2C_SCL_FILTER_EN	0x0050
	I2C_SCL_FILTER_THRES	
	I2C_SDA_FILTER_EN	
	I2C_SDA_FILTER_THRES	

26.4.6 Open-Drain Output

SCL and SDA output drivers must be configured as open-drain. There are two ways to achieve this:

1. Set `I2C_SCL_FORCE_OUT` and `I2C_SDA_FORCE_OUT`, and configure `GPIO_PINn_PAD_DRIVER` for corresponding SCL and SDA pads as open-drain.
2. Clear `I2C_SCL_FORCE_OUT` and `I2C_SDA_FORCE_OUT`.

Because these lines are configured as open-drain, the low-to-high transition time of each line is longer, determined together by the pull-up resistor and line capacitance. The output duty cycle of I2C is limited by the SDA and SCL line's pull-up speed, mainly SCL's speed.

In addition, when `I2C_SCL_FORCE_OUT` and `I2C_SCL_PD_EN` are set to 1, SCL can be forced low; when `I2C_SDA_FORCE_OUT` and `I2C_SDA_PD_EN` are set to 1, SDA can be forced low.

26.4.7 Timing Parameters Configuration

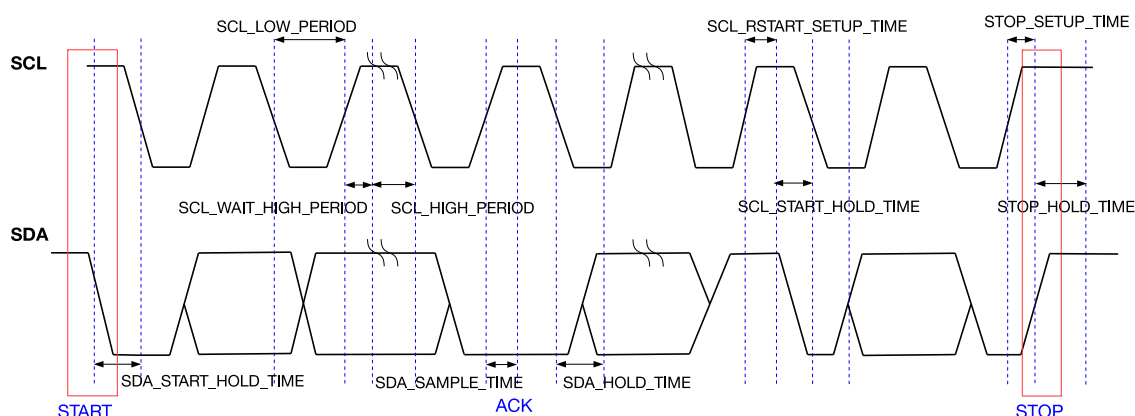


Figure 26.4-1. I2C Timing Diagram

Figure 26.4-1 shows the timing diagram of an I2C master. This figure also specifies registers used to configure the START bit, STOP bit, data hold time, data sample time, waiting time on the rising SCL edge, etc. Timing parameters are calculated as follows in I2C_SCLK clock cycles:

1. $t_{LOW} = (I2C\_SCL\_LOW\_PERIOD + 1) \cdot T_{I2C\_SCLK}$
2. $t_{HIGH} = (I2C\_SCL\_HIGH\_PERIOD + 1) \cdot T_{I2C\_SCLK}$
3. $t_{SU:STA} = (I2C\_SCL\_RSTART\_SETUP\_TIME + 1) \cdot T_{I2C\_SCLK}$
4. $t_{HD:STA} = (I2C\_SCL\_START\_HOLD\_TIME + 1) \cdot T_{I2C\_SCLK}$
5. $t_r = (I2C\_SCL\_WAIT\_HIGH\_PERIOD + 1) \cdot T_{I2C\_SCLK}$
6. $t_{SU:STO} = (I2C\_SCL\_STOP\_SETUP\_TIME + 1) \cdot T_{I2C\_SCLK}$
7. $t_{BUF} = (I2C\_SCL\_STOP\_HOLD\_TIME + 1) \cdot T_{I2C\_SCLK}$
8. $t_{HD:DAT} = (I2C\_SDA\_HOLD\_TIME + 1) \cdot T_{I2C\_SCLK}$
9. $t_{SU:DAT} = (I2C\_SCL\_LOW\_PERIOD - I2C\_SDA\_HOLD\_TIME) \cdot T_{I2C\_SCLK}$

Timing registers below are divided into two groups, depending on the mode in which these registers are active:

- Master mode only:

1. **I2C_SCL_START_HOLD_TIME**: Specifies the interval between the moment SDA is pulled low and the moment SCL is pulled low when the master generates a START condition. This interval is (**I2C_SCL_START_HOLD_TIME** + 1) in I2C_SCLK cycles. This register is active only when the I2C controller works in master mode.
2. **I2C_SCL_LOW_PERIOD**: Specifies the low period of SCL. This period lasts (**I2C_SCL_LOW_PERIOD** + 1) in I2C_SCLK cycles. This register is active only when the I2C controller works in master mode.

However, this period could be extended in the following scenarios:

- SCL is pulled low by peripheral devices when I2C acts as a master.
 - SCL is pulled low by an END command executed by the I2C controller.
 - SCL is pulled low by clock stretching when I2C acts as a slave.
3. **I2C_SCL_WAIT_HIGH_PERIOD**: Specifies the time for SCL to switch from low to high in I2C_SCLK cycles. Please make sure that SCL can be pulled high within this time period. Otherwise, the high period of SCL may be incorrect. This register is active only when the I2C controller works in master mode.
 4. **I2C_SCL_HIGH_PERIOD**: Specifies the high period of SCL in I2C_SCLK cycles. This register is active only when the I2C controller works in master mode. When SCL goes high within (**I2C_SCL_WAIT_HIGH_PERIOD** + 1) in I2C_SCLK cycles, its frequency is:

$$f_{scl} = \frac{f_{I2C\_SCLK}}{I2C\_SCL\_LOW\_PERIOD + I2C\_SCL\_HIGH\_PERIOD + I2C\_SCL\_WAIT\_HIGH\_PERIOD + 3 + I2C\_SCL\_FILTER\_THRES}$$

where 3 represents the amount of clock cycles required to synchronize the SCL. If the SCL filtering function is turned on, the delay caused by **I2C_SCL_FILTER_THRES** needs to be added. As the SCL low-to-high transition time represented by **I2C_SCL_WAIT_HIGH_PERIOD** + 1 module clock can be affected by the pull-up resistor, IO drive capability, SCL line capacitance, etc., deviation may occur between the actual frequency of the test and the theoretical frequency. At this point, deviations can be reduced by adjusting the value of **I2C_SCL_WAIT_HIGH_PERIOD**.

- Master mode and slave mode:

1. **I2C_SDA_SAMPLE_TIME**: Specifies the interval between the rising edge of SCL and the level sampling time of SDA. It is advised to set a value in the middle of SCL's high period, so as to correctly sample the level of SCL. This register is active both in master mode and slave mode.
2. **I2C_SDA_HOLD_TIME**: Specifies the interval between changing the SDA output level and the falling edge of SCL. This register is active both in master mode and slave mode.

Timing parameters limits corresponding register configuration.

1. $\frac{f_{I2C\_SCLK}}{f_{SCL}} > 20$
2. $3 \times f_{I2C\_SCLK} \leq (I2C\_SDA\_HOLD\_TIME - 4) \times f_{APB\_CLK}$
3. $I2C\_SDA\_HOLD\_TIME + I2C\_SCL\_START\_HOLD\_TIME > SDA\_FILTER\_THRES + 3$
4. $I2C\_SCL\_WAIT\_HIGH\_PERIOD < I2C\_SDA\_SAMPLE\_TIME < I2C\_SCL\_HIGH\_PERIOD$
5. $I2C\_SDA\_SAMPLE\_TIME < I2C\_SCL\_WAIT\_HIGH\_PERIOD + I2C\_SCL\_START\_HOLD\_TIME + I2C\_SCL\_RSTART\_SETUP\_TIME$
6. $I2C\_STRETCH\_PROTECT\_NUM + I2C\_SDA\_HOLD\_TIME > I2C\_SCL\_LOW\_PERIOD$

26.4.8 Timeout Control

The I2C controller has three types of timeout control, namely timeout control for SCL_FSM, for SCL_MAIN_FSM, and for the SCL line. The first two are always enabled, while the third is configurable.

When SCL_FSM remains unchanged for more than $2^{I2C\_SCL\_ST\_TO\_I2C+1}$ clock cycles, an [I2C_SCL_ST_TO_INT](#) interrupt is triggered, and then SCL_FSM goes to idle state. The value of [I2C_SCL_ST_TO_I2C](#) should be less than or equal to 23, which means SCL_FSM could remain unchanged for 2^{24} I2C_SCLK clock cycles at most before the interrupt is generated.

When SCL_MAIN_FSM remains unchanged for more than $2^{I2C\_SCL\_MAIN\_ST\_TO\_I2C+1}$ I2C_SCLK clock cycles, an [I2C_SCL_MAIN_ST_TO_INT](#) interrupt is triggered, and then SCL_MAIN_FSM goes to idle state. The value of [I2C_SCL_MAIN_ST_TO_I2C](#) should be less than or equal to 23, which means SCL_MAIN_FSM could remain unchanged for 2^{24} clock cycles at most before the interrupt is generated.

Timeout control for SCL is enabled by setting [I2C_TIME_OUT_EN](#). When the level of SCL remains unchanged for more than $2^{I2C\_TIME\_OUT\_VALUE+1}$ clock cycles, an [I2C_TIME_OUT_INT](#) interrupt is triggered, and then the I2C bus goes to idle state.

26.4.9 Command Configuration

When the I2C controller works in master mode, CMD_Controller reads commands from eight sequential command registers and controls SCL_FSM and SCL_MAIN_FSM accordingly.

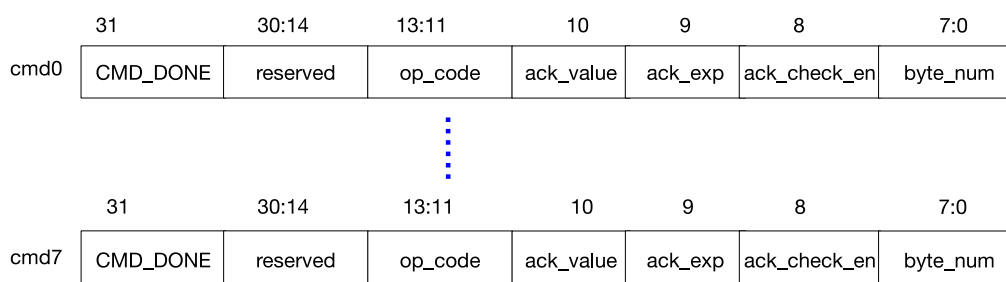


Figure 26.4-2. Structure of I2C Command Registers

Command registers, whose structure is illustrated in Figure 26.4-2, are active only when the I2C controller works in master mode. Fields of command registers are:

1. **CMD_DONE**: Indicates that a command has been executed. After each command has been executed, the CMD_DONE bit in the corresponding command register is set to 1 by hardware. By reading this bit, the software can tell if the command has been executed. When writing new commands, this bit must be cleared by software.
2. **op_code**: Indicates the command. The I2C controller supports five commands:
 - **WRITE**: op_code = 1. The I2C controller sends a slave address, a register address (only in dual address mode), and data to the slave.
 - **STOP**: op_code = 2. The I2C controller sends a STOP bit defined by the I2C protocol. This code also indicates that the command sequence has been executed, and the CMD_Controller stops reading commands. After restarted by software, the CMD_Controller resumes reading commands from command register 0.

- READ: op_code = 3. The I2C controller reads data from the slave.
 - END: op_code = 4. The I2C controller pulls the SCL line down and suspends I2C communication. This code also indicates that the command sequence has been completed, and the CMD_Controller stops executing commands. Once the software refreshes data in command registers and the RAM, the CMD_Controller can be restarted to execute commands from command register 0 again.
 - RSTART: op_code = 6. The I2C controller sends a START bit or a RSTART bit defined by the I2C protocol.
3. ack_value: Used to configure the level of the ACK bit sent by the I2C controller during a read operation. This bit is ignored in RSTART, STOP, END, and WRITE conditions.
 4. ack_exp: Used to configure the level of the ACK bit expected by the I2C controller during a write operation. This bit is ignored during RSTART, STOP, END, and READ conditions.
 5. ack_check_en: Used to enable the I2C controller during a write operation to check whether the ACK level sent by the slave matches ack_exp in the command. If this bit is set and the level received does not match ack_exp in the WRITE command, the master will generate an I2C_NACK_INT interrupt and a STOP condition for data transfer. If this bit is cleared, the controller will not check the ACK level sent by the slave. This bit is ignored during RSTART, STOP, END, and READ conditions.
 6. byte_num: Specifies the length of data (in bytes) to be read or written. It can range from 1 to 255 bytes. This bit is ignored during RSTART, STOP, and END conditions.

Each command sequence is executed starting from command register 0 and terminated by a STOP or an END. Therefore, there must be a STOP or an END command in the eight command registers.

A complete data transfer on the I2C bus should be initiated by a START and terminated by a STOP. The transfer process may be completed using multiple sequences, separated by END commands. Each sequence may differ in the direction of data transfer, clock frequency, slave addresses, data length, etc. This allows efficient use of available peripheral RAM and also achieves more flexible I2C communication.

26.4.10 TX/RX RAM Data Storage

Both TX RAM and RX RAM are 32×8 bits and can be accessed in FIFO or non-FIFO mode. If [I2C_NONFIFO_EN](#) bit is cleared, both RAMs are accessed in FIFO mode; if [I2C_NONFIFO_EN](#) bit is set, both RAMs are accessed in non-FIFO mode.

TX RAM stores data that the I2C controller needs to send. During communication, when the I2C controller needs to send data (except acknowledgment bits), it reads data from TX RAM and sends them sequentially via SDA. When the I2C controller works in master mode, all data must be stored in TX RAM in the order they need to be sent to slaves. The data stored in TX RAM include slave addresses, read/write bits, register addresses (only in dual address mode) and data to be sent. When the I2C controller works in slave mode, TX RAM only stores data to be sent.

TX RAM can be read and written by the CPU. The CPU writes to TX RAM either in FIFO mode or in non-FIFO mode (direct address). In FIFO mode, the CPU writes to TX RAM via the fixed address [I2C_DATA_REG](#), with addresses for writing in TX RAM incremented automatically by hardware. In non-FIFO mode, the CPU accesses TX RAM directly via address fields ([I2C Base Address](#) + 0x100) ~ ([I2C Base Address](#) + 0x17C). Each byte in TX RAM occupies an entire word in the address space. Therefore, the address of the first byte is [I2C Base](#)

[Address](#) + 0x100, the second byte is [I2C Base Address](#) + 0x104, the third byte is [I2C Base Address](#) + 0x108, and so on. The CPU can only read TX RAM via direct addresses. Bytes written to the TX RAM can be read back by the CPU, via the direct addresses. Addresses for reading TX RAM are the same as addresses for writing TX RAM.

RX RAM stores data the I2C controller receives during communication. When the I2C controller works in slave mode, neither slave addresses sent by the master nor register addresses (only in dual address mode) will be stored in RX RAM. Values of RX RAM can be read by software after I2C communication is completed.

RX RAM can only be read by the CPU. The CPU reads RX RAM either in FIFO mode or in non-FIFO mode (direct address). In FIFO mode, the CPU reads RX RAM via the fixed address [I2C_DATA_REG](#), with addresses for reading RX RAM incremented automatically by hardware. In non-FIFO mode, the CPU accesses TX RAM directly via address fields ([I2C Base Address](#) + 0x180) ~ ([I2C Base Address](#) + 0x1FC). Each byte in RX RAM occupies an entire word in the address space. Therefore, the address of the first byte is [I2C Base Address](#) + 0x180, the second byte is [I2C Base Address](#) + 0x184, the third byte is [I2C Base Address](#) + 0x188 and so on.

In FIFO mode, the TX RAM of a master may wrap around to send data larger than the FIFO depth. Set [I2C_FIFO_PRT_EN](#). If the size of data to be sent is smaller than [I2C_TXFIFO_WM_THRHD](#) (master), an [I2C_TXFIFO_WM_INT](#) (master) interrupt is generated. After receiving the interrupt, the software continues writing to [I2C_DATA_REG](#) (master). Please ensure that software writes to or refreshes TX RAM before the master sends data, otherwise it may result in unpredictable consequences.

In FIFO mode, the RX RAM of a slave may also wrap around to receive data larger than the FIFO depth. Set [I2C_FIFO_PRT_EN](#) and clear [I2C_RX_FULL_ACK_LEVEL](#). If data already received (to be overwritten) is larger than [I2C_RXFIFO_WM_THRHD](#) (slave), an [I2C_RXFIFO_WM_INT](#) (slave) interrupt is generated. After receiving the interrupt, the software continues reading from [I2C_DATA_REG](#) (slave).

26.4.11 Data Conversion

DATA_Shifter is used for serial/parallel conversion, converting byte data in TX RAM to an outgoing serial bitstream or an incoming serial bitstream to byte data in RX RAM. [I2C_RX_LSB_FIRST](#) and [I2C_TX_LSB_FIRST](#) can be used to select LSB- or MSB-first storage and transmission of data.

26.4.12 Addressing Mode

The ESP32-C5 I2C controller supports 7-bit and 10-bit addressing. 10-bit addressing can be mixed with 7-bit addressing. Besides, the ESP32-C5 I2C controller also supports dual address mode.

Define the slave address as SLV_ADDR. In 7-bit addressing mode, the slave address is SLV_ADDR[6:0]; in 10-bit addressing mode, the slave address is SLV_ADDR[9:0].

In 7-bit addressing mode, the master only needs to send one byte of the address, which comprises SLV_ADDR[6:0] and a $R/\overline{W}$ bit. In the 7-bit addressing mode, there is a special case called general call addressing (broadcast). It is enabled by setting [I2C_ADDR_BROADCASTING_EN](#) in a slave. When the slave receives the general call address (0x00) from the master and the $R/\overline{W}$ bit followed is 0, it responds to the master regardless of its own address.

In 10-bit addressing mode, the master needs to send two bytes of address. The first byte is slave_addr_first_7bits followed by a $R/\overline{W}$ bit, and slave_addr_first_7bits should be configured as (0x78 | SLV_ADDR[9:8]). The second byte is slave_addr_second_byte, which should be configured as SLV_ADDR[7:0].

The slave can enable 10-bit addressing by configuring [I2C_ADDR_10BIT_EN](#). [I2C_SLAVE_ADDR](#) is used to configure I2C slave address. Specifically, [I2C_SLAVE_ADDR\[14:7\]](#) should be configured as [SLV_ADDR\[7:0\]](#), and [I2C_SLAVE_ADDR\[6:0\]](#) should be configured as [\(0x78 | SLV_ADDR\[9:8\]\)](#). Since a 10-bit slave address has one more byte than a 7-bit address, [byte_num](#) of the WRITE command and the number of bytes in the RAM increase by one. Please refer to [Programming Example](#) for detailed descriptions.

When working in slave mode, the I2C controller supports dual address mode, where the first address is the address of an I2C slave, and the second one is the slave's memory address. When using dual address mode, RAM must be accessed in non-FIFO mode.

Dual address mode is enabled by setting [I2C_FIFO_ADDR_CFG_EN](#). When the slave address received by the slave is inconsistent with the internally configured slave address, the [I2C_SLAVE_ADDR_UNMATCH](#) interrupt will be generated.

26.4.13 $R/\overline{W}$ Bit Check in 10-bit Addressing Mode

In 10-bit addressing mode, when [I2C_ADDR_10BIT_RW_CHECK_EN](#) is set to 1, the I2C controller performs a check on the first byte, which consists of [slave_addr_first_7bits](#) and a $R/\overline{W}$ bit. When the $R/\overline{W}$ bit does not indicate a WRITE operation, i.e., not in line with the I2C protocol, the data transfer ends. If the check feature is not enabled, when the $R/\overline{W}$ bit does not indicate a WRITE, the data transfer still continues, but transfer failure may occur.

26.4.14 Start the I2C Controller

To start the I2C controller in master mode, after configuring the controller to master mode and command registers, write 1 to [I2C_TRANS_START](#) in order to let the master start to parse and execute command sequences. The master always executes a command sequence starting from the command register 0 to a STOP or an END. To execute another command sequence starting from command register 0, refresh commands by writing 1 again to [I2C_TRANS_START](#).

To start the I2C controller slave mode, after configuring the controller to slave mode and setting the relevant registers, write 1 to [I2C_TRANS_START](#). The slave will respond to the master when the received address matches.

For the slave device, please note:

- Always set [I2C_TRANS_START](#) before accepting any transfer. Otherwise, even if the address matches, the slave will not respond to the master.
- If the master initiates a STOP condition and prematurely terminates the transmission with the slave, the slave must first set [I2C_SLV_TX_AUTO_START_EN](#), then set [I2C_TRANS_START](#), and finally clear [I2C_SLV_TX_AUTO_START_EN](#) before the next transfer. Otherwise, an abnormal TX FIFO pointer issue will occur in the slave.

26.5 Functional Differences Between LP_I2C and I2C

LP_I2C can be used as a master to communicate with external devices when the main system sleeps. LP_I2C includes all the functions of the ESP32-C5 I2C_{master}, but doesn't include any functions of ESP32-C5 I2C_{slave}. It does not contain any registers related to the I2C_{slave}. For detailed register list, see [26.8.2 LP_I2C Register Summary](#).

The design differences between LP_I2C and I2C master are as follows:

- The size of TX/RX RAM in LP_I2C is 16x8 bit, which means the TX/RX FIFO depth is 16 bytes.
- The clock source of APB_CLK in LP_I2C is CLK_AON_FAST. The clock source for I2C_SCLK is configured via LP_CLKRST_LP_I2C_CLK_SEL. The corresponding bits are:
 - 0: CLK_ROOT_FAST
 - 1: CLK_XTALD2

Configuring LPPERI_LP_EXT_I2C_CK_EN to 1 enables the clock source of I2C_SCLK. Adjust the timing registers accordingly when the clock frequency changes.

See the programming examples of ESP32-C5 I2C_{slave} in Section 26.7 for that of LP_I2C.

26.6 Interrupts

ESP32-C5's I2C_n can generate the I2C_n_INTR interrupt signal that will be sent to the [Interrupt Matrix](#). There are several internal interrupt sources from I2C_n that can generate the above interrupt signal(s) as follows:

- I2C_SLAVE_STRETCH_INT: Triggered when one of the four stretching events occurs in slave mode.
- I2C_DET_START_INT: Triggered when the master or the slave detects a START signal.
- I2C_SCL_MAIN_ST_TO_INT: Triggered when the main state machine SCL_MAIN_FSM remains unchanged for over $2^{I2C\_SCL\_MAIN\_ST\_TO\_I2C+1}$ clock cycles.
- I2C_SCL_ST_TO_INT: Triggered when the state machine SCL_FSM remains unchanged for over $2^{I2C\_SCL\_ST\_TO\_I2C+1}$ clock cycles.
- I2C_RXFIFO_UDF_INT: Triggered when the I2C controller reads RX FIFO via the APB bus, but RX FIFO is empty.
- I2C_TXFIFO_OVF_INT: Triggered when the I2C controller writes TX FIFO via the APB bus, but TX FIFO is full.
- I2C_NACK_INT: Triggered when the ACK value received by the master is not as expected, or when the ACK value received by the slave is 1.
- I2C_TRANS_START_INT: Triggered when the I2C controller sends a START bit.
- I2C_TIME_OUT_INT: Triggered when SCL stays high or low for more than $2^{I2C\_TIME\_OUT\_VALUE+1}$ clock cycles during data transfer.
- I2C_TRANS_COMPLETE_INT: Triggered when the I2C controller detects a STOP bit.
- I2C_MST_TXFIFO_UDF_INT: Triggered when TX FIFO of the master underflows.
- I2C_ARBITRATION_LOST_INT: Triggered when the SDA's output value does not match its input value while the master's SCL is high.
- I2C_BYTE_TRANS_DONE_INT: Triggered when the I2C controller sends or receives a byte.
- I2C_END_DETECT_INT: Triggered when op_code of the master indicates an END command and an END condition is detected.
- I2C_RXFIFO_OVF_INT: Triggered when RX FIFO of the I2C controller overflows.

- I2C_TXFIFO_WM_INT: I2C TX FIFO watermark interrupt. Triggered when [I2C_FIFO_PRT_EN](#) is 1 and the pointers of TX FIFO are less than [I2C_TXFIFO_WM_THRHD\[4:0\]](#).
- I2C_RXFIFO_WM_INT: I2C RX FIFO watermark interrupt. Triggered when [I2C_FIFO_PRT_EN](#) is 1 and the pointers of RX FIFO are greater than [I2C_RXFIFO_WM_THRHD\[4:0\]](#).
- I2C_GENERAL_CALL_INT: Triggered when [the general call address](#) is received in slave mode.
- I2C_SLAVE_ADDR_UNMATCH_INT: Triggered when the received slave address is inconsistent with the internally configured slave address in slave mode.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 *Interrupt Matrix [to be added later]* > Section 3.1 *Interrupt Terminology*.

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [26.8.1 I2C Register Summary](#).

26.7 Programming Procedures

This section provides programming examples for typical communication scenarios. ESP32-C5 has two I2C controllers. For the convenience of description, I2C masters and slaves in all subsequent figures are ESP32-C5 I2C controllers. I2C master is referred to as I2C_{master}, and I2C slave is referred to as I2C_{slave}.

26.7.1 I2C_{master} Writes to I2C_{slave} with a 7-bit Address in One Command Sequence

26.7.1.1 Introduction

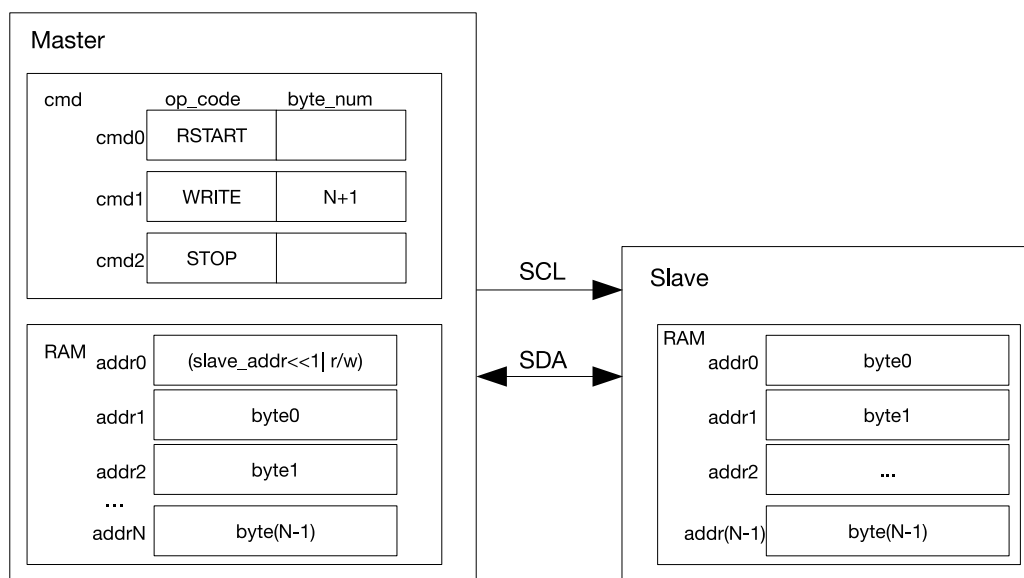


Figure 26.7-1. I2C_{master} Writing to I2C_{slave} with a 7-bit Address

Figure 26.7-1 shows how I2C_{master} writes N bytes of data to I2C_{slave} registers or RAM using 7-bit addressing. As shown in Figure 26.7-1, the first byte in the RAM of I2C_{master} is a 7-bit I2C_{slave} address followed by a $R/\overline{W}$ bit. When the $R/\overline{W}$ bit is 0, it indicates a WRITE operation. The remaining bytes are used to store data ready for transfer. The cmd box contains related command sequences.

After the command sequence is configured and data in RAM is ready, I2C_{master} enables the controller and initiates data transfer by setting the I2C_TRANS_START bit. The controller has four steps to take:

1. Wait for SCL to go high, to avoid SCL being used by other masters or slaves.
2. Execute a RSTART command by sending a START bit.
3. Execute a WRITE command by taking N+1 bytes from the RAM in order and sending them to I2C_{slave} in the same order. The first byte is the address of I2C_{slave}.
4. Execute a STOP command. Once the I2C_{master} transfers a STOP bit, an I2C_TRANS_COMPLETE_INT interrupt is generated.

26.71.2 Configuration Example

1. Configure the timing parameter registers of I2C_{master} and I2C_{slave} according to Section 26.4.7.
2. Set I2C_MS_MODE (master) to 1, and I2C_MS_MODE (slave) to 0.
3. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
4. Configure command registers of I2C_{master}.

Command register	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	ack_value	ack_exp	1	N+1
I2C_COMMAND2 (master)	STOP	—	—	—	—

5. Write the address of I2C_{slave} and data to be sent to TX RAM of I2C_{master} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
6. Write the address of I2C_{slave} to I2C_SLAVE_ADDR (slave) in I2C_SLAVE_ADDR_REG (slave) register.
7. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
8. Write 1 to I2C_TRANS_START (master) and I2C_TRANS_START (slave) to start transfer.
9. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in I2C_SLAVE_ADDR (slave). When ack_check_en (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it sends a byte. When ack_check_en (master) is 0, I2C_{master} does not check the ACK value and take I2C_{slave} as a matching slave by default.
 - Match: If the received ACK value matches ack_exp (master) (the expected ACK value) in the WRITE command, I2C_{master} continues data transfer.
 - Not match: If the received ACK value does not match ack_exp in the WRITE command, I2C_{master} generates an I2C_NACK_INT (master) interrupt and stops data transfer.
10. I2C_{master} sends data, and determines whether to check ACK value according to ack_check_en (master).
11. If data to be sent (N) is larger than TX FIFO depth, TX RAM of I2C_{master} may wrap around in FIFO mode. For details, please refer to Section 26.4.10.

- If data to be received (N) is larger than RX FIFO depth, RX RAM of I2C_{slave} may wrap around in FIFO mode. For details, please refer to Section 26.4.10.

If data to be received (N) is larger than RX FIFO depth, the other way is to enable clock stretching by setting the `I2C_SLAVE_SCL_STRETCH_EN` (slave), and clearing `I2C_RX_FULL_ACK_LEVEL`. When RX RAM is full, an `I2C_SLAVE_STRETCH_INT` (slave) interrupt is generated. In this way, I2C_{slave} can hold SCL low, in exchange for more time to read data. After the software has finished reading, you can set `I2C_SLAVE_STRETCH_INT_CLR` (slave) to 1 to clear interrupt, and set `I2C_SLAVE_SCL_STRETCH_CLR` (slave) to release the SCL line.

- After data transfer completes, I2C_{master} executes the STOP command, and generates an `I2C_TRANS_COMPLETE_INT` (master) interrupt.

26.7.2 I2C_{master} Writes to I2C_{slave} with a 10-bit Address in One Command Sequence

26.7.2.1 Introduction

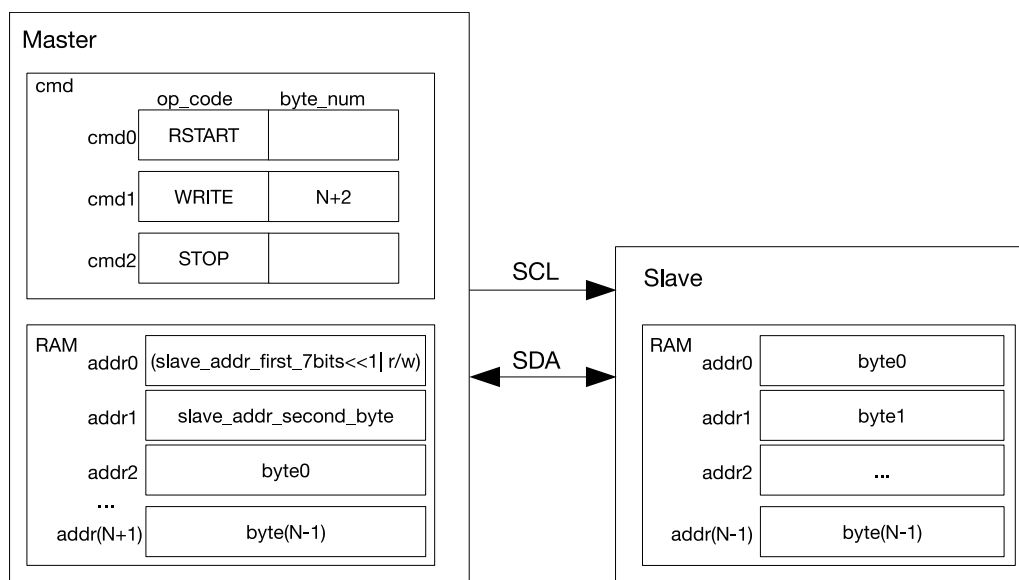


Figure 26.7-2. I2C_{master} Writing to a Slave with a 10-bit Address

Figure 26.7-2 shows how I2C_{master} writes N bytes of data using 10-bit addressing to an I2C slave. The configuration and transfer process is similar to what is described in 26.7.1, except that a 10-bit I2C_{slave} address is formed from two bytes. Since a 10-bit I2C_{slave} address has one more byte than a 7-bit I2C_{slave} address, byte_num and length of data in TX RAM increase by 1 accordingly.

26.7.2.2 Configuration Example

- Set `I2C_MS_MODE` (master) to 1, and `I2C_MS_MODE` (slave) to 0.
- Write 1 to `I2C_CONF_UPGATE` (master) and `I2C_CONF_UPGATE` (slave) to synchronize registers.
- Configure command registers of I2C_{master}.

Command registers	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	ack_value	ack_exp	1	N+2
I2C_COMMAND2 (master)	STOP	—	—	—	—

4. Configure `I2C_SLAVE_ADDR` (slave) in `I2C_SLAVE_ADDR_REG` (slave) as $I2C_{\text{slave}}$'s 10-bit address, and set `I2C_ADDR_10BIT_EN` (slave) to 1 to enable 10-bit addressing.
5. Write the address of $I2C_{\text{slave}}$ and data to be sent to TX RAM of $I2C_{\text{master}}$. The first byte of the address of $I2C_{\text{slave}}$ comprises $((0x78 | I2C\_SLAVE\_ADDR[9:8]) \ll 1)$ and a $R/\overline{W}$ bit. The second byte of the address of $I2C_{\text{slave}}$ is `I2C_SLAVE_ADDR[7:0]`. These two bytes are followed by data to be sent in FIFO or non-FIFO mode.
6. Write 1 to `I2C_CONF_UPGATE` (master) and `I2C_CONF_UPGATE` (slave) to synchronize registers.
7. Write 1 to `I2C_TRANS_START` (master) and `I2C_TRANS_START` (slave) to start transfer.
8. $I2C_{\text{slave}}$ compares the slave address sent by $I2C_{\text{master}}$ with its own address in `I2C_SLAVE_ADDR` (slave). When `ack_check_en` (master) in $I2C_{\text{master}}$'s WRITE command is 1, $I2C_{\text{master}}$ checks ACK value each time it sends a byte. When `ack_check_en` (master) is 0, $I2C_{\text{master}}$ does not check the ACK value and take $I2C_{\text{slave}}$ as a matching slave by default.
 - Match: If the received ACK value matches `ack_exp` (master) (the expected ACK value), $I2C_{\text{master}}$ continues data transfer.
 - Not match: If the received ACK value does not match `ack_exp`, $I2C_{\text{master}}$ generates an `I2C_NACK_INT` (master) interrupt and stops data transfer.
9. $I2C_{\text{master}}$ sends data, and determines whether to check ACK value according to `ack_check_en` (master).
10. If data to be sent is larger than TX FIFO depth, TX RAM of $I2C_{\text{master}}$ may wrap around in FIFO mode. For details, please refer to Section 26.4.10.
11. If data to be received is larger than RX FIFO depth, RX RAM of $I2C_{\text{slave}}$ may wrap around in FIFO mode. For details, please refer to Section 26.4.10.

If data to be received is larger than RX FIFO depth, the other way is to enable clock stretching by setting `I2C_SLAVE_SCL_STRETCH_EN` (slave), and clearing `I2C_RX_FULL_ACK_LEVEL` to 0. When RX RAM is full, an `I2C_SLAVE_STRETCH_INT` (slave) interrupt is generated. In this way, $I2C_{\text{slave}}$ can hold SCL low, in exchange for more time to read data. After the software has finished reading, you can set `I2C_SLAVE_STRETCH_INT_CLR` (slave) to 1 to clear interrupt, and set `I2C_SLAVE_SCL_STRETCH_CLR` (slave) to release the SCL line.
12. After data transfer completes, $I2C_{\text{master}}$ executes the STOP command, and generates an `I2C_TRANS_COMPLETE_INT` (master) interrupt.

26.7.3 $I2C_{\text{master}}$ Writes to $I2C_{\text{slave}}$ with Two 7-bit Addresses in One Command Sequence

26.7.3.1 Introduction

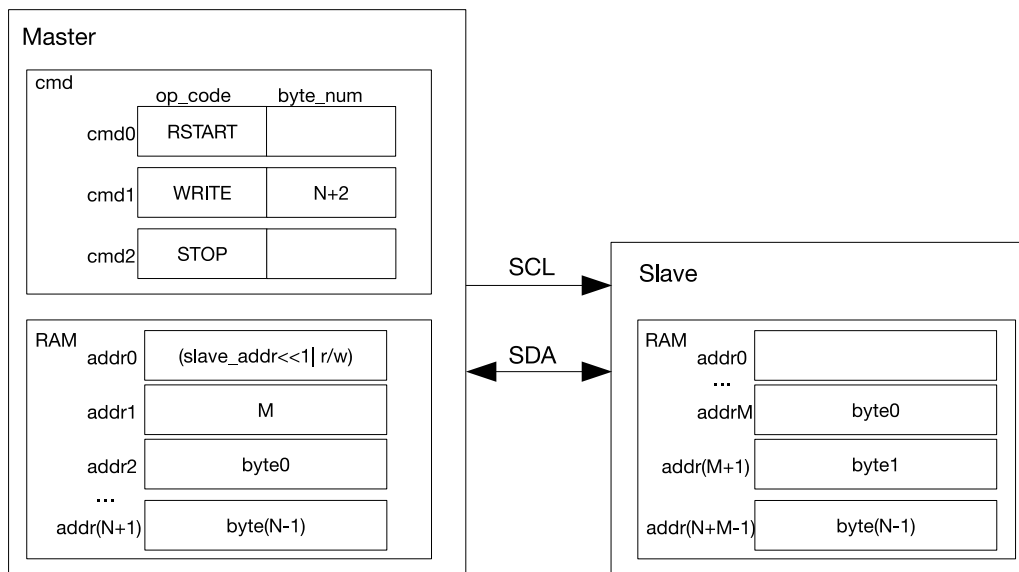


Figure 26.7-3. I2C_{master} Writing to I2C_{slave} with Two 7-bit Addresses

Figure 26.7-3 shows how I2C_{master} writes N bytes of data to I2C_{slave} registers or RAM using 7-bit double addressing. The configuration and transfer process is similar to what is described in Section 26.7.1, except that in 7-bit dual address mode I2C_{master} sends two 7-bit addresses. The first address is the address of an I2C slave, and the second one is I2C_{slave}'s memory address (i.e., addrM in Figure 26.7-3). When using double addressing, RAM must be accessed in non-FIFO mode. The I2C slave puts received byte0 ~ byte (N-1) into its RAM in an order starting from addrM. The RAM is overwritten every 32 bytes.

26.7.3.2 Configuration Example

1. Set I2C_MS_MODE (master) to 1, and I2C_MS_MODE (slave) to 0.
2. Set I2C_FIFO_ADDR_CFG_EN (slave) to 1 to enable dual address mode.
3. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
4. Configure command registers of I2C_{master}.

Command registers	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	ack_value	ack_exp	1	N+2
I2C_COMMAND2 (master)	STOP	—	—	—	—

5. Write the address of I2C_{slave} and data to be sent to TX RAM of I2C_{master} in FIFO or non-FIFO mode.
6. Write the address of I2C_{slave} to I2C_SLAVE_ADDR (slave) in I2C_SLAVE_ADDR_REG (slave) register.
7. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
8. Write 1 to I2C_TRANS_START (master) and I2C_TRANS_START (slave) to start transfer.
9. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in I2C_SLAVE_ADDR (slave).
When ack_check_en (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it

sends a byte. When `ack_check_en` (master) is 0, `I2Cmaster` does not check ACK value and take `I2Cslave` as a matching slave by default.

- Match: If the received ACK value matches `ack_exp` (master) (the expected ACK value), `I2Cmaster` continues data transfer.
- Not match: If the received ACK value does not match `ack_exp`, `I2Cmaster` generates an `I2C_NACK_INT` (master) interrupt and stops data transfer.

10. `I2Cslave` receives the RX RAM address sent by `I2Cmaster` and adds the offset.
11. `I2Cmaster` sends data, and determines whether to check ACK value according to `ack_check_en` (master).
12. If data to be sent is larger than TX FIFO depth, TX RAM of `I2Cmaster` may wrap around in FIFO mode. For details, please refer to Section [26.4.10](#).
13. If data to be received is larger than RX FIFO depth, you may enable clock stretching by setting `I2C_SLAVE_SCL_STRETCH_EN` (slave), and clearing `I2C_RX_FULL_ACK_LEVEL` to 0. When RX RAM is full, an `I2C_SLAVE_STRETCH_INT` (slave) interrupt is generated. In this way, `I2Cslave` can hold SCL low, in exchange for more time to read data. After the software has finished reading, you can set `I2C_SLAVE_STRETCH_INT_CLR` (slave) to 1 to clear interrupt, and set `I2C_SLAVE_SCL_STRETCH_CLR` (slave) to release the SCL line.
14. After data transfer completes, `I2Cmaster` executes the STOP command, and generates an `I2C_TRANS_COMPLETE_INT` (master) interrupt.

26.7.4 `I2Cmaster` Writes to `I2Cslave` with a 7-bit Address in Multiple Command Sequences

26.7.4.1 Introduction

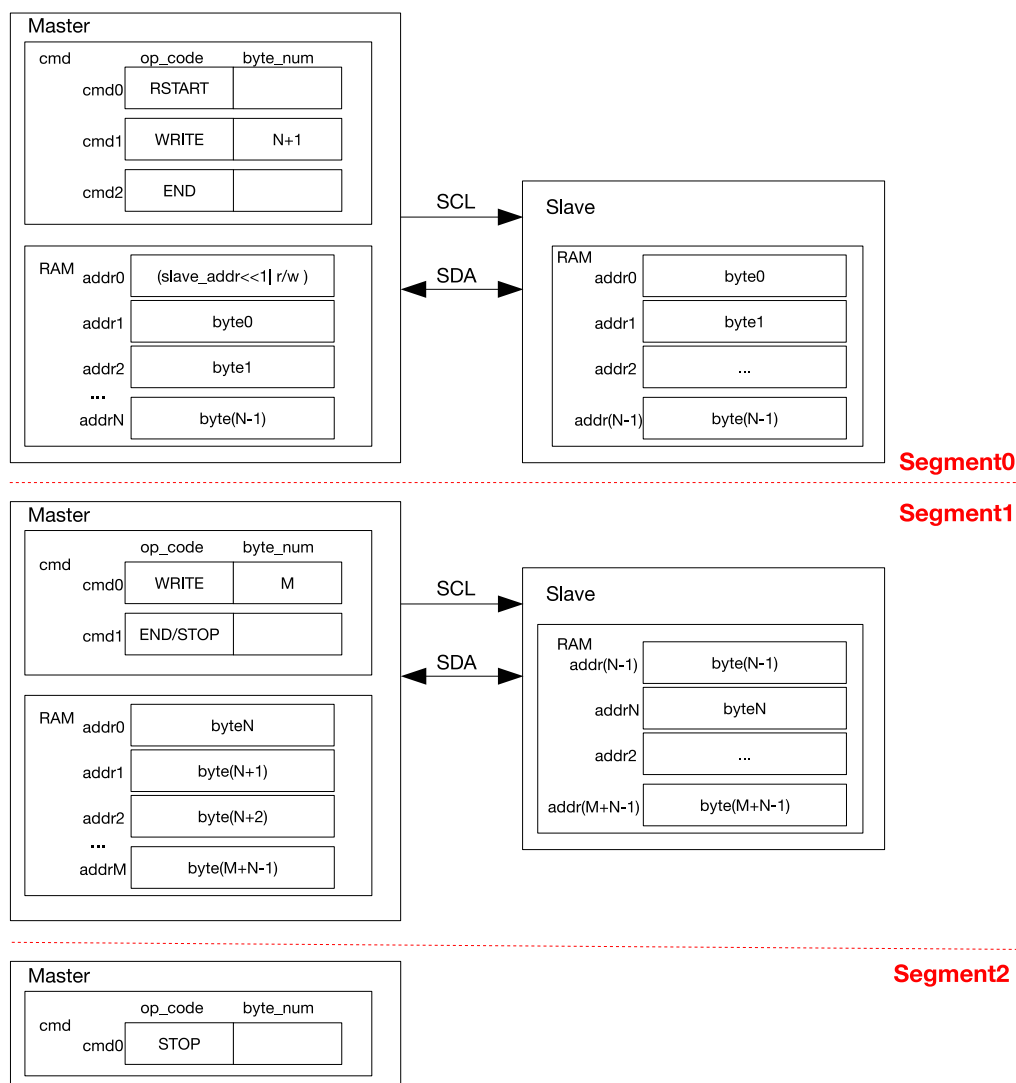


Figure 26.7-4. I2C_{master} Writing to I2C_{slave} with a 7-bit Address in Multiple Sequences

Given that the I2C Controller RAM holds only the size of TX/RX FIFO depth, when data are too large to be processed, it is advised to transmit them in multiple command sequences. Each command sequence ends with an END command. When the controller executes this END command, SCL will be pulled low, and the software can refresh command sequence registers and the RAM for the next transfer.

Figure 26.7-4 shows how I2C_{master} writes to an I2C slave in two or three segments as an example. For the first segment, the CMD_Controller registers are configured as shown in Segment0. Once data in I2C_{master}'s RAM is ready and I2C_TRANS_START is set, I2C_{master} initiates data transfer. After executing the END command, I2C_{master} turns off the SCL clock and pulls SCL low to reserve the bus. Meanwhile, the controller generates an I2C_END_DETECT_INT interrupt.

For the second segment, after detecting the I2C_END_DETECT_INT interrupt, the software refreshes the CMD_Controller registers, reloads the RAM, and clears this interrupt, as shown in Segment1. If cmd1 in the second segment is a STOP, then data is transmitted to I2C_{slave} in two segments. I2C_{master} resumes data transfer after I2C_TRANS_START is set, and terminates the transfer by sending a STOP bit.

For the third segment, after the second data transfer finishes and an I2C_END_DETECT_INT is detected, the

CMD_Controller registers of I2C_{master} are configured as shown in Segment2. Once [I2C_TRANS_START](#) is set, I2C_{master} generates a STOP bit and terminates the transfer.

Note that other I2C_{master}s will not transact on the bus between two segments. The bus is only released after a STOP command is sent. The I2C controller can be reset by setting [I2C_FSM_RST](#) field at any time. This field will later be cleared automatically by hardware.

26.7.4.2 Configuration Example

1. Set [I2C_MS_MODE](#) (master) to 1, and [I2C_MS_MODE](#) (slave) to 0.
2. Write 1 to [I2C_CONF_UPGATE](#) (master) and [I2C_CONF_UPGATE](#) (slave) to synchronize registers.
3. Configure command registers of I2C_{master}.

Command registers	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	ack_value	ack_exp	1	N+1
I2C_COMMAND2 (master)	END	—	—	—	—

4. Write the address of I2C_{slave} and data to be sent to TX RAM of I2C_{master} in either FIFO mode or non-FIFO mode according to Section [26.4.10](#).
5. Write the address of I2C_{slave} to [I2C_SLAVE_ADDR](#) (slave) in [I2C_SLAVE_ADDR_REG](#) (slave) register.
6. Write 1 to [I2C_CONF_UPGATE](#) (master) and [I2C_CONF_UPGATE](#) (slave) to synchronize registers.
7. Write 1 to [I2C_TRANS_START](#) (master) and [I2C_TRANS_START](#) (slave) to start transfer.
8. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in [I2C_SLAVE_ADDR](#) (slave). When [ack_check_en](#) (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it sends a byte. When [ack_check_en](#) (master) is 0, I2C_{master} does not check ACK value and take I2C_{slave} as a matching slave by default.
 - Match: If the received ACK value matches [ack_exp](#) (master) (the expected ACK value), I2C_{master} continues data transfer.
 - Not match: If the received ACK value does not match [ack_exp](#), I2C_{master} generates an [I2C_NACK_INT](#) (master) interrupt and stops data transfer.
9. I2C_{master} sends data, and checks ACK value or not according to [ack_check_en](#) (master).
10. After the [I2C_END_DETECT_INT](#) (master) interrupt is generated, set [I2C_END_DETECT_INT_CLR](#) (master) to 1 to clear this interrupt.
11. Update I2C_{master}'s command registers.

Command registers	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	WRITE	ack_value	ack_exp	1	M
I2C_COMMAND1 (master)	END/STOP	—	—	—	—

12. Write M bytes of data to be sent to TX RAM of I2C_{master} in FIFO or non-FIFO mode.
13. Write 1 to [I2C_TRANS_START](#) (master) bit to start the transfer and repeat step 9.

- 14. If the command is a STOP, I2C stops the transfer and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.
- 15. If the command is an END, repeat step 10.
- 16. Update I2C_{master}'s command registers.

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND1 (master)	STOP	—	—	—	—

- 17. Write 1 to I2C_TRANS_START (master) bit to start transfer.
- 18. I2C_{master} executes the STOP command and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.

26.7.5 I2C_{master} Reads I2C_{slave} with a 7-bit Address in One Command Sequence

26.7.5.1 Introduction

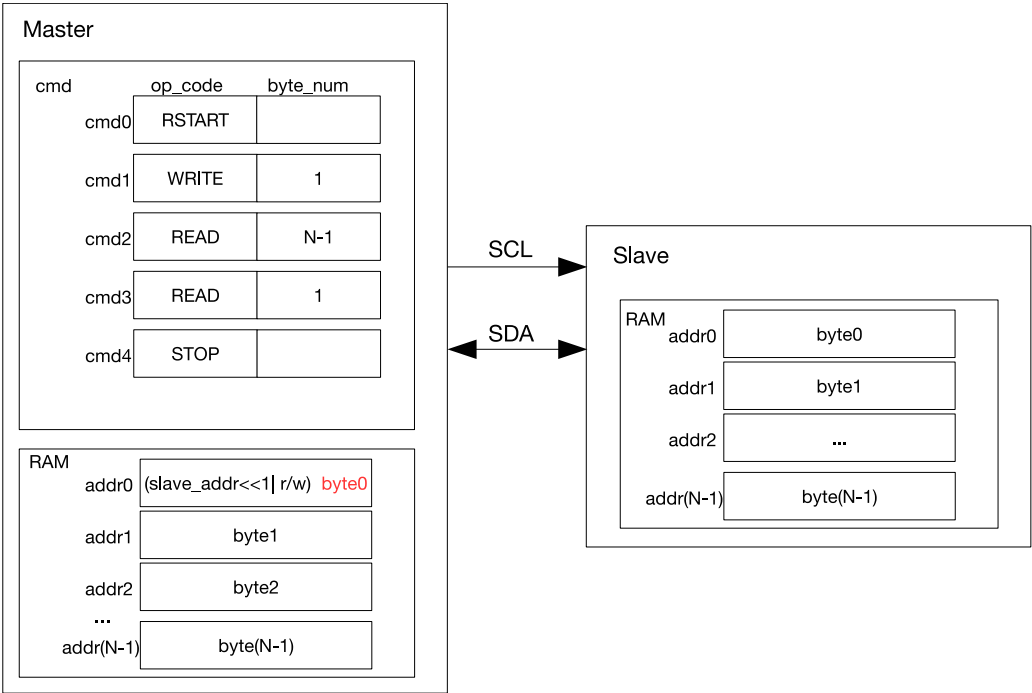


Figure 26.7-5. I2C_{master} Reading I2C_{slave} with a 7-bit Address

Figure 26.7-5 shows how I2C_{master} reads N bytes of data from an I2C slave using 7-bit addressing. cmd1 is a WRITE command, and when this command is executed I2C_{master} sends the address of I2C_{slave}. The byte sent comprises a 7-bit I2C_{slave} address and a $R/\overline{W}$ bit. When the $R/\overline{W}$ bit is 1, it indicates a READ operation. If the address of an I2C slave matches the sent address, this matching slave starts sending data to I2C_{master}. I2C_{master} generates acknowledgments according to ack_value defined in the READ command upon receiving a byte. As illustrated in Figure 26.7-5, I2C_{master} executes two READ commands: it generates ACKs for (N-1) bytes of data in cmd2, and a NACK for the last byte of data in cmd 3. This configuration may be changed as required.

I2C_{master} writes received data into the controller RAM from addr0, whose original content (the address of I2C_{slave} and a $R/\overline{W}$ bit) is overwritten by byte0 marked red in Figure 26.7-5.

26.7.5.2 Configuration Example

1. Set I2C_MS_MODE (master) to 1, and I2C_MS_MODE (slave) to 0.
2. We recommend setting I2C_SLAVE_SCL_STRETCH_EN (slave) to 1, so that SCL can be held low for more processing time when I2C_{slave} needs to send data. If this bit is not set, the software should write data to be sent to I2C_{slave}'s TX RAM before I2C_{master} initiates the transfer. The configuration below is applicable to the scenario where I2C_SLAVE_SCL_STRETCH_EN (slave) is 1.
3. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
4. Configure command registers of I2C_{master}.

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	0	0	1	1
I2C_COMMAND2 (master)	READ	0	0	1	N-1
I2C_COMMAND3 (master)	READ	1	0	1	1
I2C_COMMAND4 (master)	STOP	—	—	—	—

5. Write the address of I2C_{slave} to TX RAM of I2C_{master} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
6. Write the address of I2C_{slave} to I2C_SLAVE_ADDR (slave) in I2C_SLAVE_ADDR_REG (slave) register.
7. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
8. Write 1 to I2C_TRANS_START (master) bit to start I2C_{master}'s transfer.
9. Start I2C_{slave}'s transfer according to Section 26.4.14.
10. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in I2C_SLAVE_ADDR (slave). When ack_check_en (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it sends a byte. When ack_check_en (master) is 0, I2C_{master} does not check ACK value and take I2C_{slave} as a matching slave by default.
 - Match: If the received ACK value matches ack_exp (master) (the expected ACK value), I2C_{master} continues data transfer.
 - Not match: If the received ACK value does not match ack_exp, I2C_{master} generates an I2C_NACK_INT (master) interrupt and stops data transfer.
11. After I2C_SLAVE_STRETCH_INT (slave) is generated, the I2C_STRETCH_CAUSE bit is 0. The address of I2C_{slave} matches the address sent over SDA, and I2C_{slave} needs to send data.
12. Write data to be sent to TX RAM of I2C_{slave} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
13. Set I2C_SLAVE_SCL_STRETCH_CLR (slave) to 1 to release SCL.
14. I2C_{slave} sends data, and I2C_{master} checks ACK value or not according to ack_check_en (master) in the READ command.

15. If data to be read by I2C_{master} is larger than the TX FIFO depth of I2C_{slave}, an [I2C_SLAVE_STRETCH_INT](#) (slave) interrupt will be generated when TX RAM of I2C_{slave} becomes empty. In this way, I2C_{slave} can hold SCL low, so that software has more time to pad data in TX RAM of I2C_{slave} and read data in RX RAM of I2C_{master}. After software has finished reading, you can set [I2C_SLAVE_STRETCH_INT_CLR](#) (slave) to 1 to clear interrupt, and set [I2C_SLAVE_SCL_STRETCH_CLR](#) (slave) to release the SCL line.
16. After I2C_{master} has received the last byte of data, set ack_value (master) to 1. I2C_{slave} will stop the transfer once receiving the I2C_NACK_INT interrupt.
17. After data transfer completes, I2C_{master} executes the STOP command, and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.

26.7.6 I2C_{master} Reads I2C_{slave} with a 10-bit Address in One Command Sequence

26.7.6.1 Introduction

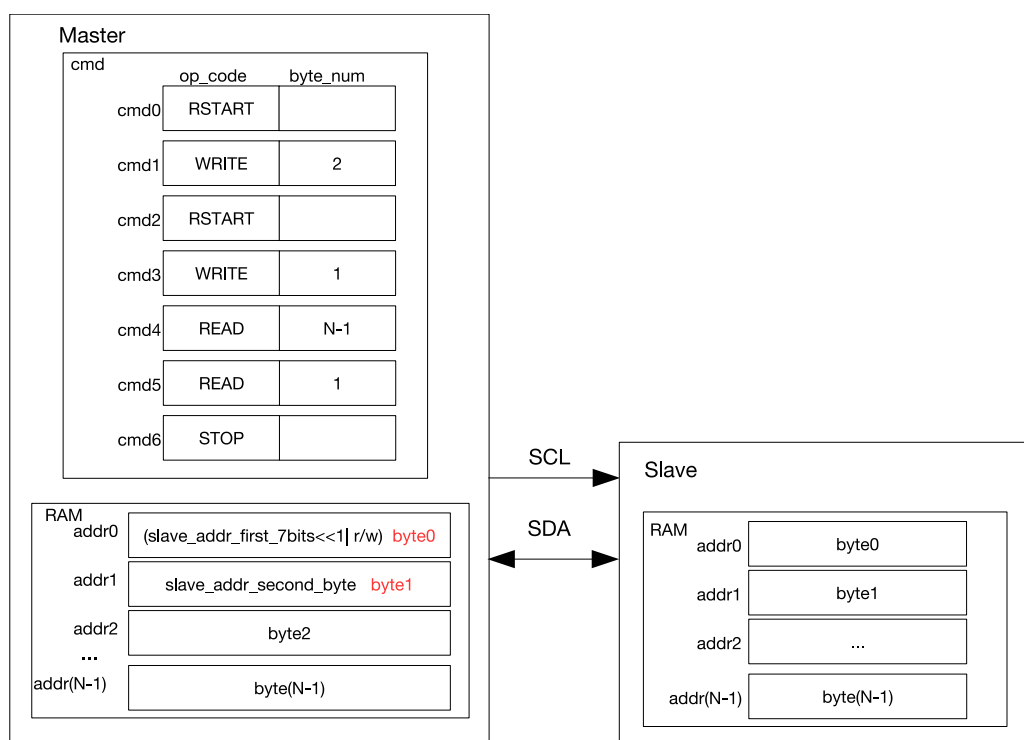


Figure 26.7-6. I2C_{master} Reading I2C_{slave} with a 10-bit Address

Figure 26.7-6 shows how I2C_{master} reads data from an I2C slave using 10-bit addressing. Unlike 7-bit addressing, in 10-bit addressing the WRITE command of the I2C_{master} is formed from two bytes, and correspondingly TX RAM of this master stores a 10-bit address of two bytes. The $R/\overline{W}$ bit in the first byte is 0, which indicates a WRITE operation. After a RSTART condition, I2C_{master} sends the first byte of address again to read data from I2C_{slave}, but the $R/\overline{W}$ bit is 1, which indicates a READ operation. The two address bytes can be configured as described in Section 26.7.2.

26.7.6.2 Configuration Example

1. Set [I2C_MS_MODE](#) (master) to 1, and [I2C_MS_MODE](#) (slave) to 0.

2. We recommend setting `I2C_SLAVE_SCL_STRETCH_EN` (slave) to 1, so that SCL can be held low for more processing time when `I2C_slave` needs to send data. If this bit is not set, the software should write data to be sent to `I2C_slave`'s TX RAM before `I2C_master` initiates the transfer. The configuration below is applicable to a scenario where `I2C_SLAVE_SCL_STRETCH_EN` (slave) is 1.
3. Write 1 to `I2C_CONF_UPGATE` (master) and `I2C_CONF_UPGATE` (slave) to synchronize registers.
4. Configure command registers of `I2C_master`.

Command registers of <code>I2C_master</code>	op_code	ack_value	ack_exp	ack_check_en	byte_num
<code>I2C_COMMAND0</code> (master)	RSTART	—	—	—	—
<code>I2C_COMMAND1</code> (master)	WRITE	0	0	1	2
<code>I2C_COMMAND2</code> (master)	RSTART	—	—	—	—
<code>I2C_COMMAND3</code> (master)	WRITE	0	0	1	1
<code>I2C_COMMAND4</code> (master)	READ	0	0	1	N-1
<code>I2C_COMMAND5</code> (master)	READ	1	0	1	1
<code>I2C_COMMAND6</code> (master)	STOP	—	—	—	—

5. Configure `I2C_SLAVE_ADDR` (slave) in `I2C_SLAVE_ADDR_REG` (slave) as `I2C_slave`'s 10-bit address, and set `I2C_ADDR_10BIT_EN` (slave) to 1 to enable 10-bit addressing.
6. Write the address of `I2C_slave` and data to be sent to TX RAM of `I2C_master` in either FIFO or non-FIFO mode. The first byte of address comprises $((0x78 | I2C\_SLAVE\_ADDR[9:8]) \ll 1)$ and a $R/\overline{W}$ bit, which is 1 and indicates a WRITE operation. The second byte of address is `I2C_SLAVE_ADDR[7:0]`. The third byte is $((0x78 | I2C\_SLAVE\_ADDR[9:8]) \ll 1)$ and a $R/\overline{W}$ bit, which is 1 and indicates a READ operation.
7. Write 1 to `I2C_CONF_UPGATE` (master) and `I2C_CONF_UPGATE` (slave) to synchronize registers.
8. Write 1 to `I2C_TRANS_START` (master) to start `I2C_master`'s transfer.
9. Start `I2C_slave`'s transfer according to Section 26.4.14.
10. `I2C_slave` compares the slave address sent by `I2C_master` with its own address in `I2C_SLAVE_ADDR` (slave). When `ack_check_en` (master) in `I2C_master`'s WRITE command is 1, `I2C_master` checks ACK value each time it sends a byte. When `ack_check_en` (master) is 0, `I2C_master` does not check ACK value and take `I2C_slave` as a matching slave by default.
 - Match: If the received ACK value matches `ack_exp` (master) (the expected ACK value), `I2C_master` continues data transfer.
 - Not match: If the received ACK value does not match `ack_exp`, `I2C_master` generates an `I2C_NACK_INT` (master) interrupt and stops data transfer.
11. `I2C_master` sends a RSTART and the third byte in TX RAM, which is $((0x78 | I2C\_SLAVE\_ADDR[9:8]) \ll 1)$ and a $R/\overline{W}$ bit that indicates READ.
12. `I2C_slave` repeats step 10. If its address matches the address sent by `I2C_master`, `I2C_slave` proceed on to the next steps.
13. After `I2C_SLAVE_STRETCH_INT` (slave) is generated, the `I2C_STRETCH_CAUSE` bit is 0. The address of `I2C_slave` matches the address sent over SDA, and `I2C_slave` needs to send data.

14. Write data to be sent to TX RAM of I2C_{slave} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
15. Set I2C_SLAVE_SCL_STRETCH_CLR (slave) to 1 to release SCL.
16. I2C_{slave} sends data, and I2C_{master} checks ACK value or not according to ack_check_en (master) in the READ command.
17. If data to be read by I2C_{master} is larger than the TX FIFO depth of I2C_{slave}, an I2C_SLAVE_STRETCH_INT (slave) interrupt will be generated when TX RAM of I2C_{slave} becomes empty. In this way, I2C_{slave} can hold SCL low, so that software has more time to pad data in TX RAM of I2C_{slave} and read data in RX RAM of I2C_{master}. After the software has finished reading, you can set I2C_SLAVE_STRETCH_INT_CLR (slave) to 1 to clear interrupt, and set I2C_SLAVE_SCL_STRETCH_CLR (slave) to release the SCL line.
18. After I2C_{master} has received the last byte of data, set ack_value (master) to 1. I2C_{slave} will stop transfer once receiving the I2C_NACK_INT interrupt.
19. After data transfer completes, I2C_{master} executes the STOP command, and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.

26.7.7 I2C_{master} Reads I2C_{slave} with Two 7-bit Addresses in One Command Sequence

26.7.7.1 Introduction

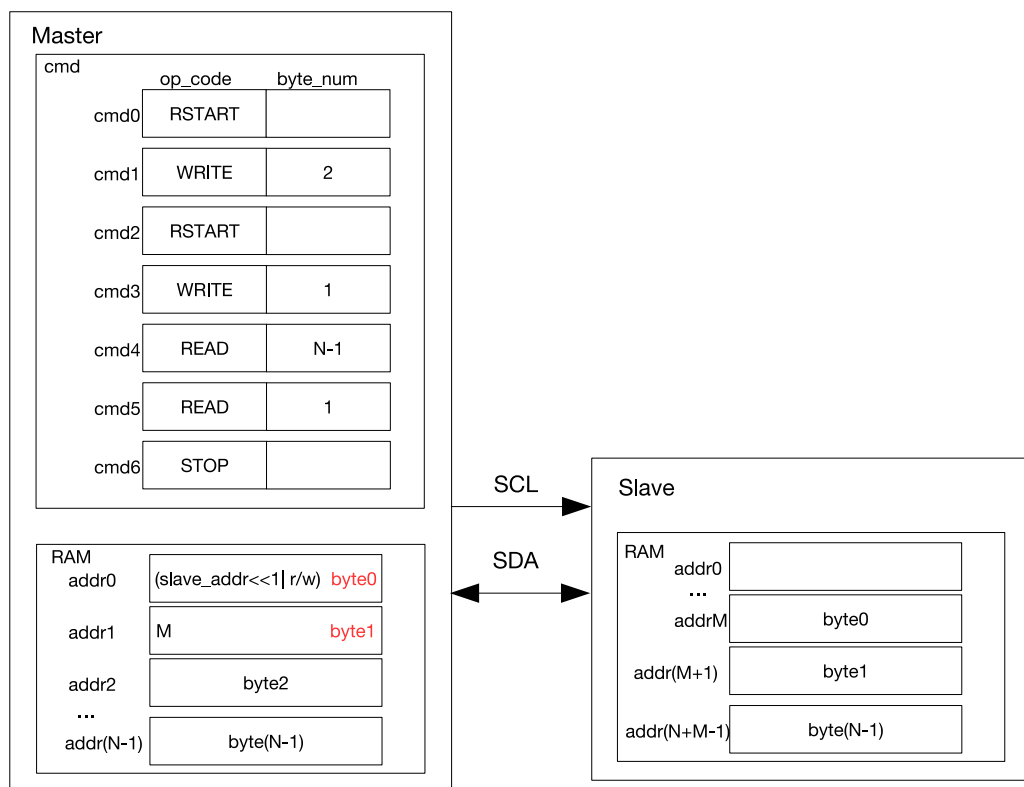


Figure 26.7-7. I2C_{master} Reading N Bytes of Data from addrM of I2C_{slave} with a 7-bit Address

Figure 26.7-7 shows how I2C_{master} reads data from specified addresses in an I2C slave. I2C_{master} sends two bytes of addresses: the first byte is a 7-bit I2C_{slave} address followed by a $R/\overline{W}$ bit, which is 0 and indicates a

WRITE; the second byte is I2C_{slave}'s memory address. After a RSTART condition, I2C_{master} sends the first byte of address again, but the $R/\overline{W}$ bit is 1 which indicates a READ. Then, I2C_{master} reads data starting from addrM.

26.7.7.2 Configuration Example

1. Set I2C_MS_MODE (master) to 1, and I2C_MS_MODE (slave) to 0.
2. We recommend setting I2C_SLAVE_SCL_STRETCH_EN (slave) to 1, so that SCL can be held low for more processing time when I2C_{slave} needs to send data. If this bit is not set, the software should write data to be sent to I2C_{slave}'s TX RAM before I2C_{master} initiates the transfer. The configuration below is applicable to the scenario where I2C_SLAVE_SCL_STRETCH_EN (slave) is 1.
3. Set I2C_FIFO_ADDR_CFG_EN (slave) to 1 to enable dual address mode.
4. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
5. Configure command registers of I2C_{master}.

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	0	0	1	2
I2C_COMMAND2 (master)	RSTART	—	—	—	—
I2C_COMMAND3 (master)	WRITE	0	0	1	1
I2C_COMMAND4 (master)	READ	0	0	1	N-1
I2C_COMMAND5 (master)	READ	1	0	1	1
I2C_COMMAND6 (master)	STOP	—	—	—	—

6. Configure I2C_SLAVE_ADDR (slave) in I2C_SLAVE_ADDR_REG (slave) register as I2C_{slave}'s 7-bit address, and set I2C_ADDR_10BIT_EN (slave) to 0 to enable 7-bit addressing.
7. Write the address of I2C_{slave} and data to be sent to TX RAM of I2C_{master} in either FIFO or non-FIFO mode according to Section 26.4.10. The first byte of address comprises (I2C_SLAVE_ADDR[6:0])«1) and a $R/\overline{W}$ bit, which is 0 and indicates a WRITE. The second byte of the address is memory address M of I2C_{slave}. The third byte is (I2C_SLAVE_ADDR[6:0])«1) and a $R/\overline{W}$ bit, which is 1 and indicates a READ.
8. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
9. Write 1 to I2C_TRANS_START (master) to start I2C_{master}'s transfer.
10. Start I2C_{slave}'s transfer according to Section 26.4.14.
11. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in I2C_SLAVE_ADDR (slave). When ack_check_en (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it sends a byte. When ack_check_en (master) is 0, I2C_{master} does not check ACK value and takes I2C_{slave} as a matching slave by default.
 - Match: If the received ACK value matches ack_exp (master) (the expected ACK value), I2C_{master} continues data transfer.
 - Not match: If the received ACK value does not match ack_exp, I2C_{master} generates an I2C_NACK_INT (master) interrupt and stops data transfer.
12. I2C_{slave} receives memory address sent by I2C_{master} and adds the offset.

13. I2C_{master} sends a RSTART and the third byte in TX RAM, which is ((0x78 | I2C_SLAVE_ADDR[9:8])«1) and a R bit.
14. I2C_{slave} repeats step 11. If its address matches the address sent by I2C_{master}, I2C_{slave} proceed on to the next steps.
15. After I2C_SLAVE_STRETCH_INT (slave) is generated, the I2C_STRETCH_CAUSE bit is 0. The address of I2C_{slave} matches the address sent over SDA, and I2C_{slave} needs to send data.
16. Write data to be sent to TX RAM of I2C_{slave} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
17. Set I2C_SLAVE_SCL_STRETCH_CLR (slave) to 1 to release SCL.
18. I2C_{slave} sends data, and I2C_{master} checks ACK value or not according to ack_check_en (master) in the READ command.
19. If data to be read by I2C_{master} is larger than the TX FIFO depth of I2C_{slave}, an I2C_SLAVE_STRETCH_INT (slave) interrupt will be generated when TX RAM of I2C_{slave} becomes empty. In this way, I2C_{slave} can hold SCL low, so that software has more time to pad data in TX RAM of I2C_{slave} and read data in RX RAM of I2C_{master}. After software has finished reading, you can set I2C_SLAVE_STRETCH_INT_CLR (slave) to 1 to clear interrupt, and set I2C_SLAVE_SCL_STRETCH_CLR (slave) to release the SCL line.
20. After I2C_{master} has received the last byte of data, set ack_value (master) to 1. I2C_{slave} will stop transfer once receiving the I2C_NACK_INT interrupt.
21. After data transfer completes, I2C_{master} executes the STOP command, and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.

26.7.8 I2C_{master} Reads I2C_{slave} with a 7-bit Address in Multiple Command Sequences

26.7.8.1 Introduction

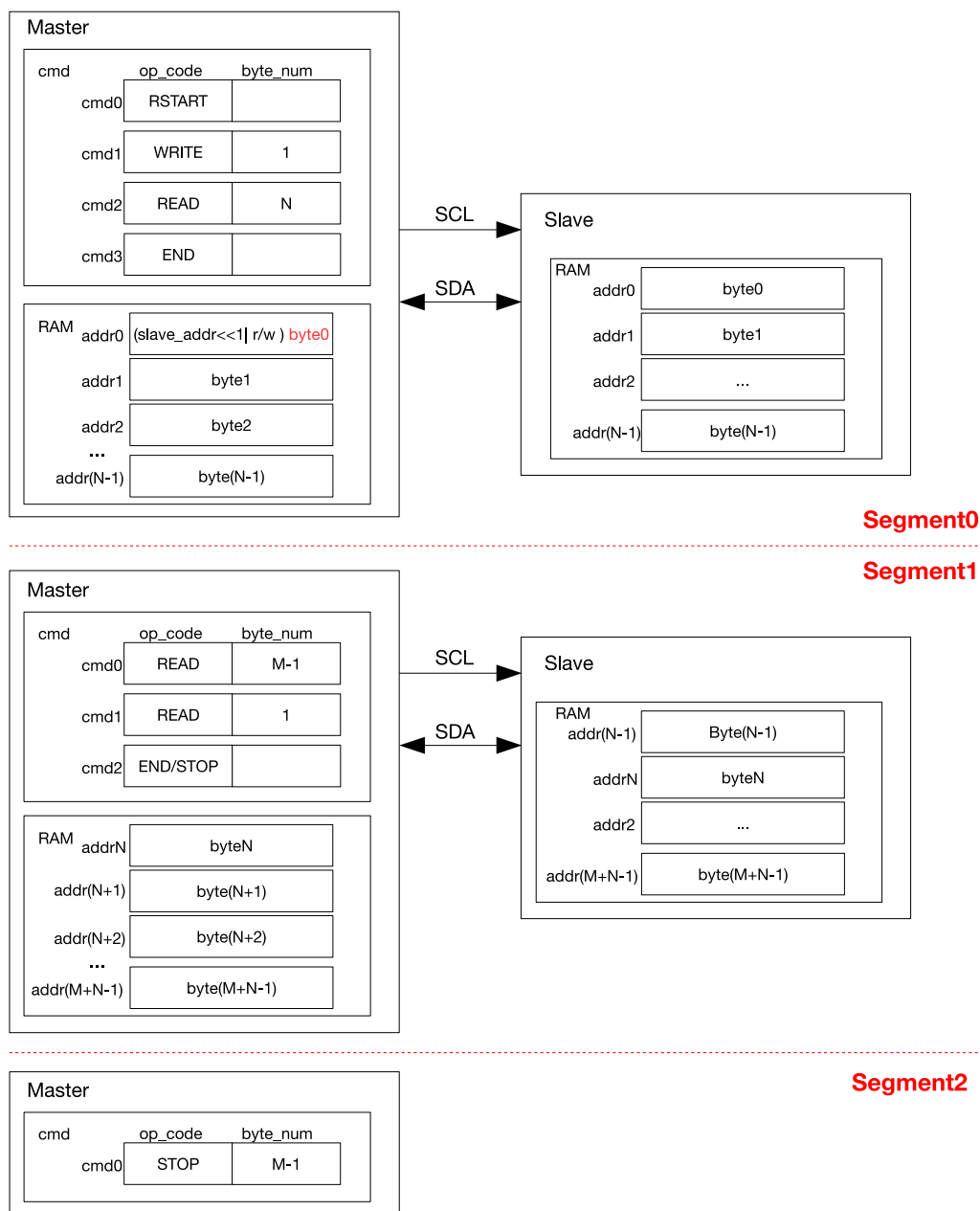


Figure 26.7-8. I2C_{master} Reading I2C_{slave} with a 7-bit Address in Segments

Figure 26.7-8 shows how I2C_{master} reads (N+M) bytes of data from an I2C slave in two/three segments separated by END commands. Configuration procedures are described as follows:

1. The procedures for Segment0 is similar to 26.7-5, except that the last command is an END.
2. Prepare data in the TX RAM of I2C_{slave}, and set I2C_TRANS_START to start data transfer. After executing the END command, I2C_{master} refreshes command registers and the RAM as shown in Segment1, and clears the corresponding I2C_END_DETECT_INT interrupt. If cmd2 in Segment1 is a STOP, then data is read from I2C_{slave} in two segments. I2C_{master} resumes data transfer by setting I2C_TRANS_START and terminates the transfer by sending a STOP bit.
3. If cmd2 in Segment1 is an END, then data is read from I2C_{slave} in three segments. After the second data

transfer finishes and an I2C_END_DETECT_INT interrupt is detected, the cmd box is configured as shown in Segment2. Once I2C_TRANS_START is set, I2C_{master} terminates the transfer by sending a STOP bit.

26.7.8.2 Configuration Example

1. Set I2C_MS_MODE (master) to 1, and I2C_MS_MODE (slave) to 0.
2. We recommend setting I2C_SLAVE_SCL_STRETCH_EN (slave) to 1, so that SCL can be held low for more processing time when I2C_{slave} needs to send data. If this bit is not set, the software should write data to be sent to I2C_{slave}'s TX RAM before I2C_{master} initiates the transfer. The configuration below is applicable to a scenario where I2C_SLAVE_SCL_STRETCH_EN (slave) is 1.
3. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
4. Configure command registers of I2C_{master}.

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	RSTART	—	—	—	—
I2C_COMMAND1 (master)	WRITE	0	0	1	1
I2C_COMMAND2 (master)	READ	0	0	1	N
I2C_COMMAND3 (master)	END	—	—	—	—

5. Write the address of I2C_{slave} to TX RAM of I2C_{master} in FIFO or non-FIFO mode.
6. Write the address of I2C_{slave} to I2C_SLAVE_ADDR (slave) in I2C_SLAVE_ADDR_REG (slave) register.
7. Write 1 to I2C_CONF_UPGATE (master) and I2C_CONF_UPGATE (slave) to synchronize registers.
8. Write 1 to I2C_TRANS_START (master) to start I2C_{master}'s transfer.
9. Start I2C_{slave}'s transfer according to Section 26.4.14.
10. I2C_{slave} compares the slave address sent by I2C_{master} with its own address in I2C_SLAVE_ADDR (slave). When ack_check_en (master) in I2C_{master}'s WRITE command is 1, I2C_{master} checks ACK value each time it sends a byte. When ack_check_en (master) is 0, I2C_{master} does not check the ACK value and takes I2C_{slave} as a matching slave by default.
 - Match: If the received ACK value matches ack_exp (master) (the expected ACK value), I2C_{master} continues data transfer.
 - Not match: If the received ACK value does not match ack_exp, I2C_{master} generates an I2C_NACK_INT (master) interrupt and stops data transfer.
11. After I2C_SLAVE_STRETCH_INT (slave) is generated, the I2C_STRETCH_CAUSE bit is 0. The address of I2C_{slave} matches the address sent over SDA, and I2C_{slave} needs to send data.
12. Write data to be sent to TX RAM of I2C_{slave} in either FIFO mode or non-FIFO mode according to Section 26.4.10.
13. Set I2C_SLAVE_SCL_STRETCH_CLR (slave) to 1 to release SCL.
14. I2C_{slave} sends data, and I2C_{master} checks ACK value or not according to ack_check_en (master) in the READ command.
15. If data to be read by I2C_{master} in one READ command (N or M) is larger than the TX FIFO depth of I2C_{slave}, an I2C_SLAVE_STRETCH_INT (slave) interrupt will be generated when TX RAM of I2C_{slave} becomes

empty. In this way, I2C_{slave} can hold SCL low so that software has more time to pad data in TX RAM of I2C_{slave} and read data in RX RAM of I2C_{master}. After the software has finished reading, you can set [I2C_SLAVE_STRETCH_INT_CLR](#) (slave) to 1 to clear interrupt, and set [I2C_SLAVE_SCL_STRETCH_CLR](#) (slave) to release the SCL line.

16. Once finishing reading data in the first READ command, I2C_{master} executes the END command and triggers an I2C_END_DETECT_INT (master) interrupt, which is cleared by setting [I2C_END_DETECT_INT_CLR](#) (master) to 1.
17. Update I2C_{master}'s command registers using one of the following two methods:

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	READ	ack_value	ack_exp	1	M
I2C_COMMAND1 (master)	END	—	—	—	—

Or

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND0 (master)	READ	0	0	1	M-1
I2C_COMMAND0 (master)	READ	1	0	1	1
I2C_COMMAND1 (master)	STOP	—	—	—	—

18. Write M bytes of data to be sent to TX RAM of I2C_{slave}. If M is larger than the TX FIFO depth, then repeat step 12 in FIFO or non-FIFO mode.
19. Write 1 to [I2C_TRANS_START](#) (master) bit to start the transfer and repeat step 14.
20. If the last command is a STOP, then set ack_value (master) to 1 after I2C_{master} has received the last byte of data. I2C_{slave} stops transfer upon the I2C_NACK_INT interrupt. I2C_{master} executes the STOP command to stop the transfer and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.
21. If the last command is an END, then repeat step 16 and proceed on to the next steps.
22. Update I2C_{master}'s command registers.

Command registers of I2C _{master}	op_code	ack_value	ack_exp	ack_check_en	byte_num
I2C_COMMAND1 (master)	STOP	—	—	—	—

23. Write 1 to [I2C_TRANS_START](#) (master) bit to start transfer.
24. I2C_{master} executes the STOP command to stop transfer, and generates an I2C_TRANS_COMPLETE_INT (master) interrupt.

26.8 Register Summary

26.8.1 I2C Register Summary

The addresses in this section are relative to the I2C Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Timing registers			
I2C_SCL_LOW_PERIOD_REG	Configures the low level width of the SCL Clock	0x0000	R/W
I2C_SDA_HOLD_REG	Configures the hold time after a negative SCL edge	0x0030	R/W
I2C_SDA_SAMPLE_REG	Configures the sample time after a positive SCL edge	0x0034	R/W
I2C_SCL_HIGH_PERIOD_REG	Configures the high level width of SCL	0x0038	R/W
I2C_SCL_START_HOLD_REG	Configures the delay between the SDA and SCL negative edge for a start condition	0x0040	R/W
I2C_SCL_RSTART_SETUP_REG	Configures the delay between the positive edge of SCL and the negative edge of SDA	0x0044	R/W
I2C_SCL_STOP_HOLD_REG	Configures the delay after the SCL clock edge for a stop condition	0x0048	R/W
I2C_SCL_STOP_SETUP_REG	Configures the delay between the SDA and SCL rising edge for a stop condition Measurement unit: i2c_sclk	0x004C	R/W
I2C_SCL_ST_TIME_OUT_REG	SCL status time out register	0x0078	R/W
I2C_SCL_MAIN_ST_TIME_OUT_REG	SCL main status time out register	0x007C	R/W
Configuration registers			
I2C_CTR_REG	Transmission setting	0x0004	varies
I2C_TO_REG	Setting time out control for receiving data	0x000C	R/W
I2C_SLAVE_ADDR_REG	Local slave address setting	0x0010	R/W
I2C_FIFO_CONF_REG	FIFO configuration register	0x0018	R/W
I2C_FILTER_CFG_REG	SCL and SDA filter configuration register	0x0050	R/W
I2C_SCL_SP_CONF_REG	Power configuration register	0x0080	varies
I2C_SCL_STRETCH_CONF_REG	Set SCL stretch of I2C slave	0x0084	varies
Status registers			
I2C_SR_REG	Describe I2C work status	0x0008	RO
I2C_FIFO_ST_REG	FIFO status register	0x0014	RO
I2C_DATA_REG	Rx FIFO read data	0x001C	HRO
Interrupt registers			
I2C_INT_RAW_REG	Raw interrupt status	0x0020	R/SS WTC
I2C_INT_CLR_REG	Interrupt clear bits	0x0024	WT

Name	Description	Address	Access
I2C_INT_ENA_REG	Interrupt enable bits	0x0028	R/W
I2C_INT_STATUS_REG	Status of captured I2C communication events	0x002C	RO
Command registers			
I2C_COMD0_REG	I2C command register 0	0x0058	varies
I2C_COMD1_REG	I2C command register 1	0x005C	varies
I2C_COMD2_REG	I2C command register 2	0x0060	varies
I2C_COMD3_REG	I2C command register 3	0x0064	varies
I2C_COMD4_REG	I2C command register 4	0x0068	varies
I2C_COMD5_REG	I2C command register 5	0x006C	varies
I2C_COMD6_REG	I2C command register 6	0x0070	varies
I2C_COMD7_REG	I2C command register 7	0x0074	varies
Version register			
I2C_DATE_REG	Version register	0x00F8	R/W
Address register			
I2C_TXFIFO_START_ADDR_REG	I2C TX FIFO base address register	0x0100	HRO
I2C_RXFIFO_START_ADDR_REG	I2C RX FIFO base address register	0x0180	HRO

26.8.2 LP_I2C Register Summary

The abbreviations given in Column **Access** are explained in Section [Access Types for Registers](#).

Name	Description	Address	Access
Timing Registers			
LP_I2C_SCL_LOW_PERIOD_REG	Configures the low level width of the SCL Clock	0x0000	R/W
LP_I2C_SDA_HOLD_REG	Configures the hold time after a negative SCL edge	0x0030	R/W
LP_I2C_SDA_SAMPLE_REG	Configures the sample time after a positive SCL edge	0x0034	R/W
LP_I2C_SCL_HIGH_PERIOD_REG	Configures the high level width of SCL	0x0038	R/W
LP_I2C_SCL_START_HOLD_REG	Configures the delay between the SDA and SCL negative edge for a start condition	0x0040	R/W
LP_I2C_SCL_RSTART_SETUP_REG	Configures the delay between the positive edge of SCL and the negative edge of SDA	0x0044	R/W
LP_I2C_SCL_STOP_HOLD_REG	Configures the delay after the SCL clock edge for a stop condition	0x0048	R/W
LP_I2C_SCL_STOP_SETUP_REG	Configures the delay between the SDA and SCL positive edge for a stop condition	0x004C	R/W
LP_I2C_SCL_ST_TIME_OUT_REG	SCL status time out register	0x0078	R/W
LP_I2C_SCL_MAIN_ST_TIME_OUT_REG	SCL main status time out register	0x007C	R/W
Configuration Registers			
LP_I2C_CTR_REG	Transmission setting	0x0004	varies
LP_I2C_TO_REG	Setting time out control for receiving data	0x000C	R/W
LP_I2C_FIFO_CONF_REG	FIFO configuration register	0x0018	R/W
LP_I2C_FILTER_CFG_REG	SCL and SDA filter configuration register	0x0050	R/W

Name	Description	Address	Access
LP_I2C_SCL_SP_CONF_REG	Power configuration register	0x0080	varies
Status Registers			
LP_I2C_SR_REG	Describe I2C work status	0x0008	RO
LP_I2C_FIFO_ST_REG	FIFO status register	0x0014	RO
LP_I2C_DATA_REG	Rx FIFO read data	0x001C	RO
Interrupt Registers			
LP_I2C_INT_RAW_REG	Raw interrupt status	0x0020	R/SS WTC
LP_I2C_INT_CLR_REG	Interrupt clear bits	0x0024	WT
LP_I2C_INT_ENA_REG	Interrupt enable bits	0x0028	R/W
LP_I2C_INT_STATUS_REG	Status of captured I2C communication events	0x002C	RO
Command Registers			
LP_I2C_COMD0_REG	I2C command register 0	0x0058	varies
LP_I2C_COMD1_REG	I2C command register 1	0x005C	varies
LP_I2C_COMD2_REG	I2C command register 2	0x0060	varies
LP_I2C_COMD3_REG	I2C command register 3	0x0064	varies
LP_I2C_COMD4_REG	I2C command register 4	0x0068	varies
LP_I2C_COMD5_REG	I2C command register 5	0x006C	varies
LP_I2C_COMD6_REG	I2C command register 6	0x0070	varies
LP_I2C_COMD7_REG	I2C command register 7	0x0074	varies
Version Register			
LP_I2C_DATE_REG	Version register	0x00F8	R/W
Address Registers			
LP_I2C_TXFIFO_START_ADDR_REG	I2C TXFIFO base address register	0x0100	HRO
LP_I2C_RXFIFO_START_ADDR_REG	I2C RXFIFO base address register	0x0180	HRO

26.9 Registers

26.9.1 I2C Registers

The addresses in this section are relative to the I2C Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 26.1. I2C_SCL_LOW_PERIOD_REG (0x0000)

(reserved)																								I2C_SCL_LOW_PERIOD																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
31																								9	8	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2C_SCL_LOW_PERIOD Configures the low level width of the SCL clock in master mode.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.2. I2C_SDA_HOLD_REG (0x0030)

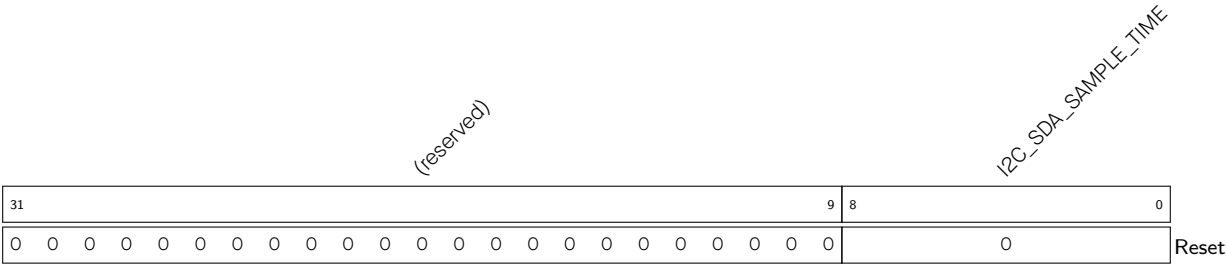
(reserved)																								I2C_SDA_HOLD_TIME															
31																								8								0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0								Reset							

I2C_SDA_HOLD_TIME Configures the time to hold the data after the falling edge of SCL.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.3. I2C_SDA_SAMPLE_REG (0x0034)



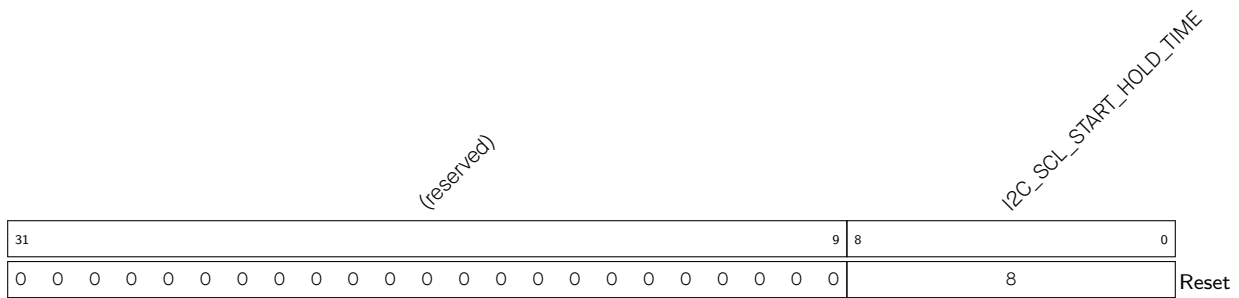
I2C_SDA_SAMPLE_TIME Configures the time for sampling SDA.
Measurement unit: I2C_SCLK clock cycles
(R/W)

Register 26.4. I2C_SCL_HIGH_PERIOD_REG (0x0038)



I2C_SCL_HIGH_PERIOD Configures for how long SCL remains high in master mode.
Measurement unit: I2C_SCLK clock cycles
(R/W)

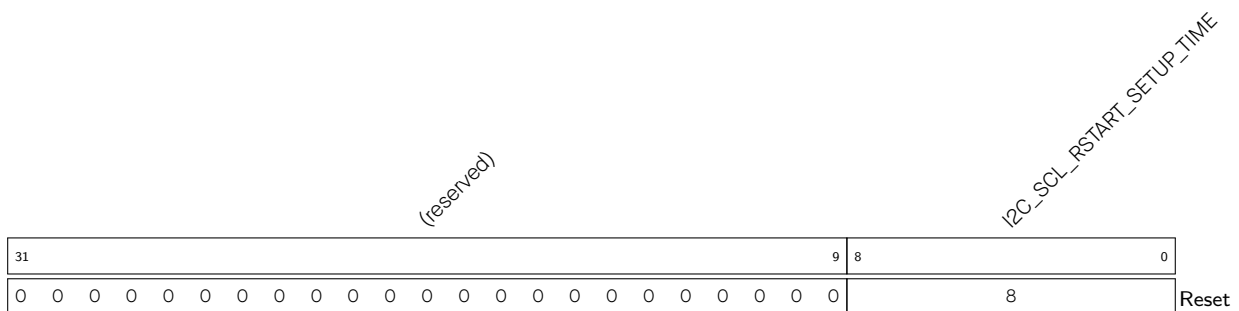
I2C_SCL_WAIT_HIGH_PERIOD Configures the SCL_FSM's waiting period for SCL high level in master mode.
Measurement unit: I2C_SCLK clock cycles
(R/W)

Register 26.5. I2C_SCL_START_HOLD_REG (0x0040)

I2C_SCL_START_HOLD_TIME Configures the time between the falling edge of SDA and the falling edge of SCL for a START condition.

Measurement unit: I2C_SCLK clock cycles

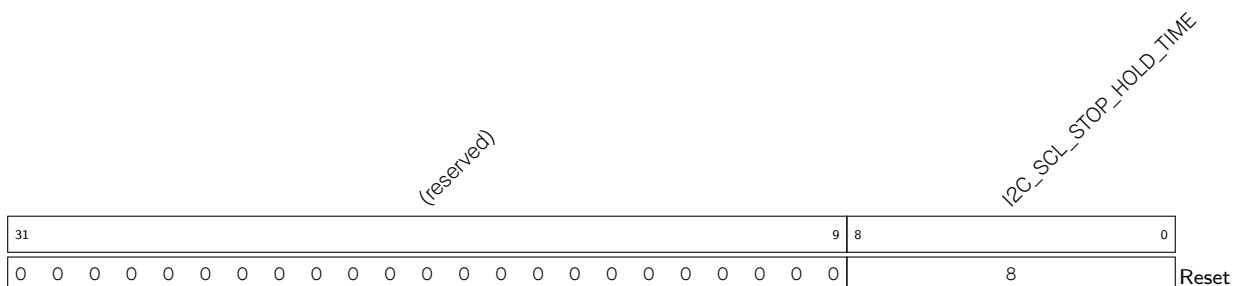
(R/W)

Register 26.6. I2C_SCL_RSTART_SETUP_REG (0x0044)

I2C_SCL_RSTART_SETUP_TIME Configures the time between the rising edge of SCL and the negative edge of SDA for a RESTART condition.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.7. I2C_SCL_STOP_HOLD_REG (0x0048)

I2C_SCL_STOP_HOLD_TIME Configures the delay after the STOP condition.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.8. I2C_SCL_STOP_SETUP_REG (0x004C)

(reserved)																I2C_SCL_STOP_SETUP_TIME																
31																0																
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																8																Reset

I2C_SCL_STOP_SETUP_TIME Configures the time between the rising edge of SCL and the rising edge of SDA.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.9. I2C_SCL_ST_TIME_OUT_REG (0x0078)

(reserved)																I2C_SCL_ST_TO_I2C																	
31																5	4	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x10																Reset	

I2C_SCL_ST_TO_I2C Configures the threshold value of SCL_FSM state unchanged period. It should be no more than 23. The actual period is $2^{(\text{reg\_time\_out\_value}+1)}$.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.10. I2C_SCL_MAIN_ST_TIME_OUT_REG (0x007C)

(reserved)																												I2C_SCL_MAIN_ST_TO_I2C					
31																												5	4	0			
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0x10				Reset	

I2C_SCL_MAIN_ST_TO_I2C Configures the threshold value of SCL_MAIN_FSM state unchanged period. It should be no more than 23. The actual period is $2^{(\text{I2C\_SCL\_MAIN\_ST\_TO\_I2C}+1)}$.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.11. I2C_CTR_REG (0x0004)

(reserved)																I2C_ADDR_BROADCASTING_EN I2C_ADDR_10BIT_RW_CHECK_EN I2C_SLV_TX_AUTO_START_EN I2C_CONF_UPGATE I2C_FSM_RST I2C_ARBITRATION_EN I2C_CLK_EN I2C_RX_LSB_FIRST I2C_TX_LSB_FIRST I2C_TRANS_START I2C_MS_MODE I2C_RX_FULL_ACK_LEVEL I2C_SAMPLE_SCL_LEVEL I2C_SCL_FORCE_OUT I2C_SDA_FORCE_OUT															
31																15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	Reset			

I2C_SDA_FORCE_OUT Configures the SDA output mode.

0: Open drain output

1: Direct output

(R/W)

I2C_SCL_FORCE_OUT Configures the SCL output mode.

0: Open drain output

1: Direct output

(R/W)

I2C_SAMPLE_SCL_LEVEL Configures the sample mode for SDA.

0: Sample SDA data on the SCL high level

1: Sample SDA data on the SCL low level

(R/W)

I2C_RX_FULL_ACK_LEVEL Configures the ACK value that needs to be sent by the master when the received RX RAM has reached the threshold.

(R/W)

I2C_MS_MODE Configures the module as an I2C Master or Slave.

0: Slave

1: Master

(R/W)

I2C_TRANS_START Configures whether the slave starts sending the data in TX FIFO.

0: No effect

1: Start

(WT)

I2C_TX_LSB_FIRST Configures to control the sending order for data needing to be sent.

0: send data from the most significant bit

1: send data from the least significant bit

(R/W)

I2C_RX_LSB_FIRST Configures to control the storage order for received data.

0: receive data from the most significant bit

1: receive data from the least significant bit

(R/W)

Continued on the next page...

Register 26.11. I2C_CTR_REG (0x0004)

Continued from the previous page...

I2C_CLK_EN Configures whether to gate clock signal for registers.

0: Support clock only when registers are read or written to by software

1: Force clock on for registers

(R/W)

I2C_ARBITRATION_EN Configures to enable I2C bus arbitration detection.

0: No effect

1: Enable

(R/W)

I2C_FSM_RST Configures to reset the SCL_FSM.

0: No effect

1: Reset (WT)

I2C_CONF_UPGATE Configures this bit for synchronization.

0: No effect

1: Synchronize (WT)

I2C_SLV_TX_AUTO_START_EN Configures to enable slave to send data automatically

0: Disable

1: Enable

(R/W)

I2C_ADDR_10BIT_RW_CHECK_EN Configures to check if the R/W bit of 10-bit addressing consists with I2C protocol.

0: Not check

1: Check (R/W)

I2C_ADDR_BROADCASTING_EN Configures to support the 7-bit general call function.

0: Not support

1: Support

(R/W)

Register 26.12. I2C_TO_REG (0x000C)

(reserved)																												I2C_TIME_OUT_EN		I2C_TIME_OUT_VALUE		
31																												6	5	4	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0	0x10		Reset	

I2C_TIME_OUT_VALUE Configures the timeout threshold period for SCL stuck at high or low level.

The actual period is $2^{(I2C\_TIME\_OUT\_VALUE+1)}$.

Measurement unit: I2C_SCLK clock cycles

(R/W)

I2C_TIME_OUT_EN Configures to enable time out control.

0: No effect

1: Enable

(R/W)

Register 26.13. I2C_SLAVE_ADDR_REG (0x0010)

I2C_ADDR_10BIT_EN																(reserved)																I2C_SLAVE_ADDR															
31	30														15	14														0																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														Reset																

I2C_SLAVE_ADDR Configure the slave address of I2C Slave.

(R/W)

I2C_ADDR_10BIT_EN Configures to enable the slave 10-bit addressing mode in master mode.

0: No effect

1: Enable

(R/W)

Register 26.14. I2C_FIFO_CONF_REG (0x0018)

(reserved)																I2C_FIFO_PRT_EN				I2C_TX_FIFO_RST				I2C_RX_FIFO_RST				I2C_FIFO_ADDR_CFG_EN				I2C_NONFIFO_EN				I2C_TXFIFO_WM_THRHD				I2C_RXFIFO_WM_THRHD			
31																15	14	13	12	11	10	9					5	4					0										
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																1	0	0	0	0	0	0x4				0xb				Reset													

Reset

I2C_RXFIFO_WM_THRHD Configures the watermark threshold of RX FIFO in non-FIFO access mode. When I2C_FIFO_PRT_EN is 1 and RX FIFO counter is bigger than I2C_RXFIFO_WM_THRHD[4:0], I2C_RXFIFO_WM_INT_RAW bit will be valid. (R/W)

I2C_TXFIFO_WM_THRHD Configures the watermark threshold of TX FIFO in non-FIFO access mode. When I2C_FIFO_PRT_EN is 1 and TC FIFO counter is bigger than I2C_TXFIFO_WM_THRHD[4:0], I2C_TXFIFO_WM_INT_RAW bit will be valid. (R/W)

I2C_NONFIFO_EN Configures to enable APB non-FIFO access. (R/W)

I2C_FIFO_ADDR_CFG_EN Configures the slave to enable dual address mode. When this mode is enabled, the byte received after the I2C address byte represents the offset address in the I2C Slave RAM.
 0: Disable
 1: Enable
 (R/W)

I2C_RX_FIFO_RST Configures to reset RX FIFO.
 0: No effect
 1: Reset
 (R/W)

I2C_TX_FIFO_RST Configures to reset TX FIFO.
 0: No effect
 1: Reset
 (R/W)

I2C_FIFO_PRT_EN Configures to enable FIFO pointer in non-FIFO access mode. This bit controls the valid bits and the TX/RX FIFO overflow, underflow, full and empty interrupts.
 0: No effect
 1: Enable
 (R/W)

Register 26.15. I2C_FILTER_CFG_REG (0x0050)

(reserved)																								I2C_SDA_FILTER_EN		I2C_SCL_FILTER_EN		I2C_SDA_FILTER_THRES		I2C_SCL_FILTER_THRES				
31																								10		9	8	7		4		3	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								1	1	0		0		Reset				

I2C_SCL_FILTER_THRES Configures the threshold pulse width to be filtered on SCL. When a pulse on the SCL input has smaller width than this register value, the I2C controller will ignore that pulse. Measurement unit: I2C_SCLK clock cycles
(R/W)

I2C_SDA_FILTER_THRES Configures the threshold pulse width to be filtered on SDA. When a pulse on the SDA input has smaller width than this register value, the I2C controller will ignore that pulse. Measurement unit: I2C_SCLK clock cycles
(R/W)

I2C_SCL_FILTER_EN Configures to enable the filter function for SCL.
 0: No effect
 1: Enable
 (R/W)

I2C_SDA_FILTER_EN Configures to enable the filter function for SDA.
 0: No effect
 1: Enable
 (R/W)

Register 26.16. I2C_SCL_SP_CONF_REG (0x0080)

(reserved)																								I2C_SDA_PD_EN		I2C_SCL_PD_EN		I2C_SCL_RST_SLV_NUM		I2C_SCL_RST_SLV_EN	
31																								8	7	6	5	1			0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0	0	0			0	Reset	

I2C_SCL_RST_SLV_EN Configures to send out SCL pulses when I2C master is IDLE. The number of pulses equals to [I2C_SCL_RST_SLV_NUM](#).

0: Invalid

1: Send out SCL pulses

(R/W/SC)

I2C_SCL_RST_SLV_NUM Configure the pulses of SCL generated in I2C master mode.

Valid when [I2C_SCL_RST_SLV_EN](#) is 1.

Measurement unit: I2C_SCLK clock cycles

(R/W)

I2C_SCL_PD_EN Configures to power down the I2C output SCL line.

0: Not power down.

1: Not work and power down.

Valid only when [I2C_SCL_FORCE_OUT](#) is 1. (R/W)

I2C_SDA_PD_EN Configures to power down the I2C output SDA line.

0: Not power down.

1: Not work and power down.

Valid only when [I2C_SDA_FORCE_OUT](#) is 1. (R/W)

Register 26.17. I2C_SCL_STRETCH_CONF_REG (0x0084)

(reserved)														I2C_SLAVE_BYTE_ACK_LVL I2C_SLAVE_BYTE_ACK_CTL_EN I2C_SLAVE_SCL_STRETCH_CLR I2C_SLAVE_SCL_STRETCH_EN							I2C_STRETCH_PROTECT_NUM																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
31														14	13	12	11	10	9											0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2C_STRETCH_PROTECT_NUM Configures the time period to release the SCL line from stretching to avoid timing violation. Usually it should be larger than the SDA setup time.

Measurement unit: I2C_SCLK clock cycles

(R/W)

I2C_SLAVE_SCL_STRETCH_EN Configures to enable slave SCL stretch function. The SCL output line will be stretched low when I2C_SLAVE_SCL_STRETCH_EN is 1 and a stretch event happens. The stretch cause can be seen in [I2C_STRETCH_CAUSE](#).

0: Disable

1: Enable

(R/W)

I2C_SLAVE_SCL_STRETCH_CLR Configures to clear the I2C slave SCL stretch function.

0: No effect

1: Clear

(WT)

I2C_SLAVE_BYTE_ACK_CTL_EN Configures to enable the function for the slave to control ACK level.

0: Disable

1: Enable

(R/W)

I2C_SLAVE_BYTE_ACK_LVL Configures the ACK level when slave controlling ACK level function is enabled.

0: Low level

1: High level

(R/W)

Register 26.18. I2C_SR_REG (0x0008)

(reserved)		I2C_SCL_STATE_LAST		(reserved)		I2C_SCL_MAIN_STATE_LAST		I2C_TXFIFO_CNT		(reserved)		I2C_STRETCH_CAUSE		I2C_RXFIFO_CNT		(reserved)		I2C_SLAVE_ADDRESSED		I2C_BUS_BUSY		I2C_ARB_LOST		(reserved)		I2C_SLAVE_RW		I2C_RESP_REC	
31	30	28	27	26	24	23	18	17	16	15	14	13	8	7	6	5	4	3	2	1	0								
0	0	0	0	0	0	0	0	0	0	0x3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset																													

I2C_RESP_REC Represents the received ACK value in master mode or slave mode.

- 0: ACK
- 1: NACK
- (RO)

I2C_SLAVE_RW Represents the transfer direction in slave mode.

- 0: Master writes to slave
- 1: Master reads from slave
- (RO)

I2C_ARB_LOST Represents whether the I2C controller loses control of SCL line.

- 0: No arbitration lost
- 1: Arbitration lost
- (RO)

I2C_BUS_BUSY Represents the I2C bus state.

- 0: The I2C bus is in an idle state.
- 1: The I2C bus is busy transferring data
- (RO)

I2C_SLAVE_ADDRESSED Represents whether the address sent by the master is equal to the address of the slave.

- Valid only when the module is configured as an I2C Slave.
- 0: Not equal
- 1: Equal
- (RO)

I2C_RXFIFO_CNT Represents the number of data bytes received in RAM. (RO)

I2C_STRETCH_CAUSE Represents the cause of SCL clocking stretching in slave mode.

- 0: Stretching SCL low when the master starts to read data.
- 1: Stretching SCL low when I2C TX FIFO is empty in slave mode.
- 2: Stretching SCL low when I2C RX FIFO is full in slave mode.
- 3: Invalid
- (RO)

I2C_TXFIFO_CNT Represents the number of data bytes to be sent. (RO)

Continued on the next page...

Register 26.18. I2C_SR_REG (0x0008)

Continued from the previous page...

I2C_SCL_MAIN_STATE_LAST Represents the states of the I2C module state machine.

- 0: Idle
- 1: Address shift
- 2: ACK address
- 3: Rx data
- 4: Tx data
- 5: Send ACK
- 6: Wait ACK
- 7: Invalid
- (RO)

I2C_SCL_STATE_LAST Represents the states of the state machine used to produce SCL.

- 0: Idle
- 1: Start
- 2: Negative edge
- 3: Low
- 4: rising edge
- 5: High
- 6: Stop
- 7: Invalid
- (RO)

Register 26.19. I2C_FIFO_ST_REG (0x0014)

(reserved)			I2C_SLAVE_RW_POINT				(reserved)			I2C_TXFIFO_WADDR				I2C_TXFIFO_RADDR				I2C_RXFIFO_WADDR				I2C_RXFIFO_RADDR											
31	30	29					22	21	20	19					15	14					10	9					5	4					0
0	0	0								0	0	0								0					0								0
																																Reset	

Reset

I2C_RXFIFO_RADDR Represents the offset address of the APB reading from RX FIFO. (RO)**I2C_RXFIFO_WADDR** Represents the offset address of the I2C module receiving data and writing to RX FIFO. (RO)**I2C_TXFIFO_RADDR** Represents the offset address of the I2C module reading from TX FIFO. (RO)**I2C_TXFIFO_WADDR** Represents the offset address of the APB bus writing to TX FIFO. (RO)**I2C_SLAVE_RW_POINT** Represents the offset address in the I2C Slave RAM addressed by I2C Master when in I2C slave mode. (RO)

Register 26.20. I2C_DATA_REG (0x001C)

(reserved)																I2C_FIFO_RDATA																	
31																8	7	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0																Reset	

I2C_FIFO_RDATA Represents the value of RX FIFO read data. (RO)

Register 26.21. I2C_INT_RAW_REG (0x0020)

(reserved)																I2C_SLAVE_ADDR_UNMATCH_INT_RAW I2C_GENERAL_CALL_INT_RAW I2C_SLAVE_STRETCH_INT_RAW I2C_DET_START_INT_RAW I2C_SCL_MAIN_ST_TO_INT_RAW I2C_SCL_ST_TO_INT_RAW I2C_RXFIFO_UDF_INT_RAW I2C_TXFIFO_UDF_INT_RAW I2C_NACK_INT_RAW I2C_TRANS_START_INT_RAW I2C_TRANS_OUT_INT_RAW I2C_MST_TXFIFO_COMPLETE_INT_RAW I2C_BYTE_ARBTRATION_LOST_INT_RAW I2C_END_DETECT_INT_RAW I2C_RXFIFO_OVF_INT_RAW I2C_TXFIFO_WM_INT_RAW																				
31																19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset								

I2C_RXFIFO_WM_INT_RAW The raw interrupt status of [I2C_RXFIFO_WM_INT](#) interrupt. (R/SS/WTC)

I2C_TXFIFO_WM_INT_RAW The raw interrupt status of [I2C_TXFIFO_WM_INT](#) interrupt. (R/SS/WTC)

I2C_RXFIFO_OVF_INT_RAW The raw interrupt status of [I2C_RXFIFO_OVF_INT](#) interrupt. (R/SS/WTC)

I2C_END_DETECT_INT_RAW The raw interrupt status of the [I2C_END_DETECT_INT](#) interrupt. (R/SS/WTC)

I2C_BYTE_TRANS_DONE_INT_RAW The raw interrupt status of the [I2C_BYTE_TRANS_DONE_INT](#) interrupt. (R/SS/WTC)

I2C_ARBITRATION_LOST_INT_RAW The raw interrupt status of the [I2C_ARBITRATION_LOST_INT](#) interrupt. (R/SS/WTC)

I2C_MST_TXFIFO_UDF_INT_RAW The raw interrupt status of [I2C_MST_TXFIFO_UDF_INT](#) interrupt. (R/SS/WTC)

I2C_TRANS_COMPLETE_INT_RAW The raw interrupt status of the [I2C_TRANS_COMPLETE_INT](#) interrupt. (R/SS/WTC)

Continued on the next page...

Register 26.21. I2C_INT_RAW_REG (0x0020)

Continued from the previous page...

I2C_TIME_OUT_INT_RAW The raw interrupt status of the [I2C_TIME_OUT_INT](#) interrupt. (R/SS/WTC)

I2C_TRANS_START_INT_RAW The raw interrupt status of the [I2C_TRANS_START_INT](#) interrupt. (R/SS/WTC)

I2C_NACK_INT_RAW The raw interrupt status of [I2C_NACK_INT](#) interrupt. (R/SS/WTC)

I2C_TXFIFO_OVF_INT_RAW The raw interrupt status of [I2C_TXFIFO_OVF_INT](#) interrupt. (R/SS/WTC)

I2C_RXFIFO_UDF_INT_RAW The raw interrupt status of [I2C_RXFIFO_UDF_INT](#) interrupt. (R/SS/WTC)

I2C_SCL_ST_TO_INT_RAW The raw interrupt status of [I2C_SCL_ST_TO_INT](#) interrupt. (R/SS/WTC)

I2C_SCL_MAIN_ST_TO_INT_RAW The raw interrupt status of [I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (R/SS/WTC)

I2C_DET_START_INT_RAW The raw interrupt status of [I2C_DET_START_INT](#) interrupt. (R/SS/WTC)

I2C_SLAVE_STRETCH_INT_RAW The raw interrupt status of [I2C_SLAVE_STRETCH_INT](#) interrupt. (R/SS/WTC)

I2C_GENERAL_CALL_INT_RAW The raw interrupt status of [I2C_GENARAL_CALL_INT](#) interrupt. (R/SS/WTC)

I2C_SLAVE_ADDR_UNMATCH_INT_RAW The raw interrupt status of [I2C_SLAVE_ADDR_UNMATCH_INT](#) interrupt. (R/SS/WTC)

Register 26.22. I2C_INT_CLR_REG (0x0024)

(reserved)																I2C_SLAVE_ADDR_UNMATCH_INT_CLR I2C_GENERAL_CALL_INT_CLR I2C_SLAVE_STRETCH_INT_CLR I2C_DET_START_INT_CLR I2C_SCL_MAIN_ST_TO_INT_CLR I2C_SCL_ST_TO_INT_CLR I2C_RXFIFO_UDF_INT_CLR I2C_NACK_INT_CLR I2C_TRANS_START_INT_CLR I2C_TIME_OUT_INT_CLR I2C_TRANS_COMPLETE_INT_CLR I2C_MST_TXFIFO_UDF_INT_CLR I2C_ARBITRATION_LOST_INT_CLR I2C_BYTE_TRANS_DONE_INT_CLR I2C_END_DETECT_INT_CLR I2C_RXFIFO_OVF_INT_CLR I2C_RXFIFO_WM_INT_CLR																			
31																19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

I2C_RXFIFO_WM_INT_CLR Write 1 to clear [I2C_RXFIFO_WM_INT](#) interrupt. (WT)

I2C_TXFIFO_WM_INT_CLR Write 1 to clear [I2C_TXFIFO_WM_INT](#) interrupt. (WT)

I2C_RXFIFO_OVF_INT_CLR Write 1 to clear [I2C_RXFIFO_OVF_INT](#) interrupt. (WT)

I2C_END_DETECT_INT_CLR Write 1 to clear the [I2C_END_DETECT_INT](#) interrupt. (WT)

I2C_BYTE_TRANS_DONE_INT_CLR Write 1 to clear the [I2C_BYTE_TRANS_DONE_INT](#) interrupt. (WT)

I2C_ARBITRATION_LOST_INT_CLR Write 1 to clear the [I2C_ARBITRATION_LOST_INT](#) interrupt. (WT)

I2C_MST_TXFIFO_UDF_INT_CLR Write 1 to clear [I2C_MST_TXFIFO_UDF_INT](#) interrupt. (WT)

I2C_TRANS_COMPLETE_INT_CLR Write 1 to clear the [I2C_TRANS_COMPLETE_INT](#) interrupt. (WT)

I2C_TIME_OUT_INT_CLR Write 1 to clear the [I2C_TIME_OUT_INT](#) interrupt. (WT)

I2C_TRANS_START_INT_CLR Write 1 to clear the [I2C_TRANS_START_INT](#) interrupt. (WT)

I2C_NACK_INT_CLR Write 1 to clear [I2C_NACK_INT](#) interrupt. (WT)

I2C_TXFIFO_OVF_INT_CLR Write 1 to clear [I2C_TXFIFO_OVF_INT](#) interrupt. (WT)

I2C_RXFIFO_UDF_INT_CLR Write 1 to clear [I2C_RXFIFO_UDF_INT](#) interrupt. (WT)

I2C_SCL_ST_TO_INT_CLR Write 1 to clear [I2C_SCL_ST_TO_INT](#) interrupt. (WT)

I2C_SCL_MAIN_ST_TO_INT_CLR Write 1 to clear [I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (WT)

I2C_DET_START_INT_CLR Write 1 to clear [I2C_DET_START_INT](#) interrupt. (WT)

I2C_SLAVE_STRETCH_INT_CLR Write 1 to clear [I2C_SLAVE_STRETCH_INT](#) interrupt. (WT)

I2C_GENERAL_CALL_INT_CLR Write 1 to clear [I2C_GENARAL_CALL_INT](#) interrupt. (WT)

I2C_SLAVE_ADDR_UNMATCH_INT_CLR Write 1 to clear [I2C_SLAVE_ADDR_UNMATCH_INT](#) interrupt. (WT)

Register 26.23. I2C_INT_ENA_REG (0x0028)

(reserved)																I2C_SLAVE_ADDR_UNMATCH_INT_ENA I2C_GENERAL_CALL_INT_ENA I2C_SLAVE_STRETCH_INT_ENA I2C_DET_START_INT_ENA I2C_SCL_MAIN_ST_TO_INT_ENA I2C_SCL_ST_TO_INT_ENA I2C_RXFIFO_UDF_INT_ENA I2C_RXFIFO_OVF_INT_ENA I2C_NACK_INT_ENA I2C_TRANS_START_INT_ENA I2C_TIME_OUT_INT_ENA I2C_TRANS_COMPLETE_INT_ENA I2C_MST_TXFIFO_UDF_INT_ENA I2C_ARBITRATION_LOST_INT_ENA I2C_BYTE_TRANS_DONE_INT_ENA I2C_END_DETECT_INT_ENA I2C_RXFIFO_OVF_INT_ENA I2C_RXFIFO_WM_INT_ENA																			
31																19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

I2C_RXFIFO_WM_INT_ENA Write 1 to enable [I2C_RXFIFO_WM_INT](#) interrupt. (R/W)

I2C_TXFIFO_WM_INT_ENA Write 1 to enable [I2C_TXFIFO_WM_INT](#) interrupt. (R/W)

I2C_RXFIFO_OVF_INT_ENA Write 1 to enable [I2C_RXFIFO_OVF_INT](#) interrupt. (R/W)

I2C_END_DETECT_INT_ENA Write 1 to enable the [I2C_END_DETECT_INT](#) interrupt. (R/W)

I2C_BYTE_TRANS_DONE_INT_ENA Write 1 to enable the [I2C_BYTE_TRANS_DONE_INT](#) interrupt.
(R/W)

I2C_ARBITRATION_LOST_INT_ENA Write 1 to enable the [I2C_ARBITRATION_LOST_INT](#) interrupt.
(R/W)

I2C_MST_TXFIFO_UDF_INT_ENA Write 1 to enable [I2C_MST_TXFIFO_UDF_INT](#) interrupt. (R/W)

I2C_TRANS_COMPLETE_INT_ENA Write 1 to enable the [I2C_TRANS_COMPLETE_INT](#) interrupt.
(R/W)

I2C_TIME_OUT_INT_ENA Write 1 to enable the [I2C_TIME_OUT_INT](#) interrupt. (R/W)

I2C_TRANS_START_INT_ENA Write 1 to enable the [I2C_TRANS_START_INT](#) interrupt. (R/W)

I2C_NACK_INT_ENA Write 1 to enable [I2C_NACK_INT](#) interrupt. (R/W)

I2C_TXFIFO_OVF_INT_ENA Write 1 to enable [I2C_TXFIFO_OVF_INT](#) interrupt. (R/W)

I2C_RXFIFO_UDF_INT_ENA Write 1 to enable [I2C_RXFIFO_UDF_INT](#) interrupt. (R/W)

I2C_SCL_ST_TO_INT_ENA Write 1 to enable [I2C_SCL_ST_TO_INT](#) interrupt. (R/W)

I2C_SCL_MAIN_ST_TO_INT_ENA Write 1 to enable [I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (R/W)

I2C_DET_START_INT_ENA Write 1 to enable [I2C_DET_START_INT](#) interrupt. (R/W)

I2C_SLAVE_STRETCH_INT_ENA Write 1 to enable [I2C_SLAVE_STRETCH_INT](#) interrupt. (R/W)

I2C_GENERAL_CALL_INT_ENA Write 1 to enable [I2C_GENERAL_CALL_INT](#) interrupt. (R/W)

I2C_SLAVE_ADDR_UNMATCH_INT_ENA Write 1 to enable [I2C_SLAVE_ADDR_UNMATCH_INT](#) interrupt. (R/W)

Register 26.24. I2C_INT_STATUS_REG (0x002C)

(reserved)																		I2C_SLAVE_ADDR_UNMATCH_INT_ST I2C_GENERAL_CALL_INT_ST I2C_SLAVE_STRETCH_INT_ST I2C_DET_START_INT_ST I2C_SCL_MAIN_INT_ST I2C_SCL_ST_TO_INT_ST I2C_RXFIFO_UDF_INT_ST I2C_TXFIFO_UDF_INT_ST I2C_NACK_INT_ST I2C_TRANS_START_INT_ST I2C_TIME_OUT_INT_ST I2C_TRANS_COMPLETE_INT_ST I2C_MST_TXFIFO_UDF_INT_ST I2C_ARBITRATION_LOST_INT_ST I2C_BYTE_TRANS_DONE_INT_ST I2C_END_DETECT_INT_ST I2C_RXFIFO_OVF_INT_ST I2C_TXFIFO_WM_INT_ST I2C_RXFIFO_WM_INT_ST																			
31																	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										

I2C_RXFIFO_WM_INT_ST The masked interrupt status of [I2C_RXFIFO_WM_INT](#) interrupt. (RO)

I2C_TXFIFO_WM_INT_ST The masked interrupt status of [I2C_TXFIFO_WM_INT](#) interrupt. (RO)

I2C_RXFIFO_OVF_INT_ST The masked interrupt status of [I2C_RXFIFO_OVF_INT](#) interrupt. (RO)

I2C_END_DETECT_INT_ST The masked interrupt status of the [I2C_END_DETECT_INT](#) interrupt. (RO)

I2C_BYTE_TRANS_DONE_INT_ST The masked interrupt status of the [I2C_BYTE_TRANS_DONE_INT](#) interrupt. (RO)

I2C_ARBITRATION_LOST_INT_ST The masked interrupt status of the [I2C_ARBITRATION_LOST_INT](#) interrupt. (RO)

I2C_MST_TXFIFO_UDF_INT_ST The masked interrupt status of [I2C_MST_TXFIFO_UDF_INT](#) interrupt. (RO)

I2C_TRANS_COMPLETE_INT_ST The masked interrupt status of the [I2C_TRANS_COMPLETE_INT](#) interrupt. (RO)

I2C_TIME_OUT_INT_ST The masked interrupt status of the [I2C_TIME_OUT_INT](#) interrupt. (RO)

I2C_TRANS_START_INT_ST The masked interrupt status of the [I2C_TRANS_START_INT](#) interrupt. (RO)

I2C_NACK_INT_ST The masked interrupt status of [I2C_NACK_INT](#) interrupt. (RO)

I2C_TXFIFO_OVF_INT_ST The masked interrupt status of [I2C_TXFIFO_OVF_INT](#) interrupt. (RO)

Continued on the next page...

Register 26.24. I2C_INT_STATUS_REG (0x002C)

Continued from the previous page...

I2C_RXFIFO_UDF_INT_ST The masked interrupt status of [I2C_RXFIFO_UDF_INT](#) interrupt. (RO)

I2C_SCL_ST_TO_INT_ST The masked interrupt status of [I2C_SCL_ST_TO_INT](#) interrupt. (RO)

I2C_SCL_MAIN_ST_TO_INT_ST The masked interrupt status of [I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (RO)

I2C_DET_START_INT_ST The masked interrupt status of [I2C_DET_START_INT](#) interrupt. (RO)

I2C_SLAVE_STRETCH_INT_ST The masked interrupt status of [I2C_SLAVE_STRETCH_INT](#) interrupt. (RO)

I2C_GENERAL_CALL_INT_ST The masked interrupt status of [I2C_GENARAL_CALL_INT](#) interrupt. (RO)

I2C_SLAVE_ADDR_UNMATCH_INT_ST The masked interrupt status of [I2C_SLAVE_ADDR_UNMATCH_INT](#) interrupt. (RO)

I2C_COMMAND_DONE

(reserved)

I2C_COMMANDO

It consists of three parts:
op_code is the command

- Byte_num represents the number of bytes that need to be sent or received.

(R/W)

0: Not done

- (R/W/SS)

I2C_COMMAND1_DONE

(reserved)

I2C_COMMAND1

See details in [I2C_COMMAND0](#). (R/W)

0: Not done

- (R/W/SS)

Register 26.27. I2C_COMMD2_REG (0x0060)

Diagram illustrating the structure of the I2C_COMMAND2 register:

- Bit 31: I2C_COMMAND2_DONE
- Bits 14-31: (reserved)
- Bits 13-0: I2C_COMMAND2

I2C_COMMAND2 Configures command 2. See details in [I2C_COMMAND0](#). (R/W)

I2C_COMMAND2_DONE Represents whether command 2 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.28. I2C_COMMD3_REG (0x0064)

I2C_COMMAND3_DONE																	(reserved)															I2C_COMMAND3														
31	30															14	13															0														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															Reset														

I2C_COMMAND3 Configures command 3. See details in [I2C_COMMAND0](#). (R/W)

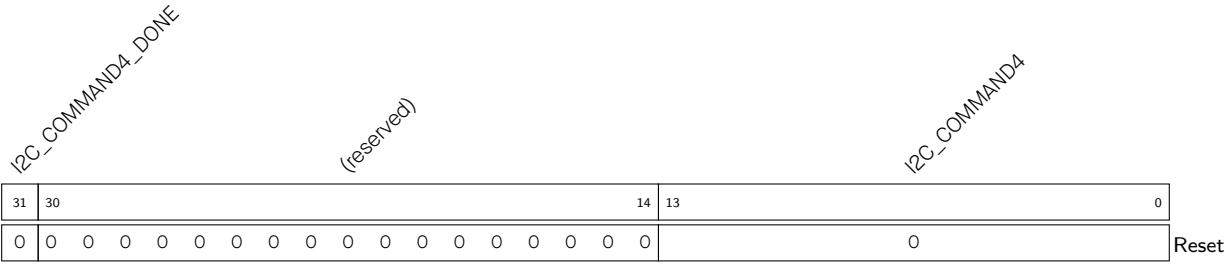
I2C_COMMAND3_DONE Represents whether command 3 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.29. I2C_COMD4_REG (0x0068)



I2C_COMMAND4 Configures command 4. See details in [I2C_COMMAND0](#). (R/W)

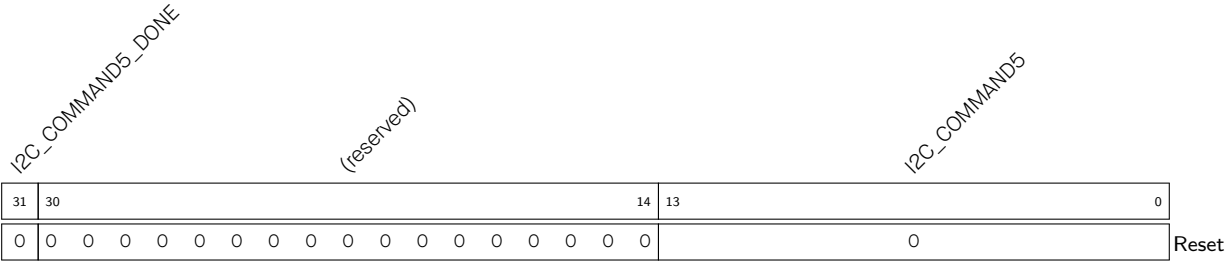
I2C_COMMAND4_DONE Represents whether command 4 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.30. I2C_COMD5_REG (0x006C)



I2C_COMMAND5 Configures command 5. See details in [I2C_COMMAND0](#). (R/W)

I2C_COMMAND5_DONE Represents whether command 5 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.31. I2C_COMMD6_REG (0x0070)

I2C_COMMAND6_DONE																	(reserved)																	I2C_COMMAND6																
31	30														14	13														0																				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														Reset																		

I2C_COMMAND6 Configures command 6. See details in [I2C_COMMAND0](#). (R/W)

I2C_COMMAND6_DONE Represents whether command 6 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.32. I2C_COMMD7_REG (0x0074)

I2C_COMMAND7_DONE																(reserved)																I2C_COMMAND7															
31	30														14	13														0																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														Reset																

I2C_COMMAND7 Configures command 7. See details in [I2C_COMMAND0](#). (R/W)

I2C_COMMAND7_DONE Represents whether command 7 is done in I2C Master mode.

0: Not done

1: Done

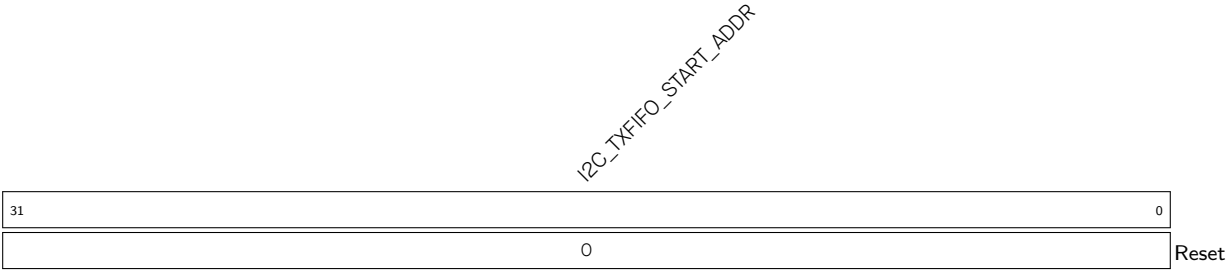
(R/W/SS)

Register 26.33. I2C_DATE_REG (0x00F8)

Diagram of the I2C_DATE register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The value 0x2401040 is shown in the register. A 'Reset' label is next to the register.

I2C_DATE Version control register. (R/W)

Register 26.34. I2C_TXFIFO_START_ADDR_REG (0x0100)



I2C_TXFIFO_START_ADDR Represents the I2C TX FIFO first address. (HRO)

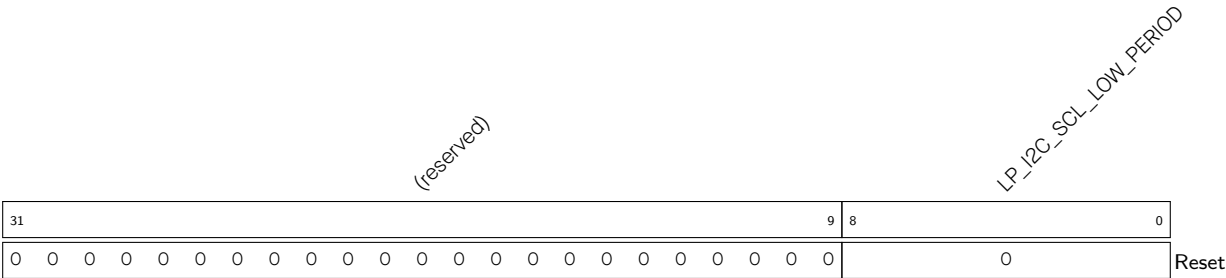
Register 26.35. I2C_RXFIFO_START_ADDR_REG (0x0180)



I2C_RXFIFO_START_ADDR Represents the I2C RX FIFO first address. (HRO)

26.9.2 LP_I2C Registers

Register 26.36. LP_I2C_SCL_LOW_PERIOD_REG (0x0000)



LP_I2C_SCL_LOW_PERIOD Configures the low level width of the SCL Clock in master mode.
Measurement unit: I2C_SCLK clock cycles
(R/W)

(reserved)

LP_I2C_SDA_HOLD_TIME

(reserved)

LP_I2C_SDA_SAMPLE_TIME

Register 26.39. LP_I2C_SCL_HIGH_PERIOD_REG (0x0038)

(reserved)																LP_I2C_SCL_WAIT_HIGH_PERIOD								LP_I2C_SCL_HIGH_PERIOD										
31																16	15								9	8								0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								0								Reset		

Reset

LP_I2C_SCL_HIGH_PERIOD Configures for how long SCL remains high in master mode.

Measurement unit: I2C_SCLK clock cycles

(R/W)

LP_I2C_SCL_WAIT_HIGH_PERIOD Configures the SCL_FSM's waiting period for SCL high level in master mode.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.40. LP_I2C_SCL_START_HOLD_REG (0x0040)

(reserved)																								LP_I2C_SCL_START_HOLD_TIME																							
31																								9								8								0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								8								Reset															

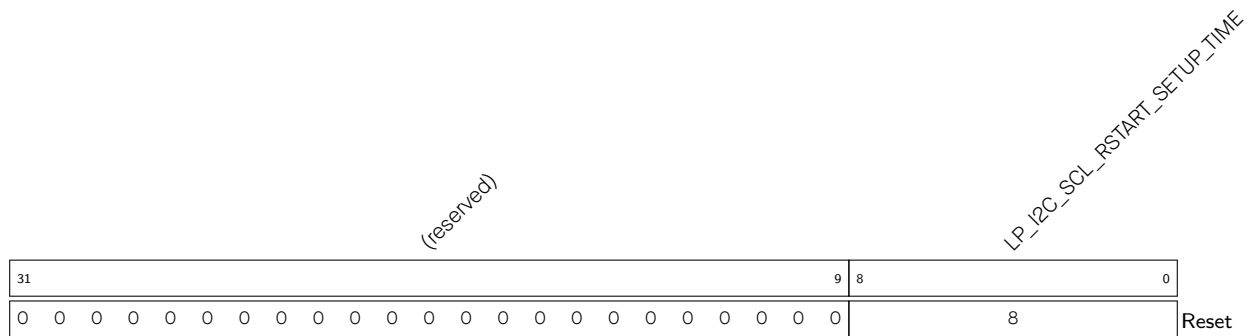
Reset

LP_I2C_SCL_START_HOLD_TIME Configures the time between the falling edge of SDA and the falling edge of SCL for a START condition.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.41. LP_I2C_SCL_RSTART_SETUP_REG (0x0044)

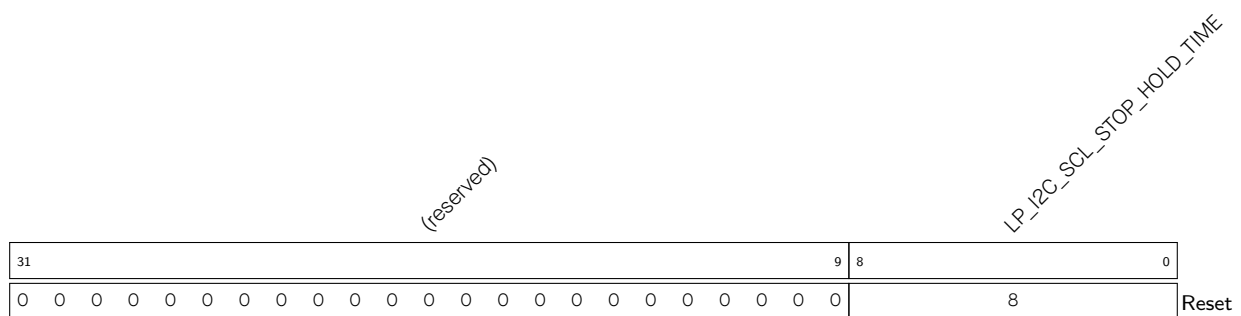


LP_I2C_SCL_RSTART_SETUP_TIME Configures the time between the rising edge of SCL and the negative edge of SDA for a RESTART condition.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.42. LP_I2C_SCL_STOP_HOLD_REG (0x0048)



LP_I2C_SCL_STOP_HOLD_TIME Configures the delay after the STOP condition.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.43. LP_I2C_SCL_STOP_SETUP_REG (0x004C)

(reserved)																LP_I2C_SCL_STOP_SETUP_TIME																	
31																9	8	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																8																Reset	

LP_I2C_SCL_STOP_SETUP_TIME Configures the time between the rising edge of SCL and the rising edge of SDA.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.44. LP_I2C_SCL_ST_TIME_OUT_REG (0x0078)

(reserved)																																LP_I2C_SCL_ST_TO_I2C																
31																															5	4	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																															0x10																Reset	

LP_I2C_SCL_ST_TO_I2C Configures the threshold value of SCL_FSM state unchanged period. It should be no more than 23.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.45. LP_I2C_SCL_MAIN_ST_TIME_OUT_REG (0x007C)

(reserved)																												LP_I2C_SCL_MAIN_ST_TO_I2C									
31																												5	4	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0x10								Reset	

LP_I2C_SCL_MAIN_ST_TO_I2C Configures the threshold value of SCL_MAIN_FSM state unchanged period. It should be no more than 23.

Measurement unit: I2C_SCLK clock cycles

(R/W)

Register 26.46. LP_I2C_CTR_REG (0x0004)

(reserved)												LP_I2C_CONF_UPGATE LP_I2C_FSM_RST LP_I2C_ARBITRATION_EN LP_I2C_CLK_EN LP_I2C_RX_LSB_FIRST LP_I2C_TX_LSB_FIRST (reserved) LP_I2C_TRANS_START LP_I2C_RX_FULL_ACK_LEVEL LP_I2C_SAMPLE_SCL_LEVEL (reserved)												
31												12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	Reset

LP_I2C_SAMPLE_SCL_LEVEL Configures the sample mode for SDA.

1: Sample SDA data on the SCL low level.

0: Sample SDA data on the SCL high level.

(R/W)

LP_I2C_RX_FULL_ACK_LEVEL Configures the ACK value that needs to be sent by master when the rx_fifo_cnt has reached the threshold. (R/W)

LP_I2C_TRANS_START Configures to start sending the data in TX FIFO for slave.

0: No effect

1: Start

(WT)

LP_I2C_TX_LSB_FIRST Configures to control the sending order for data to be sent.

1: send data from the least significant bit

0: send data from the most significant bit

(R/W)

LP_I2C_RX_LSB_FIRST Configures to control the storage order for received data.

1: receive data from the least significant bit

0: receive data from the most significant bit

(R/W)

LP_I2C_CLK_EN Configures whether to gate clock signal for registers.

0: Support clock only when registers are read or written to by software

1: Force clock on for registers.

(R/W)

LP_I2C_ARBITRATION_EN Configures to enable I2C bus arbitration detection.

0: No effect

1: Enable

(R/W)

LP_I2C_FSM_RST Configures to reset the SCL_FSM.

0: No effect

1: Reset

(WT)

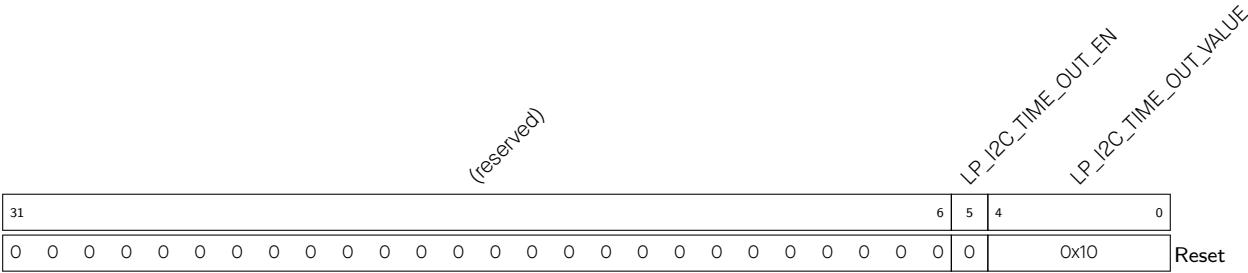
LP_I2C_CONF_UPGATE Configures this bit for synchronization.

0: No effect

1: Synchronize

(WT)

Register 26.47. LP_I2C_TO_REG (0x000C)



LP_I2C_TIME_OUT_VALUE Configures the timeout threshold period for SCL sticking at high or low level. The actual period is 2^{^(LP_I2C_TIME_OUT_VALUE+1)}.
Measurement unit: I2C_SCLK clock cycles.
(R/W)

LP_I2C_TIME_OUT_EN Configures to enable time out control.
0: No effect
1: Enable
(R/W)

Register 26.48. LP_I2C_FIFO_CONF_REG (0x0018)

(reserved)																LP_I2C_FIFO_PRT_EN				LP_I2C_TX_FIFO_RST				LP_I2C_RX_FIFO_RST				(reserved)				LP_I2C_NONFIFO_EN				LP_I2C_TXFIFO_WM_THRHD				(reserved)				LP_I2C_RXFIFO_WM_THRHD			
31																15	14	13	12	11	10	9	8					5	4	3					0												
0																0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0x2				0	0x6				Reset							

Reset

LP_I2C_RXFIFO_WM_THRHD Configures the watermark threshold of RX FIFO in nonfifo access mode. When [LP_I2C_FIFO_PRT_EN](#) is 1 and RX FIFO counter is bigger than LP_I2C_RXFIFO_WM_THRHD[3:0], LP_I2C_RXFIFO_WM_INT_RAW bit will be valid. (R/W)

LP_I2C_TXFIFO_WM_THRHD Configures the watermark threshold of TX FIFO in nonfifo access mode. When [LP_I2C_FIFO_PRT_EN](#) is 1 and RX FIFO counter is bigger than LP_I2C_TXFIFO_WM_THRHD, [LP_I2C_TXFIFO_WM_INT_RAW](#) bit will be valid. (R/W)

LP_I2C_NONFIFO_EN Configures to enable APB nonfifo access. (R/W)

LP_I2C_RX_FIFO_RST Configures to reset RX FIFO.

0: No effect

1: Reset

(R/W)

LP_I2C_TX_FIFO_RST Configures to reset TX FIFO. 0: No effect

1: Reset

(R/W)

LP_I2C_FIFO_PRT_EN Configures to enable FIFO pointer in non-fifo access mode. This bit controls the valid bits and the TX/RX FIFO overflow, underflow, full and empty interrupts.

0: No effect

1: Enable

(R/W)

Register 26.49. LP_I2C_FILTER_CFG_REG (0x0050)

(reserved)										10	9	8	7	4	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
											1	1	0		0	Reset

LP_I2C_SCL_FILTER_THRES Configures the threshold pulse width to be filtered on SCL. When a pulse on the SCL input has smaller width than this value, the I2C controller will ignore that pulse.

Measurement unit: I2C_SCLK clock cycles

(R/W)

LP_I2C_SDA_FILTER_THRES Configures the threshold pulse width to be filtered on SDA. When a pulse on the SDA input has smaller width than this value, the I2C controller will ignore that pulse.

Measurement unit: I2C_SCLK clock cycles

(R/W)

LP_I2C_SCL_FILTER_EN Configures to enable the filter function for SCL.

0: No effect

1: Enable

(R/W)

LP_I2C_SDA_FILTER_EN Configures to enable the filter function for SDA.

0: No effect

1: Enable

(R/W)

Register 26.50. LP_I2C_SCL_SP_CONF_REG (0x0080)

(reserved)																								LP_I2C_SDA_PD_EN		LP_I2C_SCL_PD_EN		LP_I2C_SCL_RST_SLV_NUM		LP_I2C_SCL_RST_SLV_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																								8	7	6	5	1				0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

LP_I2C_SCL_RST_SLV_EN Configures to send out SCL pulses when I2C master is IDLE. The number of pulses equals to [LP_I2C_SCL_RST_SLV_NUM](#). (R/W/SC)

LP_I2C_SCL_RST_SLV_NUM Configure the pulses of SCL generated in I2C master mode.
Valid when [LP_I2C_SCL_RST_SLV_EN](#) is 1.
Measurement unit: I2C_SCLK clock cycles
(R/W)

LP_I2C_SCL_PD_EN Configures to power down the I2C output SCL line.
0: Not power down
1: Not work and power down
Valid only when [LP_I2C_SCL_FORCE_OUT](#) is 1
(R/W)

LP_I2C_SDA_PD_EN Configures to power down the I2C output SDA line.
0: Not power down.
1: Not work and power down.
Valid only when [LP_I2C_SDA_FORCE_OUT](#) is 1. (R/W)

Register 26.51. LP_I2C_SR_REG (0x0008)

(reserved)		LP_I2C_SCL_STATE_LAST		(reserved)		LP_I2C_SCL_MAIN_STATE_LAST		(reserved)		LP_I2C_TXFIFO_CNT		(reserved)		LP_I2C_RXFIFO_CNT		(reserved)		LP_I2C_BUS_BUSY		LP_I2C_ARB_LOST		(reserved)		LP_I2C_RESP_REC	
31	30	28	27	26	24	23	22	18	17	13	12	8	7	5	4	3	2	1	0						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Reset																									

LP_I2C_RESP_REC Represents the received ACK value in master mode or slave mode.

0: ACK

1: NACK

(RO)

LP_I2C_ARB_LOST Represents whether the I2C controller loses control of SCL line.

0: No arbitration lost

1: Arbitration lost

(RO)

LP_I2C_BUS_BUSY Represents the I2C bus state.

1: The I2C bus is busy transferring data

0: The I2C bus is in idle state

(RO)

LP_I2C_RXFIFO_CNT Represents the number of data bytes received in RAM. (RO)

LP_I2C_TXFIFO_CNT Represents the number of data bytes to be sent. (RO)

LP_I2C_SCL_MAIN_STATE_LAST Represents the states of the I2C module state machine.

0: Idle

1: Address shift

2: ACK address

3: Rx data

4: Tx data

5: Send ACK

6: Wait ACK

(RO)

LP_I2C_SCL_STATE_LAST Represents the states of the state machine used to produce SCL.

0: Idle

1: Start

2: Negative edge

3: Low

4: rising edge

5: High

6: Stop

(RO)

Register 26.52. LP_I2C_FIFO_ST_REG (0x0014)

(reserved)																LP_I2C_TXFIFO_WADDR				(reserved)				LP_I2C_TXFIFO_RADDR				(reserved)				LP_I2C_RXFIFO_WADDR				(reserved)				LP_I2C_RXFIFO_RADDR			
31																19	18	15		14	13		10		9	8	5		4	3	0												
0 0 0 0 0 0 0 0 0 0 0 0 0 0																0		0	0		0		0		0		0		0				Reset										

Reset

LP_I2C_RXFIFO_RADDR Represents the offset address of the APB reading from RX FIFO (RO)

LP_I2C_RXFIFO_WADDR Represents the offset address of i2c module receiving data and writing to RX FIFO. (RO)

LP_I2C_TXFIFO_RADDR Represents the offset address of i2c module reading from TX FIFO. (RO)

LP_I2C_TXFIFO_WADDR Represents the offset address of APB bus writing to TX FIFO. (RO)

Register 26.53. LP_I2C_DATA_REG (0x001C)

(reserved)																								LP_I2C_FIFO_RDATA																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																								8	7	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Reset

LP_I2C_FIFO_RDATA Represents the value of RX FIFO read data. (RO)

Register 26.54. LP_I2C_INT_RAW_REG (0x0020)

(reserved)																																LP_I2C_DET_START_INT_RAW LP_I2C_SCL_MAIN_ST_TO_INT_RAW LP_I2C_SCL_ST_TO_INT_RAW LP_I2C_RXFIFO_UDF_INT_RAW LP_I2C_TXFIFO_UDF_INT_RAW LP_I2C_NACK_INT_RAW LP_I2C_TRANS_START_INT_RAW LP_I2C_TIME_OUT_INT_RAW LP_I2C_TRANS_COMPLETE_INT_RAW LP_I2C_ARBTRATION_LOST_INT_RAW LP_I2C_BYTE_TRANS_DONE_INT_RAW LP_I2C_END_DETECT_INT_RAW LP_I2C_RXFIFO_OVF_INT_RAW LP_I2C_TXFIFO_WM_INT_RAW																
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset																	

LP_I2C_RXFIFO_WM_INT_RAW The raw interrupt status of [LP_I2C_RXFIFO_WM_INT](#) interrupt. (R/SS/WTC)

LP_I2C_TXFIFO_WM_INT_RAW The raw interrupt status of [LP_I2C_TXFIFO_WM_INT](#) interrupt. (R/SS/WTC)

LP_I2C_RXFIFO_OVF_INT_RAW The raw interrupt status of [LP_I2C_RXFIFO_OVF_INT](#) interrupt. (R/SS/WTC)

LP_I2C_END_DETECT_INT_RAW The raw interrupt status of the [LP_I2C_END_DETECT_INT](#) interrupt. (R/SS/WTC)

LP_I2C_BYTE_TRANS_DONE_INT_RAW The raw interrupt status of the [LP_I2C_BYTE_TRANS_DONE_INT](#) interrupt. (R/SS/WTC)

LP_I2C_ARBTRATION_LOST_INT_RAW The raw interrupt status of the [LP_I2C_ARBTRATION_LOST_INT](#) interrupt. (R/SS/WTC)

LP_I2C_MST_TXFIFO_UDF_INT_RAW The raw interrupt status of [LP_I2C_MST_TXFIFO_UDF_INT](#) interrupt. (R/SS/WTC)

LP_I2C_TRANS_COMPLETE_INT_RAW The raw interrupt status of the [LP_I2C_TRANS_COMPLETE_INT](#) interrupt. (R/SS/WTC)

LP_I2C_TIME_OUT_INT_RAW The raw interrupt status of the [LP_I2C_TIME_OUT_INT](#) interrupt. (R/SS/WTC)

LP_I2C_TRANS_START_INT_RAW The raw interrupt status of the [LP_I2C_TRANS_START_INT](#) interrupt. (R/SS/WTC)

LP_I2C_NACK_INT_RAW The raw interrupt status of [LP_I2C_NACK_INT](#) interrupt. (R/SS/WTC)

LP_I2C_TXFIFO_OVF_INT_RAW The raw interrupt status of [LP_I2C_TXFIFO_OVF_INT](#) interrupt. (R/SS/WTC)

LP_I2C_RXFIFO_UDF_INT_RAW The raw interrupt status of [LP_I2C_RXFIFO_UDF_INT](#) interrupt. (R/SS/WTC)

LP_I2C_SCL_ST_TO_INT_RAW The raw interrupt status of [LP_I2C_SCL_ST_TO_INT](#) interrupt. (R/SS/WTC)

Continued on the next page...

Register 26.54. LP_I2C_INT_RAW_REG (0x0020)

Continued from the previous page...

LP_I2C_SCL_MAIN_ST_TO_INT_RAW The raw interrupt status of [LP_I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (R/SS/WTC)

LP_I2C_DET_START_INT_RAW The raw interrupt status of [LP_I2C_DET_START_INT](#) interrupt. (R/SS/WTC)

Register 26.55. LP_I2C_INT_CLR_REG (0x0024)

(reserved)																LP_I2C_DET_START_INT_CLR LP_I2C_SCL_MAIN_ST_TO_INT_CLR LP_I2C_SCL_ST_TO_INT_CLR LP_I2C_RXFIFO_UDF_INT_CLR LP_I2C_TXFIFO_OVF_INT_CLR LP_I2C_NACK_INT_CLR LP_I2C_TRANS_START_INT_CLR LP_I2C_TIME_OUT_INT_CLR LP_I2C_TRANS_COMPLETE_INT_CLR LP_I2C_ARBITRATION_LOST_INT_CLR LP_I2C_BYTE_TRANS_DONE_INT_CLR LP_I2C_END_DETECT_INT_CLR LP_I2C_RXFIFO_WM_INT_CLR LP_I2C_TXFIFO_WM_INT_CLR																
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset	

LP_I2C_RXFIFO_WM_INT_CLR Write 1 to clear [LP_I2C_RXFIFO_WM_INT](#) interrupt. (WT)

LP_I2C_TXFIFO_WM_INT_CLR Write 1 to clear [LP_I2C_TXFIFO_WM_INT](#) interrupt. (WT)

LP_I2C_RXFIFO_OVF_INT_CLR Write 1 to clear [LP_I2C_RXFIFO_OVF_INT](#) interrupt. (WT)

LP_I2C_END_DETECT_INT_CLR Write 1 to clear the [LP_I2C_END_DETECT_INT](#) interrupt. (WT)

LP_I2C_BYTE_TRANS_DONE_INT_CLR Write 1 to clear the [LP_I2C_BYTE_TRANS_DONE_INT](#) interrupt. (WT)

LP_I2C_ARBITRATION_LOST_INT_CLR Write 1 to clear the [LP_I2C_ARBITRATION_LOST_INT](#) interrupt. (WT)

LP_I2C_MST_TXFIFO_UDF_INT_CLR Write 1 to clear [LP_I2C_MST_TXFIFO_UDF_INT](#) interrupt. (WT)

LP_I2C_TRANS_COMPLETE_INT_CLR Write 1 to clear the [LP_I2C_TRANS_COMPLETE_INT](#) interrupt. (WT)

LP_I2C_TIME_OUT_INT_CLR Write 1 to clear the [LP_I2C_TIME_OUT_INT](#) interrupt. (WT)

LP_I2C_TRANS_START_INT_CLR Write 1 to clear the [LP_I2C_TRANS_START_INT](#) interrupt. (WT)

LP_I2C_NACK_INT_CLR Write 1 to clear [LP_I2C_NACK_INT](#) interrupt. (WT)

LP_I2C_TXFIFO_OVF_INT_CLR Write 1 to clear [LP_I2C_TXFIFO_OVF_INT](#) interrupt. (WT)

LP_I2C_RXFIFO_UDF_INT_CLR Write 1 to clear [LP_I2C_RXFIFO_UDF_INT](#) interrupt. (WT)

LP_I2C_SCL_ST_TO_INT_CLR Write 1 to clear [LP_I2C_SCL_ST_TO_INT](#) interrupt. (WT)

LP_I2C_SCL_MAIN_ST_TO_INT_CLR Write 1 to clear [LP_I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (WT)

LP_I2C_DET_START_INT_CLR Write 1 to clear [LP_I2C_DET_START_INT](#) interrupt. (WT)

Register 26.56. LP_I2C_INT_ENA_REG (0x0028)

(reserved)																LP_I2C_DET_START_INT_ENA LP_I2C_SCL_MAIN_ST_TO_INT_ENA LP_I2C_SCL_ST_TO_INT_ENA LP_I2C_RXFIFO_UDF_INT_ENA LP_I2C_TXFIFO_OVF_INT_ENA LP_I2C_NACK_INT_ENA LP_I2C_TRANS_START_INT_ENA LP_I2C_TIME_OUT_INT_ENA LP_I2C_TRANS_COMPLETE_INT_ENA LP_I2C_ARBITRATION_LOST_INT_ENA LP_I2C_BYTE_TRANS_DONE_INT_ENA LP_I2C_END_DETECT_INT_ENA LP_I2C_RXFIFO_WM_INT_ENA LP_I2C_TXFIFO_WM_INT_ENA																
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset	

LP_I2C_RXFIFO_WM_INT_ENA Write 1 to enable [LP_I2C_RXFIFO_WM_INT](#) interrupt. (R/W)

LP_I2C_TXFIFO_WM_INT_ENA Write 1 to enable [LP_I2C_TXFIFO_WM_INT](#) interrupt. (R/W)

LP_I2C_RXFIFO_OVF_INT_ENA Write 1 to enable [LP_I2C_RXFIFO_OVF_INT](#) interrupt. (R/W)

LP_I2C_END_DETECT_INT_ENA Write 1 to enable the [LP_I2C_END_DETECT_INT](#) interrupt. (R/W)

LP_I2C_BYTE_TRANS_DONE_INT_ENA Write 1 to enable the [LP_I2C_BYTE_TRANS_DONE_INT](#) interrupt. (R/W)

LP_I2C_ARBITRATION_LOST_INT_ENA Write 1 to enable the [LP_I2C_ARBITRATION_LOST_INT](#) interrupt. (R/W)

LP_I2C_MST_TXFIFO_UDF_INT_ENA Write 1 to enable [LP_I2C_MST_TXFIFO_UDF_INT](#) interrupt. (R/W)

LP_I2C_TRANS_COMPLETE_INT_ENA Write 1 to enable the [LP_I2C_TRANS_COMPLETE_INT](#) interrupt. (R/W)

LP_I2C_TIME_OUT_INT_ENA Write 1 to enable the [LP_I2C_TIME_OUT_INT](#) interrupt. (R/W)

LP_I2C_TRANS_START_INT_ENA Write 1 to enable the [LP_I2C_TRANS_START_INT](#) interrupt. (R/W)

LP_I2C_NACK_INT_ENA Write 1 to enable [LP_I2C_NACK_INT](#) interrupt. (R/W)

LP_I2C_TXFIFO_OVF_INT_ENA Write 1 to enable [LP_I2C_TXFIFO_OVF_INT](#) interrupt. (R/W)

LP_I2C_RXFIFO_UDF_INT_ENA Write 1 to enable [LP_I2C_RXFIFO_UDF_INT](#) interrupt. (R/W)

LP_I2C_SCL_ST_TO_INT_ENA Write 1 to enable [LP_I2C_SCL_ST_TO_INT](#) interrupt. (R/W)

LP_I2C_SCL_MAIN_ST_TO_INT_ENA Write 1 to enable [LP_I2C_SCL_MAIN_ST_TO_INT](#) interrupt. (R/W)

LP_I2C_DET_START_INT_ENA Write 1 to enable [LP_I2C_DET_START_INT](#) interrupt. (R/W)

Register 26.57. LP_I2C_INT_STATUS_REG (0x002C)

(reserved)																LP_I2C_DET_START_INT_ST LP_I2C_SCL_MAIN_ST_TO_INT_ST LP_I2C_SCL_ST_TO_INT_ST LP_I2C_RXFIFO_UDF_INT_ST LP_I2C_TXFIFO_UDF_INT_ST LP_I2C_NACK_INT_ST LP_I2C_TRANS_START_INT_ST LP_I2C_TIME_OUT_INT_ST LP_I2C_TRANS_COMPLETE_INT_ST LP_I2C_ARBITRATION_LOST_INT_ST LP_I2C_BYTE_TRANS_DONE_INT_ST LP_I2C_END_DETECT_INT_ST LP_I2C_RXFIFO_OVF_INT_ST LP_I2C_TXFIFO_WM_INT_ST																
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset	

LP_I2C_RXFIFO_WM_INT_ST The masked interrupt status of [LP_I2C_RXFIFO_WM_INT](#) interrupt. (RO)

LP_I2C_TXFIFO_WM_INT_ST The masked interrupt status of [LP_I2C_TXFIFO_WM_INT](#) interrupt. (RO)

LP_I2C_RXFIFO_OVF_INT_ST The masked interrupt status of [LP_I2C_RXFIFO_OVF_INT](#) interrupt. (RO)

LP_I2C_END_DETECT_INT_ST The masked interrupt status of the [LP_I2C_END_DETECT_INT](#) interrupt. (RO)

LP_I2C_BYTE_TRANS_DONE_INT_ST The masked interrupt status of the [LP_I2C_BYTE_TRANS_DONE_INT](#) interrupt. (RO)

LP_I2C_ARBTRATION_LOST_INT_ST The masked interrupt status of the [LP_I2C_ARBTRATION_LOST_INT](#) interrupt. (RO)

LP_I2C_MST_TXFIFO_UDF_INT_ST The masked interrupt status of [LP_I2C_MST_TXFIFO_UDF_INT](#) interrupt. (RO)

LP_I2C_TRANS_COMPLETE_INT_ST The masked interrupt status of the [LP_I2C_TRANS_COMPLETE_INT](#) interrupt. (RO)

LP_I2C_TIME_OUT_INT_ST The masked interrupt status of the [LP_I2C_TIME_OUT_INT](#) interrupt. (RO)

LP_I2C_TRANS_START_INT_ST The masked interrupt status of the [LP_I2C_TRANS_START_INT](#) interrupt. (RO)

LP_I2C_NACK_INT_ST The masked interrupt status of [LP_I2C_NACK_INT](#) interrupt. (RO)

LP_I2C_TXFIFO_OVF_INT_ST The masked interrupt status of [LP_I2C_TXFIFO_OVF_INT](#) interrupt. (RO)

LP_I2C_RXFIFO_UDF_INT_ST The masked interrupt status of [LP_I2C_RXFIFO_UDF_INT](#) interrupt. (RO)

LP_I2C_SCL_ST_TO_INT_ST The masked interrupt status of [LP_I2C_SCL_ST_TO_INT](#) interrupt. (RO)

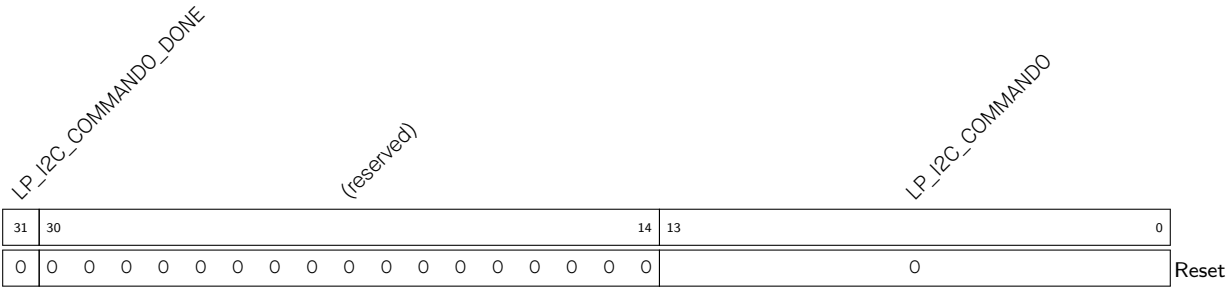
Continued on the next page...

Register 26.57. LP_I2C_INT_STATUS_REG (0x002C)

Continued from the previous page...

- LP_I2C_SCL_MAIN_ST_TO_INT_ST The masked interrupt status of LP_I2C_SCL_MAIN_ST_TO_INT interrupt. (RO)
- LP_I2C_DET_START_INT_ST The masked interrupt status of LP_I2C_DET_START_INT interrupt. (RO)

Register 26.58. LP_I2C_COMDO_REG (0x0058)



- LP_I2C_COMMANDO Configures command 0.
It consists of three parts:
op_code is the command
1: WRITE
2: STOP
3: READ
4: END
6: RSTART
Byte_num represents the number of bytes that need to be sent or received.
ack_check_en, ack_exp and ack are used to control the ACK bit. See I2C cmd structure 26.4-2 for more information. (R/W)
- LP_I2C_COMMANDO_DONE Represents whether command 0 is done in I2C Master mode.
0: Not done
1: Done
(R/W/SS)

LP_I2C_COMMAND1_DONE

(reserved)

LP_I2C_COMMAND1

See details in [LP_I2C_COMMAND0](#). (R/W)

0: Not done

(R/W/SS)

LP_I2C_COMMAND2_DONE

(reserved)

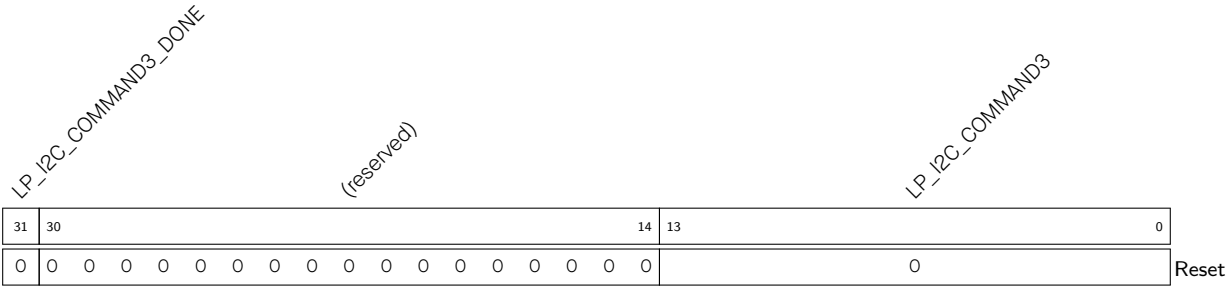
LP_I2C_COMMAND2

0: Not done

1: Done

(R/W/SS)

Register 26.61. LP_I2C_COMD3_REG (0x0064)



LP_I2C_COMMAND3 Configures command 3. See details in [LP_I2C_COMMAND0](#). (R/W)

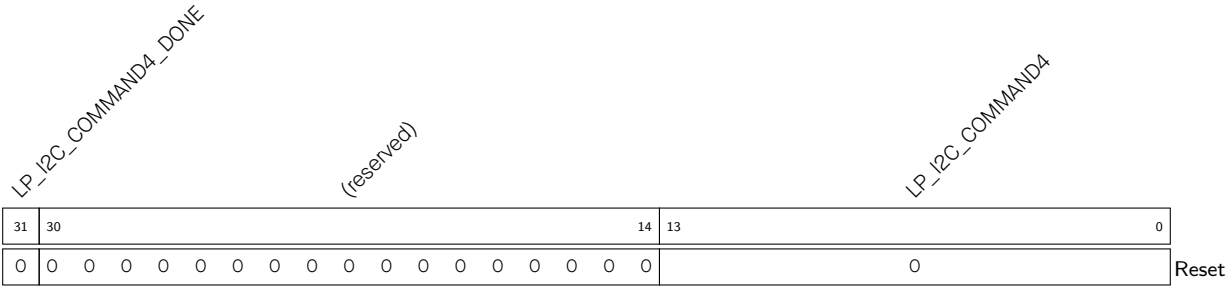
LP_I2C_COMMAND3_DONE Represents whether command 3 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.62. LP_I2C_COMD4_REG (0x0068)



LP_I2C_COMMAND4 Configures command 4. See details in [LP_I2C_COMMAND0](#). (R/W)

LP_I2C_COMMAND4_DONE Represents whether command 4 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

LP_I2C_COMMAND5_DONE

(reserved)

LP_I2C_COMMAND5

Reset

(R/W/SS)

LP_I2C_COMMAND6_DONE

(reserved)

LP_I2C_COMMAND6

Reset

(R/W/SS)

Register 26.65. LP_I2C_COMD7_REG (0x0074)

Diagram illustrating the structure of the LP_I2C_COMMAND7 register. The register is 32 bits wide, divided into three fields:

- LP_I2C_COMMAND7_DONE**: Bits 31-30 (2 bits).
- (reserved)**: Bits 14-30 (17 bits).
- LP_I2C_COMMAND7**: Bits 13-0 (14 bits).

The LP_I2C_COMMAND7 field contains a single '0' bit.

LP_I2C_COMMAND7 Configures command 7. See details in [LP_I2C_COMMAND0](#). (R/W)

LP_I2C_COMMAND7_DONE Represents whether command 7 is done in I2C Master mode.

0: Not done

1: Done

(R/W/SS)

Register 26.66. LP_I2C_DATE_REG (0x00F8)

Diagram of the LP_I2C_DATE register. The register is 32 bits wide. Bit 31 is labeled '0' and bit 0 is labeled '0'. The register is divided into two 16-bit halves. The lower 16 bits are labeled '0x2401040' and 'Reset'.

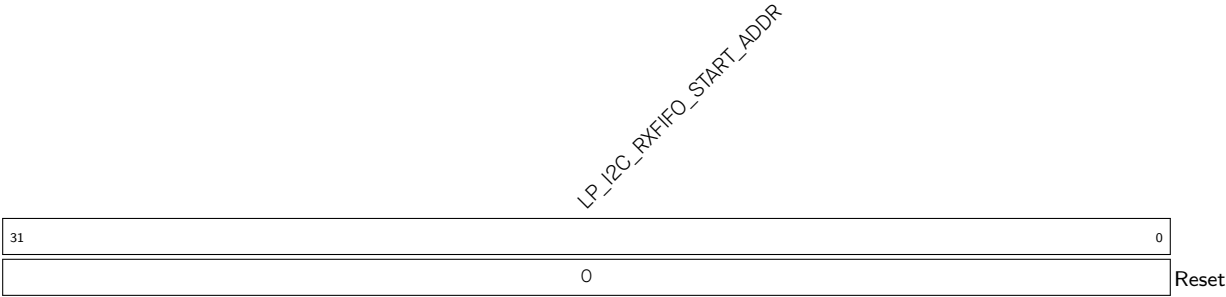
LP_I2C_DATE Version control register. (R/W)

Register 26.67. LP_I2C_TXFIFO_START_ADDR_REG (0x0100)

Diagram of the LP_I2C_TXFIFO_START_ADDR register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. Bit 0 is labeled 'Reset'.

LP_I2C_TXFIFO_START_ADDR Represents the I2C TX FIFO first address. (HRO)

Register 26.68. LP_I2C_RXFIFO_START_ADDR_REG (0x0180)



LP_I2C_RXFIFO_START_ADDR Represents the I2C RX FIFO first address. (HRO)

Chapter 27

I2S Controller (I2S)

27.1 Overview

ESP32-C5 has a built-in I2S interface, which provide flexible communication interfaces for streaming digital data in multimedia applications, especially digital audio applications.

The I2S standard bus defines three signals, namely, a bit clock signal (BCK), a channel/word select signal (WS), and a serial data signal (SD). A basic I2S data bus has one master and one slave. The roles remain unchanged throughout the communication. The I2S module on ESP32-C5 provides separate transmit (TX) and receive (RX) units for high performance.

27.2 Terminology

To better illustrate the functionality of I2S, the following terms are used in this chapter.

Master mode	As a master, I2S drives BCK/WS signals and transmits data to or receives data from a slave.
Slave mode	As a slave, I2S is driven by BCK/WS signals and receives data from or transmits data to a master.
Full-duplex	There are two separate data lines. The transmitted and received data is carried simultaneously.
Half-duplex	Only one side, the master or the slave, transmits data first, and the other side receives data. Data transmission and reception cannot occur simultaneously.
A-law and μ-law	A-law and μ -law are compression/decompression algorithms in digital pulse code modulated (PCM) non-uniform quantization, which can effectively improve the signal-to-quantization noise ratio.
TDM RX mode	In this mode, pulse code modulated (PCM) data is received utilizing time division multiplexing (TDM). The signal lines include BCK, WS, and SD. Data from 16 channels at most can be received. TDM Philips standard, TDM MSB alignment standard, and TDM PCM standard are supported in this mode, depending on user configuration.
PDM RX mode	In this mode, pulse density modulation (PDM) data is received. Used signals include WS and DATA. PDM standard is supported in this mode by user configuration.

TDM TX mode	In this mode, pulse code modulated (PCM) data is sent in a way of time division multiplexing (TDM). The signal lines include BCK, WS, and DATA. Data up to 16 channels can be sent. TDM Philips standard, TDM MSB alignment standard, and TDM PCM standard are supported in this mode, depending on user configuration.
PDM TX mode	In this mode, pulse density modulation (PDM) data is sent. The signal lines include WS and DATA. PDM standard is supported in this mode by user configuration.
PCM-to-PDM Data Format Converter	A data format converter that converts PCM data into PDM data (hereinafter referred to as PCM-to-PDM converter), which can be enabled in PDM TX master mode. When the converter is enabled, I2S fetches pulse code modulated (PCM) data from storage via GDMA, converts it to pulse density modulation (PDM) data, and then transmits it.

27.3 Features

The I2S module has the following features:

- Master mode and slave mode
- Full-duplex and half-duplex communications
- Separate TX and RX units that can work independently or simultaneously
- A variety of audio standards supported:
 - TDM Philips standard
 - TDM MSB alignment standard
 - TDM PCM standard
 - PDM standard
- Various TX/RX modes supported:
 - TDM TX mode, up to 16 channels supported
 - TDM RX mode, up to 16 channels supported
 - PDM TX mode
 - * Raw PDM data transmission
 - * PCM-to-PDM data format conversion, up to 2 channels supported
 - PDM RX mode
 - * Raw PDM data reception
- Configurable clock source with frequencies up to 240 MHz
- Configurable high-precision sample clock with a variety of sampling frequencies supported (refer to the note below for more details)
- 8/16/24/32-bit data width

- Synchronous counter in TX mode
- ETM feature
- Direct Memory Access
- Standard I2S interface interrupts

Note:

In slave mode, to ensure the sampling accuracy, the module clock frequency must be greater than or equal to 8 times the BCK clock frequency. Therefore, the maximum sampling frequency of I2S is limited by the data bit width and the number of channels. For example, since the clock source frequency is up to 240 MHz, the module clock can be configured up to 120 MHz, and the BCK clock can be configured up to 15 MHz. Therefore, when transmitting dual-channel 32-bit width data, I2S supports sample frequencies up to 234.375 kHz, e.g., 8 kHz, 16 kHz, 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 128 kHz, and 192 kHz. Please refer to Section [27.6 I2S TX/RX Clock](#) for detailed information.

27.4 System Architecture

Figure [27.4-1](#) shows the structure of the ESP32-C5 I2S module, consisting of:

- Transmit control unit (TX Unit)
- Receive control unit (RX Unit)
- Input and output timing unit (I/O Sync)
- Clock divider (Clock Generator)
- 64 x 32-bit TX FIFO
- 64 x 32-bit RX FIFO
- Compress/Decompress units

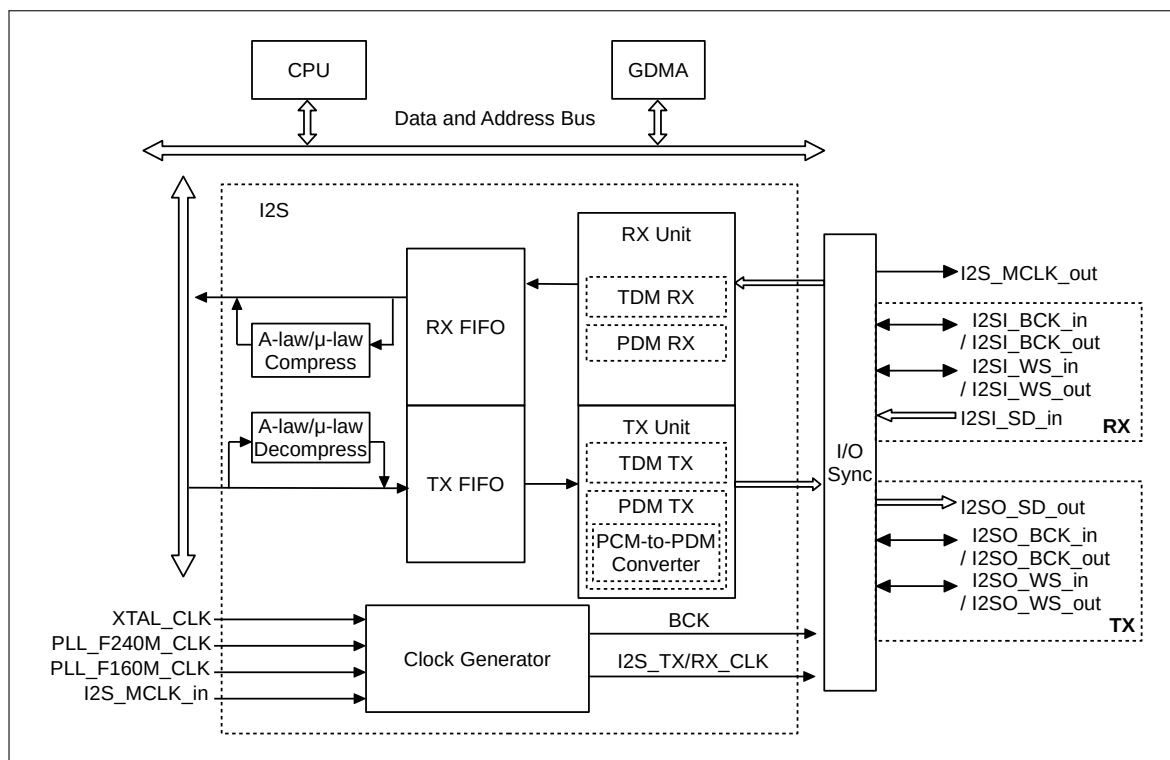


Figure 27.4-1. ESP32-C5 I2S System Diagram

The I2S module supports direct memory access (DMA) to internal memory. For more information, see Chapter 3 [GDMA Controller \(GDMA\)](#).

Both the TX unit and the RX unit have a three-line interface that uses a bit clock line (BCK), a word select line (WS), and a serial data line (SD). The SD line of the TX unit is used for data output and the SD line of the RX unit for data input. The BCK and WS signal lines of the TX and RX units are used for data output in master mode, and the BCK and WS signal lines of the TX and RX units are used for data input in slave mode.

The signal bus of the I2S module is shown at the right part of Figure 27.4-1. The naming of these signals in RX and TX units follows the pattern of I2SA_B_C such as I2SI_BCK_in.

- “A” indicates that the signal belongs to the I2S TX or RX unit, which includes:
 - “I”: Signal input to/output from the RX unit
 - “O”: Signal input to/output from the TX unit
- “B” represents the signal function, which includes:
 - BCK
 - WS
 - SD
- “C” represents the signal direction, which includes:
 - “in”: Input signal into the I2S module
 - “out”: Output signal from the I2S module

Table 27.4-1 provides a detailed description of I2S signals.

Table 27.4-1. I2S Signal Description

Signal	Direction	Function
I2SI_BCK_in	Input	In I2S slave mode, inputs BCK signal for RX unit.
I2SI_BCK_out	Output	In I2S master mode, outputs BCK signal for RX unit.
I2SI_WS_in	Input	In I2S slave mode, inputs WS signal for RX unit.
I2SI_WS_out	Output	In I2S master mode, outputs WS signal for RX unit.
I2SI_SD_in	Input	Works as the serial input data bus for I2S RX unit.
I2SO_SD_out	Output	Works as the serial output data bus for I2S TX unit.
I2SO_BCK_in	Input	In I2S slave mode, inputs BCK signal for TX unit.
I2SO_BCK_out	Output	In I2S master mode, outputs BCK signal for TX unit.
I2SO_WS_in	Input	In I2S slave mode, inputs WS signal for TX unit.
I2SO_WS_out	Output	In I2S master mode, outputs WS signal for TX unit.
I2S_MCLK_in	Input	In I2S slave mode, works as a clock source from the external master.
I2S_MCLK_out	Output	In I2S master mode, works as a clock source for the external slave.
I2SO_SD1_out	Output	When the PCM-to-PDM converter is enabled, works as the serial output data line for TX unit.

* Any required signals of I2S must be mapped to the chip's pins via GPIO matrix, see Chapter 5 [GPIO Matrix and IO MUX](#).

27.5 Supported Audio Standards

ESP32-C5 I2S supports multiple audio standards, including TDM Philips standard, TDM MSB alignment standard, TDM PCM standard, and PDM standard.

Select the needed standard by configuring the following bits:

- [I2S_TX/RX_TDM_EN](#)
 - 0: Disable TDM mode
 - 1: Enable TDM mode
- [I2S_TX/RX_PDM_EN](#)
 - 0: Disable PDM mode
 - 1: Enable PDM mode
- [I2S_TX/RX_MSB_SHIFT](#)
 - 0: WS and SD signals change simultaneously, i.e., enable MSB alignment standard
 - 1: WS signal changes one BCK clock cycle earlier than SD signal, i.e., enable Philips standard or select PCM standard

Note:

[I2S_TX/RX_TDM_EN](#) and [I2S_TX/RX_PDM_EN](#) must not be configured to 1 or 0 at the same time, otherwise ESP32-C5 I2S will transmit data incorrectly in a mode that is neither TDM nor PDM.

27.5.1 TDM Philips Standard

Philips standards require that WS signal changes one BCK clock cycle earlier than SD signal on BCK falling edge, which means that WS signal is valid from one clock cycle before transmitting the first bit of channel data and changes one clock before the end of channel data transfer. SD signal line transmits the most significant bit of audio data first.

Compared with the basic Philips standard, TDM Philips standard supports multiple channels. See Figure 27.5-1.

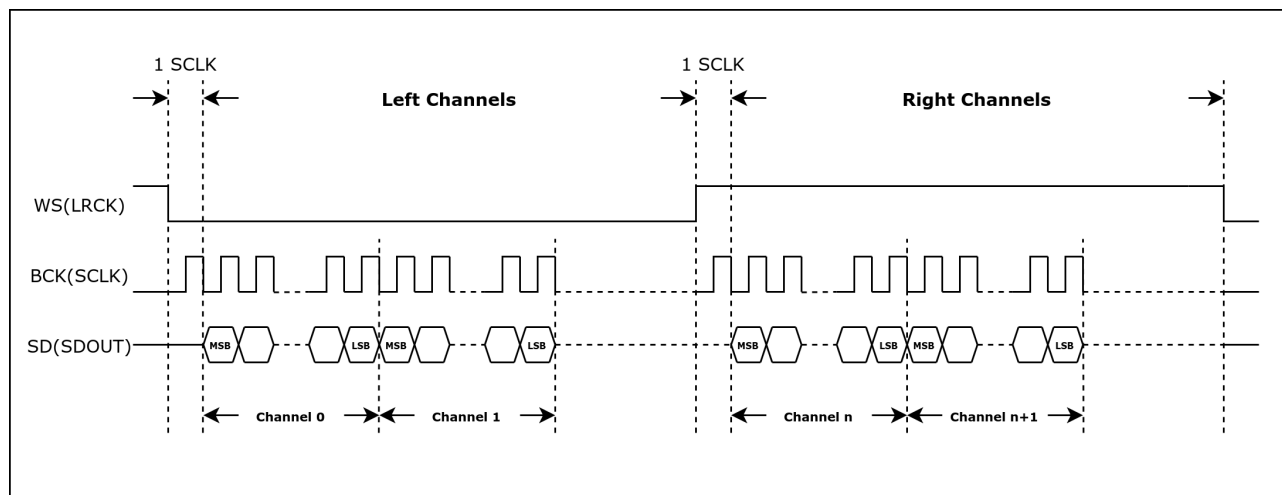


Figure 27.5-1. TDM Philips Standard Timing Diagram

27.5.2 TDM MSB Alignment Standard

MSB alignment standards require that WS and SD signals change simultaneously on the falling edge of BCK. The WS signal is valid until the end of the channel data transfer. The SD signal line transmits the most significant bit of audio data first.

Compared with the basic MSB alignment standard, TDM MSB alignment standard supports multiple channels. See Figure 27.5-2.

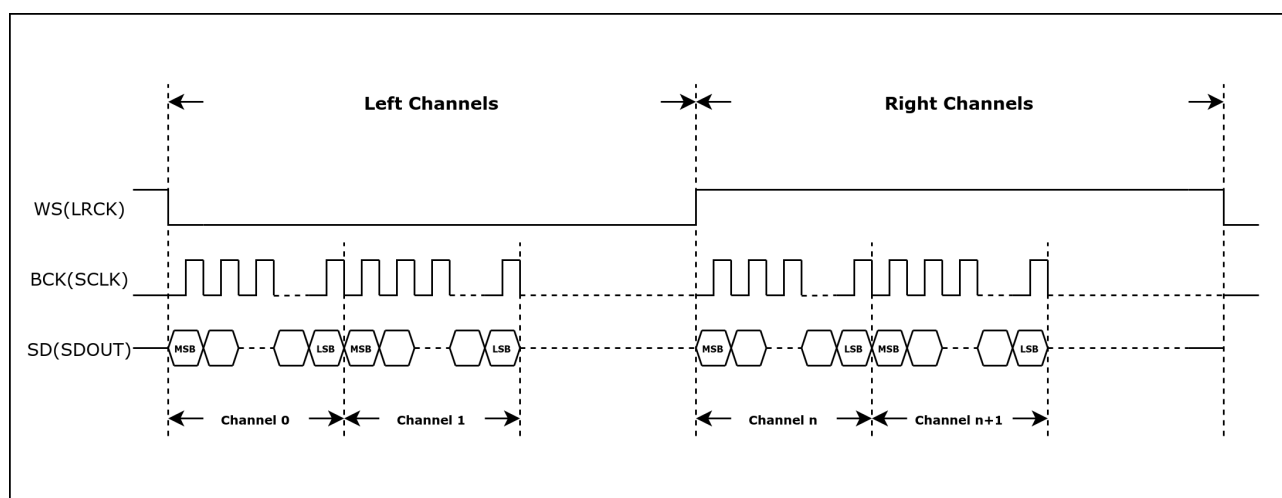


Figure 27.5-2. TDM MSB Alignment Standard Timing Diagram

27.5.3 TDM PCM Standard

Short frame synchronization under the PCM standards requires that the WS signal changes one BCK clock cycle earlier than the SD signal on the falling edge of BCK, which means that the WS signal becomes valid one clock cycle before transferring the first bit of channel data and remains unchanged in this BCK clock cycle. SD signal line first transmits the most significant bit of audio data.

Compared with the basic PCM standard, TDM PCM standard supports multiple channels. See Figure 27.5-3.

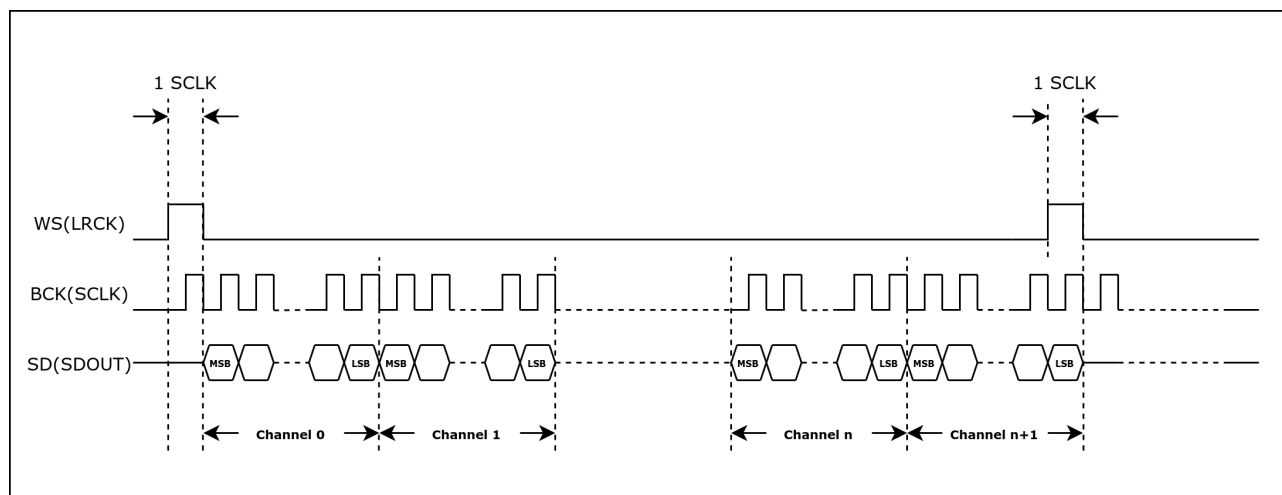


Figure 27.5-3. TDM PCM Standard Timing Diagram

27.5.4 PDM Standard

Under PDM standard, WS signal changes continuously during data transmission. The low-level and high-level of this signal indicates the left channel and right channel respectively. WS and SD signals change simultaneously. See Figure 27.5-4.

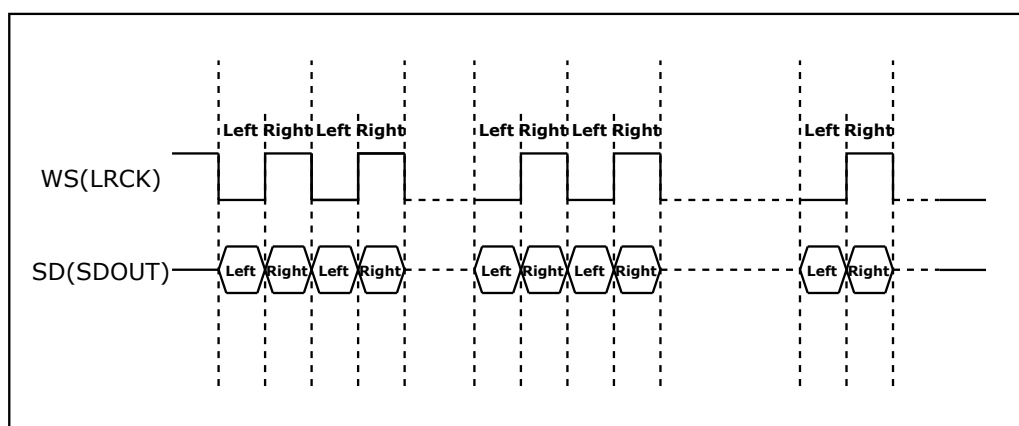


Figure 27.5-4. PDM Standard Timing Diagram

27.6 I2S TX/RX Clock

I2S_TX/RX_CLK is the master clock of I2S TX/RX unit, divided from:

- 40 MHz XTAL_CLK
- 240 MHz PLL_F240M_CLK
- 160 MHz PLL_F160M_CLK
- I2S_MCLK_in (external input clock)

The serial clock (BCK) of the I2S TX/RX unit is divided from I2S_TX/RX_CLK, as shown in Figure 27.6-1.

`PCR_I2S_TX/RX_CLKM_SEL` is used to select clock source for TX/RX unit, and `PCR_I2S_TX/RX_CLKM_EN` to enable or disable the clock source.

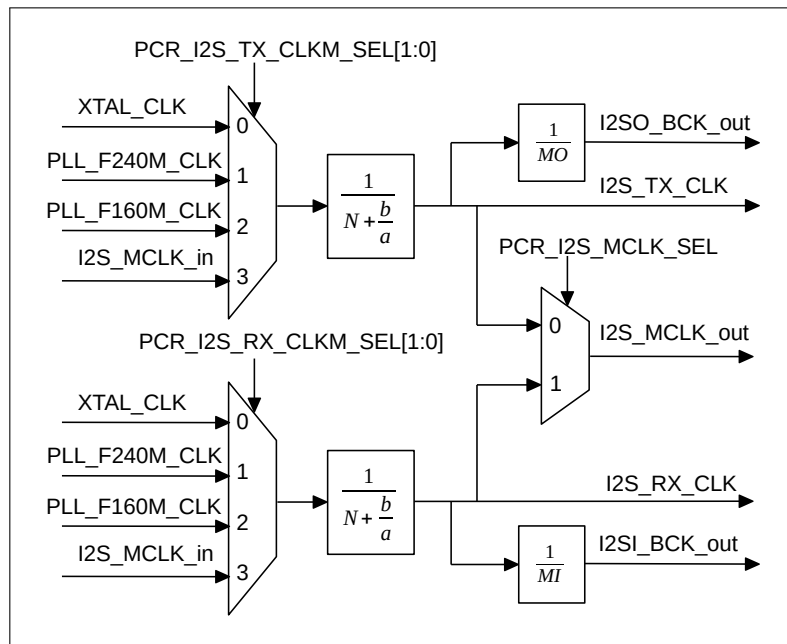


Figure 27.6-1. I2S Clock Generator

The following formula shows the relation between I2S_TX/RX_CLK frequency f_{I2S_TX/RX_CLK} and the divider clock source frequency $f_{I2S\_CLK\_S}$:

$$f_{I2S\_TX/RX\_CLK} = \frac{f_{I2S\_CLK\_S}}{N + \frac{b}{a}}$$

N is an integer value between 2 and 256. The value of N is mapped to that of `PCR_I2S_TX/RX_CLKM_DIV_NUM` as follows:

- When `PCR_I2S_TX/RX_CLKM_DIV_NUM` = 0, N = 256;
- When `PCR_I2S_TX/RX_CLKM_DIV_NUM` = 1, N = 2;
- When `PCR_I2S_TX/RX_CLKM_DIV_NUM` has any other value, N = `PCR_I2S_TX/RX_CLKM_DIV_NUM`.

The values of “a” and “b” in fractional divider depend only on x, y, z, and yn1. The corresponding formulas are as follows:

- When $b \leq \frac{a}{2}$, $yn1 = 0$, $x = \text{floor}(\lceil \frac{a}{b} \rceil) - 1$, $y = a \% b$, $z = b$;
- When $b > \frac{a}{2}$, $yn1 = 1$, $x = \text{floor}(\lceil \frac{a}{a-b} \rceil) - 1$, $y = a \% (a - b)$, $z = a - b$.

The values of x, y, z, and yn1 are configured in `PCR_I2S_TX/RX_CLKM_DIV_X`, `PCR_I2S_TX/RX_CLKM_DIV_Y`, `PCR_I2S_TX/RX_CLKM_DIV_Z`, and `PCR_I2S_TX/RX_CLKM_DIV_YN1`.

To configure the integer divider, clear `PCR_I2S_TX/RX_CLKM_DIV_X` and `PCR_I2S_TX/RX_CLKM_DIV_Z`, then set `PCR_I2S_TX/RX_CLKM_DIV_Y` to 1.

Note:

Using fractional divider may introduce some clock jitter.

In master TX mode, the serial clock BCK for I2S TX unit is `I2SO_BCK_out` divided from `I2S_TX_CLK` which is:

$$f_{I2SO\_BCK\_out} = \frac{f_{I2S\_TX\_CLK}}{MO}$$

“MO” is an integer value:

$$MO = I2S\_TX\_BCK\_DIV\_NUM + 1$$

Note:

Note that `I2S_TX_BCK_DIV_NUM` must not be configured as 1.

In master RX mode, the serial clock BCK for I2S RX unit is `I2SI_BCK_out` divided from `I2S_RX_CLK`, which is:

$$f_{I2SI\_BCK\_out} = \frac{f_{I2S\_RX\_CLK}}{MI}$$

“MI” is an integer value:

$$MI = I2S\_RX\_BCK\_DIV\_NUM + 1$$

Note:

- `I2S_RX_BCK_DIV_NUM` must not be configured as 1.
- In I2S slave mode, make sure $f_{I2S\_TX/RX\_CLK} \geq 8 * f_{BCK}$. The I2S module can output `I2S_MCLK_out` as the master clock for peripherals.

27.7 I2S Reset

The units and FIFOs in the I2S module are reset by the following bits.

- I2S TX/RX units: Reset by `I2S_TX_RESET` and `I2S_RX_RESET`;
- I2S TX/RX FIFO: Reset by `I2S_TX_FIFO_RESET` and `I2S_RX_FIFO_RESET`.

Note:

The I2S module clock must be configured first before the module and FIFO are reset.

27.8 I2S Master/Slave Mode

The ESP32-C5 I2S module can operate as a master or a slave in half-duplex and full-duplex communications, depending on the configuration of [I2S_RX_SLAVE_MOD](#) and [I2S_TX_SLAVE_MOD](#).

- [I2S_TX_SLAVE_MOD](#)
 - 0: Master TX mode
 - 1: Slave TX mode
- [I2S_RX_SLAVE_MOD](#)
 - 0: Master RX mode
 - 1: Slave RX mode

27.8.1 Master/Slave TX Mode

- I2S works as a master transmitter:
 - Set [I2S_TX_START](#) to start transmitting data. When this bit is set, the TX unit keeps driving the clock signal and serial data.
 - If [I2S_TX_STOP_EN](#) is set and all the data in FIFO is transmitted, the master stops transmitting data and clock signals.
 - If [I2S_TX_STOP_EN](#) is cleared and all the data in FIFO is transmitted, meanwhile no new data is filled into FIFO, the TX unit keeps transmitting the last data frame and clock signal.
 - Master stops transmitting data when [I2S_TX_START](#) is cleared.
- I2S works as a slave transmitter:
 - Set [I2S_TX_START](#). Wait for the master BCK clock to enable a transmit operation.
 - If [I2S_TX_STOP_EN](#) is set and all the data in FIFO is transmitted, then the slave keeps transmitting zeros, till the master stops providing BCK signal.
 - If [I2S_TX_STOP_EN](#) is cleared and all the data in FIFO is transmitted, meanwhile no new data is filled into FIFO, the TX unit keeps transmitting the last data frame.
 - If [I2S_TX_START](#) is cleared, slave keeps transmitting zeros till the master stops providing BCK clock signal.

27.8.2 Master/Slave RX Mode

- I2S works as a master receiver:
 - Set [I2S_RX_START](#) to start receiving data. When this bit is set, the RX unit keeps outputting clock signal and sampling input data.
 - Configure [I2S_RX_STOP_MODE](#) to control the suspension of data reception:
 - * 0: The RX unit only suspends data reception when [I2S_RX_START](#) is cleared.

- * 1: The RX unit suspends data reception when [I2S_RX_START](#) is cleared or the number of received bytes is greater than the value configured in [I2S_RX_EOF_NUM_REG](#). When [I2S_RX_START](#) is not cleared, data reception can be restarted by setting [I2S_RX_RESET](#) to 1.
- * 2: The RX unit suspends data reception when [I2S_RX_START](#) is cleared or GDMA RX FIFO is full. When [I2S_RX_START](#) is not cleared and the GDMA RX FIFO is no longer full, data reception can be restarted by setting [I2S_RX_RESET](#) to 1.
- RX unit stops receiving data when [I2S_RX_START](#) is cleared.
- I2S works as a slave receiver:
 - Set [I2S_RX_START](#). Wait for the master BCK signal to start receiving data.
 - Configure [I2S_RX_STOP_MODE](#) to control data reception suspension:
 - * 0: The RX unit only suspends data reception when [I2S_RX_START](#) is cleared.
 - * 1: The RX unit suspends data reception when the number of received bytes is greater than the value configured in [I2S_RX_EOF_NUM_REG](#). When [I2S_RX_START](#) is not cleared, data reception can be restarted by setting [I2S_RX_RESET](#) to 1.
 - * 2: The RX unit suspends data reception when [I2S_RX_START](#) is cleared or GDMA RX FIFO is full. When [I2S_RX_START](#) is not cleared and the GDMA RX FIFO is no longer full, data reception can be restarted by setting [I2S_RX_RESET](#) to 1.
 - The RX unit stops receiving data when [I2S_RX_START](#) is cleared.

27.9 Transmitting Data

Note:

Updating the configuration described in this and subsequent sections requires to set [I2S_TX_UPDATE](#) accordingly to synchronize registers from APB clock domain to TX clock domain. For more detailed configurations, see Section [27.13.1](#).

In TX mode, I2S first reads data through GDMA and transmits these data out via output signals according to the configured data mode and channel mode.

27.9.1 Data Format Control

Data format is controlled in the following phases:

- Phase I: Read data from memory and write it to TX FIFO;
- Phase II: Read the TX data from TX FIFO and convert the data according to the output data mode;
- Phase III: Clock out the TX data serially.

27.9.1.1 Bit Width Control of Channel Valid Data

The bit width of the valid data in each channel is determined by [I2S_TX_BITS_MOD](#) and [I2S_TX_24_FILL_EN](#). For details, see the table below.

Table 27.9-1. Bit Width of Channel Valid Data

Channel Valid Data Width	I2S_TX_BITS_MOD	I2S_TX_24_FILL_EN
32	31	x [*]
	23	1
24	23	0
16	15	x
8	7	x

* “x” represents that this value is ignored (applies to all tables in this chapter).

27.9.1.2 Endian Control of Channel Valid Data

When I2S reads data through GDMA, the data endian under various data width is controlled by [I2S_TX_BIG_ENDIAN](#). Table 27.9-2 shows how [I2S_TX_BIG_ENDIAN](#) controls the reading of the data with different valid data widths of the channel.

Table 27.9-2. Endian of Channel Valid Data

Channel Valid Data Width	Original Data	Endian of Processed Data	I2S_TX_BIG_ENDIAN
32	{B3, B2, B1, B0}	{B3, B2, B1, B0}	0
		{B0, B1, B2, B3}	1
24	{B2, B1, B0}	{B2, B1, B0}	0
		{B0, B1, B2}	1
16	{B1, B0}	{B1, B0}	0
		{B0, B1}	1
8	{B0}	{B0}	x

Note:

B0, B1, B2, B3 each represents an 8-bit data, and the symbol {} indicates that the bytes are combined together. For example, {B3, B2, B1, B0} represents a 32-bit data, wherein B0 represents bit 0-7, B1 represents bit 8-15, B2 represents bit 16-23, and B3 represents bit 24-31.

27.9.1.3 A-law/ μ -law Compression and Decompression

ESP32-C5 I2S compresses/decompresses the valid data into 32-bit by A-law or by μ -law. If the bit width of valid data is smaller than 32, zeros are filled to the extra high bits of the data to be compressed/decompressed by default.

Note:

Extra high bits here mean the bits[31: channel valid data width] of the data to be compressed/decompressed.

Configure [I2S_TX_PCM_BYPASS](#):

- 0: Compress or decompress the data
- 1: Do not compress or decompress the data

Configure [I2S_TX_PCM_CONF](#):

- 0: Decompress the data using A-law
- 1: Compress the data using A-law
- 2: Decompress the data using μ -law
- 3: Compress the data using μ -law

At this point, the first phase of data format control is completed.

27.9.1.4 Bit Width Control of Channel TX Data

The TX data width in each channel is determined by [I2S_TX_TDM_CHAN_BITS](#).

- If TX data width in each channel is larger than the valid data width, zeros will be filled to these extra bits.
Configure [I2S_TX_LEFT_ALIGN](#):
 - 0: The valid data is at the lower bits of TX data. Zeros are filled into higher bits of TX data;
 - 1: The valid data is at the higher bits of TX data. Zeros are filled into lower bits of TX data.
- If the TX data width in each channel is smaller than the valid data width, only the lower bits of valid data are sent out, and the higher bits are discarded.

At this point, the second phase of data format control is completed.

27.9.1.5 Bit Order Control of Channel Data

The data bit order in each channel is controlled by [I2S_TX_BIT_ORDER](#):

- 0: Not reverse the valid data bit order;
- 1: Reverse the valid data bit order.

At this point, the data format control is completed. The data after format control will be sent sequentially from high to low. Figure [27.9-1](#) shows the complete process of TX data format control.

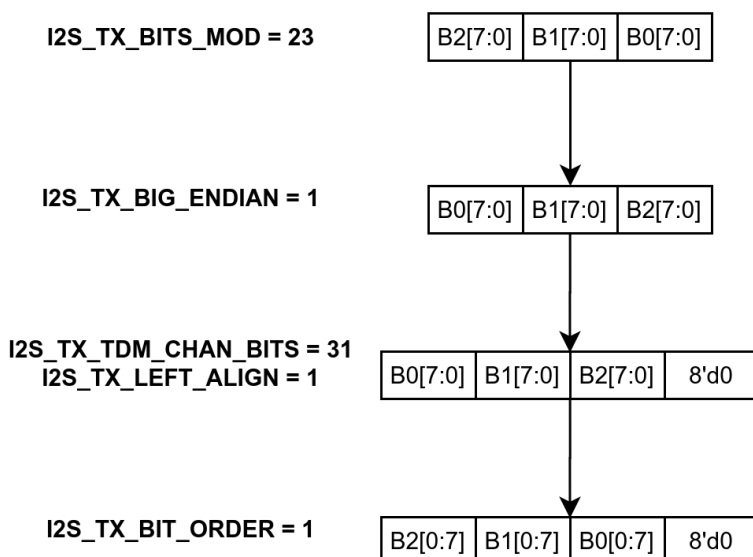


Figure 27.9-1. TX Data Format Control

27.9.2 Channel Mode Control

ESP32-C5 I2S supports both TDM TX mode and PDM TX mode. Set [I2S_TX_TDM_EN](#) to enable TDM TX mode, or set [I2S_TX_PDM_EN](#) to enable PDM TX mode.

Note:

[I2S_TX_TDM_EN](#) and [I2S_TX_PDM_EN](#) must not be cleared or set simultaneously.

27.9.2.1 TDM TX Mode

In TDM TX mode, I2S supports up to 16 channels to transmit data. The total number of TX channels in use is controlled by [I2S_TX_TDM_TOT_CHAN_NUM](#). For example, If [I2S_TX_TDM_TOT_CHAN_NUM](#) is set to 5, six channels in total (channel 0-5) will be used to transmit data. See Figure 27.9-2.

Note:

Most stereo I2S codecs can be controlled by setting the I2S module into 2-channel mode under TDM standard.

In these TX channels, if [I2S_TX_TDM_CHAN_n_EN](#) is set to:

- 1: This channel transmits the channel data out;
- 0: The TX data to be sent by this channel is controlled by [I2S_TX_CHAN_EQUAL](#):
 - 1: The data of the previous channel is sent out;
 - 0: The data stored in [I2S_SINGLE_DATA](#) is sent out.

In TDM TX master mode, WS signal is controlled by [I2S_TX_WS_IDLE_POL](#) and [I2S_TX_TDM_WS_WIDTH](#):

- [I2S_TX_WS_IDLE_POL](#): The default level of WS signal;
- [I2S_TX_TDM_WS_WIDTH](#): The cycles the WS default level lasts for when transmitting all channel data.

$I2S\_TX\_HALF\_SAMPLE\_BITS \times 2$ is equal to the BCK cycles in one WS period.

TDM Channel Configuration Example

In this example, the register configuration is as follows:

- $I2S\_TX\_TDM\_CHAN\_NUM = 5$, i.e., channel 0-5 are used to transmit data.
- $I2S\_TX\_CHAN\_EQUAL = 1$, i.e., data of the previous channel will be transmitted if the $I2S\_TX\_TDM\_CHANn\_EN$ ($n = 0-5$) is cleared.
- $I2S\_TX\_TDM\_CHAN0/2/5\_EN = 1$, i.e., these channels transmit their channel data out.
- $I2S\_TX\_TDM\_CHAN1/3/4\_EN = 0$, i.e., these channels transmit the previous channel's data out.

Once the configuration is done, data is transmitted as follows.

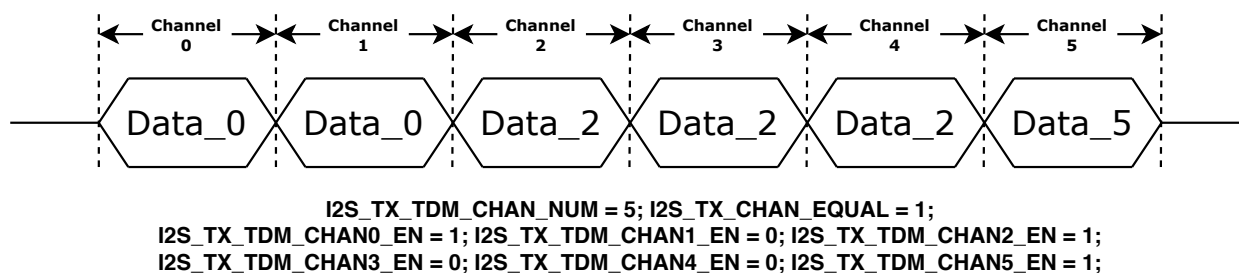


Figure 27.9-2. TDM Channel Control

27.9.2.2 PDM TX Mode

In PDM TX mode, I2S supports both PDM raw data transmission and PCM-to-PDM data format conversion.

In PDM TX mode, fetching data through GDMA is controlled by $I2S\_TX\_MONO$ and $I2S\_TX\_MONO\_FST\_VLD$. See Table 27.9-3. Configure the two bits according to the data stored in memory, be it the single-channel or dual-channel data.

Table 27.9-3. Data-Fetching Control in PDM Mode

Data-Fetching Control Option	Mode	$I2S\_TX\_MONO$	$I2S\_TX\_MONO\_FST\_VLD$
Post data-fetching request to GDMA at any edge of WS signal	Stereo mode	0	x
Post data-fetching request to GDMA only at the second half period of WS signal	Mono mode	1	0
Post data-fetching request to GDMA only at the first half period of WS signal	Mono mode	1	1

When I2S is in PDM TX master mode, the default level of WS signal is controlled by $I2S\_TX\_WS\_IDLE\_POL$, and the WS signal frequency is half of the BCK signal frequency. The configuration of WS signal is similar to that of BCK signal. Please refer to Section 27.6 and Figure 27.6.

When **transmitting raw PDM data**, the I2S channel mode is controlled by $I2S\_TX\_CHAN\_MOD$ and $I2S\_TX\_WS\_IDLE\_POL$. See the table below.

Table 27.9-4. I2S Channel Control for Raw PDM Data

Channel Control Option	Left Channel	Right Channel	Mode Control Field ¹	Channel Select Bit ²
Stereo mode	Transmit the left channel data	Transmit the right channel data	0	x
Mono mode	Transmit the left channel data	Transmit the left channel data	1	0
	Transmit the right channel data	Transmit the right channel data	1	1
	Transmit the right channel data	Transmit the right channel data	2	0
	Transmit the left channel data	Transmit the left channel data	2	1
	Transmit the value of “single” ³	Transmit the right channel data	3	0
	Transmit the left channel data	Transmit the value of “single”	3	1
	Transmit the left channel data	Transmit the value of “single”	4	0
	Transmit the value of “single”	Transmit the right channel data	4	1

¹ [I2S_TX_CHAN_MOD](#)² [I2S_TX_WS_IDLE_POL](#)³ The “single” value is equal to the value of [I2S_SINGLE_DATA](#).

If the **PCM-to-PDM converter is enabled**, the PCM data through GDMA is converted to PDM data and then output in PDM signal format. Configure [I2S_PCM2PDM_CONV_EN](#) to enable the converter. The register configuration for the PCM-to-PDM converter is as follows:

- Configure 1-line PDM output format or 1-/2-line DAC output mode as the table below:

Table 27.9-5. I2S PCM-to-PDM Data Output

Channel Output Format	I2S_TX_PDM_DAC_MODE_EN	I2S_TX_PDM_DAC_2OUT_EN
1-line PDM output format ¹	0	x
1-line DAC output format ²	1	0
2-line DAC output format	1	1

Note:

- In PDM output format, SD data of two channels is sent out in one WS period.
- In DAC output format, SD data of one channel is sent out in one WS period.
- 1-line PDM/DAC output format uses [I2SO_Data_out](#) as the data output line.
- 2-line DAC output format uses [I2SO_Data_out](#) and [I2SO_Data1_out](#) as data output lines.
- In terms of application, the PDM output format is targeted at applications that require clock signals and can decode PDM format codecs, while the DAC output format is targeted at applications that do not rely on clock signals, do not have PDM format codecs, and can recover analog waveforms directly through low-pass filtering.

- Configure sampling frequency and upsampling rate as below:

When the I2S PCM-to-PDM converter is enabled, PDM clock frequency is equal to BCK frequency. The relation of sampling frequency (f_{Sampling} , the sampling frequency of PCM data) and BCK frequency (the

sampling frequency of PDM data) is as follows:

$$f_{\text{Sampling}} = \frac{f_{\text{BCK}}}{\text{OSR}}$$

Upsampling rate (OSR) is related to `I2S_TX_PDM_SINC_OSR2` as follows:

$$\text{OSR} = \text{I2S\_TX\_PDM\_SINC\_OSR2} \times 64$$

Sampling frequency f_{Sampling} is related to `I2S_TX_PDM_FS` as follows:

$$f_{\text{Sampling}} = \text{I2S\_TX\_PDM\_FS} \times 100$$

Configure the registers according to the needed sampling frequency, upsampling rate, and PDM clock frequency.

PDM Channel Configuration Example

In this example of I2S, the register configuration is as follows.

- `I2S_PCM2PDM_CONV_EN` = 0, i.e., transmit the raw PDM data.
- `I2S_TX_MONO` = 0, i.e., data is fetched from memory via GDMA in both the high and low levels of WS.
- `I2S_TX_CHAN_MOD` = 2, i.e., mono mode is selected.
- `I2S_TX_WS_IDLE_POL` = 1, i.e., both the left and right channels transmit the left channel data, and the right channel data will be discarded.

Once the configuration is done, assume that the data in memory **after data format control** is:

Left	Right	Left	Right	...	Left	Right

Note:

1. The data above refers to the processed data after data format control instead of the original data.
2. “Left” and “Right” represent channel data, and their bit widths are channel valid data width. Please refer to Section 27.9.1□

Then the channel data is transmitted after channel mode control as follows.

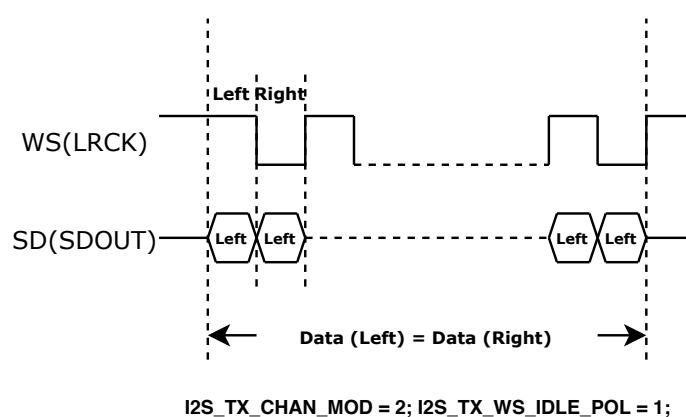


Figure 27.9-3. PDM Channel Control Example

27.9.3 Synchronous Counter

ESP32-C5 I2S TX unit contains synchronous counters to reflect the current I2S data transmission status. The counters can be used to detect I2S transmit data errors or to synchronize data when multiple devices are using I2S for data transmission.

ESP32-C5 I2S contains two types of synchronous counters:

- [I2S_TX_FIFO_CNT](#)
 - Each time the I2S TX FIFO reads data, the counter is incremented by 1.
 - The counter can be reset by setting [I2S_TX_FIFO_CNT_RST](#).
 - The counter is 31-bit wide and is automatically reset to zero on overflow.
- [I2S_TX_BCK_CNT](#)
 - Each time I2S TX transmits a BCK cycle, the counter is incremented by 1.
 - The counter can be reset by setting [I2S_TX_BCK_CNT_RST](#).
 - The counter is 31-bit wide and is automatically reset to zero on overflow.

Note:

- Each time the I2S TX FIFO transmits data, a channel data is sent. Therefore, when I2S transmits the data correctly, $I2S\_TX\_BCK\_CNT = I2S\_TX\_FIFO\_CNT * \text{the bit width of channel TX data}$ (see Section 27.9.1.4).
- When the I2S TX unit transmits the data, there is some delay between fetching data from the FIFO and transmitting it to the line, so the value of [I2S_TX_BCK_CNT](#) might be a little different from that of [I2S_TX_FIFO_CNT](#).

Application scenarios

1. I2S TX synchronization for multiple devices

When multiple devices transmit data through the I2S TX at the same time, desynchronization may occur after a long period of time due to small frequency differences in the clock sources. In this case, read the [I2S_TX_FIFO_CNT](#) value of each device to determine the location of the audio data being sent to perform synchronization.

2. I2S TX stability check

When the device is in an application scenario with high GDMA usage, there may be numbers missing in I2S TX occasionally due to multiplexing arbitration. In this case, check the ratio of [I2S_TX_BCK_CNT](#) to [I2S_TX_FIFO_CNT](#) to confirm the stability of the device when running I2S TX.

27.10 Receiving Data

In RX mode, I2S first reads data from the peripheral interface and then stores the data in memory via GDMA according to the configured channel mode and data mode.

27.10.1 Channel Mode Control

ESP32-C5 I2S supports both TDM RX mode and PDM RX mode. Set [I2S_RX_TDM_EN](#) to enable TDM RX mode, or set [I2S_RX_PDM_EN](#) to enable PDM RX mode.

Note:

`I2S_RX_TDM_EN` and `I2S_RX_PDM_EN` must not be cleared or set simultaneously.

27.10.1.1 TDM RX Mode

In TDM RX mode, I2S supports up to 16 channels to input data. The total number of RX channels in use is controlled by `I2S_RX_TDM_TOT_CHAN_NUM`. For example, if `I2S_RX_TDM_TOT_CHAN_NUM` is set to 5, channel 0-5 will be used to receive data.

In these RX channels, if `I2S_RX_TDM_CHANn_EN` is set to:

- 1: The channel data is valid and will be stored into RX FIFO;
- 0: The channel data is invalid and will not be stored into RX FIFO.

In TDM master mode, WS signal is controlled by `I2S_RX_WS_IDLE_POL` and `I2S_RX_TDM_WS_WIDTH`.

- `I2S_RX_WS_IDLE_POL`: The default level of WS signal;
- `I2S_RX_TDM_WS_WIDTH`: The cycles the WS default level lasts for when receiving all channel data.

`I2S_RX_HALF_SAMPLE_BITS` x 2 is equal to the BCK cycles in one WS period.

27.10.1.2 PDM RX Mode

In PDM RX mode, I2S supports PDM raw data reception. It converts the serial data from channels to the data to be entered into memory.

In PDM RX master mode, the default level of the WS signal is controlled by `I2S_RX_WS_IDLE_POL`. WS frequency is half of the BCK frequency. The configuration of the BCK signal is similar to that of WS signal as described in Section 27.6. Note, in PDM RX mode, the value of `I2S_RX_HALF_SAMPLE_BITS` must be same as that of `I2S_RX_BITS_MOD`.

27.10.2 Data Format Control

The data format of I2S is controlled in the following phases:

- Phase I: Serial input data is converted into the data to be saved to RX FIFO;
- Phase II: The data is read from RX FIFO and converted according to the input data mode.

27.10.2.1 Bit Order Control of Channel Data

The channel data will be stored as the data to be input in order from high to low. The data bit order in each channel is controlled by `I2S_RX_BIT_ORDER`:

- 0: The bit order of the data to be input is not reversed;
- 1: The bit order of the data to be input is reversed.

At this point, the first phase of data format control is completed. The data to be input after bit order control is stored in the RX FIFO.

27.10.2.2 Bit Width Control of Channel Storage (Valid) Data

The storage data width in each channel is controlled by [I2S_RX_BITS_MOD](#) and [I2S_RX_24_FILL_EN](#). See the table below.

Table 27.10-1. Channel Storage Data Width

Channel Storage Data Width	I2S_RX_BITS_MOD	I2S_RX_24_FILL_EN
32	31	x
	23	1
24	23	0
16	15	x
8	7	x

27.10.2.3 Bit Width Control of Channel RX Data

The RX data width in each channel is determined by [I2S_RX_TDM_CHAN_BITS](#).

- If the storage data width in each channel is smaller than the received (RX) data width, then only the bits within the storage data width is saved into memory. Configure [I2S_RX_LEFT_ALIGN](#) to:
 - 0: Only the lower bits of the received data within the storage data width is stored to memory;
 - 1: Only the higher bits of the received data within the storage data width is stored to memory.
- If the received data width is smaller than the storage data width in each channel, the higher bits of the received data will be filled with zeros and then the data is saved to memory.

27.10.2.4 Endian Control of Channel Storage Data

The received data is then converted into storage data (to be stored to memory) after some processing, such as discarding extra bits or filling zeros in missing bits. The endian of the storage data is controlled by [I2S_RX_BIG_ENDIAN](#) under various data width. See the table below.

Table 27.10-2. Channel Storage Data Endian

Channel Storage Data Width	Original Data	Endian of Processed Data	I2S_RX_BIG_ENDIAN
32	{B3, B2, B1, B0}	{B3, B2, B1, B0}	0
		{B0, B1, B2, B3}	1
24	{B2, B1, B0}	{B2, B1, B0}	0
		{B0, B1, B2}	1
16	{B1, B0}	{B1, B0}	0
		{B0, B1}	1
8	{B0}	{B0}	x

27.10.2.5 A-law/ μ -law Compression and Decompression

ESP32-C5 I2S compresses/decompresses the storage data in 32-bit by A-law or by μ -law. By default, zeros are filled into high bits.

Configure [I2S_RX_PCM_BYPASS](#):

- 0: Compress or decompress the data
- 1: Do not compress or decompress the data

Configure [I2S_RX_PCM_CONF](#):

- 0: Decompress the data using A-law
- 1: Compress the data using A-law
- 2: Decompress the data using μ -law
- 3: Compress the data using μ -law

At this point, the data format control is completed. Data then is stored into memory via GDMA.

27.11 Event Task Matrix Feature

ESP32-C5 I2S supports the Event Task Matrix (ETM) function, which allows I2S's ETM tasks to be triggered by any peripherals' ETM events, or I2S's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to I2S. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

I2S can receive the following ETM tasks:

- I2SO_TASK_START_TX: Enables I2S TX for data transfer.
- I2SO_TASK_START_RX: Enables I2S RX for data transfer.
- I2SO_TASK_STOP_TX: Stops I2S TX data transfer.
- I2SO_TASK_STOP_RX: Stops I2S RX data transfer.

I2S can generate the following ETM events:

- I2SO_EVT_TX_DONE: Indicates that I2S TX has completed data transmission. Triggered when all data in the TX FIFO has been sent.
- I2SO_EVT_RX_DONE: Can be triggered in different ways depending on the configured value of [I2S_RX_STOP_MODE](#):
 - 0: Will not be triggered;
 - 1: When triggered, indicates that the number of bytes received by I2S RX is greater than the receive length value configured by [I2S_RX_EOF_NUM_REG](#);
 - 2: When triggered, indicates that the GDMA RX FIFO is full.
- I2SO_EVT_X_WORDS_SENT: Indicates that the word number sent by I2S TX is equal to or larger than the value set by [I2S_ETM_TX_SEND_WORD_NUM](#).
- I2SO_EVT_X_WORDS_RECEIVED: Indicates that the word number received by I2S RX is equal to or larger than the value set by [I2S_ETM_RX_RECEIVE_WORD_NUM](#).

In practical applications, I2S's ETM events can trigger its own ETM tasks. For example, the I2S_EVT_X_WORDS_SENT event can trigger the I2S_TASK_STOP_TX task, and in this way stop the I2S operation through ETM.

27.12 I2S Interrupts

ESP32-C5's I2S can generate the I2S_INTR interrupt signal that will be sent to the [Interrupt Matrix](#). There are several internal interrupt sources from I2S that can generate the above interrupt signal(s) as follows:

- I2S_TX_HUNG_INT: Triggered when data transmission is timed out. For example, if the I2S module is configured as TX slave mode, but the master does not provide BCK or WS signals for a long time (specified in [I2S_LC_HUNG_CONF_REG](#)), this interrupt will be triggered.
- I2S_RX_HUNG_INT: Triggered when the data reception is timed out. For example, if the I2S module is configured as RX slave mode, but the master does not transmit data for a long time (specified in [I2S_LC_HUNG_CONF_REG](#)), this interrupt will be triggered.
- I2S_TX_DONE_INT: Triggered when sending data is complete, i.e., all data in I2S TX FIFO has been sent.
- I2S_RX_DONE_INT: Can be triggered in different ways depending on the configured value of [I2S_RX_DONE_MODE](#):
 - 0: Triggered when the number of bytes received by I2S RX is greater than the receive length value configured by [I2S_RX_EOF_NUM_REG](#);
 - 1: Triggered when the GDMA RX FIFO is full.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [27.14 Register Summary](#).

27.13 Software Configuration Process

27.13.1 Configure I2S as TX Mode

Follow the steps below to configure I2S as TX mode via software:

1. Configure the clock as described in Section [27.6](#).
2. Configure signal pins according to Table [27.4-1](#).
3. Select the mode needed by configuring [I2S_TX_SLAVE_MOD](#).
 - 0: Master TX mode
 - 1: Slave TX mode
4. Set needed TX data mode and TX channel mode as described in Section [27.9](#), and then set [I2S_TX_UPDATE](#).
5. Reset TX unit and TX FIFO as described in Section [27.7](#).
6. Enable corresponding interrupts. See Section [27.12](#).
7. Configure GDMA outlink.

8. Set [I2S_TX_STOP_EN](#) if needed. For more information, please refer to Section [27.8.1](#).
9. Start transmitting data:
 - In master mode, wait till I2S slave gets ready, then set [I2S_TX_START](#) to start transmitting data;
 - In slave mode, set [I2S_TX_START](#). When the I2S master supplies BCK and WS signals, I2S slave starts transmitting data.
10. Wait for the interrupt signals set in Step [6](#), or check whether the transfer is completed by querying [I2S_TX_IDLE](#):
 - 0: Transmitter is working;
 - 1: Transmitter is in idle state.
11. Clear [I2S_TX_START](#) to stop data transfer.

27.13.2 Configure I2S as RX Mode

Follow the steps below to configure I2S as RX mode via software:

1. Configure the clock as described in Section [27.6](#).
2. Configure signal pins according to Table [27.4-1](#).
3. Select the mode needed by configuring [I2S_RX_SLAVE_MOD](#).
 - 0: Master RX mode
 - 1: Slave RX mode
4. Set needed RX data mode and RX channel mode as described in Section [27.10](#), and then set [I2S_RX_UPDATE](#).
5. Reset the RX unit and its FIFO according to Section [27.7](#).
6. Enable the corresponding interrupts. See Section [27.12](#).
7. Configure GDMA inlink and set the length of RX data by configuring [I2S_RX_EOF_NUM_REG](#).
8. Start receiving data:
 - In master mode, when the slave is ready, set [I2S_RX_START](#) to start receiving data.
 - In slave mode, set [I2S_RX_START](#) to start receiving data when BCK and WS signals are received from the master.
9. The received data is then stored to the specified memory address according to the configuration of GDMA. Then the corresponding interrupt set in step [6](#) is generated.

27.14 Register Summary

The addresses in this section are relative to I2S base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Interrupt registers			
I2S_INT_RAW_REG	I2S interrupt raw register	0x000C	R/SS/WTC
I2S_INT_ST_REG	I2S interrupt status register	0x0010	RO
I2S_INT_ENA_REG	I2S interrupt enable register	0x0014	R/W
I2S_INT_CLR_REG	I2S interrupt clear register	0x0018	WT
RX Control and configuration registers			
I2S_RX_CONF_REG	I2S RX configuration register	0x0020	varies
I2S_RX_CONF1_REG	I2S RX configuration register 1	0x0028	R/W
I2S_RX_TDM_CTRL_REG	I2S TX TDM mode control register	0x0050	R/W
I2S_RXEOF_NUM_REG	I2S RX data number control register	0x0064	R/W
TX Control and configuration registers			
I2S_TX_CONF_REG	I2S TX configuration register	0x0024	varies
I2S_TX_CONF1_REG	I2S TX configuration register 1	0x002C	R/W
I2S_TX_PCM2PDM_CONF_REG	I2S TX PCM-to-PDM configuration register	0x0044	R/W
I2S_TX_PCM2PDM_CONF1_REG	I2S TX PCM-to-PDM configuration register	0x0048	R/W
I2S_TX_TDM_CTRL_REG	I2S TX TDM mode control register	0x0054	R/W
RX clock and timing registers			
I2S_RX_TIMING_REG	I2S RX timing control register	0x0058	R/W
TX clock and timing registers			
I2S_TX_TIMING_REG	I2S TX timing control register	0x005C	R/W
Control and configuration registers			
I2S_LC_HUNG_CONF_REG	I2S timeout configuration register.	0x0060	R/W
I2S_CONF_SIGLE_DATA_REG	I2S single data register	0x0068	R/W
TX status registers			
I2S_STATE_REG	I2S TX status register	0x006C	RO
ETM register			
I2S_ETM_CONF_REG	I2S ETM configuration register	0x0070	R/W
Sync counter registers			
I2S_FIFO_CNT_REG	I2S sync counter register	0x0074	varies
I2S_BCK_CNT_REG	I2S sync counter register	0x0078	varies
Clock registers			
I2S_CLK_GATE_REG	Clock gate register	0x007C	R/W
Version register			
I2S_DATE_REG	Version control register	0x0080	R/W

27.15 Registers

The addresses in this section are relative to I2S base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 27.1. I2S_INT_RAW_REG (0x000C)

(reserved)																																I2S_TX_HUNG_INT_RAW I2S_RX_HUNG_INT_RAW I2S_TX_DONE_INT_RAW I2S_RX_DONE_INT_RAW																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
31																															4	3	2	1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2S_RX_DONE_INT_RAW The raw interrupt status of the [I2S_RX_DONE_INT](#) interrupt. (R/SS/WTC)

I2S_TX_DONE_INT_RAW The raw interrupt status of the [I2S_TX_DONE_INT](#) interrupt.
(R/SS/WTC)

I2S_RX_HUNG_INT_RAW The raw interrupt status of the [I2S_RX_HUNG_INT](#) interrupt. (R/SS/WTC)

I2S_TX_HUNG_INT_RAW The raw interrupt status of the [I2S_TX_HUNG_INT](#) interrupt. (R/SS/WTC)

Register 27.2. I2S_INT_ST_REG (0x0010)

(reserved)																												I2S_TX_HUNG_INT_ST I2S_RX_HUNG_INT_ST I2S_TX_DONE_INT_ST I2S_RX_DONE_INT_ST					
31																												4	3	2	1	0	Reset
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												0	0	0	0		

I2S_RX_DONE_INT_ST The masked interrupt status of the [I2S_RX_DONE_INT](#) interrupt. (RO)

I2S_TX_DONE_INT_ST The masked interrupt status of the [I2S_TX_DONE_INT](#) interrupt. (RO)

I2S_RX_HUNG_INT_ST The masked interrupt status of the [I2S_RX_HUNG_INT](#) interrupt. (RO)

I2S_TX_HUNG_INT_ST The masked interrupt status of the [I2S_TX_HUNG_INT](#) interrupt. (RO)

Register 27.3. I2S_INT_ENA_REG (0x0014)

(reserved)																												I2S_TX_HUNG_INT_ENA I2S_RX_HUNG_INT_ENA I2S_TX_DONE_INT_ENA I2S_RX_DONE_INT_ENA					
31																												4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

I2S_RX_DONE_INT_ENA Write 1 to enable the [I2S_RX_DONE_INT](#) interrupt. (R/W)

I2S_TX_DONE_INT_ENA Write 1 to enable the [I2S_TX_DONE_INT](#) interrupt. (R/W)

I2S_RX_HUNG_INT_ENA Write 1 to enable the [I2S_RX_HUNG_INT](#) interrupt. (R/W)

I2S_TX_HUNG_INT_ENA Write 1 to enable the [I2S_TX_HUNG_INT](#) interrupt. (R/W)

Register 27.4. I2S_INT_CLR_REG (0x0018)

(reserved)																																I2S_TX_HUNG_INT_CLR I2S_RX_HUNG_INT_CLR I2S_TX_DONE_INT_CLR I2S_RX_DONE_INT_CLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31																															4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

I2S_RX_DONE_INT_CLR Write 1 to clear the [I2S_RX_DONE_INT](#) interrupt. (WT)

I2S_TX_DONE_INT_CLR Write 1 to clear the [I2S_TX_DONE_INT](#) interrupt. (WT)

I2S_RX_HUNG_INT_CLR Write 1 to clear the [I2S_RX_HUNG_INT](#) interrupt. (WT)

I2S_TX_HUNG_INT_CLR Write 1 to clear the [I2S_TX_HUNG_INT](#) interrupt. (WT)

Register 27.5. I2S_RX_CONF_REG (0x0020)

[illegible]

I2S_RX_RESET Configures whether to reset RX unit.

0: No effect

1: Reset

(WT)

I2S_RX_FIFO_RESET Configures whether to reset RX FIFO.

0: No effect

1: Reset

(WT)

I2S_RX_START Configures whether to start receiving data.

0: No effect

1: Start

(R/W/SC)

I2S_RX_SLAVE_MOD Configures whether to enable slave RX mode.

0: Enable master mode

1: Enable slave mode

(R/W)

I2S_RX_STOP_MODE Configures when I2S RX stop data reception.

0: Only stops when I2S_RX_START is cleared

1: Stops when `I2S_RX_START` is cleared or the number of received bytes is greater than the value configured in `I2S_RX_EOF_NUM_REG`

2: Stops when I2S_RX_START is cleared or GDMA RX FIFO is full

(R/W)

I2S_RX_MONO Configures whether to enable RX unit in mono mode.

0: Disable

1: Enable

(R/W)

I2S_RX_BIG_ENDIAN Configures I2S RX byte endian.

0: Low address data is saved to low address

1: Low address data is saved to high address

(R/W)

Continued on the next page...

Register 27.5. I2S_RX_CONF_REG (0x0020)

Continued from the previous page...

I2S_RX_UPDATE Configures whether to update I2S RX registers from APB clock domain to I2S RX clock domain.

0: No effect

1: Update

This bit will be cleared by hardware after the register update is done.

(R/W/SC)

I2S_RX_MONO_FST_VLD Configures the valid data channel in I2S RX mono mode.

0: The second channel data valid

1: The first channel data valid

(R/W)

I2S_RX_PCM_CONF Configures I2S RX compress/decompress mode.

0 (atol): A-law decompress

1 (ltoa): A-law compress

2 (utol): μ -law decompress

3 (ltou): μ -law compress

(R/W)

I2S_RX_PCM_BYPASS Configures whether to bypass the Compress/Decompress units for received data.

0: No effect

1: Bypass

(R/W)

I2S_RX_MSB_SHIFT Configures the timing between the WS signal and the MSB of data.

0: Align at the rising edge

1: The WS signal changes one BCK clock earlier

(R/W)

I2S_RX_DONE_MODE Configures when to trigger the [I2S_RX_DONE_INT](#) interrupt.

0: When RX FIFO is full

1: When IN_SUC_EOF is 1

(R/W)

I2S_RX_LEFT_ALIGN Configures I2S RX alignment mode.

0: Right alignment mode

1: Left alignment mode

(R/W)

I2S_RX_24_FILL_EN Configures the bit number that the 24-bit channel data is stored to.

0: Store 24-bit channel data to 24 bits

1: Store 24-bit channel data to 32 bits (Extra bits are filled with zeros)

(R/W)

Continued on the next page...

Register 27.5. I2S_RX_CONF_REG (0x0020)

Continued from the previous page...

I2S_RX_WS_IDLE_POL Configures the relationship between WS level and which channel data to receive.

0: WS remains low when receiving left channel data and high when receiving right channel data

1: WS remains high when receiving left channel data and low when receiving right channel data

(R/W)

I2S_RX_BIT_ORDER Configures whether to reverse the bit order of the I2S RX data to be received.

0: Not reverse

1: Reverse

(R/W)

I2S_RX_TDM_EN Configures whether to enable I2S TDM RX mode.

0: Disable

1: Enable

(R/W)

I2S_RX_PDM_EN Configures whether to enable I2S PDM RX mode.

0: Disable

1: Enable

(R/W)

I2S_RX_BCK_DIV_NUM Configures the divider of BCK in RX mode. Note this divider must not be configured to 1. (R/W)

Register 27.6. I2S_RX_CONF1_REG (0x0028)

I2S_RX_TDM_CHAN_BITS				I2S_RX_HALF_SAMPLE_BITS				I2S_RX_BITS_MOD				(reserved)				I2S_RX_TDM_WS_WIDTH			
31	27	26	19	18	14	13	9	8	0										
0xf		0xf			0xf		0	0	0	0	0	0x0			Reset				

I2S_RX_TDM_WS_WIDTH Configures the width of I2SI_WS_out (WS default level) at idle level in TDM mode. Width of I2SI_WS_out at idle level in TDM mode = (I2S_RX_TDM_WS_WIDTH[8:0] + 1) x T_BCK. (R/W)

I2S_RX_BITS_MOD Configures the valid data bit length of I2S RX channel.

- 7: All the valid channel data is in 8-bit mode
 - 15: All the valid channel data is in 16-bit mode
 - 23: All the valid channel data is in 24-bit mode
 - 31: All the valid channel data is in 32-bit mode
 - Other values are invalid.
- (R/W)

I2S_RX_HALF_SAMPLE_BITS Configures I2S RX sample bits. BCK cycles in one WS period = I2S_RX_HALF_SAMPLE_BITS x 2. (R/W)

I2S_RX_TDM_CHAN_BITS Configures RX bit number for each channel in TDM mode. Bit number expected = I2S_RX_TDM_CHAN_BITS + 1. (R/W)

Register 27.7. I2S_RX_TDM_CTRL_REG (0x0050)

(reserved)												I2S_RX_TDM_TOT_CHAN_NUM I2S_RX_TDM_CHAN15_EN I2S_RX_TDM_CHAN14_EN I2S_RX_TDM_CHAN13_EN I2S_RX_TDM_CHAN12_EN I2S_RX_TDM_CHAN11_EN I2S_RX_TDM_CHAN10_EN I2S_RX_TDM_CHAN9_EN I2S_RX_TDM_CHAN8_EN I2S_RX_TDM_CHAN7_EN I2S_RX_TDM_CHAN6_EN I2S_RX_TDM_CHAN5_EN I2S_RX_TDM_CHAN4_EN I2S_RX_TDM_CHAN3_EN I2S_RX_TDM_CHAN2_EN I2S_RX_TDM_CHAN0_EN																						
31												20	19	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x0												1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Reset

I2S_RX_TDM_PDM_CHAN n _EN (n : 0-7) Configures whether to enable the valid data input of I2S RX TDM or PDM channel n .

0: Disable. Channel *n* only inputs 0

1: Enable

(R/W)

I2S_RX_TDM_CHAN n _EN (n : 8-15) Configures whether to enable the valid data input of I2S RX TDM channel n .

0: Disable. Channel *n* only inputs 0

1: Enable

(R/W)

I2S_RX_TDM_TOT_CHAN_NUM Configures the total number of channels in use in I2S RX TDM mode. Total channel number in use = I2S_RX_TDM_TOT_CHAN_NUM + 1. (R/W)

Register 27.8. I2S_RXEOF_NUM_REG (0x0064)

(reserved)																I2S_RX_EOF_NUM															
31																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x40															

I2S_RX_EOF_NUM Configures the bit length of RX data. Bit length of RX data = (I2S_RX_BITS_MOD + 1) x (I2S_RX_EOF_NUM + 1). Once the received data reaches such bit length, an **AHB_DMA_IN_SUC_EOF_CHn_INT** interrupt is triggered in the configured GDMA RX channel. (R/W)

Register 27.9. I2S_TX_CONF_REG (0x0024)

(reserved)		I2S_SIG_LOOPBACK				I2S_TX_CHAN_MOD				I2S_TX_BCK_DIV_NUM				I2S_TX_PDM_EN		I2S_TX_TDM_EN		I2S_TX_BIT_ORDER		I2S_TX_WS_IDLE_POL		I2S_TX_24_FILL_EN		I2S_TX_LEFT_ALIGN		I2S_TX_BCK_MSB_SHIFT		I2S_TX_PCM_BYPASS		I2S_TX_PCM_CONF		I2S_TX_MONO_FST_VLD		I2S_TX_UPDATE		I2S_TX_BIG_ENDIAN		I2S_TX_MONO		I2S_TX_CHAN_EQUAL		I2S_TX_STOP_EN		I2S_TX_SLAVE_MOD		I2S_TX_FIFO_RESET																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	6	0	0	0	0	0	0	0	1	1	1	1	0x0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2S_TX_RESET Configures whether to reset TX unit.

0: No effect

1: Reset

(WT)

I2S_TX_FIFO_RESET Configures whether to reset TX FIFO.

0: No effect

1: Reset

(WT)

I2S_TX_START Configures whether to start transmitting data.

0: No effect

1: Start

(R/W/SC)

I2S_TX_SLAVE_MOD Configures whether to enable slave TX mode.

0: Enable master mode

1: Enable slave mode

(R/W)

I2S_TX_STOP_EN Configures whether to stop outputting the BCK signal and the WS signal when TX FIFO is empty.

0: No effect

1: Stop

(R/W)

I2S_TX_CHAN_EQUAL Configures whether to equalize left channel data and right channel data in I2S TX mono mode or TDM mode.

0: The I2S_SINGLE_DATA is invalid channel data in I2S TX mono mode or TDM mode

1: The left channel data is equal to right channel data in I2S TX mono mode or TDM mode

(R/W)

I2S_TX_MONO Configures whether to enable TX unit in mono mode.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 27.9. I2S_TX_CONF_REG (0x0024)

Continued from the previous page...

I2S_TX_BIG_ENDIAN Configures I2S TX byte endian.

0: Low address with low address value

1: Low address value to high address

(R/W)

I2S_TX_UPDATE Configures whether to update I2S TX registers from APB clock domain to I2S TX clock domain. This bit will be cleared by hardware after update register done. This bit will be cleared by hardware after update register done.

0: No effect

1: Update

(R/W/SC)

I2S_TX_MONO_FST_VLD Configures the valid data channel in I2S TX mono mode.

0: The second channel data valid

1: The first channel data valid

(R/W)

I2S_TX_PCM_CONF Configures the I2S TX compress/decompress mode.

0 (atol): A-law decompress

1 (ltoa): A-law compress

2 (utol): μ -law decompress

3 (ltou): μ -law compress

(R/W)

I2S_TX_PCM_BYPASS Configures whether to bypass Compress/Decompress units for transmitted data.

0: No effect

1: Bypass

(R/W)

I2S_TX_MSB_SHIFT Configures the timing between the WS signal and the MSB of data.

0: Align at the rising edge

1: WS signal changes one BCK clock earlier

(R/W)

I2S_TX_BCK_NO_DLY Configures the source of the BCK rising and falling edges in master mode.

0: Rising and falling edges are constructed based on the BCK input from I2SO_BCK_in

1: Rising and falling edges are constructed by dividing the clock of I2S TX

(R/W)

I2S_TX_LEFT_ALIGN Configures I2S TX alignment mode.

0: Right alignment mode

1: Left alignment mode

(R/W)

Continued on the next page...

Register 27.9. I2S_TX_CONF_REG (0x0024)

Continued from the previous page...

I2S_TX_24_FILL_EN Configures the bit number that the 24 channel bits are stored to.

0: Store 24-bit channel data to 24 bits

1: Store 24-bit channel data to 32 bits (Extra bits are filled with zeros)

(R/W)

I2S_TX_WS_IDLE_POL Configures the relationship between WS and which channel data to transmit.

0: WS remains low when transmitting left channel data and high when transmitting right channel data

1: WS remains high when transmitting left channel data and low when transmitting right channel data

(R/W)

I2S_TX_BIT_ORDER Configures whether to reverse the bit order of valid data to be sent by the I2S TX.

0: Not reverse

1: Reverse

(R/W)

I2S_TX_TDM_EN Configures whether to enable I2S TDM TX mode.

0: Disable

1: Enable

(R/W)

I2S_TX_PDM_EN Configures whether to enable I2S PDM TX mode.

0: Disable

1: Enable

(R/W)

I2S_TX_BCK_DIV_NUM Configures the divider of BCK in TX mode. Note this divider must not be configured to 1. (R/W)

I2S_TX_CHAN_MOD Configures I2S TX channel mode. For more information, see Table [27.9-4](#).

(R/W)

I2S_SIG_LOOPBACK Configures whether to enable TX unit and RX unit sharing the same WS and BCK signals.

0: Disable

1: Enable

(R/W)

Register 27.10. I2S_TX_CONF1_REG (0x002C)

<i>I2S_TX_TDM_CHAN_BITS</i>					<i>I2S_TX_HALF_SAMPLE_BITS</i>					<i>I2S_TX_BITS_MOD</i>					<i>(reserved)</i>					<i>I2S_TX_TDM_WS_WIDTH</i>						
31					27	26				19	18				14	13				9	8					0
0xf					0xf					0xf					0 0 0 0 0					0x0					Reset	

I2S_TX_TDM_WS_WIDTH Configures the width of I2SO_WS_out (WS default level) at idle level in TDM mode. The width of I2SO_WS_out at idle level in TDM mode = (I2S_TX_TDM_WS_WIDTH[8:0] + 1) x T_BCK. (R/W)

I2S_TX_BITS_MOD Configures the valid data bit length of I2S TX channel.

- 7: All the valid channel data is in 8-bit mode
- 15: All the valid channel data is in 16-bit mode
- 23: All the valid channel data is in 24-bit mode
- 31: All the valid channel data is in 32-bit mode
- Other values are invalid.
- (R/W)

I2S_TX_HALF_SAMPLE_BITS Configures I2S TX sample bits. BCK cycles in one WS period = I2S_TX_HALF_SAMPLE_BITS x 2. (R/W)

I2S_TX_TDM_CHAN_BITS Configures TX bit number for each channel in TDM mode. Bit number expected = I2S_TX_TDM_CHAN_BITS + 1. (R/W)

Register 27.11. I2S_TX_PCM2PDM_CONF_REG (0x0044)

(reserved)						I2S_PCM2PDM_CONV_EN						I2S_TX_PDM_DAC_MODE_EN						I2S_TX_PDM_DAC_2OUT_EN						(reserved)						I2S_TX_PDM_SINC_OSR2						(reserved)					
31	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	(reserved)						5	4	(reserved)		1	0														
0	0	0	0	0	0	0	0	0	1	0	0x1	0x1	0x1	0x1	0x0						0x2		0		Reset																

Reset

I2S_TX_PDM_SINC_OSR2 Configures I2S TX PDM OSR value. (R/W)**I2S_TX_PDM_DAC_2OUT_EN** Configures whether to enable I2S TX PDM DAC mode.

0: Enable 1-line DAC output mode

1: Enable 2-line DAC output mode

Only valid when [I2S_TX_PDM_DAC_MODE_EN](#) is set.

(R/W)

I2S_TX_PDM_DAC_MODE_EN Configures whether to enable 1-line PDM output mode or DAC output mode.

0: Enable 1-line PDM output mode

1: Enable DAC output mode

(R/W)

I2S_PCM2PDM_CONV_EN Configures whether to enable the I2S TX PCM-to-PDM converter.

0: Disable

1: Enable

(R/W)

Register 27.12. I2S_TX_PCM2PDM_CONF1_REG (0x0048)

(reserved)							(reserved)		(reserved)		I2S_TX_PDM_FS										(reserved)										
31						26	25	23	22	20	19						10	9											0		
0	0	0	0	0	0	0	7		7		480										960										Reset

Reset

I2S_TX_PDM_FS Configures I2S PDM TX upsampling parameter. (R/W)

Register 27.13. I2S_TX_TDM_CTRL_REG (0x0054)

[illegible]

I2S_TX_TDM_CHAN n _EN (n : 0-15) Configures whether to enable the valid data output of I2S TX TDM channel n .

0: Channel TX data is controlled by `I2S_TX_CHAN_EQUAL` and `I2S_SINGLE_DATA`. See Section 27.9.2.1

1: Enable

(R/W)

I2S_TX_TDM_TOT_CHAN_NUM Configures the total number of channels in use in I2S TX TDM mode.
Total channel number in use = I2S_TX_TDM_TOT_CHAN_NUM + 1. (R/W)

I2S_TX_TDM_SKIP_MSK_EN Configures the data to be sent in GDMA TX buffer.

Q: Data stored in GDMA TX buffer is used by enabled channels and will not be read by channels that are not enabled

1: Data stored in GDMA TX buffer is read by all channels and will be skipped by channels that are not enabled

(R/W)

Register 27.14. I2S_RX_TIMING_REG (0x0058)

(reserved)		I2S_RX_BCK_IN_DM		(reserved)		I2S_RX_WS_IN_DM		(reserved)		I2S_RX_BCK_OUT_DM		(reserved)		I2S_RX_WS_OUT_DM		(reserved)		I2S_RX_SD3_IN_DM		(reserved)		I2S_RX_SD2_IN_DM		(reserved)		I2S_RX_SD1_IN_DM		(reserved)		I2S_RX_SD_IN_DM	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0	0x0	0	0

Reset

I2S_RX_SD_IN_DM Configures the delay mode of I2S RX SD input signal.

0: Bypass

1: Delay by rising edge

2: Delay by falling edge

3: Invalid

(R/W)

I2S_RX_SD1_IN_DM Configures the delay mode of I2S RX SD1 input signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_SD2_IN_DM Configures the delay mode of I2S RX SD2 input signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_SD3_IN_DM Configures the delay mode of I2S RX SD3 input signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_WS_OUT_DM Configures the delay mode of I2S RX WS output signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_BCK_OUT_DM Configures the delay mode of I2S RX BCK output signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_WS_IN_DM Configures the delay mode of I2S RX WS input signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

I2S_RX_BCK_IN_DM Configures the delay mode of I2S RX BCK input signal. For detailed configuration values, please refer to [I2S_RX_SD_IN_DM](#). (R/W)

Register 27.15. I2S_TX_TIMING_REG (0x005C)

(reserved)		I2S_TX_BCK_IN_DM		(reserved)		I2S_TX_WS_IN_DM		(reserved)		I2S_TX_BCK_OUT_DM		(reserved)		I2S_TX_WS_OUT_DM		(reserved)										I2S_TX_SD1_OUT_DM		(reserved)		I2S_TX_SD_OUT_DM																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	6										5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
0	0	0x0		0	0	0x0		0	0	0x0		0	0	0x0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

I2S_TX_SD_OUT_DM Configures the delay mode of I2S TX SD output signal.

- 0: Bypass
 - 1: Delay by rising edge
 - 2: Delay by falling edge
 - 3: Invalid
- (R/W)

I2S_TX_SD1_OUT_DM Configures the delay mode of I2S TX SD1 output signal. For detailed configuration values, please refer to [I2S_TX_SD_OUT_DM](#). (R/W)

I2S_TX_WS_OUT_DM Configures the delay mode of I2S TX WS output signal. For detailed configuration values, please refer to [I2S_TX_SD_OUT_DM](#). (R/W)

I2S_TX_BCK_OUT_DM Configures the delay mode of I2S TX BCK output signal. For detailed configuration values, please refer to [I2S_TX_SD_OUT_DM](#). (R/W)

I2S_TX_WS_IN_DM Configures the delay mode of I2S TX WS input signal. For detailed configuration values, please refer to [I2S_TX_SD_OUT_DM](#). (R/W)

I2S_TX_BCK_IN_DM Configures the delay mode of I2S TX BCK input signal. For detailed configuration values, please refer to [I2S_TX_SD_OUT_DM](#). (R/W)

Register 27.16. I2S_LC_HUNG_CONF_REG (0x0060)

(reserved)																				I2S_LC_FIFO_TIMEOUT_ENA				I2S_LC_FIFO_TIMEOUT_SHIFT				I2S_LC_FIFO_TIMEOUT			
31																	12	11	10	8	7					0					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0x10				Reset							

I2S_LC_FIFO_TIMEOUT Configures the FIFO timeout threshold. The FIFO hung counter is incremented by one each time I2S is in an error state and the tick counter reaches its threshold. [I2S_TX_HUNG_INT](#) or [I2S_RX_HUNG_INT](#) interrupt will be triggered when FIFO hung counter is equal to the FIFO timeout threshold. (R/W)

I2S_LC_FIFO_TIMEOUT_SHIFT Configures the tick counter threshold. The tick counter is incremented by one each time I2S is in an error state and it is on the rising edge in each system clock (APB). The tick counter is reset when counter value $\geq 88000/2^{I2S\_LC\_FIFO\_TIMEOUT\_SHIFT}$. (R/W)

I2S_LC_FIFO_TIMEOUT_ENA Configures whether to enable FIFO timeout.
 0: Disable
 1: Enable
 (R/W)

Register 27.17. I2S_CONF_SIGLE_DATA_REG (0x0068)

I2S_SINGLE_DATA																															
31																															0
0																															Reset

I2S_SINGLE_DATA Configures constant channel data to be sent out. (R/W)

Register 27.18. I2S_STATE_REG (0x006C)

(reserved)																															I2S_TX_IDLE			
31																															1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Reset

I2S_TX_IDLE Represents the TX unit state.

0: I2S TX unit is working

1: I2S TX unit is in idle state

(RO)

Register 27.19. I2S_ETM_CONF_REG (0x0070)

(reserved)				I2S_ETM_RX_RECEIVE_WORD_NUM										I2S_ETM_TX_SEND_WORD_NUM												
31	28	27										14	13													0
0	0	0	0	0x40										0x40												Reset

I2S_ETM_TX_SEND_WORD_NUM Configures the threshold of triggering ETM [I2S_TX_X_WORDS_SENT](#) event. When transmitting word number of I2S_ETM_TX_SEND_WORD_NUM [9:0], I2S will trigger the corresponding ETM event. (R/W)

I2S_ETM_RX_RECEIVE_WORD_NUM Configures the threshold of triggering ETM [I2S_RX_X_WORDS_RECEIVED](#) event. When receiving word number of I2S_ETM_RX_RECEIVE_WORD_NUM [9:0], I2S will trigger the corresponding ETM event. (R/W)

Register 27.20. I2S_FIFO_CNT_REG (0x0074)

31		30		0	
0		0x000000		Reset	

I2S_TX_FIFO_CNT Configures the TX FIFO counter value. (RO)

I2S_TX_FIFO_CNT_RST Configure whether to reset the TX FIFO counter.

0: No effect

1: Reset

(WT)

Register 27.21. I2S_BCK_CNT_REG (0x0078)

31		30		0	
0		0x000000		Reset	

I2S_TX_BCK_CNT Configures the TX BCK counter value. (RO)

I2S_TX_BCK_CNT_RST Configures whether to reset TC BCK counter.

0: No effect

1: Reset

(WT)

Register 27.22. I2S_CLK_GATE_REG (0x007C)

31		1		0	
0		0		0	

I2S_CLK_EN Configures whether to enable clock gate.

0: Disable

1: Enable

(R/W)

Register 27.23. I2S_DATE_REG (0x0080)

(reserved)				I2S_DATE																									
31		28	27																										0
0	0	0	0	0x2307190																									

Reset

I2S_DATE Version control register. (R/W)

Chapter 28

USB Serial/JTAG Controller

ESP32-C5 contains a USB Serial/JTAG Controller. This unit can be used to program the SoC's flash, read program output, as well as attach a debugger to the running program. All of these are possible for any computer with a USB host (hereafter referred to as 'host') without any active external components.

28.1 Overview

While programming and debugging an ESP32-C5 project using the UART and JTAG functionality is certainly possible, it has a few downsides. First of all, both UART and JTAG take up IO pins and as such, fewer pins are left usable for controlling external signals in software. Additionally, an external chip or adapter is needed for both UART and JTAG to interface with a host computer, which means it will be necessary to integrate these two functionalities in the form of external chips or debugging adapters.

In order to alleviate these issues, ESP32-C5 provides a USB Serial/JTAG Controller, which integrates the functionality of both a USB-to-serial converter as well as a USB-to-JTAG adapter. As this device directly interfaces with an external USB host using only the two data lines required by USB 2.0, only two pins are required to be dedicated to this functionality for debugging ESP32-C5.

28.2 Feature List

The USB Serial/JTAG controller has the following features:

- USB Full-speed device; Hardwired for CDC-ACM (Communication Device Class - Abstract Control Model) and JTAG adapter functionality
- CDC-ACM:
 - Integrates CDC-ACM adherent serial port emulation (plug-and-play on most modern OSes)
 - Supports host controllable chip reset and entry into download mode
- JTAG adapter functionality:
 - Allows fast communication with CPU debugging core using a compact representation of JTAG instructions
- A control endpoint, a dummy interrupt endpoint, two bulk input endpoints, and two bulk output endpoints; Up to 64-byte data payload size
- Internal PHY: very few or no external components needed to connect to a host computer

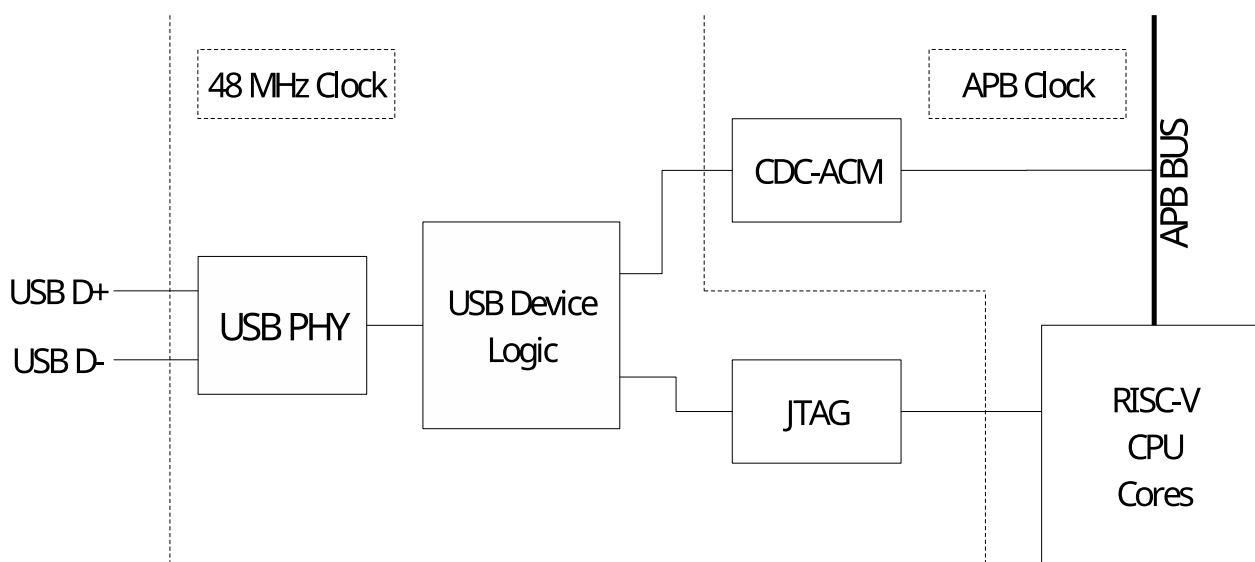


Figure 28.2-1. USB Serial/JTAG High Level Diagram

As shown in Figure 28.2-1, the USB Serial/JTAG controller consists of a USB PHY, a USB device interface, a JTAG command processor, a response capture unit, and the CDC-ACM registers. The PHY and device interface are clocked from a 48 MHz clock derived from the baseband PLL (BBPLL); the software-accessible side of the CDC-ACM block is clocked from APB_CLK. The JTAG command processor is connected to the JTAG debugging unit of the main processor; the CDC-ACM registers are connected to the APB bus and as such can be read from and written to by software running on the main CPU.

Note that while the USB Serial/JTAG device supports USB 2.0 standard, it only supports Full-speed (12 Mbps) mode but not other modes that the USB 2.0 standard introduced, e.g., the High-speed (480 Mbps) mode.

Figure 28.2-2 shows the internal details of the USB Serial/JTAG controller on the USB side. The USB Serial/JTAG controller consists of a USB 2.0 Full-speed device. It contains a control endpoint, a dummy interrupt endpoint, two bulk input endpoints, and two bulk output endpoints. Together, these form a USB composite device, which consists of a CDC-ACM USB class device as well as a vendor-specific device implementing the JTAG interface. On the SoC side, the JTAG interface is directly connected to the RISC-V CPU's debugging interface, allowing debugging of programs running on that core. Meanwhile, the CDC-ACM device is exposed as a set of registers, allowing a program on the CPU to read and write from it. Additionally, the ROM startup code of the SoC contains code that allows the user to reprogram attached flash memory using this interface.

Note:

To get specific information about what endpoint numbers are used for what functionality, please parse the relevant USB descriptors as returned by the device. This guarantees compatibility with other devices with the same functionality but a different implementation.

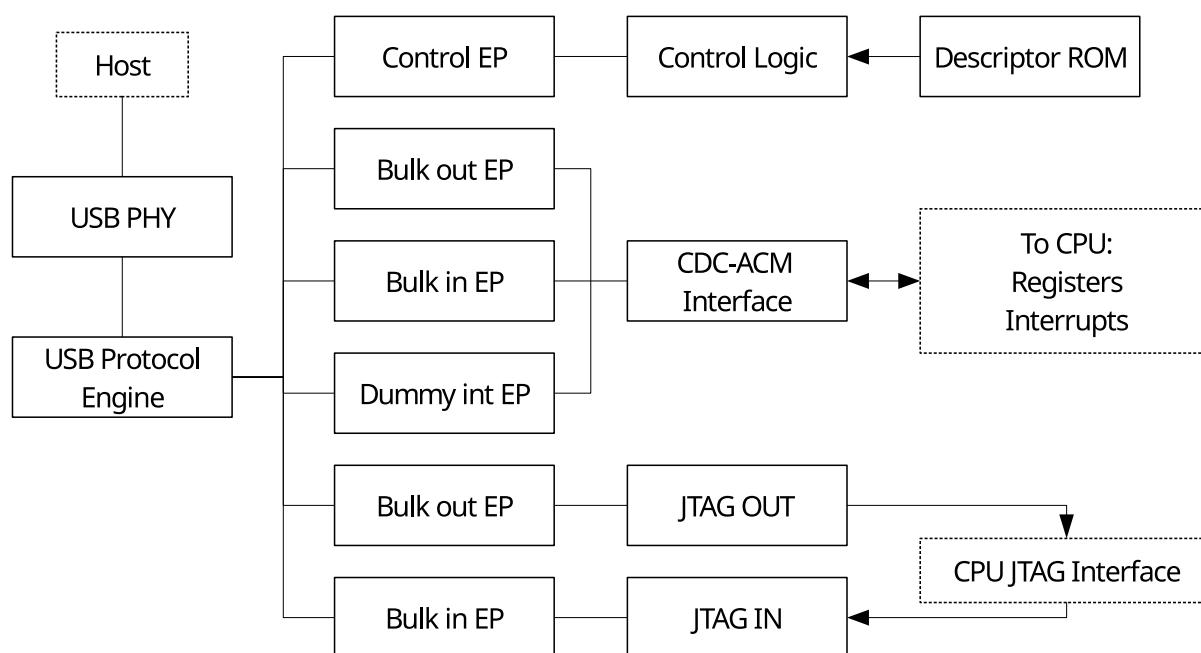


Figure 28.2-2. USB Serial/JTAG Block Diagram

28.3 Functional Description

The USB Serial/JTAG controller interfaces with a USB host processor on one side, and with the CPU debugging hardware as well as the software that communicates through the CDC-ACM port on the other side.

28.3.1 CDC-ACM USB Interface Functional Description

The CDC-ACM interface adheres to the standard USB CDC-ACM class for serial port emulation. It contains a dummy interrupt endpoint (which will never send any events, as they are not implemented nor needed) and a Bulk IN as well as a Bulk OUT endpoint for the host's received and sent serial data respectively. These endpoints can handle 64-bytes packet at a time, allowing high throughput. As CDC-ACM is a standard USB device class, a host generally can function without any special installation procedures. That is to say, when the USB debugging device is properly connected to a host, the operating system should show a new serial port moments later.

The CDC-ACM interface accepts the following standard CDC-ACM control requests:

Table 28.3-1. Standard CDC-ACM Control Requests

Command	Action
SEND_BREAK	Accepted but ignored (dummy)
SET_LINE_CODING	Accepted, value sent is readable in software
GET_LINE_CODING	By default, returns 9600 baud, no parity, 8 data bits, 1 stop bit (Can be changed through software)
SET_CONTROL_LINE_STATE	Set the state of the RTS/DTR lines. See Table 28.3-2

Aside from general-purpose communication, the CDC-ACM interface can also be used to reset ESP32-C5 and optionally make it enter download mode to flash new firmware. This can be realized by setting the RTS and DTR lines on the virtual serial port.

Table 28.3-2. CDC-ACM Settings with RTS and DTR

RTS	DTR	Action
0	0	Clear download mode flag
0	1	Set download mode flag
1	0	Reset ESP32-C5
1	1	No action

Note that if the download mode flag is set when ESP32-C5 is reset, ESP32-C5 will reboot into download mode. When this flag is cleared and the chip is reset, ESP32-C5 will boot from flash. For specific sequences, please refer to Section 28.5. All these functions can also be disabled by programming various eFuses. Please refer to Chapter 2 *eFuse Controller (eFuse) [to be added later]* for more details.

28.3.2 CDC-ACM Firmware Interface Functional Description

The CPU can interact with the USB Serial/JTAG controller as the module is connected to the internal APB bus of ESP32-C5. This is mainly used to read and write data from and to the virtual serial port on the attached host.

USB CDC-ACM serial data is sent to and received from the host in packets of 0 to 64 bytes in size. When enough CDC-ACM data has accumulated in the host, the host sends a packet to the CDC-ACM receive endpoint, and the USB Serial/JTAG controller accepts this packet if it has a free buffer. Conversely, the host checks periodically if the USB Serial/JTAG controller has a packet ready to be sent to the host, and if so, receives this packet.

Firmware can get notified of new data from the host in one of the following two ways. First of all, the [USB_SERIAL_JTAG_SERIAL_OUT_EP_DATA_AVAIL](#) bit will remain set as long as there still is unread host data in the buffer. Secondly, the availability of data will trigger the [USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#) interrupt. When data is available, it can be read by firmware through repeatedly reading bytes from [USB_SERIAL_JTAG_EP1_REG](#). The amount of bytes to read can be determined by checking the [USB_SERIAL_JTAG_SERIAL_OUT_EP_DATA_AVAIL](#) bit after reading each byte to see if there is more data to read. After all data is read, the USB device is automatically readied to receive a new data packet from the host.

Generally, after sending data, the attached host will block until the data is actually read by firmware in one of the described methods. However, it is possible that a given firmware program does not read data at all. This situation generally is not expected by host programs which are written to talk to a discrete USB-serial converter; the program e.g. stops executing until the data is read. The ESP32-C5 USB-serial-JTAG adapter can simulate the behavior of a discrete USB-serial-JTAG converter: by writing a timeout to [USB_SERIAL_JTAG_SERIAL_TIMEOUT_MAX](#) and writing an 1 to [USB_SERIAL_JTAG_SERIAL_TIMEOUT_EN](#), any unread data will be automatically flushed after the given timeout.

When the firmware has data to send, it can put the data in the send buffer and trigger a flush to allow the host to receive the data in a USB packet. In order to do so, there needs to be space available in the send buffer. Firmware can check this by reading [USB_REG_SERIAL_IN_EP_DATA_FREE](#). A 1 in this register field indicates

there is still free room in the buffer, and firmware can fill the buffer by writing bytes to the [USB_SERIAL_JTAG_EP1_REG](#) register. Writing the buffer does not immediately trigger sending data to the host until the buffer is flushed. After the flush, the entire buffer will be ready to be received by the USB host at once. A flush can be triggered in two ways: 1) after the 64th byte is written to the buffer, the USB hardware will automatically flush the buffer to the host; or 2) firmware can trigger a flush by writing 1 to [USB_REG_SERIAL_WR_DONE](#).

Regardless of how a flush is triggered, the send buffer will be unavailable for firmware to write into until it has been fully read by the host. As soon as the send buffer has been fully read, the [USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#) interrupt will be triggered, indicating that the send buffer can receive another 64 bytes.

It is possible to handle some out-of-band serial requests in software, specifically, the host setting DTR and RTS and changing the line state. If the CDC-ACM interface receives a SET_LINE_CODING request, the peripheral can be configured to trigger a [USB_SERIAL_JTAG_SET_LINE_CODE_INT](#) interrupt, at which point the line coding can be read from the [USB_SERIAL_JTAG_SET_LINE_CODE_WO_REG](#) register. Similarly, SET_CONTROL_LINE_STATE requests will trigger [USB_SERIAL_JTAG_RTS_CHG_INT](#) and [USB_SERIAL_JTAG_DTR_CHG_INT](#) interrupts if they change the state of these lines. Software can then read the specific state through the [USB_SERIAL_JTAG_RTS](#) and [USB_SERIAL_JTAG_DTR](#) bits. Note that as described earlier, certain RTS/DTR sequences lead to hardware reset of ESP32-C5. Software can disable hardware recognition of these DTR/RTS sequences by setting the [USB_SERIAL_JTAG_USB_UART_CHIP_RST_DIS](#) bit, allowing software to interpret these signals freely.

Finally, the host can read the current line state using GET_LINE_CODING. This event sends back the data in the [USB_SERIAL_JTAG_GET_LINE_CODE_WO_REG](#) register and triggers a [USB_SERIAL_JTAG_GET_LINE_CODE_INT](#) interrupt.

28.3.3 USB-to-JTAG Interface: JTAG Command Processor

The USB-to-JTAG interface uses a vendor-specific class for its implementation. It consists of two endpoints, one to receive commands and another to send responses. Additionally, some less time-sensitive commands can be given as control requests.

Commands from the host to the JTAG interface are interpreted by the JTAG command processor. Internally, the JTAG command processor implements a full four-wire JTAG bus, consisting of the TCK, TMS and TDI output lines to the RISC-V CPU, as well as the TDO line signalling back from the CPU to the JTAG response capture unit. These signals adhere to the IEEE 1149.1 JTAG standards. Additionally, there is an SRST line to reset ESP32-C5.

Optionally, software can set [USB_SERIAL_JTAG_USB_JTAG_BRIDGE_EN](#) in order to redirect these signals to the GPIO matrix instead, where they can be routed to IO pads on ESP32-C5. This also allows external devices to be debugged via the USB Serial/JTAG peripheral.

The JTAG command processor parses each received nibble (4-bit value) as a command. As USB data is received in 8-bit bytes, this means each byte contains two commands. The USB command processor will execute high-nibble first and low-nibble second. The commands are used to control the TCK, TMS, TDI, and SRST lines of the internal JTAG bus, as well as to signal the JTAG response capture unit the state of the TDO line (which is driven by the CPU debugging logic) that needs to be captured.

In the internal JTAG bus, TCK, TMS, TDI, and TDO are connected directly to the JTAG debugging logic of the

RISC-V CPU. SRST is connected to the reset logic of the digital circuitry in ESP32-C5 and a high level on this line will cause a digital system reset. Note that the USB Serial/JTAG controller itself is not affected by SRST.

A nibble can contain the following commands:

Table 28.3-3. Commands of a Nibble

bit	3	2	1	0
CMD_CLK	0	cap	tms	tdi
CMD_RST	1	0	0	srst
CMD_FLUSH	1	0	1	0
CMD_RSV	1	0	1	1
CMD_REP	1	1	R1	R0

- CMD_CLK will set the TDI and TMS as the indicated values and emit one clock pulse on TCK. If the CAP bit is 1, it will instruct the JTAG response capture unit to capture the state of the TDO line. This instruction forms the basis of JTAG communication.
- CMD_RST will set the state of the SRST line as the indicated value. This can be used to reset ESP32-C5.
- CMD_FLUSH will instruct the JTAG response capture unit to flush the buffer of all bits it collected so the host is able to read them. Note that in some cases, a JTAG transaction will end in an odd number of commands and as such an odd number of nibbles. In this case, it is allowed to repeat the CMD_FLUSH command to get an even number of nibbles fitting an integer number of bytes.
- CMD_RSV is reserved in the current implementation. This command will be ignored when received by ESP32-C5.
- CMD_REP repeats the last (non-CMD_REP) command for a certain number of times. The purpose is to compress command streams which repeat the CMD_CLK instruction for multiple times. A command such as CMD_CLK can be followed by multiple CMD_REP commands. The number of repetitions done by one CMD_REP can be expressed as $repetition\_count = (R1 \times 2 + R0) \times (4^{cmd\_rep\_count})$, where *cmd_rep_count* indicates the number of the CMD_REP instruction that went directly before it. Note that the CMD_REP command is only intended to repeat a CMD_CLK command. Specifically, using it on a CMD_FLUSH command may lead to an unresponsive USB device, and a USB reset will be required to recover it.

28.3.4 USB-to-JTAG Interface: CMD_REP Usage Example

Here is a list of commands as an illustration of the usage of CMD_REP. Note that each command is a nibble, and in this example, the bitwise command stream would be 0x0D 0x5E 0xCF.

1. 0x0 (CMD_CLK: cap=0, tdi=0, tms=0)
2. 0xD (CMD_REP: R1=0, R0=1)
3. 0x5 (CMD_CLK: cap=1, tdi=0, tms=1)
4. 0xE (CMD_REP: R1=1, R0=0)
5. 0xC (CMD_REP: R1=0, R0=0)

6. 0xF (CMD_REP: R1=1, R0=1)

The following shows what happens at every step:

1. TCK is clocked with the TDI and TMS lines set to 0. No data is captured.
2. TCK is clocked another $(0 \times 2 + 1) \times (4^0) = 1$ time with the same settings as step 1.
3. TCK is clocked with the TDI line set to 0 and TMS set to 1. Data on the TDO line is captured.
4. TCK is clocked another $(1 \times 2 + 0) \times (4^0) = 2$ times with the same settings as step 3.
5. Nothing happens: $(0 \times 2 + 0) \times (4^1) = 0$. Note that this increases cmd_rep_count in the next step.
6. TCK is clocked another $(1 \times 2 + 1) \times (4^2) = 48$ times with the same settings as step 3.

In other words, this example stream has the same net effect as that of executing command 1 twice, then repeating command 3 for 51 times.

28.3.5 USB-to-JTAG Interface: Response Capture Unit

The response capture unit reads the TDO line of the internal JTAG bus and captures its value when the command parser executes a CMD_CLK with cap=1. It puts this bit into an internal shift register, and writes a byte into the USB buffer when 8 bits have been collected. Of these 8 bits, the least significant one is the one that is read from TDO the earliest.

As soon as either 64 bytes (512 bits) have been collected or a CMD_FLUSH command is executed, the response capture unit will make the buffer available for the host to receive. Note that the interface to the USB logic is double-buffered. Therefore, as long as the USB throughput is sufficient, the response capture unit can always receive more data. That is to say, while one of the buffers is waiting to be sent to the host, the other can receive more data. When the host has received data from its buffer and the response capture unit flushes its buffer, the two buffers exchange position.

This also means that a command stream can cause at most 128 bytes of capture data generated (less if there are flush commands in the stream) without the host acting to receive the generated data. If more data is generated anyway, the command stream will pause and the device will not accept more commands until the generated capture data is read out.

Note that in general, the logic of the response capture unit tries not to send zero-byte responses. For instance, sending a series of CMD_FLUSH commands will not cause a series of 0-byte USB responses to be sent. However, in the current implementation, some zero-0 responses may be generated in extraordinary circumstances. It is recommended to ignore these responses.

28.3.6 USB-to-JTAG Interface: Control Transfer Requests

Aside from the command processor and the response capture unit, the USB-to-JTAG interface also understands some control requests, as documented in the table below:

Table 28.3-4. USB-to-JTAG Control Requests

bmRequestType	bRequest	wValue	wIndex	wLength	Data
01000000b	0 (VEND_JTAG_SETDIV)	[divider]	interface	0	None
01000000b	1 (VEND_JTAG_SETIO)	[iobits]	interface	0	None
11000000b	2 (VEND_JTAG_GETTDO)	0	interface	1	[iostate]
10000000b	6 (GET_DESCRIPTOR)	0x2000	0	256	[jtag cap desc]

- VEND_JTAG_SETDIV sets the divider used. This directly affects the duration of a TCK clock pulse. The TCK clock pulses are derived from a base clock of 48 MHz, which is divided down using an internal divider. This control request allows the host to set this divider. Note that on startup, the divider is set to 2, which means the TCK clock rate will generally be 24 MHz.
- VEND_JTAG_SETIO can bypass the JTAG command processor to set the internal TDI, TDO, TMS, and SRST lines to given values. These values are encoded in the wValue field in the format of 11'b0, srst, trst, tck, tms, tdi.
- VEND_JTAG_GETTDO can bypass the JTAG response capture unit to read the internal TDO signal directly. This request returns one byte of data, of which the least significant bit represents the status of the TDO line.
- GET_DESCRIPTOR is a standard USB request. However, it can also be used with a vendor-specific wValue of 0x2000 to get the JTAG capabilities descriptor. This returns a certain amount of bytes representing the following fixed structure, which describes the capabilities of the USB-to-JTAG adapter (as shown in Table 28.3-5). This structure allows host software to automatically support future revisions of the hardware without the need for an update.

The JTAG capability descriptors of ESP32-C5 are as follows. Note that all 16-bit values are little-endian.

Table 28.3-5. JTAG Capability Descriptors

Byte	Value	Description
0	1	JTAG protocol capability structure version
1	10	Total length of JTAG protocol capabilities
2	1	Type of this struct: 1 for speed capability struct
3	8	Length of this speed capabilities struct
4 ~ 5	4800	JTAG base clock speed in 10 kHz increments. Note that the maximum TCK speed is half of this value
6 ~ 7	1	Minimum divider value settable by the VEND_JTAG_SETDIV request
8 ~ 9	255	Maximum divider value settable by the VEND_JTAG_SETDIV request

28.4 Interrupts

ESP32-C5's USB Serial/JTAG Controller can generate the USB_SERIAL_JTAG_INTR interrupt signal that will be sent to the [Interrupt Matrix](#). There are several internal interrupt sources from the USB Serial/JTAG Controller that can generate this interrupt signal.

- USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT: triggered when flush cmd is received for the JTAG bulk out endpoint.

- `USB_SERIAL_JTAG_SOF_INT`: triggered when SOF frame is received.
- `USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT`: triggered when Serial Port OUT Endpoint receives one packet.
- `USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT`: triggered when Serial Port IN Endpoint is empty.
- `USB_SERIAL_JTAG_PID_ERR_INT`: triggered when PID error is detected.
- `USB_SERIAL_JTAG_CRC5_ERR_INT`: triggered when CRC5 error is detected.
- `USB_SERIAL_JTAG_CRC16_ERR_INT`: triggered when CRC16 error is detected.
- `USB_SERIAL_JTAG_STUFF_ERR_INT`: triggered when a bit stuffing error is detected.
- `USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT`: triggered when IN token for IN endpoint 1 is received.
- `USB_SERIAL_JTAG_USB_BUS_RESET_INT`: triggered when USB bus reset is detected.
- `USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT`: triggered when OUT endpoint 1 receives packet with zero payload.
- `USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT`: triggered when OUT endpoint 2 receives packet with zero payload.
- `USB_SERIAL_JTAG_RTS_CHG_INT`: triggered when level of RTS from USB serial channel is changed.
- `USB_SERIAL_JTAG_DTR_CHG_INT`: triggered when level of DTR from USB serial channel is changed.
- `USB_SERIAL_JTAG_GET_LINE_CODE_INT`: triggered when level of GET_LINE_CODING request is received.
- `USB_SERIAL_JTAG_SET_LINE_CODE_INT`: triggered when level of SET_LINE_CODING request is received.

28.5 Programming Procedures

Little setup is needed for using the USB Serial/JTAG device. The USB-to-JTAG hardware itself does not need any setup aside from the standard USB initialization that the host operating system already does. Apart from that, the CDC-ACM emulation on the host side is also plug-and-play.

On the firmware side, very little initialization is needed either. The USB hardware is self-initialized and after boot-up, if a host is connected and listening on the CDC-ACM interface, data can be exchanged as described above without any specific setup except for the situation when the firmware optionally sets up an interrupt service handler.

One thing to note is that there may be situations where either the host is not attached or the CDC-ACM virtual port is not opened. In such cases, the packets that are flushed to the host will never be picked up and the send buffer will never be empty. It is important to detect these situations and implement timeout, as this is the only way to reliably detect whether the port on the host side is closed or not.

Another thing to note is that the USB device is dependent on the BBPLL for the 48 MHz USB PHY clock. If this PLL is disabled, the USB communication will cease to function.

One scenario where this happens is Deep-sleep. The USB Serial/JTAG controller (as well as the attached RISC-V CPU) will be entirely powered down in Deep-sleep mode. If a device needs to be debugged in this

mode, it may be preferable to use an external JTAG debugger and a serial interface instead.

The CDC-ACM interface can also be used to reset the SoC and take it into or out of download mode. Generating the correct sequence of handshake signals can be a bit complicated, since most operating systems only allow setting or resetting DTR and RTS separately, but not in tandem. Additionally, some drivers (e.g., the standard CDC-ACM driver on Windows) do not set DTR until RTS is set and the user needs to explicitly set RTS in order to 'propagate' the DTR value. The recommended procedures are introduced below.

To reset the SoC into download mode:

Table 28.5-1. Reset SoC into Download Mode

Action	Internal state	Note
Clear DTR	RTS=?, DTR=0	Initialize to known values
Clear RTS	RTS=0, DTR=0	-
Set DTR	RTS=0, DTR=1	Set download mode flag
Clear RTS	RTS=0, DTR=1	Propagate DTR
Set RTS	RTS=1, DTR=1	-
Clear DTR	RTS=1, DTR=0	Reset SoC
Set RTS	RTS=1, DTR=0	Propagate DTR
Clear RTS	RTS=0, DTR=0	Clear download flag

To reset the SoC into booting from flash:

Table 28.5-2. Reset SoC into Booting from flash

Action	Internal state	Note
Clear DTR	RTS=?, DTR=0	-
Clear RTS	RTS=0, DTR=0	Clear download flag
Set RTS	RTS=1, DTR=0	Reset SoC
Clear RTS	RTS=0, DTR=0	Exit reset

28.6 Register Summary

The addresses in this section are relative to USB Serial/JTAG controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
USB_SERIAL_JTAG_EP1_REG	FIFO access for the CDC-ACM data IN and OUT endpoints	0x0000	R/W
USB_SERIAL_JTAG_EP1_CONF_REG	Configuration and control registers for the CDC-ACM FIFOs	0x0004	varies
USB_SERIAL_JTAG_CONFO_REG	PHY hardware configuration	0x0018	R/W
USB_SERIAL_JTAG_TEST_REG	Registers used for debugging the PHY	0x001C	varies
USB_SERIAL_JTAG_MISC_CONF_REG	Clock enable control	0x0044	R/W
USB_SERIAL_JTAG_MEM_CONF_REG	Memory power control	0x0048	R/W
USB_SERIAL_JTAG_CHIP_RST_REG	CDC-ACM chip reset control	0x004C	varies
USB_SERIAL_JTAG_GET_LINE_CODE_W0_REG	W0 of GET_LINE_CODING command	0x0058	R/W
USB_SERIAL_JTAG_GET_LINE_CODE_W1_REG	W1 of GET_LINE_CODING command	0x005C	R/W
USB_SERIAL_JTAG_CONFIG_UPDATE_REG	Configuration registers' value update	0x0060	WT
USB_SERIAL_JTAG_SER_AFIFO_CONFIG_REG	Serial AFIFO configure register	0x0064	varies
USB_SERIAL_JTAG_SERIAL_EP_TIMEOUT0_REG	USB UART out endpoint timeout configuration	0x006C	varies
USB_SERIAL_JTAG_SERIAL_EP_TIMEOUT1_REG	USB UART out endpoint timeout configuration	0x0070	R/W
Interrupt Registers			
USB_SERIAL_JTAG_INT_RAW_REG	Interrupt raw status register	0x0008	R/WTC/SS
USB_SERIAL_JTAG_INT_ST_REG	Interrupt status register	0x000C	RO
USB_SERIAL_JTAG_INT_ENA_REG	Interrupt enable status register	0x0010	R/W
USB_SERIAL_JTAG_INT_CLR_REG	Interrupt clear status register	0x0014	WT
Status Registers			
USB_SERIAL_JTAG_JFIFO_ST_REG	JTAG FIFO status and control registers	0x0020	varies
USB_SERIAL_JTAG_FRAM_NUM_REG	Last received SOF frame index register	0x0024	RO
USB_SERIAL_JTAG_IN_EPO_ST_REG	Control IN endpoint status information	0x0028	RO
USB_SERIAL_JTAG_IN_EP1_ST_REG	CDC-ACM IN endpoint status information	0x002C	RO
USB_SERIAL_JTAG_IN_EP2_ST_REG	CDC-ACM interrupt IN endpoint status information	0x0030	RO
USB_SERIAL_JTAG_IN_EP3_ST_REG	JTAG IN endpoint status information	0x0034	RO
USB_SERIAL_JTAG_OUT_EPO_ST_REG	Control OUT endpoint status information	0x0038	RO

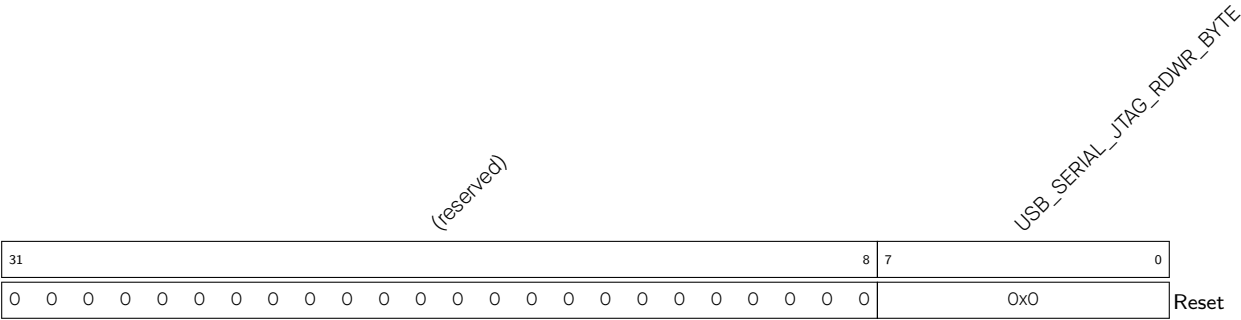
Name	Description	Address	Access
USB_SERIAL_JTAG_OUT_EP1_ST_REG	CDC-ACM OUT endpoint status information	0x003C	RO
USB_SERIAL_JTAG_OUT_EP2_ST_REG	JTAG OUT endpoint status information	0x0040	RO
USB_SERIAL_JTAG_SET_LINE_CODE_W0_REG	W0 of SET_LINE_CODING command	0x0050	RO
USB_SERIAL_JTAG_SET_LINE_CODE_W1_REG	W1 of SET_LINE_CODING command	0x0054	RO
USB_SERIAL_JTAG_BUS_RESET_ST_REG	USB Bus reset status register	0x0068	RO
Version Registers			
USB_SERIAL_JTAG_DATE_REG	Date register	0x0080	R/W

28.7 Registers

The addresses in this section are relative to USB Serial/JTAG controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

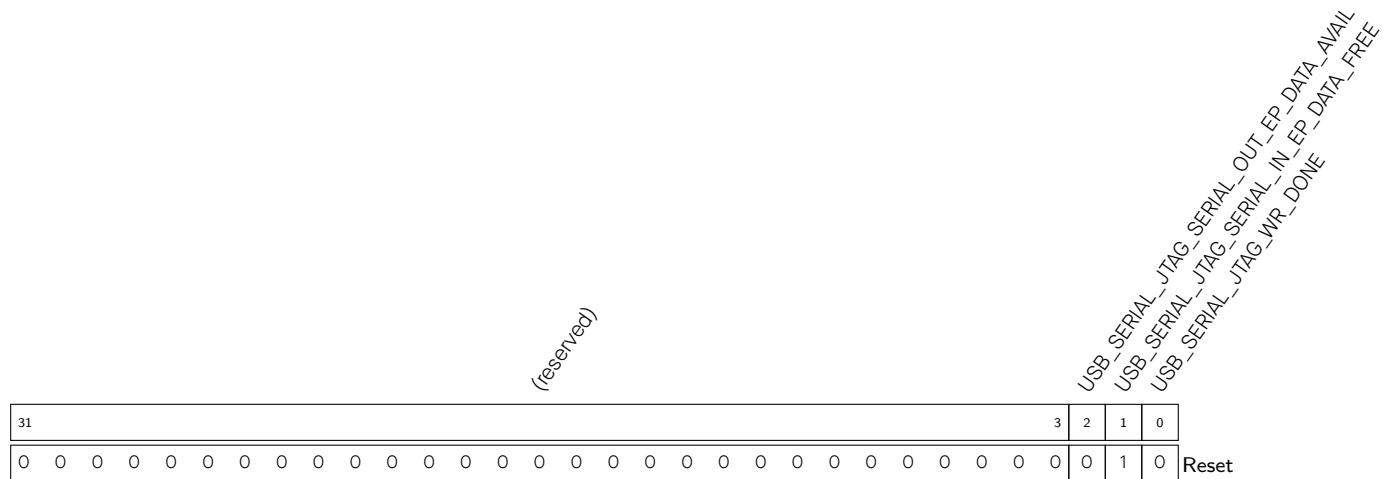
For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 28.1. USB_SERIAL_JTAG_EP1_REG (0x0000)



USB_SERIAL_JTAG_RDWR_BYTE A write to this register pushes the written data into the CDC TX FIFO; a read from this register pops a byte from the CDC RX FIFO and returns it.
When [USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#) is set, users can write data (up to 64 bytes) into CDC-ACM TX FIFO through this register.
When [USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#) is set, users can check how many data is received through [USB_SERIAL_JTAG_OUT_EP1_WR_ADDR](#), then read data from CDC-ACM RX FIFO through this register.
(R/W)

Register 28.2. USB_SERIAL_JTAG_EP1_CONF_REG (0x0004)



USB_SERIAL_JTAG_WR_DONE Configures whether to represent writing byte data to CDC-ACM TX FIFO is done.

0: No effect

1: Represents writing byte data to CDC-ACM TX FIFO is done

This bit then stays 0 until data in CDC-ACM TX FIFO is read by the USB Host.

(WT)

USB_SERIAL_JTAG_SERIAL_IN_EP_DATA_FREE Represents whether CDC-ACM TX FIFO has space available.

0: CDC-ACM TX FIFO is full and no data should be written into it

1: CDC-ACM TX FIFO is not full and data can be written into it

After writing `USB_SERIAL_JTAG_WR_DONE`, this bit will be 0 until data in CDC-ACM TX FIFO is read by USB Host.

(RO)

USB_SERIAL_JTAG_SERIAL_OUT_EP_DATA_AVAIL Represents whether there is data in CDC-ACM RX FIFO.

0: There is no data in CDC-ACM RX FIFO

1: There is data in CDC-ACM RX FIFO

(RO)

Register 28.3. USB_SERIAL_JTAG_CONFO_REG (0x0018)

(reserved)																USB_SERIAL_JTAG_USB_JTAG_BRIDGE_EN USB_SERIAL_JTAG_USB_PAD_ENABLE USB_SERIAL_JTAG_PULLUP_VALUE USB_SERIAL_JTAG_DM_PULLDOWN USB_SERIAL_JTAG_DP_PULLUP USB_SERIAL_JTAG_DP_PULLDOWN USB_SERIAL_JTAG_PAD_PULLUP USB_SERIAL_JTAG_VREF_OVERRIDE USB_SERIAL_JTAG_VREFL USB_SERIAL_JTAG_VREFH USB_SERIAL_JTAG_EXCHG_PINS USB_SERIAL_JTAG_PHY_SEL																																	
31																16																15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0																0																0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	Reset

USB_SERIAL_JTAG_PHY_SEL Configures whether to select internal or external PHY.

0: Internal PHY

1: External PHY

(R/W)

USB_SERIAL_JTAG_EXCHG_PINS_OVERRIDE Configures whether to enable software control USB

D+ D- exchange.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_EXCHG_PINS Configures whether to enable USB D+ D- exchange.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_VREFH Configures single-end input high threshold.

0: 1.76 V

1: 1.84 V

2: 1.92 V

3: 2.00 V

(R/W)

USB_SERIAL_JTAG_VREFL Configures single-end input low threshold.

0: 0.80 V

1: 0.88 V

2: 0.96 V

3: 1.04 V

(R/W)

USB_SERIAL_JTAG_VREF_OVERRIDE Configures whether to enable software control input threshold.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 28.3. USB_SERIAL_JTAG_CONFO_REG (0x0018)

Continued from the previous page...

USB_SERIAL_JTAG_PAD_PULL_OVERRIDE Configures whether to enable software to control USB D+ D- pullup and pulldown.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_DP_PULLUP Configures whether to enable USB D+ pull up when [USB_SERIAL_JTAG_PAD_PULL_OVERRIDE](#) is 1.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_DP_PULLDOWN Configures whether to enable USB D+ pull down when [USB_SERIAL_JTAG_PAD_PULL_OVERRIDE](#) is 1.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_DM_PULLUP Configures whether to enable USB D- pull up when [USB_SERIAL_JTAG_PAD_PULL_OVERRIDE](#) is 1.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_DM_PULLDOWN Configures whether to enable USB D- pull down when [USB_SERIAL_JTAG_PAD_PULL_OVERRIDE](#) is 1.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_PULLUP_VALUE Configures the pull up value when [USB_SERIAL_JTAG_PAD_PULL_OVERRIDE](#) is 1.

0: 2.2 K

1: 1.1 K

(R/W)

USB_SERIAL_JTAG_USB_PAD_ENABLE Configures whether to enable USB pad function.

0: Disable

1: Enable

(R/W)

USB_SERIAL_JTAG_USB_JTAG_BRIDGE_EN Configures whether to disconnect USB_JTAG and internal JTAG.

0: USB_JTAG is connected to the internal JTAG port of CPU

1: USB_JTAG and the internal JTAG are disconnected, MTMS, MTDI, and MTCK are output through GPIO Matrix, and MTDO is input through GPIO Matrix

(R/W)

Register 28.4. USB_SERIAL_JTAG_TEST_REG (0x001C)

(reserved)																										7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	Reset

USB_SERIAL_JTAG_TEST_ENABLE Configures whether to enable the test mode of the USB pad.

0: Resume normal operation

1: Enable the test mode of the USB pad

Enabling the test mode of the USB pad allows the USB pad to be controlled/read using the other bits in this register.

(R/W)

USB_SERIAL_JTAG_TEST_USB_OE Configures whether to enable USB pad output.

0: Set D+ and D- to high impedance

1: Output the values set in `USB_SERIAL_JTAG_TEST_TX_DP` and `USB_SERIAL_JTAG_TEST_TX_DM` on the D+ and D- pins

(R/W)

USB_SERIAL_JTAG_TEST_TX_DP Configures value of USB D+ in test mode when [USB_SERIAL_JTAG_TEST_USB_OE](#) is 1. (R/W)

USB_SERIAL_JTAG_TEST_TX_DM Configures value of USB D- in test mode when [USB_SERIAL_JTAG_TEST_USB_OE](#) is 1. (R/W)

USB_SERIAL_JTAG_TEST_RX_RCV Represents the current logical level of the voltage difference between USB D- and USB D+ pads in test mode.

0: USB D- voltage is higher than USB D+

1: USB D+ voltage is higher than USB D-

(RO)

USB_SERIAL_JTAG_TEST_RX_DP Represents the logical level of the USB D+ pad in test mode.

(RO)

USB_SERIAL_JTAG_TEST_RX_DM Represents the logical level of the USB D- pad in test mode.

(RO)

Register 28.5. USB_SERIAL_JTAG_MISC_CONF_REG (0x0044)

(reserved)																															USB_SERIAL_JTAG_CLK_EN				
31																															1	0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

USB_SERIAL_JTAG_CLK_EN Configures whether to force clock on for register.

0: Support clock only when application writes registers

1: Force clock on for register

(R/W)

Register 28.6. USB_SERIAL_JTAG_MEM_CONF_REG (0x0048)

(reserved)																												USB_SERIAL_JTAG_USB_MEM_CLK_EN		USB_SERIAL_JTAG_USB_MEM_PD	
31																											2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	Reset	

USB_SERIAL_JTAG_USB_MEM_PD Configures whether to power down USB memory.

0: No effect

1: Power down

(R/W)

USB_SERIAL_JTAG_USB_MEM_CLK_EN Configures whether to force clock on for USB memory.

0: No effect

1: Force

(R/W)

Register 28.7. USB_SERIAL_JTAG_CHIP_RST_REG (0x004C)

Diagram of the Reset register (0x00000000). The register is 32 bits wide. Bits 31-3 are reserved. Bit 2 is USB_SERIAL_JTAG_USB_UART_CHIP_RST_DIS. Bit 1 is USB_SERIAL_JTAG_DTR. Bit 0 is USB_SERIAL_JTAG_RTS. The register is labeled 'Reset'.

USB_SERIAL_JTAG_RTS Represents the state of RTS signal as set by the most recent SET_LINE_CODING command. (RO)

USB_SERIAL_JTAG_DTR Represents the state of DTR signal as set by the most recent SET_LINE_CODING command. (RO)

USB_SERIAL_JTAG_USB_UART_CHIP_RST_DIS Configures whether to disable chip reset from USB serial channel.

0: No effect

1: Disable

(R/W)

Register 28.8. USB_SERIAL_JTAG_GET_LINE_CODE_WO_REG (0x0058)

USB_SERIAL_JTAG_GET_DW_DTE_RATE Configures the value of dwDTERate set by software, which is requested by GET_LINE_CODING command. (R/W)

Register 28.9. USB_SERIAL_JTAG_GET_LINE_CODE_W1_REG (0x005C)

(reserved)								USB_SERIAL_JTAG_GET_BCHAR_FORMAT								USB_SERIAL_JTAG_GET_BPARITY_TYPE								USB_SERIAL_JTAG_GET_BDATA_BITS								
3124								2316								158								70								
00000000								0								0								0								Reset

USB_SERIAL_JTAG_GET_BDATA_BITS Configures the value of bDataBits set by software, which is requested by GET_LINE_CODING command. (R/W)

USB_SERIAL_JTAG_GET_BPARITY_TYPE Configures the value of bParityType set by software, which is requested by GET_LINE_CODING command. (R/W)

USB_SERIAL_JTAG_GET_BCHAR_FORMAT Configures the value of bCharFormat set by software, which is requested by GET_LINE_CODING command. (R/W)

Register 28.10. USB_SERIAL_JTAG_CONFIG_UPDATE_REG (0x0060)

(reserved)																															USB_SERIAL_JTAG_CONFIG_UPDATE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																															1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_SERIAL_JTAG_CONFIG_UPDATE Configures whether to update the value of configuration registers from APB clock domain to 48 MHz clock domain.

0: No effect

1: Update

(WT)

Register 28.11. USB_SERIAL_JTAG_SER_AFIFO_CONFIG_REG (0x0064)

(reserved)																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_WFULL					
USB_SERIAL__JTAG_SERIAL_OUT_AFIFO_EMPTY																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_RD					
USB_SERIAL__JTAG_SERIAL_OUT_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_RD					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR					
USB_SERIAL__JTAG_SERIAL_IN_AFIFO_RESET_WR																								USB_SERIAL__JTAG_SERIAL_IN_AFIFO					

USB_SERIAL_JTAG_SERIAL_IN_AFIFO_WFULL
 USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_EMPTY
 USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_RESET_RD
 USB_SERIAL_JTAG_SERIAL_IN_AFIFO_RESET_WR
 USB_SERIAL_JTAG_SERIAL_IN_AFIFO_RESET_RD
 USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_RESET_WR

USB_SERIAL_JTAG_SERIAL_IN_AFIFO_RESET_WR Configures whether to reset CDC_ACM IN async FIFO write clock domain.

0: No effect

1: Reset

(R/W)

USB_SERIAL_JTAG_SERIAL_IN_AFIFO_RESET_RD Configures whether to reset CDC_ACM IN async FIFO read clock domain.

0: No effect

1: Reset

(R/W)

USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_RESET_WR Configures whether to reset CDC_ACM OUT async FIFO write clock domain.

0: No effect

1: Reset

(R/W)

USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_RESET_RD Configures whether to reset CDC_ACM OUT async FIFO read clock domain.

0: No effect

1: Reset

(R/W)

USB_SERIAL_JTAG_SERIAL_OUT_AFIFO_EMPTY Represents CDC_ACM OUT async FIFO empty signal in read clock domain. (RO)

USB_SERIAL_JTAG_SERIAL_IN_AFIFO_WFULL Represents CDC_ACM IN async FIFO full signal in write clock domain. (RO)

Register 28.12. USB_SERIAL_JTAG_SERIAL_EP_TIMEOUTO_REG (0x006C)

Diagram of the USB_SERIAL_JTAG_SERIAL_TIMEOUT_STATUS_CLR register. The register is 32 bits wide. Bits 31-3 are reserved. Bits 2-0 are the USB_SERIAL_JTAG_SERIAL_TIMEOUT_STATUS_CLR field. The register is reset to 0.

31	3	2	1	0
0	0	0	0	0

Reset

USB_SERIAL_JTAG_SERIAL_TIMEOUT_EN USB serial out endpoint timeout enable. When a timeout event occurs, serial out endpoint buffer is automatically cleared and **USB_SERIAL_JTAG_SERIAL_TIMEOUT_STATUS** is asserted.

0: timeout disabled, buffer will not be cleared automatically

1: timeout enabled

(R/W)

USB SERIAL JTAG SERIAL TIMEOUT STATUS Timeout status bit

0: No timeout occurred.

1: Serial out endpoint has triggered a timeout event.

(R/WTC/SS)

USB_SERIAL_JTAG_SERIAL_TIMEOUT_STATUS_CLR Write 1 to clear

USB_SERIAL_JTAG_SERIAL_TIMEOUT_STATUS. (WT)

Register 28.13. USB_SERIAL_JTAG_SERIAL_EP_TIMEOUT1_REG (0x0070)

Diagram of the `USB_SERIAL_JTAG_SERIAL_TIMEOUT_MAX` register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The value `0x494100` is shown in the register field. A label "Reset" points to the register.

USB_SERIAL_JTAG_SERIAL_TIMEOUT_MAX USB serial out endpoint timeout max threshold value, indicates the maximum time that waiting for chips to take away data in memory. This value is in steps of 20.83 ns. (R/W)

Register 28.14. USB_SERIAL_JTAG_INT_RAW_REG (0x0008)

(reserved)																USB_SERIAL_JTAG_SET_LINE_CODE_INT_RAW USB_SERIAL_JTAG_GET_LINE_CODE_INT_RAW USB_SERIAL_JTAG_DTR_CHG_INT_RAW USB_SERIAL_JTAG_RTS_CHG_INT_RAW USB_SERIAL_JTAG_OUT_EP2_INT_RAW USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_RAW USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_RAW USB_SERIAL_JTAG_USB_BUS_RESET_INT_RAW USB_SERIAL_JTAG_STUFF_ERR_INT_RAW USB_SERIAL_JTAG_CRC16_ERR_INT_RAW USB_SERIAL_JTAG_PID_ERR_INT_RAW USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_RAW USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_RAW USB_SERIAL_JTAG_SOF_INT_RAW USB_SERIAL_JTAG_IN_FLUSH_INT_RAW																
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	Reset								

USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_SOF_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_SOF_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_PID_ERR_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_PID_ERR_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_CRC5_ERR_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_CRC5_ERR_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_CRC16_ERR_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_CRC16_ERR_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_STUFF_ERR_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_STUFF_ERR_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_USB_BUS_RESET_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_USB_BUS_RESET_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT](#). (R/WTC/SS)

Continued on the next page...

Register 28.14. USB_SERIAL_JTAG_INT_RAW_REG (0x0008)

Continued from the previous page...

USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_RTS_CHG_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_RTS_CHG_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_DTR_CHG_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_DTR_CHG_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_GET_LINE_CODE_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_GET_LINE_CODE_INT](#). (R/WTC/SS)

USB_SERIAL_JTAG_SET_LINE_CODE_INT_RAW The raw interrupt status of [USB_SERIAL_JTAG_SET_LINE_CODE_INT](#). (R/WTC/SS)

Register 28.15. USB_SERIAL_JTAG_INT_ST_REG (0x000C)

(reserved)																USB_SERIAL_JTAG_SET_LINE_CODE_INT_ST USB_SERIAL_JTAG_GET_LINE_CODE_INT_ST USB_SERIAL_JTAG_DTR_CHG_INT_ST USB_SERIAL_JTAG_RTS_CHG_INT_ST USB_SERIAL_JTAG_OUT_EP2_INT_ST USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_ST USB_SERIAL_JTAG_USB_BUS_RESET_INT_ST USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_ST USB_SERIAL_JTAG_STUFF_ERR_INT_ST USB_SERIAL_JTAG_CRC16_ERR_INT_ST USB_SERIAL_JTAG_CRC5_ERR_INT_ST USB_SERIAL_JTAG_PID_ERR_INT_ST USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_ST USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_ST																		
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset								

USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT](#). (RO)

USB_SERIAL_JTAG_SOF_INT_ST The masked interrupt status of [USB_SERIAL_JTAG_SOF_INT](#).
(RO)

USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_ST The masked interrupt status of the
[USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#). (RO)

USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#). (RO)

USB_SERIAL_JTAG_PID_ERR_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_PID_ERR_INT](#). (RO)

USB_SERIAL_JTAG_CRC5_ERR_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_CRC5_ERR_INT](#). (RO)

USB_SERIAL_JTAG_CRC16_ERR_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_CRC16_ERR_INT](#). (RO)

USB_SERIAL_JTAG_STUFF_ERR_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_STUFF_ERR_INT](#). (RO)

USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT](#). (RO)

USB_SERIAL_JTAG_USB_BUS_RESET_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_USB_BUS_RESET_INT](#). (RO)

USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT](#) (RO)

USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_ST The masked interrupt status of
[USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT](#). (RO)

Continued on the next page...

Register 28.15. USB_SERIAL_JTAG_INT_ST_REG (0x000C)

Continued from the previous page...

USB_SERIAL_JTAG_RTS_CHG_INT_ST USB_SERIAL_JTAG_RTS_CHG_INT . (RO)	The	masked	interrupt	status	of
USB_SERIAL_JTAG_DTR_CHG_INT_ST USB_SERIAL_JTAG_DTR_CHG_INT . (RO)	The	masked	interrupt	status	of
USB_SERIAL_JTAG_GET_LINE_CODE_INT_ST USB_SERIAL_JTAG_GET_LINE_CODE_INT . (RO)	The	masked	interrupt	status	of
USB_SERIAL_JTAG_SET_LINE_CODE_INT_ST USB_SERIAL_JTAG_SET_LINE_CODE_INT . (RO)	The	masked	interrupt	status	of

Register 28.16. USB_SERIAL_JTAG_INT_ENA_REG (0x0010)

(reserved)																USB_SERIAL_JTAG_SET_LINE_CODE_INT_ENA USB_SERIAL_JTAG_GET_LINE_CODE_INT_ENA USB_SERIAL_JTAG_DTR_CHG_INT_ENA USB_SERIAL_JTAG_RTS_CHG_INT_ENA USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_ENA USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_ENA USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_ENA USB_SERIAL_JTAG_USB_BUS_RESET_INT_ENA USB_SERIAL_JTAG_STUFF_ERR_INT_ENA USB_SERIAL_JTAG_CRC16_ERR_INT_ENA USB_SERIAL_JTAG_CRC5_ERR_INT_ENA USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_ENA USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_ENA USB_SERIAL_JTAG_SOF_INT_ENA USB_SERIAL_JTAG_IN_FLUSH_INT_ENA																	
31																	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT](#). (R/W)

USB_SERIAL_JTAG_SOF_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_SOF_INT](#). (R/W)

USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#). (R/W)

USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#). (R/W)

USB_SERIAL_JTAG_PID_ERR_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_PID_ERR_INT](#). (R/W)

USB_SERIAL_JTAG_CRC5_ERR_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_CRC5_ERR_INT](#).
(R/W)

USB_SERIAL_JTAG_CRC16_ERR_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_CRC16_ERR_INT](#).
(R/W)

USB_SERIAL_JTAG_STUFF_ERR_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_STUFF_ERR_INT](#).
(R/W)

USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT](#). (R/W)

USB_SERIAL_JTAG_USB_BUS_RESET_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_USB_BUS_RESET_INT](#). (R/W)

USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT](#). (R/W)

USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT](#). (R/W)

Continued on the next page...

Register 28.16. USB_SERIAL_JTAG_INT_ENA_REG (0x0010)

Continued from the previous page...

USB_SERIAL_JTAG_RTS_CHG_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_RTS_CHG_INT](#).
(R/W)

USB_SERIAL_JTAG_DTR_CHG_INT_ENA Write 1 to enable [USB_SERIAL_JTAG_DTR_CHG_INT](#).
(R/W)

USB_SERIAL_JTAG_GET_LINE_CODE_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_GET_LINE_CODE_INT](#). (R/W)

USB_SERIAL_JTAG_SET_LINE_CODE_INT_ENA Write 1 to enable
[USB_SERIAL_JTAG_SET_LINE_CODE_INT](#). (R/W)

Register 28.17. USB_SERIAL_JTAG_INT_CLR_REG (0x0014)

(reserved)																USB_SERIAL_JTAG_SET_LINE_CODE_INT_CLR USB_SERIAL_JTAG_GET_LINE_CODE_INT_CLR USB_SERIAL_JTAG_DTR_CHG_INT_CLR USB_SERIAL_JTAG_RTS_CHG_INT_CLR USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_CLR USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_CLR USB_SERIAL_JTAG_USB_BUS_RESET_INT_CLR USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_CLR USB_SERIAL_JTAG_CRC16_ERR_INT_CLR USB_SERIAL_JTAG_CRC5_ERR_INT_CLR USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_CLR USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_CLR USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT_CLR																	
31																16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset								

USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_JTAG_IN_FLUSH_INT](#). (WT)

USB_SERIAL_JTAG_SOF_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_SOF_INT](#). (WT)

USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#). (WT)

USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_SERIAL_IN_EMPTY_INT](#). (WT)

USB_SERIAL_JTAG_PID_ERR_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_PID_ERR_INT](#). (WT)

USB_SERIAL_JTAG_CRC5_ERR_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_CRC5_ERR_INT](#). (WT)

USB_SERIAL_JTAG_CRC16_ERR_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_CRC16_ERR_INT](#). (WT)

USB_SERIAL_JTAG_STUFF_ERR_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_STUFF_ERR_INT](#). (WT)

USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_IN_TOKEN_REC_IN_EP1_INT](#). (WT)

USB_SERIAL_JTAG_USB_BUS_RESET_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_USB_BUS_RESET_INT](#). (WT)

USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_OUT_EP1_ZERO_PAYLOAD_INT](#). (WT)

USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_OUT_EP2_ZERO_PAYLOAD_INT](#). (WT)

Continued on the next page...

Register 28.17. USB_SERIAL_JTAG_INT_CLR_REG (0x0014)

Continued from the previous page...

USB_SERIAL_JTAG_RTS_CHG_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_RTS_CHG_INT](#). (WT)

USB_SERIAL_JTAG_DTR_CHG_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_DTR_CHG_INT](#). (WT)

USB_SERIAL_JTAG_GET_LINE_CODE_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_GET_LINE_CODE_INT](#).
(WT)

USB_SERIAL_JTAG_SET_LINE_CODE_INT_CLR Write 1 to clear [USB_SERIAL_JTAG_SET_LINE_CODE_INT](#).
(WT)

Register 28.18. USB_SERIAL_JTAG_JFIFO_ST_REG (0x0020)

(reserved)																						USB_SERIAL_JTAG_OUT_FIFO_RESET USB_SERIAL_JTAG_IN_FIFO_RESET USB_SERIAL_JTAG_OUT_FIFO_FULL USB_SERIAL_JTAG_OUT_FIFO_EMPTY USB_SERIAL_JTAG_IN_FIFO_CNT USB_SERIAL_JTAG_IN_FIFO_FULL USB_SERIAL_JTAG_IN_FIFO_EMPTY											
31																						10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	Reset										

Reset

USB_SERIAL_JTAG_IN_FIFO_CNT Represents JTAG IN FIFO counter. (RO)

USB_SERIAL_JTAG_IN_FIFO_EMPTY Represents whether JTAG IN FIFO is empty.

0: Not empty

1: Empty

(RO)

USB_SERIAL_JTAG_IN_FIFO_FULL Represents whether JTAG IN FIFO is full.

0: Not full

1: Full

(RO)

USB_SERIAL_JTAG_OUT_FIFO_CNT Represents JTAG OUT FIFO counter. (RO)

USB_SERIAL_JTAG_OUT_FIFO_EMPTY Represents whether JTAG OUT FIFO is empty.

0: Not empty

1: Empty

(RO)

USB_SERIAL_JTAG_OUT_FIFO_FULL Represents whether JTAG OUT FIFO is full.

0: Not full

1: Full

(RO)

USB_SERIAL_JTAG_IN_FIFO_RESET Configures whether to reset JTAG IN FIFO.

0: No effect

1: Reset

(R/W)

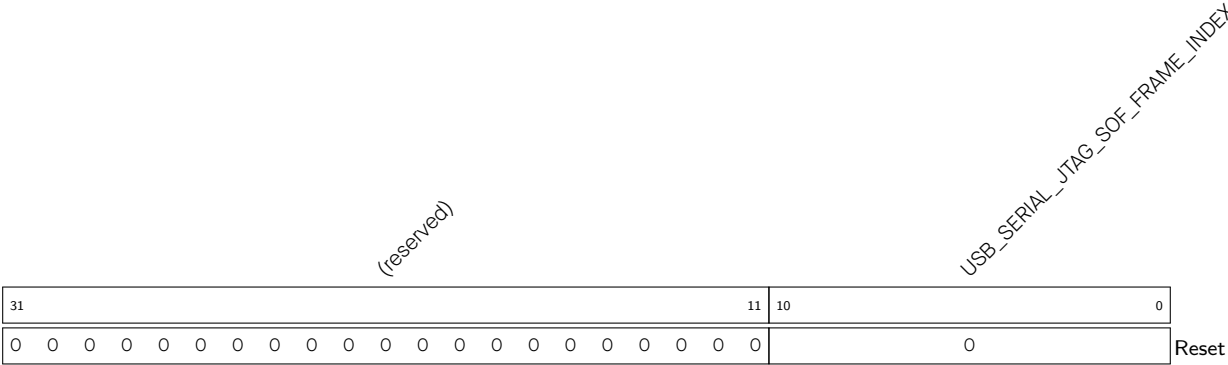
USB_SERIAL_JTAG_OUT_FIFO_RESET Configures whether to reset JTAG OUT FIFO.

0: No effect

1: Reset

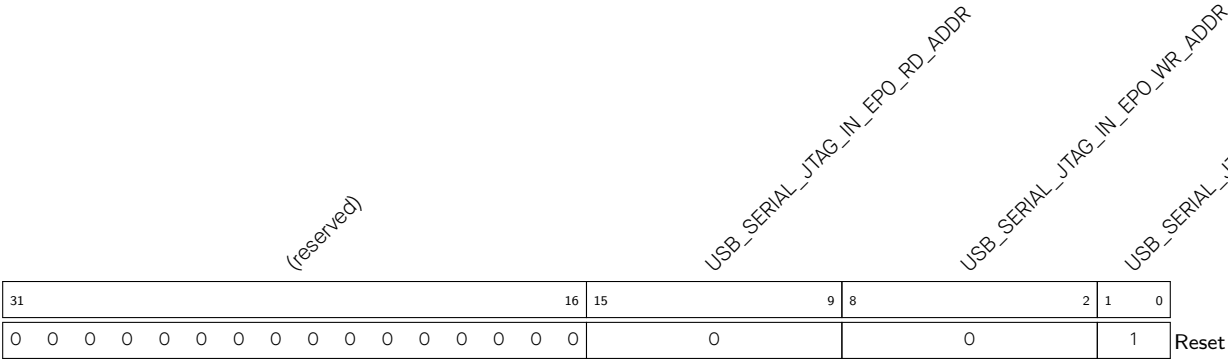
(R/W)

Register 28.19. USB_SERIAL_JTAG_FRAM_NUM_REG (0x0024)



USB_SERIAL_JTAG_SOF_FRAME_INDEX Represents frame index of received SOF frame. (RO)

Register 28.20. USB_SERIAL_JTAG_IN_EPO_ST_REG (0x0028)



USB_SERIAL_JTAG_IN_EPO_STATE Represents state of IN Endpoint 0. (RO)

USB_SERIAL_JTAG_IN_EPO_WR_ADDR Represents write data address of IN endpoint 0. (RO)

USB_SERIAL_JTAG_IN_EPO_RD_ADDR Represents read data address of IN endpoint 0. (RO)

Register 28.21. USB_SERIAL_JTAG_IN_EP1_ST_REG (0x002C)

(reserved)																USB_SERIAL_JTAG_IN_EP1_RD_ADDR								USB_SERIAL_JTAG_IN_EP1_WR_ADDR								USB_SERIAL_JTAG_IN_EP1_STATE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																16																15																9																8																2																1																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0															

Reset

USB_SERIAL_JTAG_IN_EP1_STATE Represents state of IN Endpoint 1. (RO)**USB_SERIAL_JTAG_IN_EP1_WR_ADDR** Represents write data address of IN endpoint 1. (RO)**USB_SERIAL_JTAG_IN_EP1_RD_ADDR** Represents read data address of IN endpoint 1. (RO)

Register 28.22. USB_SERIAL_JTAG_IN_EP2_ST_REG (0x0030)

(reserved)																USB_SERIAL_JTAG_IN_EP2_RD_ADDR								USB_SERIAL_JTAG_IN_EP2_WR_ADDR								USB_SERIAL_JTAG_IN_EP2_STATE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																16																15																9																8																2																1																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0															

Reset

USB_SERIAL_JTAG_IN_EP2_STATE Represents state of IN Endpoint 2. (RO)**USB_SERIAL_JTAG_IN_EP2_WR_ADDR** Represents write data address of IN endpoint 2. (RO)**USB_SERIAL_JTAG_IN_EP2_RD_ADDR** Represents read data address of IN endpoint 2. (RO)

Register 28.23. USB_SERIAL_JTAG_IN_EP3_ST_REG (0x0034)

(reserved)																USB_SERIAL_JTAG_IN_EP3_RD_ADDR								USB_SERIAL_JTAG_IN_EP3_WR_ADDR								USB_SERIAL_JTAG_IN_EP3_STATE							
31																16	15								9	8							2	1	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							0							1		0					

Reset

USB_SERIAL_JTAG_IN_EP3_STATE Represents state of IN Endpoint 3. (RO)**USB_SERIAL_JTAG_IN_EP3_WR_ADDR** Represents write data address of IN endpoint 3. (RO)**USB_SERIAL_JTAG_IN_EP3_RD_ADDR** Represents read data address of IN endpoint 3. (RO)

Register 28.24. USB_SERIAL_JTAG_OUT_EPO_ST_REG (0x0038)

(reserved)																USB_SERIAL_JTAG_OUT_EPO_RD_ADDR								USB_SERIAL_JTAG_OUT_EPO_WR_ADDR								USB_SERIAL_JTAG_OUT_EPO_STATE							
31																16	15								9	8							2	1	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							0							0		0					

Reset

USB_SERIAL_JTAG_OUT_EPO_STATE Represents state of OUT Endpoint 0. (RO)**USB_SERIAL_JTAG_OUT_EPO_WR_ADDR** Represents write data address of OUT endpoint 0.

When [USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT](#) is detected, there are (USB_SERIAL_JTAG_OUT_EPO_WR_ADDR – 2) bytes data in OUT endpoint 0.

(RO)

USB_SERIAL_JTAG_OUT_EPO_RD_ADDR Represents read data address of OUT endpoint 0. (RO)

Register 28.25. USB_SERIAL_JTAG_OUT_EP1_ST_REG (0x003C)

(reserved)										USB_SERIAL_JTAG_OUT_EP1_REC_DATA_CNT										USB_SERIAL_JTAG_OUT_EP1_RD_ADDR										USB_SERIAL_JTAG_OUT_EP1_WR_ADDR										USB_SERIAL_JTAG_OUT_EP1_STATE																																																											
31										23										22										16										15										9										8										2										1										0									
0 0 0 0 0 0 0 0 0 0										0										0										0										0										0										Reset																																							

Reset

USB_SERIAL_JTAG_OUT_EP1_STATE Represents state of OUT Endpoint 1. (RO)**USB_SERIAL_JTAG_OUT_EP1_WR_ADDR** Represents write data address of OUT endpoint 1.

When **USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT** is detected, there are
 (USB_SERIAL_JTAG_OUT_EP1_WR_ADDR – 2) bytes data in OUT endpoint 1.
 (RO)

USB_SERIAL_JTAG_OUT_EP1_RD_ADDR Represents read data address of OUT endpoint 1. (RO)

USB_SERIAL_JTAG_OUT_EP1_REC_DATA_CNT Represents data count in OUT endpoint 1 when one
 packet is received. (RO)

Register 28.26. USB_SERIAL_JTAG_OUT_EP2_ST_REG (0x0040)

(reserved)																USB_SERIAL_JTAG_OUT_EP2_RD_ADDR																USB_SERIAL_JTAG_OUT_EP2_WR_ADDR																USB_SERIAL_JTAG_OUT_EP2_STATE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																16																15																9																8																2																1																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0															

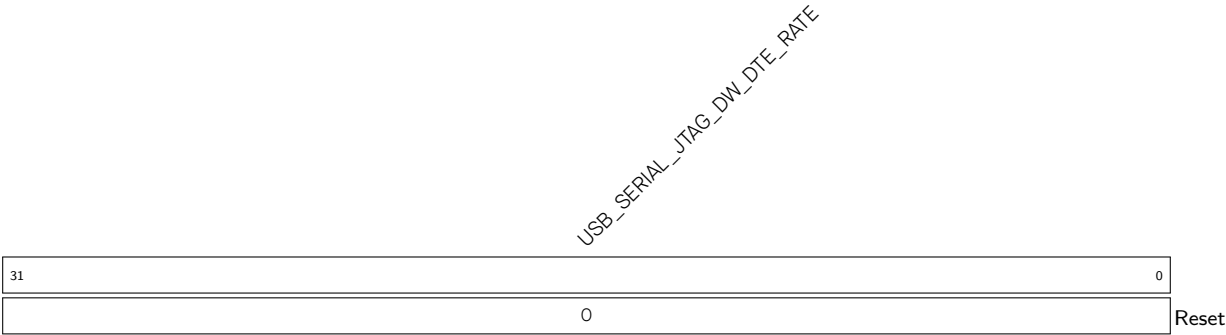
Reset

USB_SERIAL_JTAG_OUT_EP2_STATE Represents state of OUT Endpoint 2. (RO)**USB_SERIAL_JTAG_OUT_EP2_WR_ADDR** Represents write data address of OUT endpoint 2.

When **USB_SERIAL_JTAG_SERIAL_OUT_RECV_PKT_INT** is detected there are
 (USB_SERIAL_JTAG_OUT_EP2_WR_ADDR – 2) bytes data in OUT endpoint 2.
 (RO)

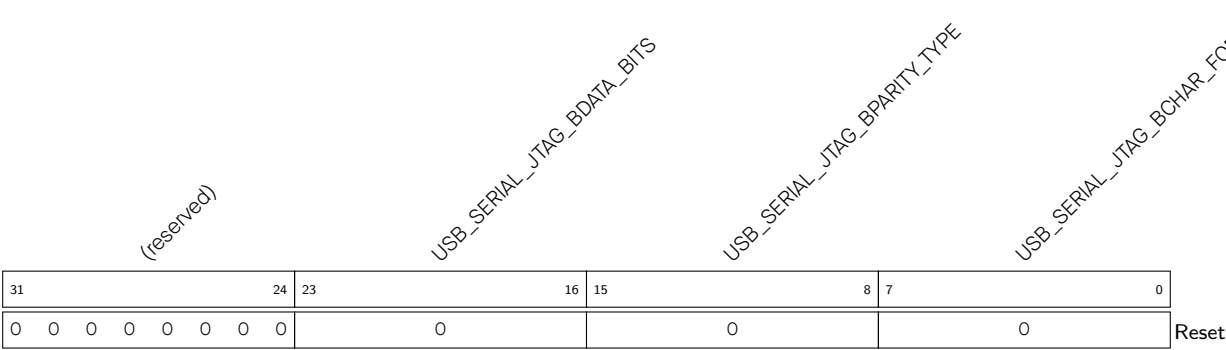
USB_SERIAL_JTAG_OUT_EP2_RD_ADDR Represents read data address of OUT endpoint 2. (RO)

Register 28.27. USB_SERIAL_JTAG_SET_LINE_CODE_W0_REG (0x0050)



USB_SERIAL_JTAG_DW_DTE_RATE Represents the value of dwDTERate set by host through SET_LINE_CODING command. (RO)

Register 28.28. USB_SERIAL_JTAG_SET_LINE_CODE_W1_REG (0x0054)

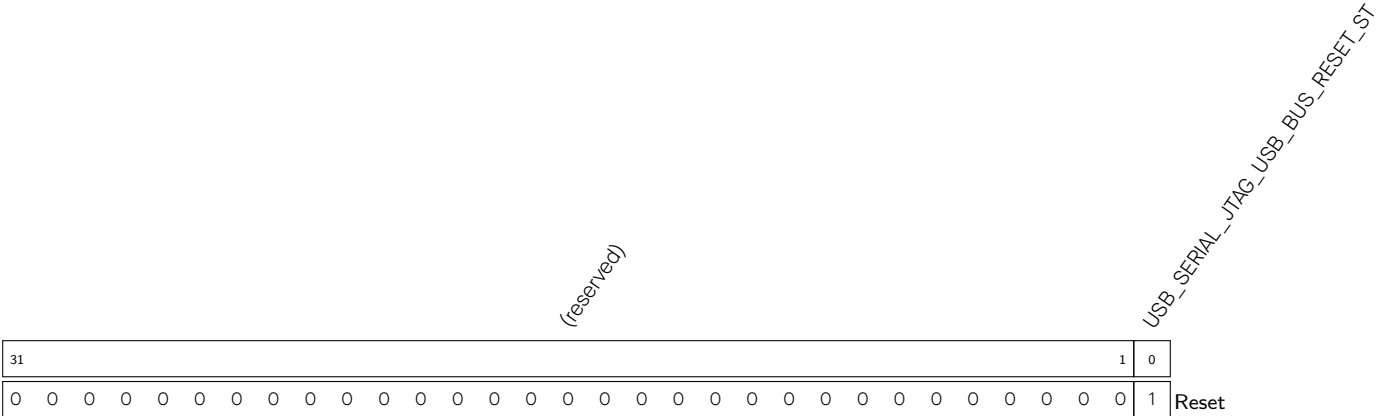


USB_SERIAL_JTAG_BCHAR_FORMAT Represents the value of bCharFormat set by host through SET_LINE_CODING command. (RO)

USB_SERIAL_JTAG_BPARITY_TYPE Represents the value of bParityTpye set by host through SET_LINE_CODING command. (RO)

USB_SERIAL_JTAG_BDATA_BITS Represents the value of bDataBits set by host through SET_LINE_CODING command. (RO)

Register 28.29. USB_SERIAL_JTAG_BUS_RESET_ST_REG (0x0068)

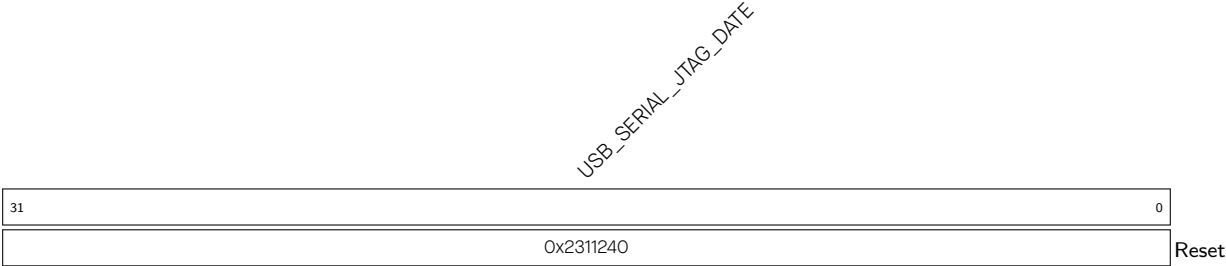


USB_SERIAL_JTAG_USB_BUS_RESET_ST Represents whether USB bus reset is released.

- 0: USB Serial/JTAG is in USB bus reset status
- 1: USB bus reset is released

(RO)

Register 28.30. USB_SERIAL_JTAG_DATE_REG (0x0080)



USB_SERIAL_JTAG_DATE Version control register. (R/W)

Chapter 29

SDIO Slave Controller (SDIO)

29.1 Overview

The ESP32-C5 features hardware support for the Secure Digital Input/Output (SDIO) device interface that conforms to the SDIO Specification V2.00. This allows an SDIO host to access the ESP32-C5 via an SDIO bus protocol.

The SDIO host can read ESP32-C5 SDIO interface registers directly or access shared memory via the Direct Memory Access (DMA) engine, thus reducing processor's overhead while keeping high performance.

29.2 Features

The SDIO Slave Controller has the following features:

- Compatible with SD Physical Layer Specification V2.00 and SDIO V2.00 specifications
- Support for two IO functions (except function 0)
- Support for SPI, 1-bit SDIO, and 4-bit SDIO transfer modes
- Clock range of 0–50 MHz
- Configurable sample and drive clock edge
- Integrated and SDIO-accessible registers for information interaction
- Support for SDIO interrupt mechanism
- Automatic padding data and discarding the padded data on the SDIO bus
- Block size up to 512 bytes
- Interrupt vector between the host and slave for bidirectional interrupt
- Support DMA for data transfer
- Support for wake-up from sleep when connection is retained

29.3 Architecture Overview

The functional block diagram of the SDIO slave module is shown in Figure [29.3-1](#).

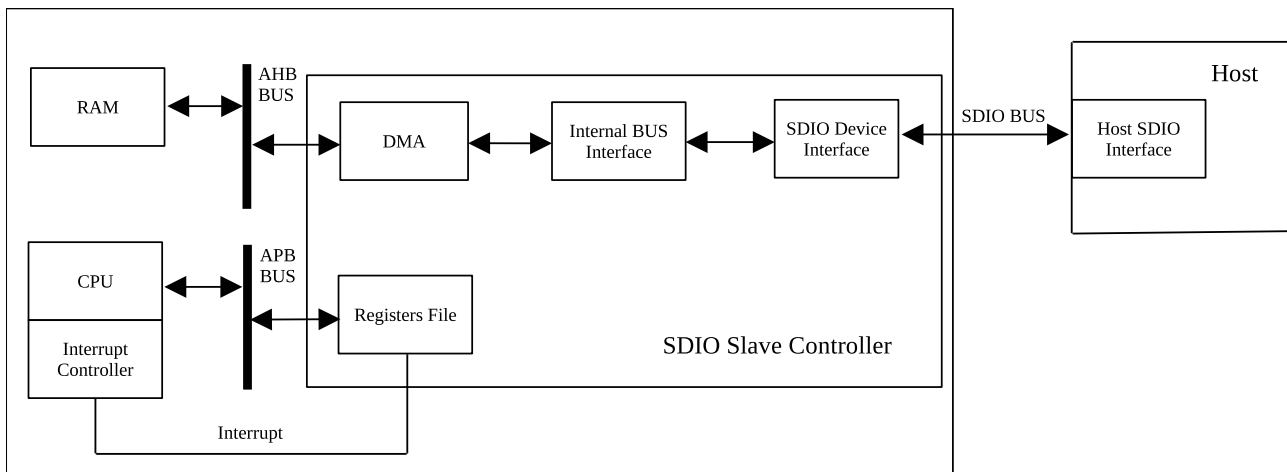


Figure 29.3-1. SDIO Slave Block Diagram

In the above figure, Host represents any host device that is compatible with SDIO Specification V2.00. It interacts with the ESP32-C5 (configured as the SDIO slave) via the standard SDIO bus implementation.

The SDIO Device Interface block enables effective communication with the external Host by directly providing SDIO interface registers and enabling DMA operation for high-speed data transfer over the Advanced High-Performance Bus (AHB) without engaging the CPU.

29.4 Standards Compliance

The ESP32-C5 SDIO Slave Controller conforms to the following standards:

- SD Specifications Part 1 Physical Layer Specification Version 2.00 (referred to as Physical Layer Specification V2.00 in this chapter)
- SD Specifications Part E1 SDIO Specification Version 2.00, January 30, 2007 (referred to as SDIO Specification V2.00 in this chapter)

29.5 Functional Description

29.5.1 Physical Bus

- Bus mode: SPI, 1-bit and 4-bit SDIO transfer modes.
- Bus signal: The physical bus signals of the standard SDIO Specification V2.00, including CS/DI/SCLK/DO/IRQ in the SPI transmission mode, CMD/CLK/DAT[3:0] in the SDIO 1-bit transmission mode, and CMD/CLK/DAT[3:0] in the SDIO 4-bit transmission mode.
- Bus speed mode: full-speed mode of 0–50 MHz clock range and low-speed mode of 0–400 kHz clock range.
- IO functions: two IO functions in addition to function 0. Function 0 is only used for CCCR, FBR, and CIS operations. Function 1 and 2 can be used at the same time to transfer application data packets (such as Wi-Fi packets and Bluetooth packets) and to access SLC Host registers.

For more information, please refer to Physical Layer Specification V2.00 and SDIO Specification V2.00.

29.5.2 Supported Commands

The SDIO Slave Controller mainly supports the IO_RW_DIRECT (CMD52) and IO_RW_EXTENDED (CMD53) data transfer commands.

IO_RW_DIRECT (CMD52) can be used to access registers and transfer data. Figure 29.5-1 shows its fields. For the meaning of each field, please refer to the SDIO Specification V2.00.

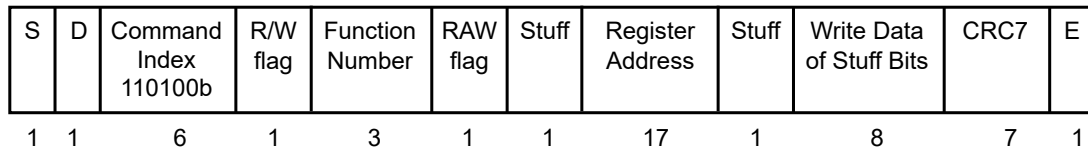


Figure 29.5-1. CMD52 Content

IO_RW_EXTENDED (CMD53) is used to initiate the transfer of packets of an arbitrary length. Figure 29.5-2 shows its fields. For the meaning of each field, please refer to the SDIO Specification V2.00.

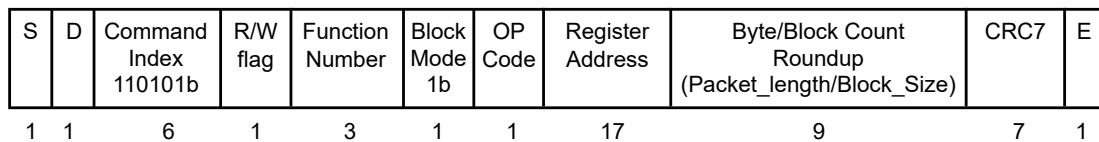


Figure 29.5-2. CMD53 Content

29.5.3 I/O Function 0 Address Space

I/O function 0 is only used for Card Common Control Registers (CCCR), Function Basic Registers (FBR), and Card Information Structure (CIS) operations. Figure 29.5-3 shows its address space map, which is specified by the SDIO Specification. For the meaning of each section in this map, please refer to the SDIO Specification V2.00.

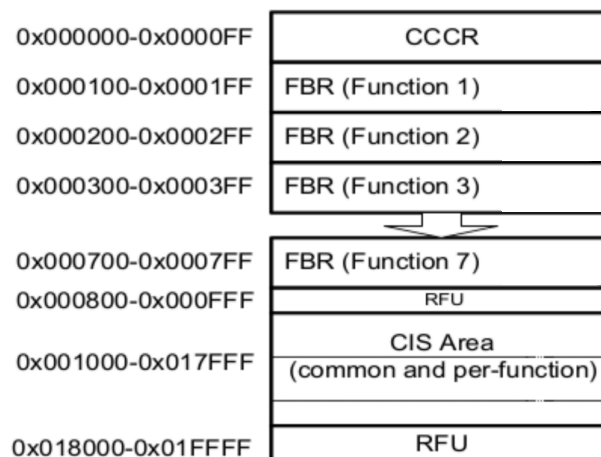


Figure 29.5-3. Function 0 Address Space

As defined in the SDIO Specification, CCCR are common control registers, FBR are control configuration registers for each function, and CIS are status registers for storing card information, such as version, power consumption, and manufacturer. The functions of these registers are optional, and their meanings are detailed in the SDIO Specification.

The CCCR configuration of ESP32-C5 SDIO slave is shown in Table 29.5-1, and the FBR configuration is shown Table 29.5-2.

Table 29.5-1. SDIO Slave CCCR Configuration

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00	CCCR/SDIO Revision	Set SDIO bit[3:0] using HINF_SDIO_VER[7:4] in HINF_CFG_DATA1_REG				Set CCCR bit[3:0] using HINF_SDIO_VER[3:0] in HINF_CFG_DATA1_REG			
0x01	SD Specification Revision	0 (RFU)				Set SD bit[3:0] using HINF_SDIO_VER[11:8] in HINF_CFG_DATA1_REG			
0x02	I/O Enable	0 (IOE[7:3])					R/W (IOE[2:1])		0 (RFU)
0x03	I/O Ready	0 (IOR[7:3])					R (IOR[2:1])		0 (RFU)
0x04	Int Enable	0 (IEN[7:3])					R/W (IEN[2:1])		R/W (IENM)
0x05	Int Pending	0 (INT[7:3])					R (INT[2:1])		0 (RFU)
0x06	I/O Abort	0 (RFU)				W (RES)	W (AS[2:0])		
0x07	Bus Interface Control	R/W (CD Disable)	1 (SCSI)	R/W (ECSI)	0 (RFU)			R/W (Bus Width[1:0])	
0x08	Card Capability	0 (4BLS)	0 (LSC)	R/W (E4MI)	1 (S4MI)	0 (SBS)	1 (SRW)	1 (SMB)	1 (SDC)
0x09-0x0B	Common CIS Pointer	Address 0x09: 0x0; Address 0x0A: 0x10; Address 0x0B: 0x0 (Pointer to card's common CIS)							
0x0C	Bus Suspend	0 (RFU)						0 (BR)	0 (BS)
0x0D	Function Select	0 (DF)	0 (RFU)			0 (FS[3:0])			
0x0E	Exec Flags	0 (EX[7:1])							0 (EXM)
0x0F	Ready Flags	0 (RF[7:1])							0 (RFM)
0x10-0x11	FNO Block Size	R/W (Supported range: 0-512) (I/O block size for Function 0)							

Cont'd on next page

Table 29.5-1. SDIO Slave CCCR Configuration – cont'd from previous page

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x12	Power Control	0 (RFU)						R/W (EMPC)	1 (SMPC)
0x13	High-Speed	0 (RFU)						R/W (EHS)	Note ^a (SHS)

^a Set SHS using HINF_HIGHSPEED_ENABLE in [HINF_CFG_DATA1_REG](#).

Table 29.5-2. SDIO Slave FBR Configuration

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x100	0 (Function 1 CSA enable)	0 (Function 1 supports CSA)	0 (RFU)		0 (Function 1 Standard SDIO Function interface code)			
0x101	0 (Function 1 Extended standard SDIO Function interface code)							
0x102	0 (RFU)						R/W (EPS)	0 (SPS)
0x109-0x10B	Address 0x109: 0x0; Address 0x10A: 0x11; Address 0x10B: 0x0 (Pointer to Function 1 CIS)							
0x110-0x111	R/W (Supported range: 0-512) (I/O block size for Function 1)							
0x200	0 (Function 2 CSA enable)	0 (Function 2 supports CSA)	0 (RFU)		0x2 (Function 2 Standard SDIO Function interface code)			
0x201	0 (Function 2 Extended standard SDIO Function interface code)							
0x202	0 (RFU)						R/W (EPS)	0 (SPS)
0x209-0x20B	Address 0x209: 0x0; Address 0x20A: 0x12; Address 0x20B: 0x0 (Pointer to Function 2 CIS)							
0x210-0x211	R/W (Supported range: 0-512) (I/O block size for Function 2)							

29.5.4 I/O Function 1/2 Address Space Map

I/O function 1 and function 2 have exactly the same functions and permissions. They can be used at the same time or independently to transmit application data (such as Wi-Fi data and Bluetooth data) in fixed-address packets or incremental-address packets. They can also access the same set of SLC Host registers. Figure 29.5-4 shows their address space map. All segments in this space can be accessed by the host.

0x0	Fixed address Packet
0x1 -0x3F	Reserved
0x40 -0x3FF	SLC Host Register
0x400 - 0x1F7FF	Incr address Packet

Figure 29.5-4. Function 1/2 Address Space Map

29.5.4.1 Accessing SLC HOST Register Space

For effective interaction, the host can access the registers that are in contiguous address from 0x40 to 0x3FF in the slave via the I/O function 1/2. To access them, the host simply needs to set the Register Address field of CMD52 or CMD53 to the low 10 bits of their address. Besides, CMD53 allows the host to access multiple registers at one go for a higher transfer rate.

From SLCHOST_CONF_WO_REG and SLCHOST_CONF_W15_REG, there are 52 bytes of fields that the host and slave can access and change, thus facilitating the information interaction.

The software on both the host and the slave sides can access the SLC Host register space at the same time, so an upper-layer mechanism should be designed to avoid the error caused by such behavior.

29.5.4.2 Transferring Incremental-Address Packets

When the host uses the address 0x400 - 0x1F7FF to continuously transmit multiple application data packets (such as Wi-Fi packets), the address field in CMD53 should be set to increment mode and the OP Code field to 1.

For example, if the host wants to use CMD53 to transfer (send or receive) three data blocks starting from the base address 0x500, then it should:

- Set the Block Mode field in CMD53 to 1, indicating data unit is block
- Set the OP Code field to 1, indicating incremental address mode
- Set the Register Address field to 0x500, indicating the base address is 0x500
- Set the Byte/Block Count field to 0x3, indicating 3 data blocks
- Set other fields according to the SDIO Specification

When the packet is transmitted (slave sends to host, or slave receives from host) through CMD53, the slave will determine whether all the valid data of the current packet has been transmitted so as to pad (when slave sends to host) or discard (when slave receives from host) the invalid data. For more information about data padding and discarding, please refer to Section 29.5.5.3.

29.5.4.3 Transferring Fixed-Address Packets

When the host uses address 0x0 to transmit application data packets (such as Bluetooth packets), the address field and OP Code field in CMD53 should be set to 0.

For example, if the host wants to use CMD53 to transfer (send or receive) three data blocks starting from the fixed address 0x0, then it should:

- Set the Block Mode field in CMD53 to 1, indicating data unit is block
- Set the OP Code field to 0, indicating fixed address mode
- Set the Register Address field to 0x0, indicating the fixed address is 0x0
- Set the Byte/Block Count field to 0x3, indicating 3 data blocks
- Set other fields according to the SDIO Specification

When the packet is transmitted (slave sends to host, or slave receives from host) between the host and the slave through CMD53, the slave will determine whether all the valid data of the current packet has been transmitted so as to pad (when slave sends to host) or discard (when slave receives from host) the invalid data. For more information about data padding and discarding, please refer to Section [29.5.5.3](#).

29.5.5 DMA

The SDIO Slave Controller uses a dedicated DMA to access transfer data from RAM or store it to RAM. As shown in Figure [29.3-1](#), RAM is accessed over the AHB. For the RAM space accessible by the Controller, please refer to Chapter [4 System and Memory](#). To set the RAM address range that can be accessed for one transfer, please configure the *SHAREMEM*_REG fields of SLC register described in Section [29.8](#).

DMA has two channels, SLC0 and SLC1. They are used to transfer incremental-address packets and fixed-address packets, respectively. For the convenience of users, the SDIO slave provides function 1/2 for data transmission. The address range of I/O function 1/2 is detailed in Section [29.5.4](#). It is recommended to transmit incremental-address packets via SLC0 using function 1 and fixed-address packets via SLC1 using function 2.

DMA accesses RAM over AHB. You can configure whether the AHB interface can use burst operation and which burst operation type to use by configuring the relevant fields of [SDIO_SLCCONFO_REG](#) and [SDIO_SLC_BURST_LEN_REG](#). For more information, please refer to Section [29.8](#).

29.5.5.1 Linked List

The slave software can use the DMA engine by mounting linked lists. DMA sends the data from the RAM address space configured in the RX (slave to host) linked list and stores the received data into the address space configured in the TX (host to slave) linked list. A linked list consists of several descriptors.

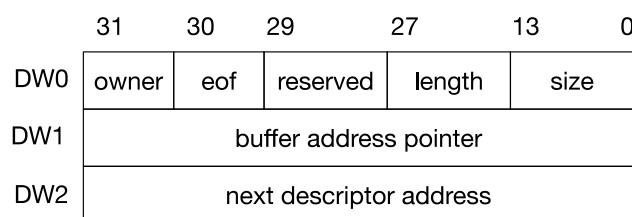


Figure 29.5-5. DMA Linked List Descriptor Structure of the SDIO Slave

The TX linked list descriptor and the RX linked list descriptor have the same structure, which is shown in Figure 29.5-5. The descriptor consists of 3 words. The meaning of each field is as follows:

- owner (DWO) [31]: Indicates who is allowed to access the buffer that this descriptor points to.
0: CPU
1: DMA engine
Slave software should set this field to 1 when creating the descriptor. After the DMA write-back permission is enabled and the corresponding buffer is used by the DMA, the field is cleared to 0.
- eof (DWO) [30]: Indicates the end of a data packet.
0: The current descriptor is not the last descriptor of the packet
1: The current descriptor is the last descriptor of the packet
When the host sends a packet to the slave, the slave software should set the field to 0 while creating the descriptor. DMA sets the field of the last descriptor in the packet to 1; When the host receives packets from the slave, the slave software configures the field depending on whether this descriptor is the last descriptor of the packet.
- reserved (DWO) [29:28]: reserved.
Slave software should set this field to 0x0.
- length (DWO) [27:14]: Indicates the number of valid bytes in the corresponding buffer. When the DMA engine is reading data from the buffer, it indicates the number of bytes that can be read; when DMA engine is storing data in the buffer, it indicates the number of bytes of the stored data.
When the host sends a packet to the slave, the slave software should set this field to 0x0 while creating the descriptor. DMA writes back the field after the corresponding buffer is used up; when the host receives the packet from the slave and the slave creates the descriptor, the slave software should set this field to the number of bytes that can be read by the corresponding buffer.
- size (DWO) [13:0]: Indicates the size of the corresponding buffer. Unit: byte.
Slave software should configure this field when creating the descriptor.
Note: This field must be word-aligned.
- buffer address pointer (DW1): Buffer address pointer.
Slave software should configure this field when creating the descriptor.
Note: This field must be word-aligned.
- next descriptor address (DW2): Address of the next descriptor. When the next descriptor does not exist, the value is 0.
Slave software should configure this field when creating the descriptor.

For more information on DMA write-back linked list descriptor fields, please refer to Section 29.5.5.2.

The slave software can combine multiple descriptors into a linked list using the next descriptor address (DW2) field. The SDIO slave DMA linked list is shown in Figure 29.5-6.

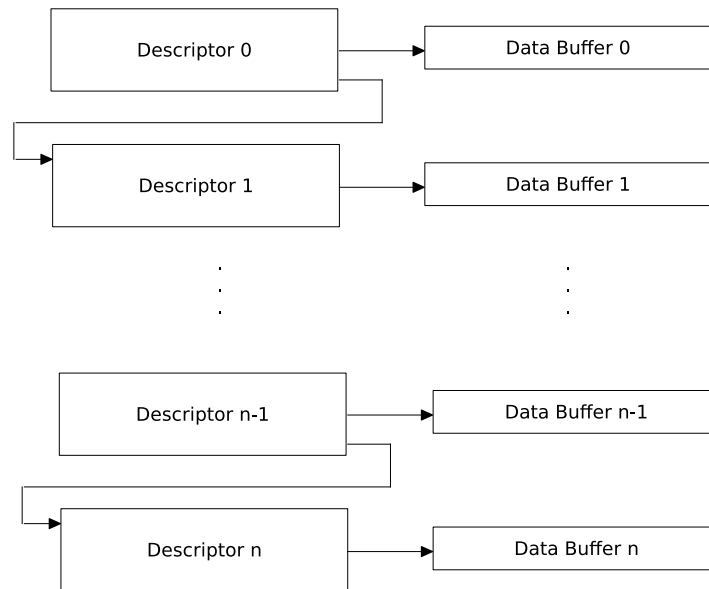


Figure 29.5-6. DMA Linked List of the SDIO Slave

An example is provided below to facilitate understanding of the linked list and the eof bit. Suppose the slave software creates a linked list that contains 3 descriptors:

- Descriptor 0 points to 500 bytes of data and its eof bit is 0.
- Descriptor 1 points to 200 bytes of data and its eof bit is 1.
- Descriptor 2 points to 200 bytes of data and its eof bit is 1.

1. If the first CMD53 needs to read 400 bytes, then DMA sends the first 400 bytes of descriptor 0 to the host.
2. If the second CMD53 needs to read 400 bytes, firstly DMA sends the remaining 100 bytes of descriptor 0 to the host. Secondly, it sends 200 bytes of descriptor 1 to the host. Since the eof bit of descriptor 1 is 1, DMA considers the valid data of the current CMD53 is over. So, lastly DMA sends 100 bytes of invalid data 0x0 to the host.
3. If the third CMD53 needs to read 400 bytes, firstly DMA sends the 200 bytes of descriptor 2 to the host. Since the eof bit of descriptor 2 is 1, DMA considers the valid data of current CMD53 is over. So, DMA sends 200 bytes of invalid data 0x0 to the host.

29.5.5.2 Write-Back of Linked List

In the process of sending packets from the host to the slave, when the buffer specified by a linked list descriptor is full, or when a packet transmission ends, the DMA engine needs to jump to the next descriptor to store subsequent data. Before the jump, DMA writes back the current descriptor. In DWO, DMA updates the eof and length bits to the latest value. The value of the owner bit is determined by `SDIO_SLC0/1_TX_LOOP_TEST` in `SDIO_SLCCONFO_REG`.

In the process of receiving packets from the slave, when the host reads all the data in the buffer specified by a linked list descriptor, DMA engine needs to jump to the next descriptor to read subsequent data. Before the jump, the slave software can set `SDIO_SLC0/1_RX_AUTO_WBACK` in `SDIO_SLCCONFO_REG` to 1 so that the DMA will write back the current descriptor. The value to write to the owner bit is determined by

SDIO_SLC0/1_RX_LOOP_TEST in [SDIO_SLCCONFO_REG](#). Values of other bits in DWO remain unchanged.

The relevant register fields are described in Section [29.8](#).

29.5.5.3 Data Padding and Discarding

In order to transfer data in blocks, both the host and the slave need to pad the data sent on the SDIO bus into entire blocks. The slave will automatically pad data when sending the packet, and automatically discard the padded data after receiving packets.

- When the host sends a data packet to the slave through CMD53 and the amount of data reaches the length of the data packet, the SDIO slave considers that the valid data of the current data packet is over. At this time, DMA will write back the current linked list descriptor, set the eof bit of the current descriptor to 1, and generate [SLC0/1_TX_SUC_EOF_INT](#) interrupt. After it determines that the valid data is over, the remaining data of the current packet will be considered as invalid data, and will not be received into the buffer by DMA. The slave will not restart receiving data into the next buffer until the next CMD53.
 - For incremental-address packets, the slave determines valid data based on the address. The data with the address greater than or equal to 0x1F800 is considered as invalid data and will be discarded. Therefore, the host should set the CMD53 start address field to 0x1F800 – Packet_length (unit: byte). The data flow of incremental-address packets on SDIO bus is shown in Figure [29.5-7](#).

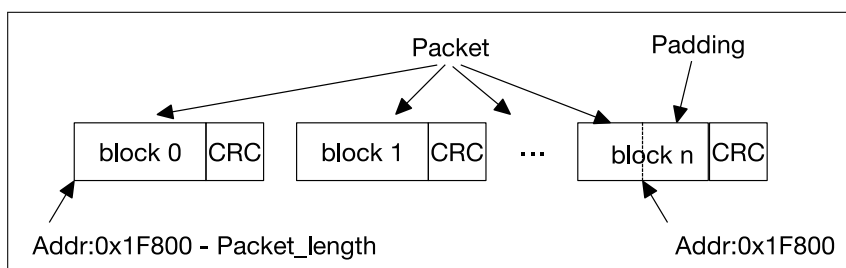


Figure 29.5-7. Data Flow of Sending Incremental-address Packets From Host to Slave

- For fixed-address packets, the slave considers the first 3 bytes of the data packet as the packet length (including the first 3 bytes, which will also be stored in the buffer specified by the linked list). After the length of the received data reaches the packet length, the subsequent data will be considered as invalid and discarded.
- When the host receives data packets (including incremental-address packets and fixed-address packets) from the slave through CMD53 command and DMA reads the last byte of a buffer and the eof bit of the DMA linked list descriptor is 1, the SDIO slave will consider the valid data of the current packet is over. At this time, DMA will write back the current descriptor and generate [SLC0/1_RX_EOF_INT](#) interrupt. After the SDIO slave determines that the valid data is over, the remaining bits of the current data packet will be padded with invalid data 0x0, and will not be read from the buffer via DMA. The slave will restart to read data from the buffer via DMA when the next CMD53 command is sent by the host.

Note: When the host receives either incremental-address or fixed-address data packets from the slave, the eof bit of the DMA linked list descriptor is always considered as the basis for determining the end of data, rather than the address 0x1F800. Therefore, when the host sends multiple CMD53 commands to

obtain multiple data packets, as long as the DMA does not encounter the eof bit set to 1 in the descriptors, the slave will obtain the data from buffers in sequence according to the linked list and then transmit it to the host; when the DMA encounters the eof bit set to 1, the data will be fetched from the corresponding buffer, and then invalid data will be padded to complete the current CMD53 command, and the next CMD53 command will take data from the buffer pointed to by the next descriptor.

29.5.6 SDIO Bus Timing

The SDIO bus operates at a very high speed and the PCB trace length usually affects signal integrity by introducing latency. To ensure that the timing characteristics conform to the desired bus timing, the SDIO slave module supports configuration of input sampling clock edge and output driving clock edge.

When the incoming data changes near the rising edge of the clock, the slave will perform sampling on the falling edge of the clock, or vice versa, as Figure 29.5-8 shows.

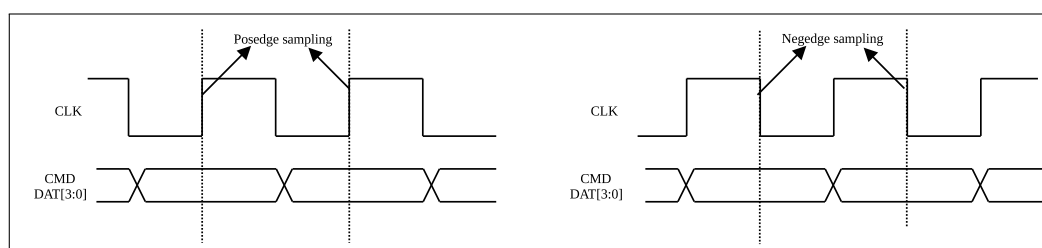


Figure 29.5-8. Sampling Timing Diagram

By default, the voltage level of GPIO25 strapping pin determines the slave pin's sampling edge. However, you can decide the sampling edge by configuring the `SLCHOST_CONF_REG` register, with priority from high to low: (1) Set `SLCHOST_FRC_POS_SAMP` to sample the corresponding signal at the rising edge; (2) Set `SLCHOST_FRC_NEG_SAMP` to sample the corresponding signal at the falling edge.

`SLCHOST_FRC_POS_SAMP` and `SLCHOST_FRC_NEG_SAMP` fields are five bits wide. The bits correspond to the CMD line and four DATA lines (0-3). Setting a bit causes the corresponding line to be sampled for input at the rising clock edge or falling clock edge.

The slave can also select which edge to drive the output lines, in order to accommodate for any latency caused by the physical signal path. The output timing is shown in Figure 29.5-9.

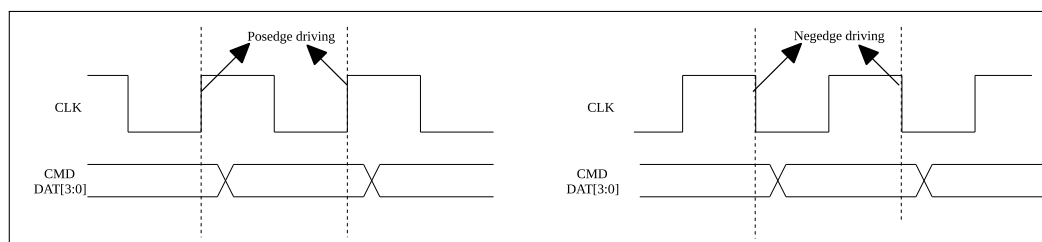


Figure 29.5-9. Output Timing Diagram

By default, the voltage level of MTDI strapping pin determines the slave pin's output driving edge. However, you can decide the output driving edge by configuring the following registers, with priority from high to low: (1) Set `SLCHOST_FRC_SDIO11` in `SLCHOST_CONF_REG` to output the corresponding signal at the falling clock edge; (2) Set `SLCHOST_FRC_SDIO22` in `SLCHOST_CONF_REG` to output the corresponding signal at the rising clock edge.

clock edge; (3) Set HINF_HIGHSPEED_ENABLE in [HINF_CFG_DATA1_REG](#) and SLCHOST_HSPEED_CON_EN in [SLCHOST_CONF_REG](#), then set the EHS (Enable High-Speed) bit in CCCR at the host side to output the corresponding signal at the rising clock edge.

SLCHOST_FRC_SDIO11 and SLCHOST_FRC_SDIO22 fields are five bits wide. The bits correspond to the CMD line and four DATA lines (0-3). Setting a bit causes the corresponding line to output at the rising clock edge or falling clock edge.

Notes on priority setting: The configuration of strapping pins has the lowest priority when controlling the sampling edge or driving edge. The lower-priority configuration takes effect only when the higher-priority configuration is not set. For example, the GPIO25 strapping value determines the sampling edge only when SCLHOST_FRC_POS_SAMP and SCLHOST_FRC_NEG_SAMP are not set.

29.6 Interrupt

The host and the slave can interrupt each other via the interrupt vector. There are 8 interrupt vectors between the host and each DMA SLC channel of the slave. To send an interrupt to the other side, the enable bit of the interrupt vector register should be set to 1.

29.6.1 Host Interrupt

- *SLCHOST_SLC0/1_RX_NEW_PACKET_INT* : The slave has a packet to send. Any of the cases below can trigger the interrupt.
 - When [SDIO_SLC0_RXLINK_START](#) or [SDIO_SLC1_RXLINK_START](#) is set to enable DMA
 - When a new RX linked list descriptor is coming after DMA processes the RX linked list descriptor with the eof bit being 1
 - When a packet needs to be retransmitted
- *SLCHOST_SLC0/1_TX_OVF_INT* : Slave TX (transmit) buffer overflow interrupt.
- *SLCHOST_SLC0/1_RX_UDF_INT* : Slave RX (receive) buffer underflow interrupt.
- *SLCHOST_SLC0/1_TOHOST_BIT_n_INT* (*n*: 0-7): Slave interrupts the host.

29.6.2 Slave Interrupt

- *SLCO/1_TX_DSCR_ERR_INT* : Slave TX linked list descriptor error.
- *SLCO/1_RX_DSCR_ERR_INT* : Slave receiving linked list descriptor error.
- *SLCO/1_RX_EOF_INT* : Slave receiving operation is finished.
- *SLCO/1_RX_DONE_INT* : A single buffer is received by the slave.
- *SLCO/1_TX_SUC_EOF_INT* : Slave transmitting operation is finished.
- *SLCO/1_TX_DONE_INT* : A single buffer is finished during transmitting operation.
- *SLCO/1_TX_OVF_INT* : Slave transmitting buffer overflow interrupt.
- *SLCO/1_RX_UDF_INT* : Slave receiving buffer underflow interrupt.
- *SLCO/1_TX_START_INT* : Slave transmitting start interrupt.

- *SLCO/1_RX_START_INT* : Slave receiving start interrupt.
- *SLC_FRHOST_BIT_n_INT* (*n*: 0–15): The host interrupts the slave via the SLC0 channel if interrupt vector Bit[7:0] is set or via the SLC1 channel if interrupt vector Bit[15:8] is set.

29.7 Packet Sending and Receiving Procedure

The SDIO host and slave devices need to follow specific data transfer procedures to successfully exchange data over the SDIO interface. Besides SDIO Specifications, ESP32-C5 should also follow the procedures below to transmit data over higher abstraction layers, such as Wi-Fi and Bluetooth data.

29.7.1 Sending Packets to SDIO Host

The transmission of packets from the slave to the host is initiated by the slave. The host will be notified with an interrupt (for detailed information on interrupts, please refer to SDIO Specification). After the host reads the relevant information from the slave, the host will initiate an SDIO bus transmission accordingly. The whole procedure is illustrated in Figure [29.7-1](#).

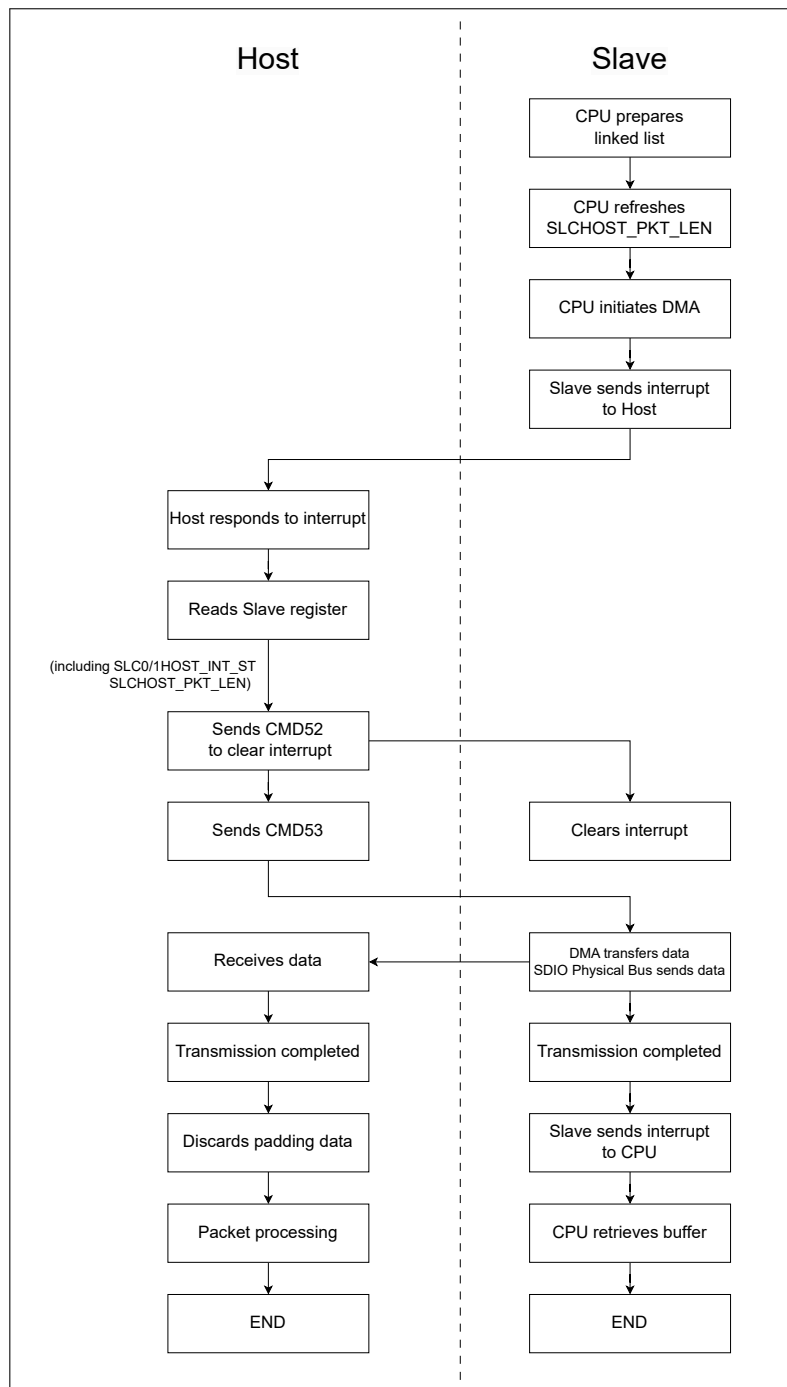


Figure 29.7-1. Procedure of Slave Sending Packets to Host

1. The slave CPU creates the linked list for the data packets that it will send to the host. For how to create a linked list, please refer to Section [29.5.5.1](#).
2. The slave CPU updates the length of data that will be sent to the host using the register [SDIO_SLC0_LEN_CONF_REG](#).
3. The slave CPU starts DMA by writing the 32-bit address of the first descriptor in linked list to [SDIO_SLCORX_LINK_ADDR_REG](#) or [SDIO_SLC1RX_LINK_ADDR_REG](#) and then configuring [SDIO_SLC0_RXLINK_START](#) or [SDIO_SLC1_RXLINK_START](#) to start DMA. For more information on DMA, please refer to Section [29.5.5](#).
4. The slave DMA sends an interrupt to the host.

5. After the host received the interrupt, it reads from [SLCHOST_SLC0HOST_INT_ST_REG](#), [SLCHOST_SLC1HOST_INT_ST_REG](#), and [SLCHOST_PKT_LEN_REG](#) for the following information:
 - [SLCHOST_SLC0/1HOST_INT_ST_REG](#): Interrupt status register. The [SLCHOST_SLC0/1_RX_NEW_PACKET_INT_ST](#) bit set to 1 indicates that cause of the interrupt is the slave sending packets.
 - [SLCHOST_PKT_LEN_REG](#): Packet length accumulator register. The current value minus the value of last time equals the packet length sent this time.
6. The host clears the interrupt by writing to the register [SLCHOST_SLC0/1HOST_INT_CLR_REG](#) after the CMD52 command.
7. The host fetches packets from the slave after the CMD53 command. During the transmission, when the slave determines that the valid data of the current packet is over, the subsequent bits will be padded with invalid data 0x0. For how to determine the end of valid data, please refer to Section [29.5.5.3](#).
8. After the packets are transmitted, the slave DMA sends an interrupt to the CPU, and the CPU can recycle the buffer at this time.

Notes:

- It is not recommended to set all of the eof bits to 0 in the linked list. Otherwise, the DMA may mistakenly send the next packet's data as part of the current command, causing an error. In cases where all of the eof bits are set to 0, the slave software should align the length of each packet to the size of the data block by padding data, to prevent this potential error. When the host sends CMD53 to read data, it should accurately control the number of data blocks in each packet, and not read more or less data blocks. Besides, the host should be able to identify the padded data.
- It is recommended that a CMD53 command should transmit only one data packet and each data packet should use only one linked list so as to avoid unknown exceptions caused by complex transmission.
- It is not recommended to send multiple packets through one linked list because it may complicate the host and software's ability to distinguish between the data packets. In cases where this has to be done, the software should set the eof bit when creating the linked list to split the data packets so that the DMA can pad data packets accordingly (the length of each data packet is recommended to be aligned to the size of the data block to avoid data padding by DMA), and the host should be able to identify the padded data.

29.7.2 Receiving Packets from SDIO Host

Transmission of packets from the host to slave is initiated by the host. The slave receives data via DMA and stores it in RAM. After transmission is completed, the CPU will be interrupted to process the data. The whole procedure is demonstrated in Figure [29.7-2](#).

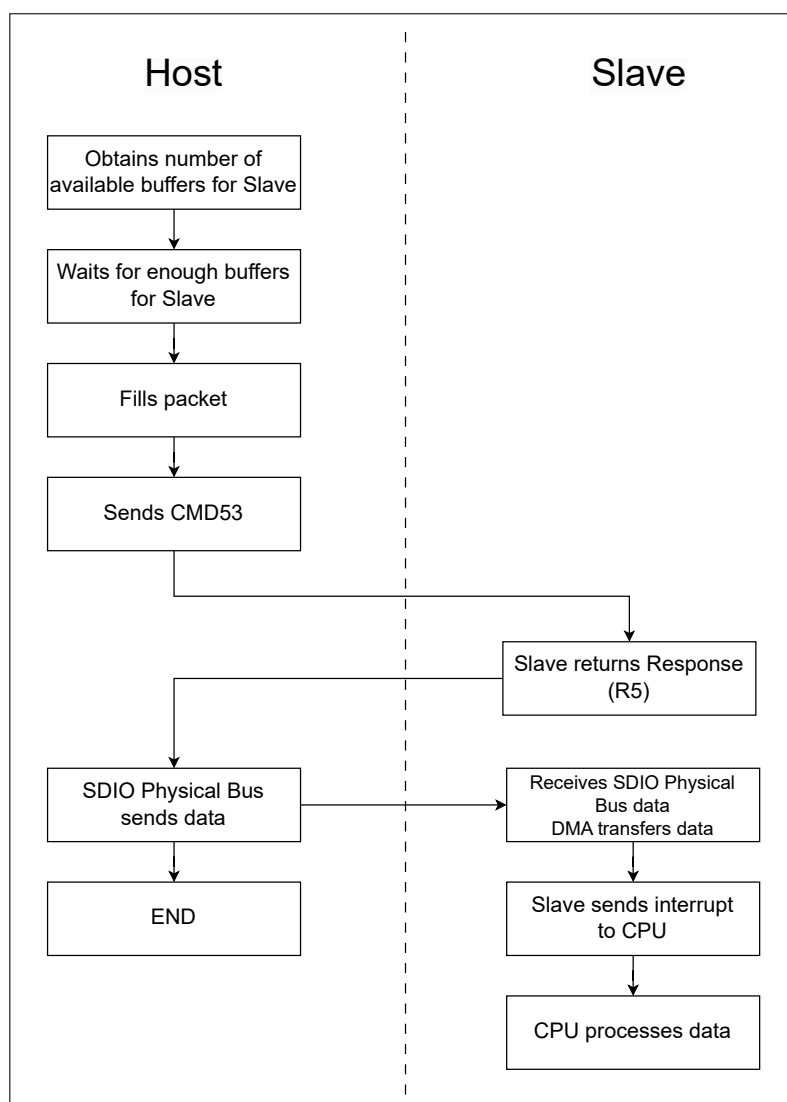


Figure 29.7-2. Procedure of Slave Receiving Packets from Host

The host obtains the number of available receiving buffers from the slave by accessing [SLCHOST_SLC0HOST_TOKEN_RDATA_REG](#) or [SLCHOST_SLC1HOST_TOKEN_RDATA_REG](#). The slave CPU should update the value of the register after the receiving DMA linked list is prepared.

[SLCHOST_HOSTSLCHOST_SLC0_TOKEN1](#) or [SLCHOST_HOSTSLCHOST_SLC1_TOKEN1](#) stores the accumulated number of available buffers. The host can figure out the available buffer space, using the register value minus the number of buffers already used. If the buffers are not enough, the host needs to constantly poll the register until there are enough buffers available.

During the transmission of packets to the slave through the CMD53 command, when a buffer specified by a linked list descriptor is written full, or when a packet transmission ends, the DMA will jump to the next buffer to store subsequent data. When the slave determines that the valid data of the current packet is over, the remaining data will be considered invalid and discarded. DMA will write back the current linked list descriptor, set the eof bit of the current descriptor to 1, and the [SLC0/1_TX_SUC_EOF_INT](#) interrupt will be generated.

For more information about DMA functions, linked list, and data discarding, please refer to Section [29.5.5](#).

To ensure sufficient receiving buffers, the slave CPU must constantly load buffers on the receiving linked list. The process is shown in Figure 29.7-3.

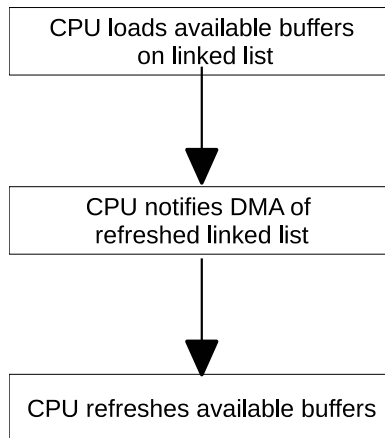


Figure 29.7-3. Loading Receiving Buffer

The CPU first needs to append new buffer segments at the end of the linked list that is being used by DMA and is available for receiving data.

The CPU then needs to notify the DMA that the linked list has been updated. This can be done by setting [SDIO_SLC0_TXLINK_RESTART](#) or [SDIO_SLC1_TXLINK_RESTART](#). Please note that when the CPU initiates DMA to receive packets for the first time, [SDIO_SLC0_TXLINK_START](#) or [SDIO_SLC1_TXLINK_START](#) should be set to 1.

Notes: Use the *_RESTART field to restart DMA only in the two scenarios:

- DMA is suspended by configuration of the *_STOP field. You can restart it after configuration of the *_RESTART field.
- DMA is suspended as a result of insufficient linked list descriptors. You can restart it by adding descriptors and configuring the *_RESTART field.

Lastly, the CPU refreshes any available buffer information by writing to the [SDIO_SLC0TOKEN1_REG](#) or [SDIO_SLC1TOKEN1_REG](#) register.

29.8 Register Summary

The addresses in this section are relative to **SDIO Slave Controller** base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

29.8.1 HINF Register Summary

Name	Description	Address	Access
Configuration registers			
HINF_CFG_DATA0_REG	SDIO CIS configuration	0x0000	R/W
HINF_CFG_DATA1_REG	SDIO configuration	0x0004	R/W
HINF_CFG_DATA7_REG	SDIO configuration	0x001C	varies
HINF_CIS_CONF_Wn_REG(n: 0-7)	SDIO CIS configuration	0x0020+0x4*n	R/W
HINF_CFG_DATA16_REG	SDIO CIS configuration	0x0040	R/W
Status registers			
HINF_CONF_STATUS_REG	SDIO CIS function 0 config0 status	0x0054	RO

29.8.2 SLC Register Summary

Name	Description	Address	Access
Configuration registers			
SDIO_SLCCONF0_REG	DMA configuration	0x0000	R/W
SDIO_SLCORX_LINK_REG	SCLO RX linked list configuration	0x003C	varies
SDIO_SLCORX_LINK_ADDR_REG	SCLO RX linked list address	0x0040	R/W
SDIO_SLCOTX_LINK_REG	SCLO TX linked list configuration	0x0044	varies
SDIO_SLCOTX_LINK_ADDR_REG	SCLO TX linked list address	0x0048	R/W
SDIO_SLC1RX_LINK_REG	SCL1 RX linked list configuration	0x004C	varies
SDIO_SLC1RX_LINK_ADDR_REG	SCL1 RX linked list address	0x0050	R/W
SDIO_SLC1TX_LINK_REG	SCL1 TX linked list configuration	0x0054	varies
SDIO_SLC1TX_LINK_ADDR_REG	SCL1 TX linked list address	0x0058	R/W
SDIO_SLC0TOKEN1_REG	SLC0 receiving buffer configuration	0x0064	varies
SDIO_SLC1TOKEN1_REG	SLC1 receiving buffer configuration	0x006C	varies
SDIO_SLCCONF1_REG	DMA configuration	0x0070	R/W
SDIO_SLC_RX_DSCR_CONF_REG	DMA slave to host configuration register	0x00A8	R/W
SDIO_SLC0_LEN_CONF_REG	Length control of transmitting packets	0x00F4	varies
SDIO_SLC0_TX_SHAREMEM_START_REG	SLC0 AHB TX start address range	0x0154	R/W
SDIO_SLC0_TX_SHAREMEM_END_REG	SLC0 AHB TX end address range	0x0158	R/W
SDIO_SLC0_RX_SHAREMEM_START_REG	SLC0 AHB RX start address range	0x015C	R/W
SDIO_SLC0_RX_SHAREMEM_END_REG	SLC0 AHB RX end address range	0x0160	R/W
SDIO_SLC1_TX_SHAREMEM_START_REG	SLC1 AHB TX start address range	0x0164	R/W
SDIO_SLC1_TX_SHAREMEM_END_REG	SLC1 AHB TX end address range	0x0168	R/W
SDIO_SLC1_RX_SHAREMEM_START_REG	SLC1 AHB RX start address range	0x016C	R/W

Name	Description	Address	Access
SDIO_SLC1_RX_SHAREMEM_END_REG	SLC1 AHB RX end address range	0x0170	R/W
SDIO_SLC_BURST_LEN_REG	DMA AHB burst type configuration	0x017C	R/W
Interrupt registers			
SDIO_SLC0INT_RAW_REG	SLC0 to slave raw interrupt status	0x0004	varies
SDIO_SLC0INT_ST_REG	SLC0 to slave masked interrupt status	0x0008	RO
SDIO_SLC0INT_ENA_REG	SLC0 to slave interrupt enable	0x000C	R/W
SDIO_SLC0INT_CLR_REG	SLC0 to slave interrupt clear	0x0010	WT
SDIO_SLC1INT_RAW_REG	SLC1 to slave raw interrupt status	0x0014	varies
SDIO_SLC1INT_CLR_REG	SLC1 to slave interrupt clear	0x0020	WT
SDIO_SLCINTVEC_TOHOST_REG	Slave to host interrupt vector set	0x005C	WT
SDIO_SLC1INT_ST1_REG	SLC1 to slave masked interrupt status	0x014C	RO
SDIO_SLC1INT_ENA1_REG	SLC1 to slave interrupt enable	0x0150	R/W
Status registers			
SDIO_SLC0_LENGTH_REG	Length of transmitting packets	0x00F8	RO

29.8.3 SLC Host Register Summary

Name	Description	Address	Access
Configuration registers			
SLCHOST_CONF_REG	Edge configuration	0x01F0	R/W
Interrupt registers			
SLCHOST_SLC0HOST_INT_RAW_REG	SLC0 to host raw interrupt status	0x0050	varies
SLCHOST_SLC1HOST_INT_RAW_REG	SLC1 to host raw interrupt status	0x0054	varies
SLCHOST_SLC0HOST_INT_ST_REG	SLC0 to host masked interrupt status	0x0058	RO
SLCHOST_SLC1HOST_INT_ST_REG	SLC1 to host masked interrupt status	0x005C	RO
SLCHOST_CONF_W7_REG	Host to slave interrupt vector set	0x008C	R/W
SLCHOST_SLC0HOST_INT_CLR_REG	SLC0 to host interrupt clear	0x00D4	WT
SLCHOST_SLC1HOST_INT_CLR_REG	SLC1 to host interrupt clear	0x00D8	WT
SLCHOST_SLC0HOST_FUNC1_INT_ENA_REG	SLC0 to host interrupt enable	0x00DC	R/W
SLCHOST_SLC1HOST_FUNC1_INT_ENA_REG	SLC0 to host interrupt enable	0x00E0	R/W
Status registers			
SLCHOST_SLC0HOST_TOKEN_RDATA_REG	Accumulated number of SLC0 receiving buffers	0x0044	RO
SLCHOST_PKT_LEN_REG	Length of the transmitting packets	0x0060	RO
SLCHOST_SLC1HOST_TOKEN_RDATA_REG	Accumulated number of SLC1 receiving buffers	0x00C4	RO
Communication Registers			
SLCHOST_CONF_Wn_REG(n: 0-2)	Host and slave communication	0x006C+0x4* n	R/W
SLCHOST_CONF_W3_REG	Host and slave communication	0x0078	R/W
SLCHOST_CONF_W4_REG	Host and slave communication	0x007C	R/W

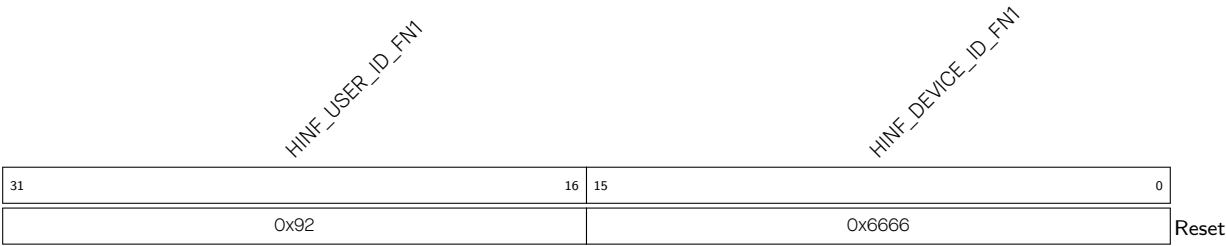
Name	Description	Address	Access
SLCHOST_CONF_W6_REG	Host and slave communication	0x0088	R/W
SLCHOST_CONF_W n _REG(n : 8-15)	Host and slave communication	0x009C+0x4*(n -8)	R/W

29.9 Registers

The addresses in this section are relative to **SDIO Slave Controller** base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

29.9.1 HINF Registers

Register 29.1. HINF_CFG_DATA0_REG (0x0000)



HINF_DEVICE_ID_FN1 Configures device ID of function 1 in SDIO CIS. (R/W)

HINF_USER_ID_FN1 Configures user ID of function 1 in SDIO CIS. (R/W)

Register 29.2. HINF_CFG_DATA1_REG (0x0004)

(reserved)				HINF_FUNC2_EPS				HINF_SDIO_VER				HINF_IOENABLE1				HINF_FUNC1_EPS				HINF_CD_DISABLE				HINF_IOENABLE2				HINF_SDIO_IOREADY2				HINF_HIGH SPEED MODE				HINF_HIGH SPEED_ENABLE				(reserved)			
31	25	24	23									12	11	10	9	8	7	6	5	4	3	2	1	0																			
0			0	0x232								0	0	0	0	0	0	0	0	1	0	0	0	1	Reset																		

HINF_SDIO_IOREADY1 Configures the field IOR1 in SDIO CCCR and the field function 1 ready in SDIO CIS.

0: The function 1 is not ready

1: The function 1 is ready

Please refer to SDIO Specification for details.

(R/W)

HINF_HIGH_SPEED_ENABLE Configures whether to support SHS in SDIO CCCR.

0: Not support High-Speed mode

1: Support High-Speed mode

Please refer to SDIO Specification for details.

(R/W)

HINF_HIGH_SPEED_MODE Represents whether EHS status is enabled in SDIO CCCR.

0: Disabled

1: Enabled

Please refer to SDIO Specification for details.

(RO)

HINF_SDIO_CD_ENABLE Configures whether to enable SDIO card detection.

0: Disable

1: Enable

(R/W)

HINF_SDIO_IOREADY2 Configures the field IOR2 in SDIO CCCR and the field function 2 ready in SDIO CIS.

0: The function 2 is not ready

1: The function 2 is ready

Please refer to SDIO Specification for details.

(R/W)

HINF_IOENABLE2 Represents whether IOE2 is enabled in SDIO CCCR.

0: The function 2 is disabled

1: The function 2 is enabled

Please refer to SDIO Specification for details.

(RO)

Continued on the next page...

Register 29.2. HINF_CFG_DATA1_REG (0x0004)

Continued from the previous page...

HINF_CD_DISABLE Represents whether CD is disabled in SDIO CCCR.

0: Enabled

1: Disabled

Please refer to SDIO Specification for details.

(RO)

HINF_FUNC1_EPS Represents function 1 EPS status in SDIO FBR.

0: The function 1 operates in Higher Current Mode

1: The function 1 works in Lower Current Mode

Please refer to SDIO Specification for details.

(RO)

HINF_EMP Represents EMPC status in SDIO CCCR.

0: Master Power Control is disabled

1: Master Power Control is enabled

Please refer to SDIO Specification for details.

(RO)

HINF_IOENABLE1 Represents IOE1 status in SDIO CCCR.

0: The function 1 is disabled

1: The function 1 is enabled

Please refer to SDIO Specification for details.

(RO)

HINF_SDIO_VER Configures SD bit[3:0], SDIO bit[3:0], CCCR bit[3:0] in SDIO CCCR.

HINF_SDIO_VER[11:8] mapping to SD bit[3:0]

HINF_SDIO_VER[7:4] mapping to SDIO bit[3:0]

HINF_SDIO_VER[3:0] mapping to CCCR bit[3:0]

Please refer to SDIO Specification for details.

(R/W)

HINF_FUNC2_EPS Represents function 2 EPS status in SDIO FBR.

0: The function 2 operates in Higher Current Mode

1: The function 2 works in Lower Current Mode

Please refer to SDIO Specification for details.

(RO)

Register 29.3. HINF_CFG_DATA7_REG (0x001C)

31		30		29																20				19	18	17	16	15	8													7	0															
0		0		0x238																0				0x1	0	0x0													0x0																Reset			

HINF_PIN_STATE Configures SDIO CIS address 318 and 574. Please refer to SDIO Specification for details. (R/W)

HINF_CHIP_STATE Configures SDIO CIS address 312, 315, 568, and 571. Please refer to SDIO Specification for details. (R/W)

HINF_SDIO_RST Configures whether to reset the SDIO slave module.

0: No effect

1: Reset

(R/W)

HINF_ESDIO_DATA1_INT_EN Configures whether to enable SDIO interrupt on data1 line.

0: Disable

1: Enable

(R/W)

HINF_SDIO_WAKEUP_CLR Configures whether to clear wake up signal after the chip is waken up by the SDIO slave.

0: No effect

1: Clear

(WT)

Register 29.4. HINF_CIS_CONF_Wn_REG(*n*: 0-7) (0x0020+0x4**n*)

Diagram illustrating the structure of the `HNF_CIS_CONF_Wn` register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The value `0xffffffff` is shown in the register box. The label `HNF_CIS_CONF_Wn` is written diagonally above the register box, with a red `n` indicating a bit field.

HINF_CIS_CONF_W_n Configures SDIO CIS address $(39+4*n) \sim (36+4*n)$. Please refer to SDIO Specification for details. (R/W)

Register 29.5. HINF_CFG_DATA16_REG (0x0040)

<i>HINF_USER_ID_FN2</i>																<i>HINF_DEVICE_ID_FN2</i>																																											
31																16	15														0																												
0x92																0x7777																																											
Reset																																																											

HINF_DEVICE_ID_FN2 Configures device ID of function 2 in SDIO CIS. (R/W)

HINF_USER_ID_FN2 Configures user ID of function 2 in SDIO CIS. (R/W)

Register 29.6. HINF_CONF_STATUS_REG (0x0054)

(reserved)																HINF_FUNC0_CONFIG0																
31																8	7														0	
0x0																0x0																Reset

HINF_FUNC0_CONFIG0 Represents SDIO CIS function 0 config0 (addr: 0x20f0) status. Please refer to SDIO Specification for details. (RO)

29.9.2 SLC Registers

Register 29.7. SDIO_SLCCONFO_REG (0x0000)

(reserved) SDIO_SLC1_TOKEN_AUTO_CLR SDIO_SLC1_TXDATA_BURST_EN SDIO_SLC1_TXDSCR_BURST_EN (reserved) SDIO_SLC1_RXDATA_BURST_EN SDIO_SLC1_RXDSCR_BURST_EN SDIO_SLC1_RX_NO_RESTART_CLR SDIO_SLC1_RX_AUTO_WBACK SDIO_SLC1_TX_LOOP_TEST (reserved) SDIO_SLC1_RX_RST SDIO_SLC1_TX_RST (reserved) SDIO_SLC0_TOKEN_AUTO_CLR SDIO_SLC0_TXDATA_BURST_EN SDIO_SLC0_TXDSCR_BURST_EN (reserved) SDIO_SLC0_RXDATA_BURST_EN SDIO_SLC0_RXDSCR_BURST_EN SDIO_SLC0_RX_NO_RESTART_CLR SDIO_SLC0_RX_AUTO_WBACK SDIO_SLC0_TX_LOOP_TEST (reserved) SDIO_SLC0_RX_RST SDIO_SLC0_TX_RST																																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	1	1	1	Ox3	1	1	O	O	1	1	Ox3	O	O	1	1	1	1	Ox3	1	1	O	O	O	O	O	O	O	OxO	O	O	Reset	

Reset

SDIO_SLC0_TX_RST Configures whether to reset TX (host to slave) FSM (finite state machine) in SLC0.

0: No effect

1: Reset

(R/W)

SDIO_SLC0_RX_RST Configures whether to reset RX (slave to host) FSM in SLC0.

0: No effect

1: Reset

(R/W)

SDIO_SLC0_TX_LOOP_TEST Configures whether SLC0 loops around when the slave buffer finishes receiving packets from the host.

0: Not loop around

1: Loop around, and hardware will not change the owner bit in the linked list

(R/W)

SDIO_SLC0_RX_LOOP_TEST Configures whether SLC0 loops around when the slave buffer finishes sending packets to the host.

0: Not loop around

1: Loop around, and hardware will not change the owner bit in the linked list

(R/W)

SDIO_SLC0_RX_AUTO_WBACK Configures whether SLC0 changes the owner bit of RX linked list.

0: Not change

1: Change

(R/W)

SDIO_SLC0_RX_NO_RESTART_CLR Please initialize to 1, and do not modify it. (R/W)

SDIO_SLC0_RXDSCR_BURST_EN Configures whether SLC0 can use AHB burst operation when reading the RX linked list from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

Continued on the next page...

Register 29.7. SDIO_SLCCONFO_REG (0x0000)

Continued from the previous page...

SDIO_SLC0_RXDATA_BURST_EN Configures whether SCL0 can use AHB burst operation when read data from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC0_TXDSCR_BURST_EN Configures whether SCL0 can use AHB burst operation when read the TX linked list from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC0_TXDATA_BURST_EN Configures whether SCL0 can use AHB burst operation when send data to memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC0_TOKEN_AUTO_CLR Please initialize to 0, and do not modify it. (R/W)

SDIO_SLC1_TX_RST Configures whether to reset TX FSM in SLC1.

0: No effect

1: Reset

(R/W)

SDIO_SLC1_RX_RST Configures whether to reset RX FSM in SLC1.

0: No effect

1: Reset

(R/W)

SDIO_SLC1_TX_LOOP_TEST Configures whether SCL1 loops around when the slave buffer finishes receiving packets from the host.

0: Not loop around

1: Loop around, and hardware will not change the owner bit in the linked list

(R/W)

SDIO_SLC1_RX_LOOP_TEST Configures whether SCL1 loops around when the slave buffer finishes sending packets to the host.

0: Not loop around

1: Loop around, and hardware will not change the owner bit in the linked list

(R/W)

Continued on the next page...

Register 29.7. SDIO_SLCCONFO_REG (0x0000)

Continued from the previous page...

SDIO_SLC1_RX_AUTO_WRBK Configures whether SCL1 changes the owner bit of the RX linked list.

0: Not change

1: Change

(R/W)

SDIO_SLC1_RX_NO_RESTART_CLR Please initialize to 1, and do not modify it. (R/W)

SDIO_SLC1_RXDSCR_BURST_EN Configures whether SCL1 can use AHB burst operation when read the RX linked list from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC1_RXDATA_BURST_EN Configures whether SCL1 can use AHB burst operation when reading data from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC1_TXDSCR_BURST_EN Configures whether SCL1 can use AHB burst operation when read the TX linked list from memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC1_TXDATA_BURST_EN Configures whether SCL1 can use AHB burst operation when send data to memory.

0: Only use single operation

1: Can use burst operation

(R/W)

SDIO_SLC1_TOKEN_AUTO_CLR Please initialize to 0, and do not modify it. (R/W)

Register 29.8. SDIO_SLCORX_LINK_REG (0x003C)

31	30	29	28	27	0

1	0	0	0	0x0	Reset

SDIO_SLC0_RXLINK_STOP Configures whether to stop SLC0 RX linked list operation.

0: No effect

1: Stop the operation

(R/W/SC)

SDIO_SLC0_RXLINK_START Configures whether to start SLC0 RX linked list operation from the address indicated by SDIO_SLC0_RXLINK_ADDR.

0: No effect

1: Start the operation

(R/W/SC)

SDIO_SLC0_RXLINK_RESTART Configures whether to restart and continue SLC0 RX linked list operation.

0: No effect

1: Restart the operation

(R/W/SC)

SDIO_SLC0_RXLINK_PARK Represents SLC0 RX linked list FSM state.

0: The FSM not in idle state

1: The FSM in idle state

(RO)

Register 29.9. SDIO_SLCORX_LINK_ADDR_REG (0x0040)

Diagram of the SDIO_SLO_RXLINK_ADDR register. The register is 32 bits wide, with bit 31 on the left and bit 0 on the right. The register is labeled "SDIO_SLO_RXLINK_ADDR" and has a "Reset" value of "0x0".

SDIO_SLC0_RXLINK_ADDR Configures SLC0 RX linked list initial address. (R/W)

Register 29.10. SDIO_SLCOTX_LINK_REG (0x0044)

31	30	29	28	27	0
					0x0
					Reset

SDIO_SLC0_TXLINK_STOP Configures whether to stop SLC0 TX linked list operation.

0: No effect

1: Stop the operation

(R/W/SC)

SDIO_SLC0_TXLINK_START Configures whether to start SLC0 TX linked list operation from the address indicated by SDIO_SLC0_TXLINK_ADDR.

0: No effect

1: Start the operation

(R/W/SC)

SDIO_SLC0_TXLINK_RESTART Configures whether to restart and continue SLC0 TX linked list operation.

0: No effect

1: Restart the operation

(R/W/SC)

SDIO_SLC0_TXLINK_PARK Represents SLC0 TX linked list FSM state.

0: The FSM not in idle state

1: The FSM in idle state

(RO)

Register 29.11. SDIO_SLCOTX_LINK_ADDR_REG (0x0048)

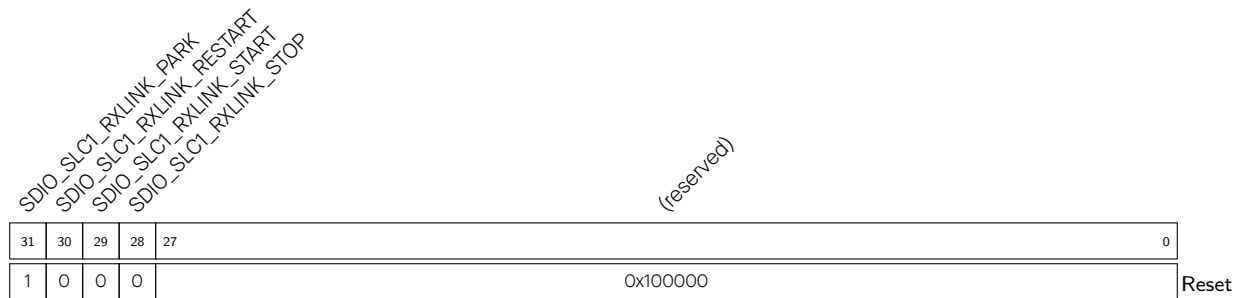
SDIO_SLO_TYLINK_ADDR

31 0

0x0 Reset

SDIO_SLC0_TXLINK_ADDR Configures SLC0 TX linked list initial address. (R/W)

Register 29.12. SDIO_SLC1RX_LINK_REG (0x004C)



SDIO_SLC1_RXLINK_STOP Configures whether to stop SLC1 RX linked list operation.

0: No effect

1: Stop the operation

(R/W/SC)

SDIO_SLC1_RXLINK_START Configures whether to start SLC1 RX linked list operation from the address indicated by SDIO_SLC1_RXLINK_ADDR.

0: No effect

1: Start the operation

(R/W/SC)

SDIO_SLC1_RXLINK_RESTART Configures whether to restart and continue SLC1 RX linked list operation.

0: No effect

1: Restart the operation

(R/W/SC)

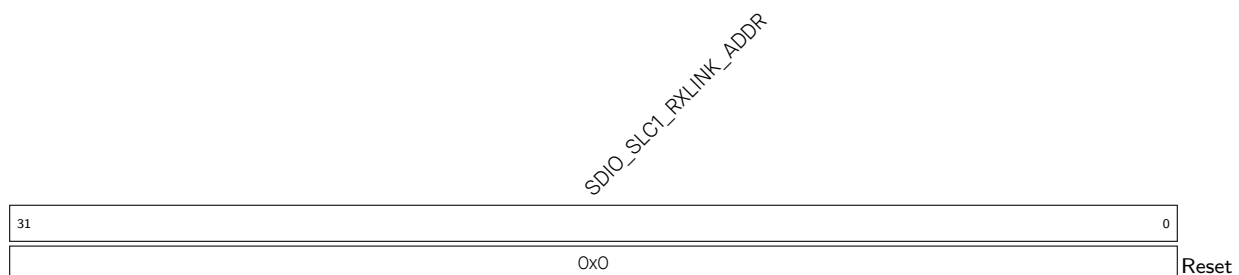
SDIO_SLC1_RXLINK_PARK Represents SLC1 RX linked list FSM state.

0: The FSM not in idle state

1: The FSM in idle state

(RO)

Register 29.13. SDIO_SLC1RX_LINK_ADDR_REG (0x0050)



SDIO_SLC1_RXLINK_ADDR Configures SLC1 RX linked list initial address. (R/W)

Register 29.14. SDIO_SLC1TX_LINK_REG (0x0054)

SDIO_SLC1_TXLINK_STOP
SDIO_SLC1_TXLINK_START
SDIO_SLC1_TXLINK_RESTART
SDIO_SLC1_TXLINK_PARK
(reserved)

31	30	29	28	27	0
1	0	0	0	0x0	Reset

SDIO_SLC1_TXLINK_STOP Configures whether to stop SLC1 TX linked list operation.

0: No effect

1: Stop the operation

(R/W/SC)

SDIO_SLC1_TXLINK_START Configures whether to start SLC1 TX linked list operation from the address indicated by SDIO_SLC1_TXLINK_ADDR.

0: No effect

1: Start the operation

(R/W/SC)

SDIO_SLC1_TXLINK_RESTART Configures whether to restart and continue SLC1 TX linked list operation.

0: No effect

1: Restart the operation

(R/W/SC)

SDIO_SLC1_TXLINK_PARK Represents SLC1 TX linked list FSM state.

0: The FSM not in idle state

1: The FSM in idle state

(RO)

Register 29.15. SDIO_SLC1TX_LINK_ADDR_REG (0x0058)

SDIO_SLC1_TXLINK_ADDR

31 0

0x0 Reset

SDIO_SLC1_TXLINK_ADDR Configures SLC1 TX linked list initial address. (R/W)

Register 29.16. SDIO_SLC0TOKEN1_REG (0x0064)

(reserved)				SDIO_SLC0_TOKEN1				(reserved)				SDIO_SLC0_TOKEN1_INC_MORE SDIO_SLC0_TOKEN1_INC SDIO_SLC0_TOKEN1_WR				SDIO_SLC0_TOKEN1_WDATA										
31	28	27												16	15	14	13	12	11					0		
0x0		0x0														0	0	0	0	0x0						Reset

SDIO_SLC0_TOKEN1_WDATA Configures SLC0 token 1 value. (WT)

SDIO_SLC0_TOKEN1_WR Configures this bit to 1 to write SDIO_SLC0_TOKEN1_WDATA into SDIO_SLC0_TOKEN1. (WT)

SDIO_SLC0_TOKEN1_INC Configures this bit to 1 to add 1 to SDIO_SLC0_TOKEN1. (WT)

SDIO_SLC0_TOKEN1_INC_MORE Configures this bit to 1 to add the value of SDIO_SLC0_TOKEN1_WDATA to SDIO_SLC0_TOKEN1. (WT)

SDIO_SLC0_TOKEN1 Represents the SLC0 accumulated number of buffers for receiving packets. (RO)

Register 29.17. SDIO_SLC1TOKEN1_REG (0x006C)

(reserved)				SDIO_SLC1_TOKEN1				(reserved)				SDIO_SLC1_TOKEN1_INC_MORE SDIO_SLC1_TOKEN1_INC SDIO_SLC1_TOKEN1_WR				SDIO_SLC1_TOKEN1_WDATA																
31	28	27												16	15	14	13	12	11												0	
0x0				0x0												0	0	0	0	0x0												Reset

SDIO_SLC1_TOKEN1_WDATA Configures SLC1 token1 value. (WT)

SDIO_SLC1_TOKEN1_WR Configures this bit to 1 to write SDIO_SLC1_TOKEN1_WDATA into SDIO_SLC1_TOKEN1. (WT)

SDIO_SLC1_TOKEN1_INC Configures this bit to 1 to add 1 to SDIO_SLC1_TOKEN1. (WT)

SDIO_SLC1_TOKEN1_INC_MORE Configures this bit to 1 to add the value of SDIO_SLC1_TOKEN1_WDATA to SDIO_SLC1_TOKEN1. (WT)

SDIO_SLC1_TOKEN1 Represents SLC1 accumulated number of buffers for receiving packets. (RO)

Register 29.18. SDIO_SLCCONF1_REG (0x0070)

(reserved)																SDIO_SLC1_RX_STTCH_EN				SDIO_SLC1_TX_STTCH_EN				SDIO_HOST_INT_LEVEL_SEL				(reserved)																SDIO_SLC0_RX_STTCH_EN				SDIO_SLC0_TX_STTCH_EN				SDIO_SLC0_LEN_AUTO_CLR				SDIO_SDIO_CMD_HOLD_EN				(reserved)															
31																22						21	20	19	18	7																6	5	4	3	2	0																												
0x0																1	1	0	0x0																1	1	1	1	0x0																Reset																				

SDIO_SDIO_CMD_HOLD_EN Please initialize to 0, and do not modify it. (R/W)

SDIO_SLC0_LEN_AUTO_CLR Please initialize to 0, and do not modify it. (R/W)

SDIO_SLC0_TX_STITCH_EN Please initialize to 0, and do not modify it. (R/W)

SDIO_SLCO_RX_STITCH_EN Please initialize to 0, and do not modify it. (R/W)

SDIO_HOST_INT_LEVEL_SEL Configures the polarity of interrupt to host.

0: Low active

1: High active

(R/W)

SDIO_SLC1_TX_STITCH_EN Please initialize to 0, and do not modify it. (R/W)

SDIO_SLC1_RX_STITCH_EN Please initialize to 0, and do not modify it. (R/W)

Register 29.19. SDIO_SLC_RX_DSCR_CONF_REG (0x00A8)

31	1	0
0x101b80d	0	Reset

SDIO_SLCO_TOKEN_NO_REPLACE Please initialize to 1, and do not modify it. (R/W)

Register 29.20. SDIO_SLC0_LEN_CONF_REG (0x00F4)

(reserved)										SDIO_SLC0_LEN_INC_MORE										SDIO_SLC0_LEN_INC										SDIO_SLC0_LEN_WR										SDIO_SLC0_LEN_WDATA																													
31										23										22										21										20										19										0									
0x20										0										0										0										0x0										Reset																			

SDIO_SLC0_LEN_WDATA Configures the length of the data that the slave wants to send. (WT)

SDIO_SLC0_LEN_WR Configures this bit to 1 to write SDIO_SLC0_LEN_WDATA into SDIO_SLC0_LEN and SLCHOST_HOSTSLCHOST_SLC0_LEN. (WT)

SDIO_SLCO_LEN_INC Configures this bit to 1 to add 1 to SDIO_SLCO_LEN and SLCHOST_HOSTSLCHOST_SLCO_LEN. (WT)

SDIO_SLCO_LEN_INC_MORE Configures this bit to 1 to add the value of SDIO_SLCO_LEN_WDATA to SDIO_SLCO_LEN and SLCHOST_HOSTSLCHOST_SLCO_LEN. (WT)

Register 29.21. SDIO_SLC0_TX_SHAREMEM_START_REG (0x0154)

SDIO_SHAREMEM_START_ADDR

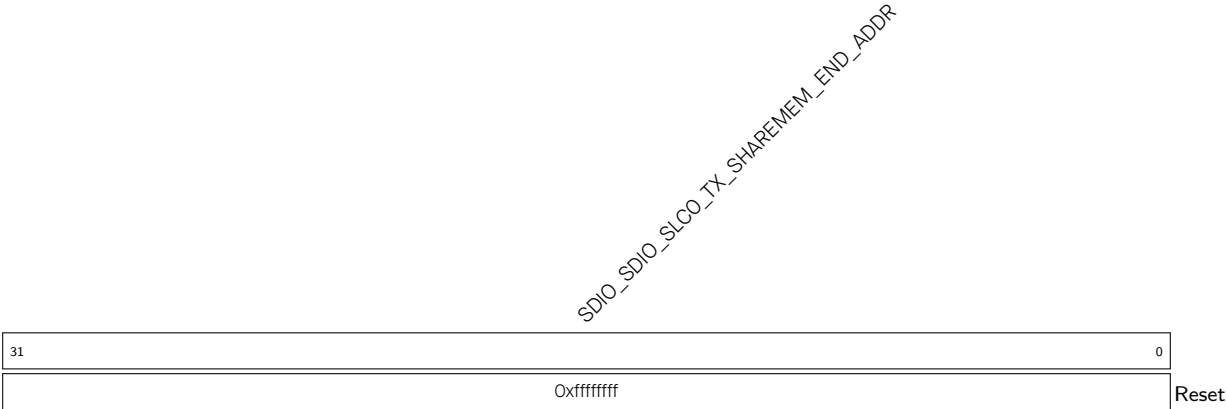
31 0

0x0

Reset

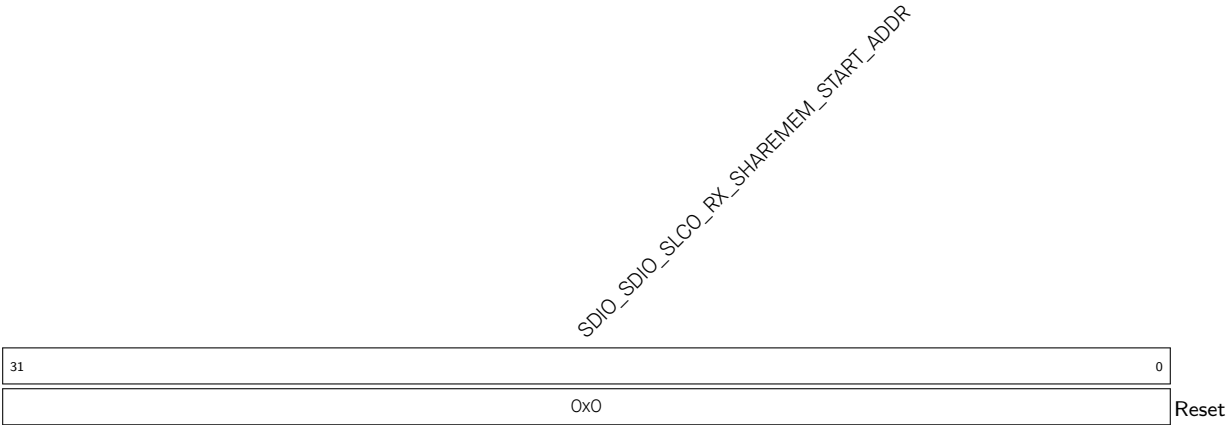
SDIO_SDIO_SLCO_TX_SHAREMEM_START_ADDR	Configures SLCO host to slave channel AHB start address boundary. (R/W)

Register 29.22. SDIO_SLC0_TX_SHAREMEM_END_REG (0x0158)



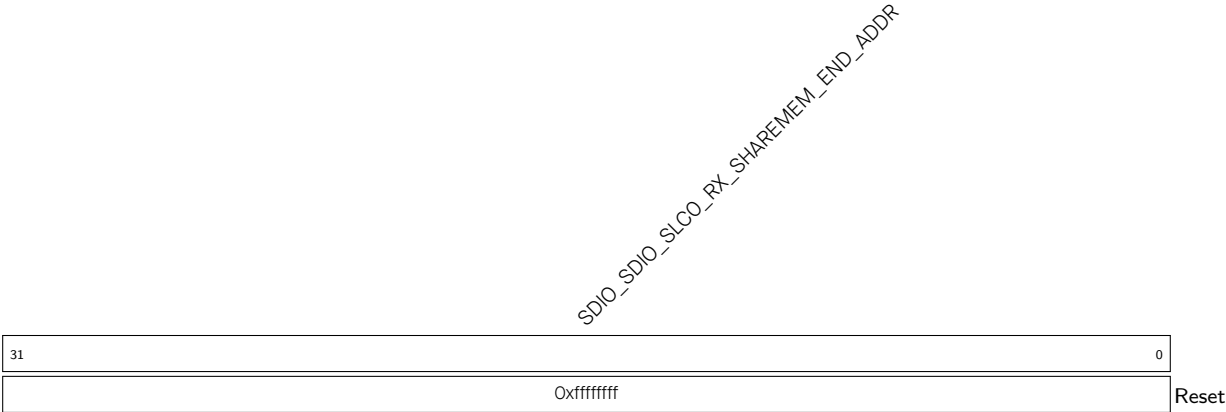
SDIO_SDIO_SLC0_TX_SHAREMEM_END_ADDR Configures SLC0 host to slave channel AHB end address boundary. (R/W)

Register 29.23. SDIO_SLC0_RX_SHAREMEM_START_REG (0x015C)



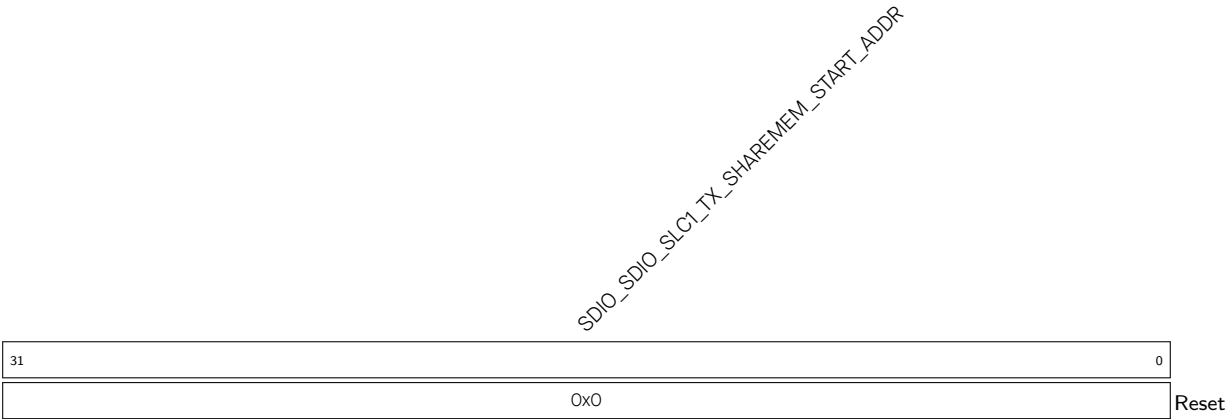
SDIO_SDIO_SLC0_RX_SHAREMEM_START_ADDR Configures SLC0 slave to host channel AHB start address boundary. (R/W)

Register 29.24. SDIO_SLC0_RX_SHAREMEM_END_REG (0x0160)



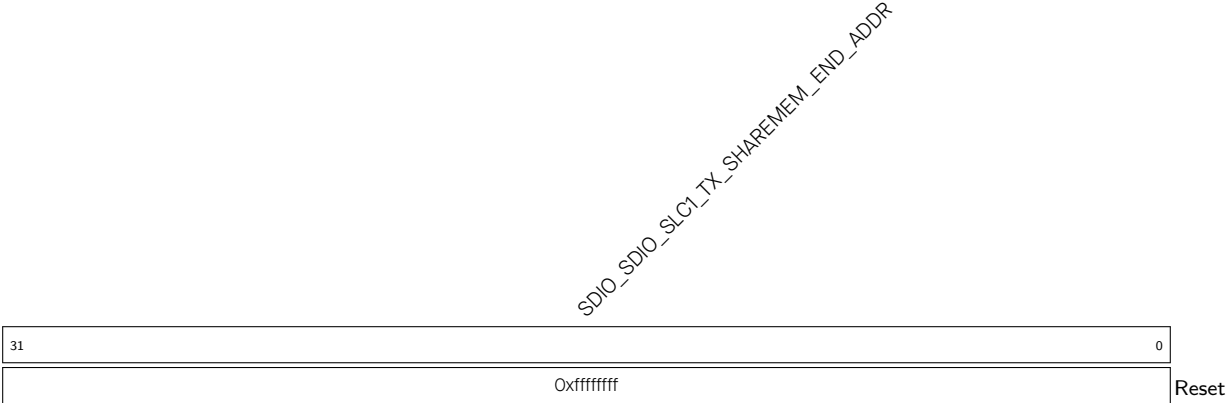
SDIO_SDIO_SLC0_RX_SHAREMEM_END_ADDR Configures SLC0 slave to host channel AHB end address boundary. (R/W)

Register 29.25. SDIO_SLC1_TX_SHAREMEM_START_REG (0x0164)



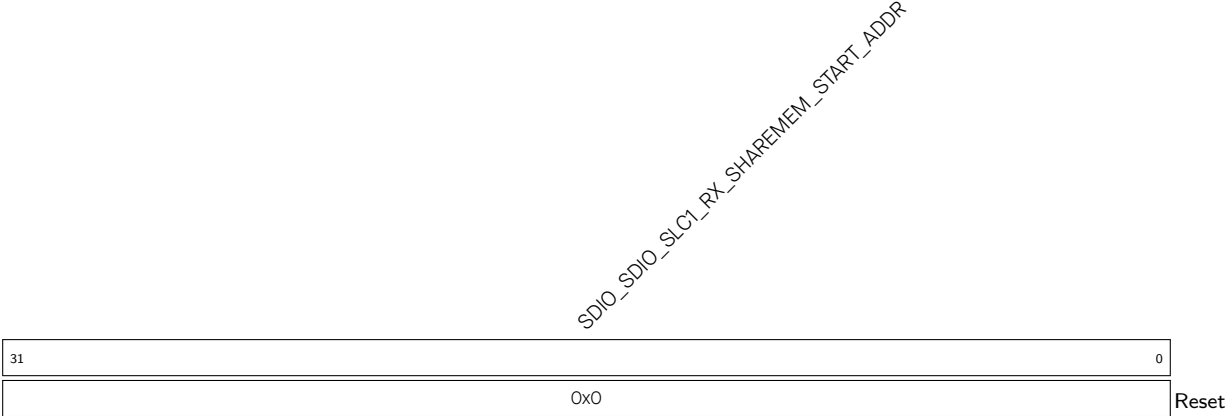
SDIO_SDIO_SLC1_TX_SHAREMEM_START_ADDR Configures SLC1 host to slave channel AHB start address boundary. (R/W)

Register 29.26. SDIO_SLC1_TX_SHAREMEM_END_REG (0x0168)



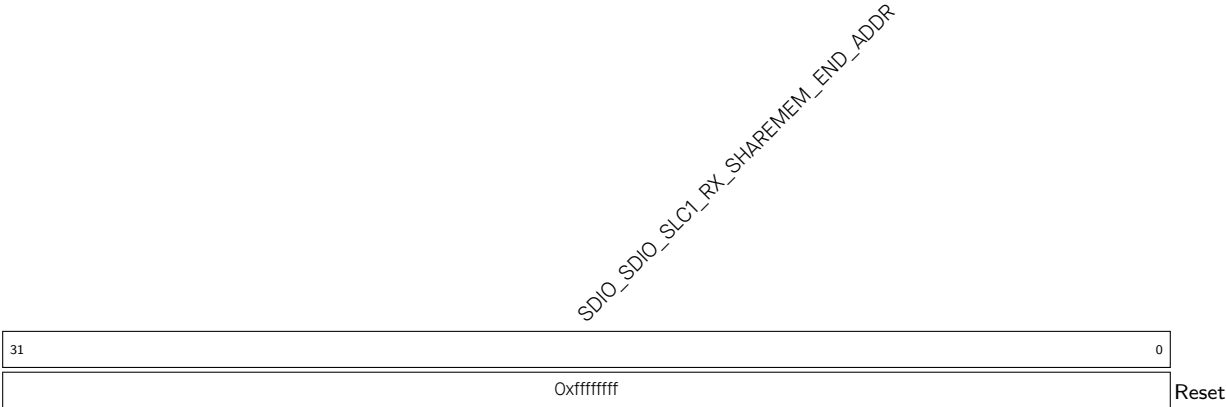
SDIO_SDIO_SLC1_TX_SHAREMEM_END_ADDR Configures SLC1 host to slave channel AHB end address boundary. (R/W)

Register 29.27. SDIO_SLC1_RX_SHAREMEM_START_REG (0x016C)



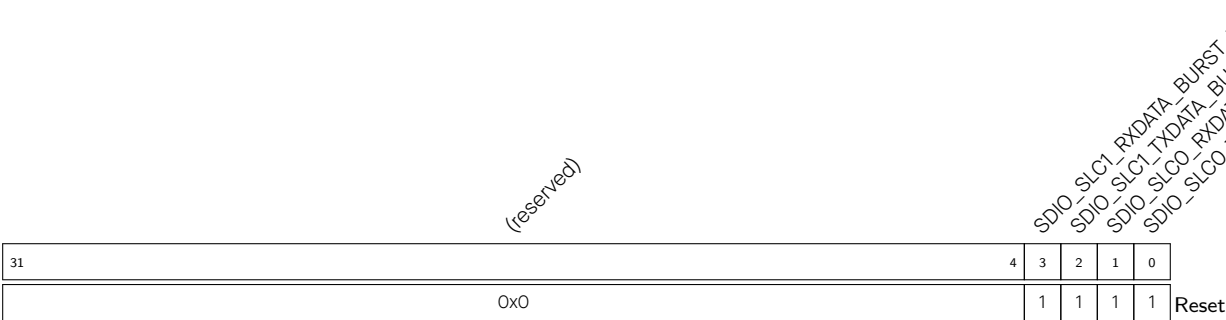
SDIO_SDIO_SLC1_RX_SHAREMEM_START_ADDR Configures SLC1 slave to host channel AHB start address boundary. (R/W)

Register 29.28. SDIO_SLC1_RX_SHAREMEM_END_REG (0x0170)



SDIO_SDIO_SLC1_RX_SHAREMEM_END_ADDR Configures SLC1 slave to host channel AHB end address boundary. (R/W)

Register 29.29. SDIO_SLC_BURST_LEN_REG (0x017C)



SDIO_SLC0_TXDATA_BURST_LEN Configures SLC0 host to slave channel AHB burst type.

- 0: Can use incr4
- 1: Can use incr8

(R/W)

SDIO_SLC0_RXDATA_BURST_LEN Configures SLC0 slave to host channel AHB burst type.

- 0: Can use incr and incr4
- 1: Can use incr and incr8

(R/W)

SDIO_SLC1_TXDATA_BURST_LEN Configures SLC1 host to slave channel AHB burst type.

- 0: Can use incr4
- 1: Can use incr8

(R/W)

SDIO_SLC1_RXDATA_BURST_LEN Configures SLC1 slave to host channel AHB burst type.

- 0: Can use incr and incr4
- 1: Can use incr and incr8

(R/W)

Register 29.30. SDIO_SLC0INT_RAW_REG (0x0004)

(reserved)												SDIO_SLC0_RX_DSCR_ERR_INT_RAW SDIO_SLC0_TX_DSCR_ERR_INT_RAW (reserved) SDIO_SLC0_RX_EOF_INT_RAW SDIO_SLC0_TX_DONE_INT_RAW SDIO_SLC0_TX_SUC_EOF_INT_RAW (reserved) SDIO_SLC0_TX_OVF_INT_RAW SDIO_SLC0_RX_UDF_INT_RAW SDIO_SLC0_TX_START_INT_RAW SDIO_SLC0_RX_START_INT_RAW SDIO_SLC0_FRHOST_BIT7_INT_RAW SDIO_SLC0_FRHOST_BIT6_INT_RAW SDIO_SLC0_FRHOST_BIT5_INT_RAW SDIO_SLC0_FRHOST_BIT4_INT_RAW SDIO_SLC0_FRHOST_BIT3_INT_RAW SDIO_SLC0_FRHOST_BIT2_INT_RAW SDIO_SLC0_FRHOST_BIT1_INT_RAW SDIO_SLC0_FRHOST_BIT0_INT_RAW																							
31											21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
0x0												0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SDIO_SLC_FRHOST_BIT n _INT_RAW (n : 0-7) The raw interrupt status of [SLC_FRHOST_BIT \$n\$ _INT \(\$n\$: 0-7\)](#). (R/WTC/SS)

SDIO_SLC0_RX_START_INT_RAW The raw interrupt status of [SLCO_RX_START_INT](#). (R/WTC/SS)

SDIO_SLC0_TX_START_INT_RAW The raw interrupt status of [SLCO_TX_START_INT](#). (R/WTC/SS)

SDIO_SLC0_RX_UDF_INT_RAW The raw interrupt status of [SLCO_RX_UDF_INT](#). (R/WTC/SS)

SDIO_SLC0_TX_OVF_INT_RAW The raw interrupt status of [SLCO_TX_OVF_INT](#). (R/WTC/SS)

SDIO_SLC0_TX_DONE_INT_RAW The raw interrupt status of [SLCO_TX_DONE_INT](#). (R/WTC/SS)

SDIO_SLC0_TX_SUC_EOF_INT_RAW The raw interrupt status of [SLCO_TX_SUC_EOF_INT](#). (R/WTC/SS)

SDIO_SLC0_RX_DONE_INT_RAW The raw interrupt status of [SLCO_RX_DONE_INT](#). (R/WTC/SS)

SDIO_SLC0_RX_EOF_INT_RAW The raw interrupt status of [SLCO_RX_EOF_INT](#). (R/WTC/SS)

SDIO_SLC0_TX_DSCR_ERR_INT_RAW The raw interrupt status of [SLCO_TX_DSCR_ERR_INT](#). (R/WTC/SS)

SDIO_SLC0_RX_DSCR_ERR_INT_RAW The raw interrupt status of [SLCO_RX_DSCR_ERR_INT](#). (R/WTC/SS)

Register 29.31. SDIO_SLCOINT_ST_REG (0x0008)

(reserved)																							SDIO_SLCO_RX_DSCR_ERR_INT_ST SDIO_SLCO_TX_DSCR_ERR_INT_ST (reserved) SDIO_SLCO_RX_EOF_INT_ST SDIO_SLCO_RX_DONE_INT_ST SDIO_SLCO_TX_SUC_EOF_INT_ST SDIO_SLCO_TX_DONE_INT_ST (reserved) SDIO_SLCO_TX_OVF_INT_ST SDIO_SLCO_RX_UDF_INT_ST SDIO_SLCO_TX_START_INT_ST SDIO_SLC_FRHOST_BIT7_INT_ST SDIO_SLC_FRHOST_BIT6_INT_ST SDIO_SLC_FRHOST_BIT5_INT_ST SDIO_SLC_FRHOST_BIT4_INT_ST SDIO_SLC_FRHOST_BIT3_INT_ST SDIO_SLC_FRHOST_BIT2_INT_ST SDIO_SLC_FRHOST_BIT1_INT_ST SDIO_SLC_FRHOST_BIT0_INT_ST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																							21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0x0																							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDIO_SLC_FRHOST_BIT n _INT_ST (n : 0-7) The masked interrupt status of [SLC_FRHOST_BIT \$n\$ _INT](#) (n : 0-7). (RO)

SDIO_SLCO_RX_START_INT_ST The masked interrupt status of [SLCO_RX_START_INT](#). (RO)

SDIO_SLCO_TX_START_INT_ST The masked interrupt status bit of [SLCO_TX_START_INT](#). (RO)

SDIO_SLCO_RX_UDF_INT_ST The masked interrupt status of [SLCO_RX_UDF_INT](#). (RO)

SDIO_SLCO_TX_OVF_INT_ST The masked interrupt status of [SLCO_TX_OVF_INT](#). (RO)

SDIO_SLCO_TX_DONE_INT_ST The masked interrupt status of [SLCO_TX_DONE_INT](#). (RO)

SDIO_SLCO_TX_SUC_EOF_INT_ST The masked interrupt status of [SLCO_TX_SUC_EOF_INT](#). (RO)

SDIO_SLCO_RX_DONE_INT_ST The masked interrupt status of [SLCO_RX_DONE_INT](#). (RO)

SDIO_SLCO_RX_EOF_INT_ST The masked interrupt status bit of [SLCO_RX_EOF_INT](#). (RO)

SDIO_SLCO_TX_DSCR_ERR_INT_ST The masked interrupt status of [SLCO_TX_DSCR_ERR_INT](#). (RO)

SDIO_SLCO_RX_DSCR_ERR_INT_ST The masked interrupt status of [SLCO_RX_DSCR_ERR_INT](#). (RO)

Register 29.32. SDIO_SLC0INT_ENA_REG (0x000C)

(reserved)												SDIO_SLC0_RX_DSCR_ERR_INT_ENA SDIO_SLC0_TX_DSCR_ERR_INT_ENA (reserved) SDIO_SLC0_RX_EOF_INT_ENA SDIO_SLC0_RX_DONE_INT_ENA SDIO_SLC0_TX_SUC_EOF_INT_ENA SDIO_SLC0_TX_DONE_INT_ENA (reserved) SDIO_SLC0_TX_OVF_INT_ENA SDIO_SLC0_RX_UDF_INT_ENA SDIO_SLC0_TX_START_INT_ENA SDIO_SLC0_RX_START_INT_ENA SDIO_SLC0_FRHOST_BIT7_INT_ENA SDIO_SLC0_FRHOST_BIT6_INT_ENA SDIO_SLC0_FRHOST_BIT5_INT_ENA SDIO_SLC0_FRHOST_BIT4_INT_ENA SDIO_SLC0_FRHOST_BIT3_INT_ENA SDIO_SLC0_FRHOST_BIT2_INT_ENA SDIO_SLC0_FRHOST_BIT1_INT_ENA SDIO_SLC0_FRHOST_BIT0_INT_ENA																									
31											21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
0x0												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SDIO_SLC0_FRHOST_BIT n _INT_ENA (n : 0-7) Write 1 to enable interrupt [SLC0_FRHOST_BIT \$n\$ _INT](#) (n : 0-7). (R/W)

SDIO_SLC0_RX_START_INT_ENA Write 1 to enable interrupt [SLC0_RX_START_INT](#). (R/W)

SDIO_SLC0_TX_START_INT_ENA Write 1 to enable interrupt [SLC0_TX_START_INT](#). (R/W)

SDIO_SLC0_RX_UDF_INT_ENA Write 1 to enable interrupt [SLC0_RX_UDF_INT](#). (R/W)

SDIO_SLC0_TX_OVF_INT_ENA Write 1 to enable interrupt [SLC0_TX_OVF_INT](#). (R/W)

SDIO_SLC0_TX_DONE_INT_ENA Write 1 to enable interrupt [SLC0_TX_DONE_INT](#). (R/W)

SDIO_SLC0_TX_SUC_EOF_INT_ENA Write 1 to enable interrupt [SLC0_TX_SUC_EOF_INT](#). (R/W)

SDIO_SLC0_RX_DONE_INT_ENA Write 1 to enable interrupt [SLC0_RX_DONE_INT](#). (R/W)

SDIO_SLC0_RX_EOF_INT_ENA Write 1 to enable interrupt [SLC0_RX_EOF_INT](#). (R/W)

SDIO_SLC0_TX_DSCR_ERR_INT_ENA Write 1 to enable interrupt [SLC0_TX_DSCR_ERR_INT](#). (R/W)

SDIO_SLC0_RX_DSCR_ERR_INT_ENA Write 1 to enable interrupt [SLC0_RX_DSCR_ERR_INT](#). (R/W)

Register 29.33. SDIO_SLC0INT_CLR_REG (0x0010)

(reserved)																SDIO_SLC0_RX_DSCR_ERR_INT_CLR SDIO_SLC0_TX_DSCR_ERR_INT_CLR (reserved) SDIO_SLC0_RX_EOF_INT_CLR SDIO_SLC0_RX_DONE_INT_CLR SDIO_SLC0_TX_SUC_EOF_INT_CLR SDIO_SLC0_TX_DONE_INT_CLR (reserved) SDIO_SLC0_TX_OVF_INT_CLR SDIO_SLC0_RX_UDF_INT_CLR SDIO_SLC0_TX_START_INT_CLR SDIO_SLC0_RX_START_INT_CLR SDIO_SLC_FRHST_BIT7_INT_CLR SDIO_SLC_FRHST_BIT6_INT_CLR SDIO_SLC_FRHST_BIT5_INT_CLR SDIO_SLC_FRHST_BIT4_INT_CLR SDIO_SLC_FRHST_BIT3_INT_CLR SDIO_SLC_FRHST_BIT2_INT_CLR SDIO_SLC_FRHST_BIT1_INT_CLR SDIO_SLC_FRHST_BIT0_INT_CLR															
31	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
0x0		0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

SDIO_SLC_FRHOST_BIT n _INT_CLR (n : 0-7) Write 1 to clear interrupt [SLC_FRHOST_BIT \$n\$ _INT \(\$n\$: 0-7\)](#). (WT)

SDIO_SLC0_RX_START_INT_CLR Write 1 to clear interrupt [SLCO_RX_START_INT](#). (WT)

SDIO_SLC0_TX_START_INT_CLR Write 1 to clear interrupt [SLCO_TX_START_INT](#). (WT)

SDIO_SLC0_RX_UDF_INT_CLR Write 1 to clear interrupt [SLCO_RX_UDF_INT](#). (WT)

SDIO_SLC0_TX_OVF_INT_CLR Write 1 to clear interrupt [SLCO_TX_OVF_INT](#). (WT)

SDIO_SLC0_TX_DONE_INT_CLR Write 1 to clear interrupt [SLCO_TX_DONE_INT](#). (WT)

SDIO_SLC0_TX_SUC_EOF_INT_CLR Write 1 to clear interrupt [SLCO_TX_SUC_EOF_INT](#). (WT)

SDIO_SLC0_RX_DONE_INT_CLR Write 1 to clear interrupt [SLCO_RX_DONE_INT](#). (WT)

SDIO_SLC0_RX_EOF_INT_CLR Write 1 to clear interrupt [SLCO_RX_EOF_INT](#). (WT)

SDIO_SLC0_TX_DSCR_ERR_INT_CLR Write 1 to clear interrupt [SLCO_TX_DSCR_ERR_INT](#). (WT)

SDIO_SLC0_RX_DSCR_ERR_INT_CLR Write 1 to clear interrupt [SLCO_RX_DSCR_ERR_INT](#). (WT)

Register 29.34. SDIO_SLC1INT_RAW_REG (0x0014)

(reserved)												SDIO_SLC1_RX_DSCR_ERR_INT_RAW SDIO_SLC1_TX_DSCR_ERR_INT_RAW (reserved) SDIO_SLC1_RX_EOF_INT_RAW SDIO_SLC1_RX_DONE_INT_RAW SDIO_SLC1_TX_SUC_EOF_INT_RAW SDIO_SLC1_TX_DONE_INT_RAW (reserved) SDIO_SLC1_TX_OVF_INT_RAW SDIO_SLC1_RX_UDF_INT_RAW SDIO_SLC1_TX_START_INT_RAW SDIO_SLC1_RX_START_INT_RAW SDIO_SLC_FRHST_BIT15_INT_RAW SDIO_SLC_FRHST_BIT14_INT_RAW SDIO_SLC_FRHST_BIT13_INT_RAW SDIO_SLC_FRHST_BIT12_INT_RAW SDIO_SLC_FRHST_BIT11_INT_RAW SDIO_SLC_FRHST_BIT10_INT_RAW SDIO_SLC_FRHST_BIT9_INT_RAW SDIO_SLC_FRHST_BIT8_INT_RAW												
31	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0x0		0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SDIO_SLC_FRHOST_BIT n _INT_RAW (n : 8-15) The raw interrupt status of [SLC_FRHOST_BIT \$n\$ _INT](#) (n : 8-15). (R/WTC/SS)

SDIO_SLC1_RX_START_INT_RAW The raw interrupt status of [SLC1_RX_START_INT](#). (R/WTC/SS)

SDIO_SLC1_TX_START_INT_RAW The raw interrupt status of [SLC1_TX_START_INT](#). (R/WTC/SS)

SDIO_SLC1_RX_UDF_INT_RAW The raw interrupt status of [SLC1_RX_UDF_INT](#). (R/WTC/SS)

SDIO_SLC1_TX_OVF_INT_RAW The raw interrupt status of [SLC1_TX_OVF_INT](#). (R/WTC/SS)

SDIO_SLC1_TX_DONE_INT_RAW The raw interrupt status of [SLC1_TX_DONE_INT](#). (R/WTC/SS)

SDIO_SLC1_TX_SUC_EOF_INT_RAW The raw interrupt status of [SLC1_TX_SUC_EOF_INT](#). (R/WTC/SS)

SDIO_SLC1_RX_DONE_INT_RAW The raw interrupt status of [SLC1_RX_DONE_INT](#). (R/WTC/SS)

SDIO_SLC1_RX_EOF_INT_RAW The raw interrupt status of [SLC1_RX_EOF_INT](#). (R/WTC/SS)

SDIO_SLC1_TX_DSCR_ERR_INT_RAW The raw interrupt status of [SLC1_TX_DSCR_ERR_INT](#). (R/WTC/SS)

SDIO_SLC1_RX_DSCR_ERR_INT_RAW The raw interrupt status of [SLC1_RX_DSCR_ERR_INT](#). (R/WTC/SS)

Register 29.35. SDIO_SLC1INT_CLR_REG (0x0020)

[illegible]

SDIO_SLC_FRHOST_BIT n _INT_CLR (n : 8-15) Write 1 to clear interrupt SLC_FRHOST_BIT n _INT (n : 8-15). (WT)

SDIO_SLC1_RX_START_INT_CLR Write 1 to clear interrupt [SLC1_RX_START_INT](#). (WT)

SDIO_SLC1_TX_START_INT_CLR Write 1 to clear interrupt [SLC1_TX_START_INT](#). (WT)

SDIO_SLC1_RX_UDF_INT_CLR Write 1 to clear interrupt [SLC1_RX_UDF_INT](#). (WT)

SDIO_SLC1_TX_OVF_INT_CLR Write 1 to clear interrupt [SLC1_TX_OVF_INT](#). (WT)

SDIO_SLC1_TX_DONE_INT_CLR Write 1 to clear interrupt [SLC1_TX_DONE_INT](#). (WT)

SDIO_SLC1_TX_SUC_EOF_INT_CLR Write 1 to clear interrupt [SLC1_TX_SUC_EOF_INT](#). (WT)

SDIO_SLC1_RX_DONE_INT_CLR Write 1 to clear interrupt [SLC1_RX_DONE_INT](#). (WT)

SDIO_SLC1_RX_EOF_INT_CLR Write 1 to clear interrupt **SLC1_RX_EOF_INT**. (WT)

SDIO_SLC1_TX_DSCR_ERR_INT_CLR Write 1 to clear interrupt [SLC1_TX_DSCR_ERR_INT](#). (WT)

SDIO_SLC1_RX_DSCR_ERR_INT_CLR Write 1 to clear interrupt [SLC1_RX_DSCR_ERR_INT](#). (WT)

Register 29.36. SDIO_SLCINTVEC_TOHOST_REG (0x005C)

(reserved)		SDIO_SLC1_TOHOST_INTVEC		(reserved)		SDIO_SLC0_TOHOST_INTVEC	
31	24	23	16	15	8	7	0
0x0		0x0		0x0		0x0	

Reset

SDIO_SLC0_TOHOST_INTVEC The interrupt set bit of **SLCHOST_SLC0_TOHOST_BIT n _INT** (n : 0-7).
These bits will be cleared automatically. (WT)

SDIO_SLC1_TOHOST_INTVEC The interrupt set bit of [SLCHOST_SLC1_TOHOST_BIT \$n\$ _INT](#) (n : 0-7).
These bits will be cleared automatically. (WT)

Register 29.37. SDIO_SLC1INT_ST1_REG (0x014C)

(reserved)																SDIO_SLC1_RX_DSCR_ERR_INT_ST1 SDIO_SLC1_TX_DSCR_ERR_INT_ST1 (reserved) SDIO_SLC1_RX_EOF_INT_ST1 SDIO_SLC1_RX_DONE_INT_ST1 SDIO_SLC1_TX_SUC_EOF_INT_ST1 SDIO_SLC1_TX_DONE_INT_ST1 (reserved) SDIO_SLC1_TX_OVF_INT_ST1 SDIO_SLC1_RX_UDF_INT_ST1 SDIO_SLC1_TX_START_INT_ST1 SDIO_SLC1_RX_START_INT_ST1 SDIO_SLC_FRHOST_BIT15_INT_ST1 SDIO_SLC_FRHOST_BIT14_INT_ST1 SDIO_SLC_FRHOST_BIT13_INT_ST1 SDIO_SLC_FRHOST_BIT12_INT_ST1 SDIO_SLC_FRHOST_BIT11_INT_ST1 SDIO_SLC_FRHOST_BIT10_INT_ST1 SDIO_SLC_FRHOST_BIT9_INT_ST1 SDIO_SLC_FRHOST_BIT8_INT_ST1																					
31																21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0								0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

SDIO_SLC_FRHOST_BIT n _INT_ST1 (n : 8-15) The masked interrupt status of [SLC_FRHOST_BIT \$n\$ _INT](#) (n : 8-15). (RO)

SDIO_SLC1_RX_START_INT_ST1 The masked interrupt status of [SLC1_RX_START_INT](#). (RO)

SDIO_SLC1_TX_START_INT_ST1 The masked interrupt status of [SLC1_TX_START_INT](#). (RO)

SDIO_SLC1_RX_UDF_INT_ST1 The masked interrupt status of [SLC1_RX_UDF_INT](#). (RO)

SDIO_SLC1_TX_OVF_INT_ST1 The masked interrupt status of [SLC1_TX_OVF_INT](#). (RO)

SDIO_SLC1_TX_DONE_INT_ST1 The masked interrupt status of [SLC1_TX_DONE_INT](#). (RO)

SDIO_SLC1_TX_SUC_EOF_INT_ST1 The masked interrupt status of [SLC1_TX_SUC_EOF_INT](#). (RO)

SDIO_SLC1_RX_DONE_INT_ST1 The masked interrupt status of [SLC1_RX_DONE_INT](#). (RO)

SDIO_SLC1_RX_EOF_INT_ST1 The masked interrupt status of [SLC1_RX_EOF_INT](#). (RO)

SDIO_SLC1_TX_DSCR_ERR_INT_ST1 The masked interrupt status of [SLC1_TX_DSCR_ERR_INT](#). (RO)

SDIO_SLC1_RX_DSCR_ERR_INT_ST1 The masked interrupt status of [SLC1_RX_DSCR_ERR_INT](#). (RO)

Register 29.38. SDIO_SLC1INT_ENA1_REG (0x0150)

(reserved)																SDIO_SLC1_RX_DSCR_ERR_INT_ENA1 SDIO_SLC1_TX_DSCR_ERR_INT_ENA1 (reserved) SDIO_SLC1_RX_EOF_INT_ENA1 SDIO_SLC1_RX_DONE_INT_ENA1 SDIO_SLC1_TX_SUC_EOF_INT_ENA1 SDIO_SLC1_TX_DONE_INT_ENA1 (reserved) SDIO_SLC1_TX_OVF_INT_ENA1 SDIO_SLC1_RX_UDF_INT_ENA1 SDIO_SLC1_TX_START_INT_ENA1 SDIO_SLC1_RX_START_INT_ENA1 SDIO_SLC1_FRHOST_BIT15_INT_ENA1 SDIO_SLC1_FRHOST_BIT14_INT_ENA1 SDIO_SLC1_FRHOST_BIT13_INT_ENA1 SDIO_SLC1_FRHOST_BIT12_INT_ENA1 SDIO_SLC1_FRHOST_BIT11_INT_ENA1 SDIO_SLC1_FRHOST_BIT10_INT_ENA1 SDIO_SLC1_FRHOST_BIT9_INT_ENA1 SDIO_SLC1_FRHOST_BIT8_INT_ENA1																	
31											21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x0											0	0	0	0	0	0	0	0x0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Reset

SDIO_SLC_FRHOST_BIT n _INT_ENA1 (n : 8-15) Write 1 to enable interrupt [SLC_FRHOST_BIT \$n\$ _INT](#) (n : 8-15). (R/W)

SDIO_SLC1_RX_START_INT_ENA1 Write 1 to enable interrupt [SLC1_RX_START_INT](#). (R/W)

SDIO_SLC1_TX_START_INT_ENA1 Write 1 to enable interrupt [SLC1_TX_START_INT](#). (R/W)

SDIO_SLC1_RX_UDF_INT_ENA1 Write 1 to enable interrupt [SLC1_RX_UDF_INT](#). (R/W)

SDIO_SLC1_TX_OVF_INT_ENA1 Write 1 to enable interrupt [SLC1_TX_OVF_INT](#). (R/W)

SDIO_SLC1_TX_DONE_INT_ENA1 Write 1 to enable interrupt [SLC1_TX_DONE_INT](#). (R/W)

SDIO_SLC1_TX_SUC_EOF_INT_ENA1 Write 1 to enable interrupt [SLC1_TX_SUC_EOF_INT](#). (R/W)

SDIO_SLC1_RX_DONE_INT_ENA1 Write 1 to enable interrupt [SLC1_RX_DONE_INT](#). (R/W)

SDIO_SLC1_RX_EOF_INT_ENA1 Write 1 to enable interrupt [SLC1_RX_EOF_INT](#). (R/W)

SDIO_SLC1_TX_DSCR_ERR_INT_ENA1 Write 1 to enable interrupt [SLC1_TX_DSCR_ERR_INT](#). (R/W)

SDIO_SLC1_RX_DSCR_ERR_INT_ENA1 Write 1 to enable interrupt [SLC1_RX_DSCR_ERR_INT](#). (R/W)

Register 29.39. SDIO_SLC0_LENGTH_REG (0x00F8)

(reserved)												SDIO_SLC0_LEN												
31											20	19											0	
0x0												0x0												Reset

Reset

SDIO_SLC0_LEN Represents the accumulated length of data that the slave wants to send. (RO)

29.9.3 SLC Host Registers

Register 29.40. SLCHOST_CONF_REG (0x01F0)

(reserved)				SLCHOST_HSPEED_CON_EN				(reserved)				SLCHOST_FRC_POS_SAMP				SLCHOST_FRC_NEG_SAMP				SLCHOST_FRC_SDIO20				SLCHOST_FRC_SDIO11				
31	28	27	26	20	19	15	14	10	9	5	4	0																
0x0				0	0x0				0x0				0x0				0x0				0x0				Reset			

Reset

SLCHOST_FRC_SDIO11 Configure 1 to bit[4] to force drive CMD signal at the falling clock edge. Configures 1 to bit[3:0] corresponding bit to force drive DAT[3:0] signal corresponding bit at the falling clock edge. (R/W)

SLCHOST_FRC_SDIO20 Configure 1 to bit[4] to force drive CMD signal at the rising clock edge. Configures 1 to bit[3:0] corresponding bit to force drive DAT[3:0] signal corresponding bit at the rising clock edge. (R/W)

SLCHOST_FRC_NEG_SAMP Configure 1 to bit[4] to force sample CMD signal at the falling clock edge. Configures 1 to bit[3:0] corresponding bit to force sample DAT[3:0] signal corresponding bit at the falling clock edge. (R/W)

SLCHOST_FRC_POS_SAMP Configure 1 to bit[4] to force sample CMD signal at the rising clock edge. Configures 1 to bit[3:0] corresponding bit to force sample DAT[3:0] signal corresponding bit at the rising clock edge. (R/W)

SLCHOST_HSPEED_CON_EN Configures 1 to this bit, configures 1 to [HINF_HIGHSPEED_ENABLE](#), and then the host configures 1 to EHS in CCCR to force drive CMD and DAT signals at the rising clock edge. (R/W)

Register 29.41. SLCHOST_SLC0HOST_INT_RAW_REG (0x0050)

[illegible]

SLCHOST_SLC0_TOHOST_BIT n _INT_RAW (n : 0-7) The raw interrupt status of SLCHOST_SLC0_TOHOST_BIT n _INT (n : 0-7). (R/WTC/SS)

SLCHOST_SLCO_RX_UDF_INT_RAW The raw interrupt status of [SLCHOST_SLCO_RX_UDF_INT](#).
(R/WTC/SS)

SLCHOST_SLCO_TX_OVF_INT_RAW The raw interrupt status of [SLCHOST_SLCO_TX_OVF_INT](#).
(R/WTC/SS)

SLCHOST_SLCO_RX_NEW_PACKET_INT_RAW The raw interrupt status of [SLCHOST_SLCO_RX_NEW_PACKET_INT](#). (R/WTC/SS)

Register 29.42. SLCHOST_SLC1HOST_INT_RAW_REG (0x0054)

(reserved)				SLCHOST_SLC1_RX_NEW_PACKET_INT_RAW				(reserved)				SLCHOST_SLC1_TX_OVF_INT_RAW SLCHOST_SLC1_RX_UDF_INT_RAW				(reserved)				SLCHOST_SLC1_TOHOST_BIT7_INT_RAW SLCHOST_SLC1_TOHOST_BIT6_INT_RAW SLCHOST_SLC1_TOHOST_BIT5_INT_RAW SLCHOST_SLC1_TOHOST_BIT4_INT_RAW SLCHOST_SLC1_TOHOST_BIT3_INT_RAW SLCHOST_SLC1_TOHOST_BIT2_INT_RAW SLCHOST_SLC1_TOHOST_BIT1_INT_RAW SLCHOST_SLC1_TOHOST_BIT0_INT_RAW							
31	26	25	24	18	17	16	15	8	7	6	5	4	3	2	1	0											
0x0			0	0x0			0	0	0x0			0	0	0	0	0	0	Reset									

SLCHOST_SLC1_TOHOST_BIT n _INT_RAW (n : 0-7) The raw interrupt status of [SLCHOST_SLC1_TOHOST_BIT \$n\$ _INT \(\$n\$: 0-7\)](#). (R/WTC/SS)

SLCHOST_SLC1_RX_UDF_INT_RAW The raw interrupt status of [SLCHOST_SLC1_RX_UDF_INT](#). (R/WTC/SS)

SLCHOST_SLC1_TX_OVF_INT_RAW The raw interrupt status of [SLCHOST_SLC1_TX_OVF_INT](#). (R/WTC/SS)

SLCHOST_SLC1_RX_NEW_PACKET_INT_RAW The raw interrupt status of [SLCHOST_SLC1_RX_NEW_PACKET_INT](#). (R/WTC/SS)

Register 29.43. SLCHOST_SLCOHST_INT_ST_REG (0x0058)

[illegible]

SLCHOST_SLC0_TOHOST_BIT*n*_INT_ST (*n*: 0-7) The masked interrupt status of SLCHOST_SLC0_TOHOST_BIT*n*_INT (*n*: 0-7). (RO)

SLCHOST_SLC0_RX_UDF_INT_ST The masked interrupt status of [SLCHOST_SLC0_RX_UDF_INT](#).
(RO)

SLCHOST_SLC0_TX_OVF_INT_ST The masked interrupt status of [SLCHOST_SLC0_TX_OVF_INT](#).
(RO)

SLCHOST_SLC0_RX_NEW_PACKET_INT_ST The masked interrupt status of [SLCHOST_SLC0_RX_NEW_PACKET_INT](#). (RO)

Register 29.44. SLCHOST_SLC1HOST_INT_ST_REG (0x005C)

(reserved)				SLCHOST_SLC1_RX_NEW_PACKET_INT_ST				(reserved)				SLCHOST_SLC1_TX_OVF_INT_ST SLCHOST_SLC1_RX_UDF_INT_ST				(reserved)				SLCHOST_SLC1_TOHOST_BIT7_INT_ST SLCHOST_SLC1_TOHOST_BIT6_INT_ST SLCHOST_SLC1_TOHOST_BIT5_INT_ST SLCHOST_SLC1_TOHOST_BIT4_INT_ST SLCHOST_SLC1_TOHOST_BIT3_INT_ST SLCHOST_SLC1_TOHOST_BIT2_INT_ST SLCHOST_SLC1_TOHOST_BIT1_INT_ST SLCHOST_SLC1_TOHOST_BIT0_INT_ST							
31	26	25	24	18	17	16	15	8	7	6	5	4	3	2	1	0	Reset										
0x0			0	0x0			0	0	0x0			0	0	0	0	0		0									

SLCHOST_SLC1_TOHOST_BIT n _INT_ST (n : 0-7) The masked interrupt status of **SLCHOST_SLC1_TOHOST_BIT n _INT** (n : 0-7). (RO)

SLCHOST_SLC1_RX_UDF_INT_ST The masked interrupt status of **SLCHOST_SLC1_RX_UDF_INT**. (RO)

SLCHOST_SLC1_TX_OVF_INT_ST The masked interrupt status of **SLCHOST_SLC1_TX_OVF_INT**. (RO)

SLCHOST_SLC1_RX_NEW_PACKET_INT_ST The masked interrupt status of **SLCHOST_SLC1_RX_NEW_PACKET_INT**. (RO)

Register 29.45. SLCHOST_CONF_W7_REG (0x008C)

SLCHOST_SLCHOST_CONF31								(reserved)								SLCHOST_SLCHOST_CONF29								(reserved)							
31	24	23	16	15	8	7	0	Reset																							
0x0								0x0								0x0								0x0							

SLCHOST_SLCHOST_CONF29 The interrupt set bits of **SLCINT_SLC_FRHOST_BIT n _INT** (n : 0-7). These bits will not be cleared automatically. (R/W)

SLCHOST_SLCHOST_CONF31 The interrupt set bits of **SLCINT_SLC_FRHOST_BIT n _INT** (n : 8-15). These bits will not be cleared automatically. (R/W)

Register 29.46. SLCHOST_SLCOHST_INT_CLR_REG (0x00D4)

[illegible]

SLCHOST_SLC0_TOHOST_BIT n _INT_CLR (n : 0-7) Write 1 to clear interrupt SL-
CHOST_SLC0_TOHOST_BIT n _INT (n : 0-7). (WT)

SLCHOST_SLCO_RX_UDF_INT_CLR Write 1 to clear interrupt [SLCHOST_SLCO_RX_UDF_INT](#). (WT)

SLCHOST_SLC0_TX_OVF_INT_CLR Write 1 to clear interrupt [SLCHOST_SLC0_TX_OVF_INT](#). (WT)

SLCHOST_SLC0_RX_NEW_PACKET_INT_CLR Write 1 to clear interrupt SL-
CHOST_SLC0_RX_NEW_PACKET_INT. (WT)

Register 29.47. SLCHOST_SLC1HOST_INT_CLR_REG (0x00D8)

[illegible]

SLCHOST_SLC1_TOHOST_BIT*n*_INT_CLR (*n*: 0-7) Write 1 to clear interrupt [SLCHOST_SLC1_TOHOST_BIT](#)*n*_INT (*n*: 0-7). (WT)

SLCHOST_SLC1_RX_UDF_INT_CLR Write 1 to clear interrupt [SLCHOST_SLC1_RX_UDF_INT](#). (WT)

SLCHOST_SLC1_TX_OVF_INT_CLR Write 1 to clear interrupt [SLCHOST_SLC1_TX_OVF_INT](#). (WT)

SLCHOST_SLC1_RX_NEW_PACKET_INT_CLR Write 1 to clear interrupt SL-
CHOST_SLC1_RX_NEW_PACKET_INT. (WT)

Register 29.48. SLCHOST_SLCOHST_FUNC1_INT_ENA_REG (0x00DC)

[illegible]

SLCHOST_FN1_SLC0_TOHOST_BIT n _INT_ENA (n : 0-7) Write 1 to enable SL-
CHOST_SLC0_TOHOST_BIT n _INT (n : 0-7). (R/W)

SLCHOST_FN1_SLC0_RX_UDF_INT_ENA Write 1 to enable [SLCHOST_SLC0_RX_UDF_INT](#). (R/W)

SLCHOST_FN1_SLC0_TX_OVF_INT_ENA Write 1 to enable [SLCHOST_SLC0_TX_OVF_INT](#). (R/W)

SLCHOST_FN1_SLC0_RX_NEW_PACKET_INT_ENA Write 1 to enable SL-
CHOST_SLC0_RX_NEW_PACKET_INT. (R/W)

Register 29.49. SLCHOST_SLC1HOST_FUNC1_INT_ENA_REG (0x00E0)

[illegible]

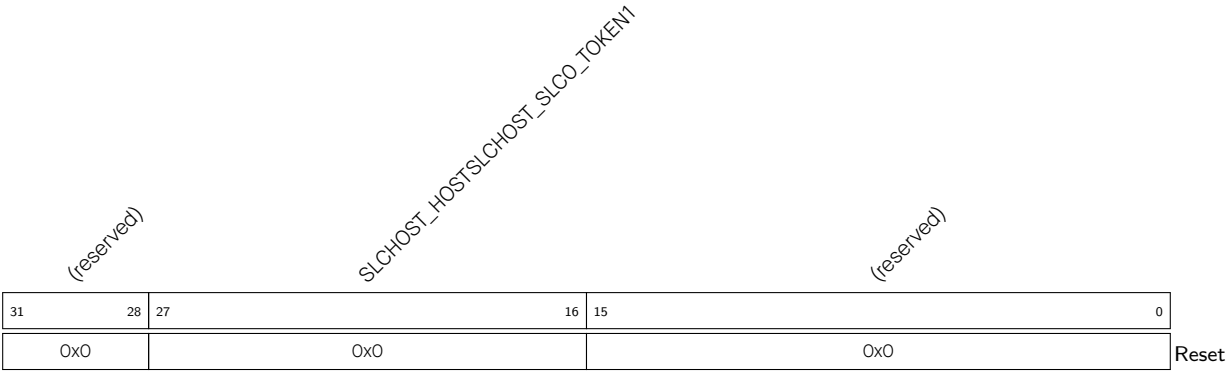
SLCHOST_FN1_SLC1_TOHOST_BIT n _INT_ENA (n : 0-7) Write 1 to enable SL-
CHOST_SLC1_TOHOST_BIT n _INT (n : 0-7). (R/W)

SLCHOST_FN1_SLC1_RX_UDF_INT_ENA Write 1 to enable [SLCHOST_SLC1_RX_UDF_INT](#). (R/W)

SLCHOST_FN1_SLC1_TX_OVF_INT_ENA Write 1 to enable [SLCHOST_SLC1_TX_OVF_INT](#). (R/W)

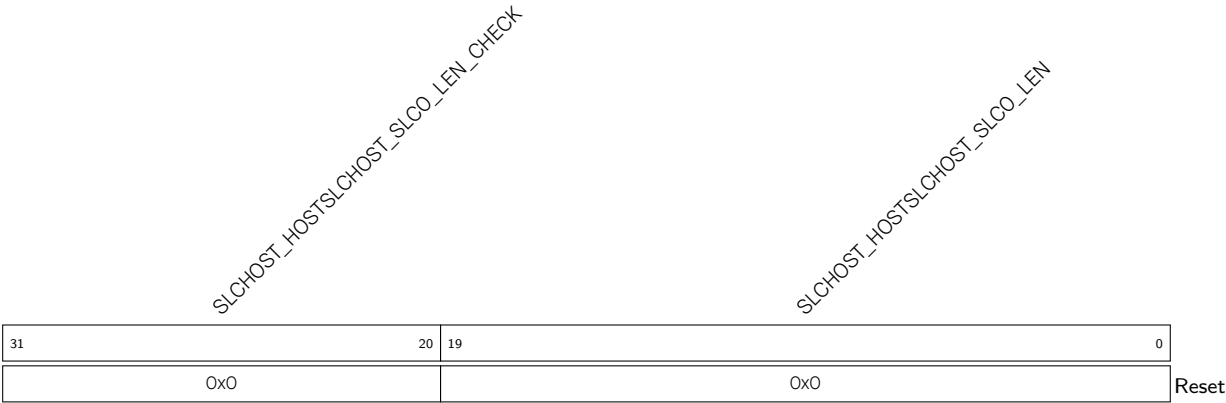
SLCHOST_FN1_SLC1_RX_NEW_PACKET_INT_ENA Write 1 to enable SL-
CHOST_SLC1_RX_NEW_PACKET_INT. (R/W)

Register 29.50. SLCHOST_SLC0HOST_TOKEN_RDATA_REG (0x0044)



SLCHOST_HOSTSLCHOST_SLC0_TOKEN1 Represents the SLC0 accumulated number of buffers for receiving data. (RO)

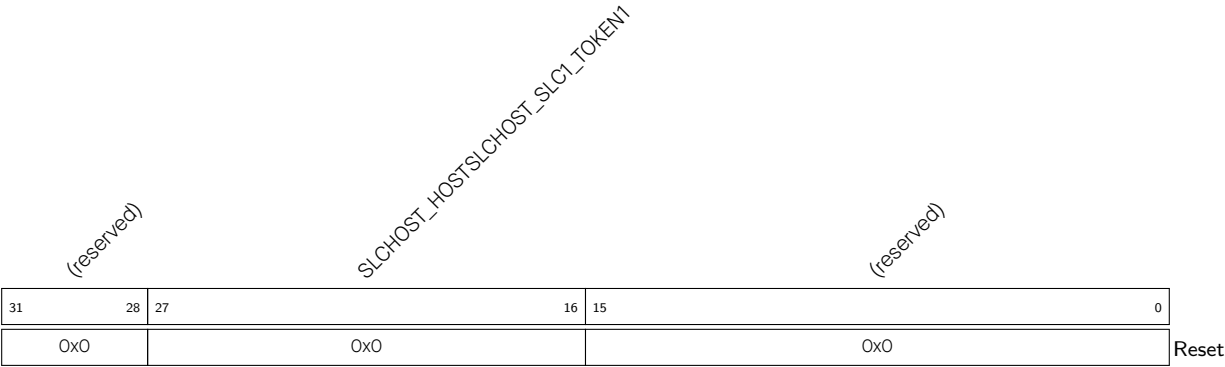
Register 29.51. SLCHOST_PKT_LEN_REG (0x0060)



SLCHOST_HOSTSLCHOST_SLC0_LEN Represents the accumulated length of data that the slave wants to send. The value gets updated only when the host reads it. (RO)

SLCHOST_HOSTSLCHOST_SLC0_LEN_CHECK Check SLCHOST_HOSTSLCHOST_SLC0_LEN. Its value is SLCHOST_HOSTSLCHOST_SLC0_LEN bit[9:0] plus bit[19:10]. (RO)

Register 29.52. SLCHOST_SLC1HOST_TOKEN_RDATA_REG (0x00C4)



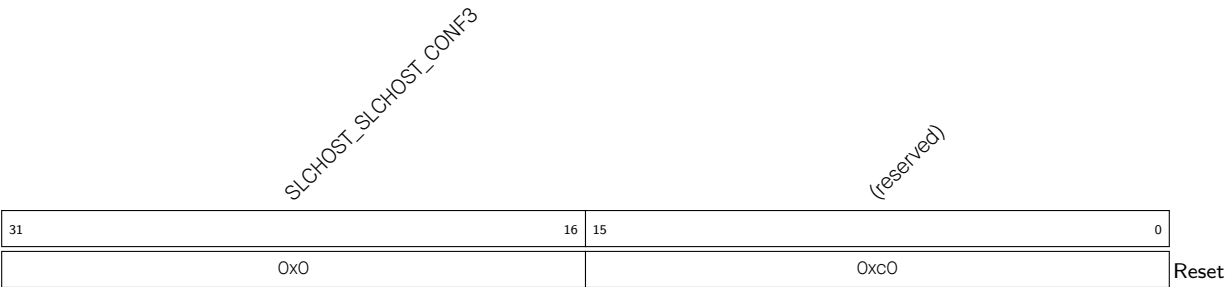
SLCHOST_HOSTSLCHOST_SLC1_TOKEN1 Represents the SLC1 accumulated number of buffers for receiving data. (RO)

Register 29.53. SLCHOST_CONF_Wn_REG(*n*: 0-2) (0x006C+0x4**n*)



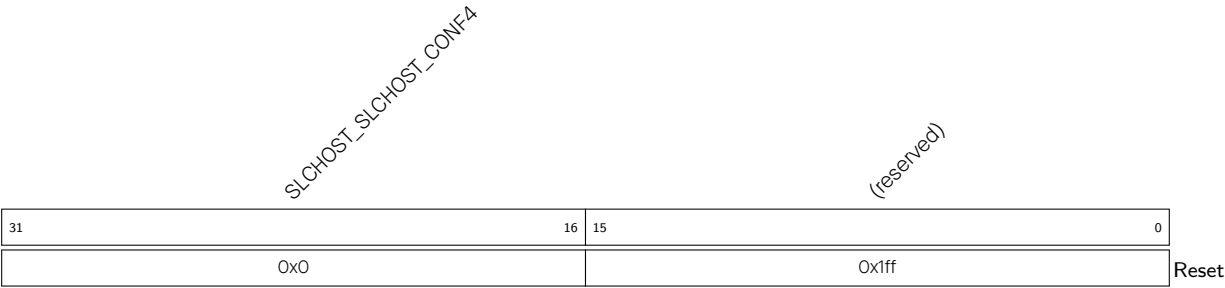
SLCHOST_SLCHOST_CONFn The information interaction register between the host and slave. Both of them can access it. (R/W)

Register 29.54. SLCHOST_CONF_W3_REG (0x0078)



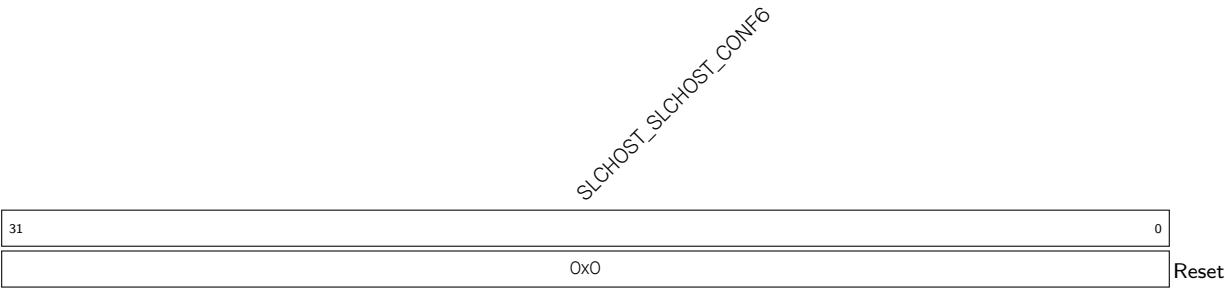
SLCHOST_SLCHOST_CONF3 The information interaction register between the host and slave. Both of them can access it. (R/W)

Register 29.55. SLCHOST_CONF_W4_REG (0x007C)



SLCHOST_SLCHOST_CONF4 The information interaction register between the host and slave. Both of them can access it. (R/W)

Register 29.56. SLCHOST_CONF_W6_REG (0x0088)



SLCHOST_SLCHOST_CONF6 The information interaction register between the host and slave. Both of them can access it. (R/W)

Register 29.57. SLCHOST_CONF_Wn_REG(*n*: 8-15) (0x009C+0x4*(*n*-8))



SLCHOST_SLCHOST_CONFn The information interaction register between the host and slave. Both of them can access it. (R/W)

Chapter 30

LED PWM Controller (LEDC)

30.1 Overview

The LED PWM Controller is a peripheral designed to generate PWM signals for LED control. It has specialized features such as automatic duty cycle fading. However, the LED PWM Controller can also be used to generate PWM signals for other purposes.

30.2 Feature List

The LED PWM Controller has the following features:

- Six independent PWM generators (i.e. six channels)
- Maximum PWM duty cycle resolution: 20 bits
- Four independent timers that support fractional division
- Adjustable phase of PWM signal output
- PWM duty cycle dithering
- Automatic duty cycle fading — gradual increase/decrease of a PWM's duty cycle without interference from the processor. An interrupt will be generated upon fade completion
- Up to 16 duty cycle ranges for each PWM generator to generate gamma curve signals — each range can be independently configured in terms of fading direction (increase or decrease), fading amount (the amount by which the duty cycle increases or decreases each time), the number of fades (how many times the duty cycle fades in one range), and fading frequency
- PWM signal output in low-power mode (Light-sleep mode)
- Event generation and task response related to the Event Task Matrix (ETM) peripheral

Note that the four timers are identical regarding their features and operation. The following sections refer to the timers collectively as Timer x (where x ranges from 0 to 3). Likewise, the six PWM generators are also identical in features and operation, and thus are collectively referred to as PWM n (where n ranges from 0 to 5).

30.3 Architectural Overview

Figure 30.3-1 shows the architecture of the LED PWM Controller.

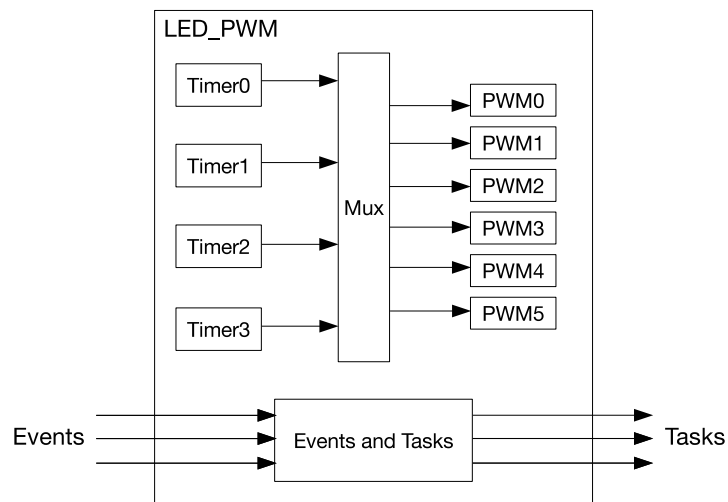


Figure 30.3-1. LED PWM Architecture

Each of the four timers has an internal timebase counter (i.e., a counter that counts on cycles of a reference clock) and thus can be independently configured (i.e., configurable clock divider, and counter overflow value). Each PWM generator selects one of the timers by configuring the `LEDC_TIMER_SEL_CHn`, and uses the timer's counter value `timerx_cnt` as a reference to generate its PWM signal.

Figure 30.3-2 illustrates the main functional blocks of the timer and the PWM generator.

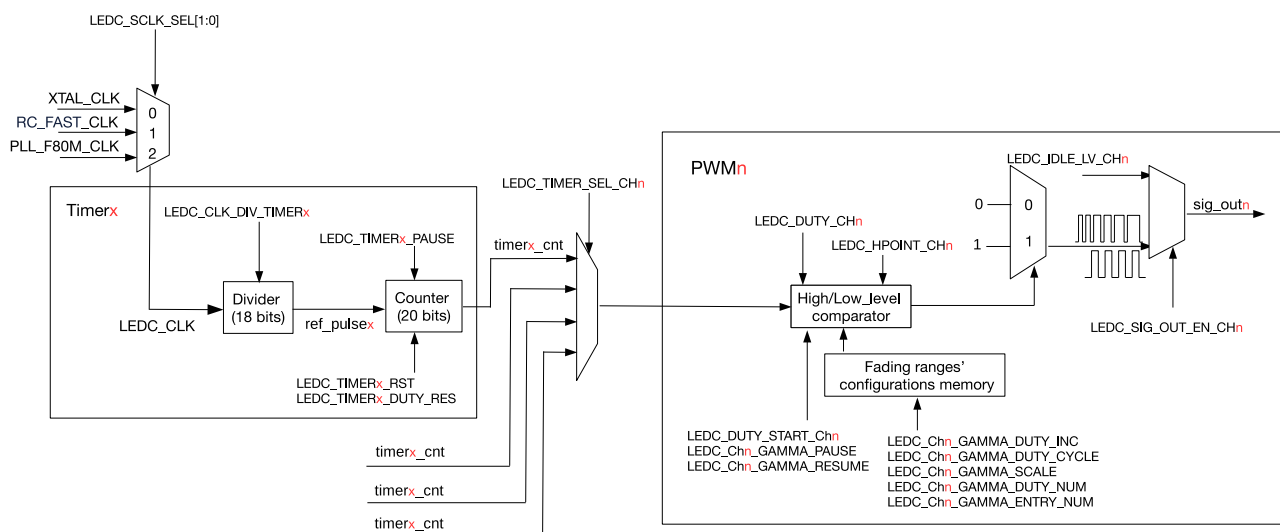


Figure 30.3-2. Timer and PWM Generator Block Diagram

30.4 Functional Description

30.4.1 Timers

Each timer in LED PWM Controller internally maintains a timebase counter. Referring to Figure 30.3-2, this clock signal used by the timebase counter is named `ref_pulsex`. All timers use the same clock source `LEDC_CLK`, which is then passed through a clock divider to generate `ref_pulsex` for the counter.

30.4.1.1 Clock Source

LED PWM registers configured by software are clocked by APB_CLK. To use the LED PWM peripheral, the APB_CLK signal going to the LED PWM has to be enabled. The APB_CLK signal to LED PWM can be enabled by setting the `PCR_LEDC_CLK_EN` field in the `PCR_LEDC_CONF_REG` register. The LEDC_CLK signal to LED PWM can be enabled by setting the `PCR_LEDC_SCLK_EN` field in the `PCR_LEDC_SCLK_CONF_REG` register. The LED PWM peripheral can be reset via software by setting the `PCR_LEDC_RST_EN` field in the `PCR_LEDC_CONF_REG` register.

Timers in the LED PWM Controller choose their common clock source from one of the following clock signals: XTAL_CLK, RC_FAST_CLK, and PLL_F80M_CLK. The procedure for selecting a clock source signal for LEDC_CLK is described below:

- XTAL_CLK: Set `PCR_LEDC_SCLK_SEL[1:0]` to 0
- RC_FAST_CLK: Set `PCR_LEDC_SCLK_SEL[1:0]` to 1
- PLL_F80M_CLK: Set `PCR_LEDC_SCLK_SEL[1:0]` to 2

The LEDC_CLK signal will then be passed through the clock divider.

For more information, please refer to Chapter 6 *Reset and Clock*.

30.4.1.2 Clock Divider Configuration

The LEDC_CLK signal is passed through a clock divider to generate the `ref_pulsex` signal for the counter. The frequency of `ref_pulsex` is equal to the frequency of LEDC_CLK divided by the divisor LEDC_CLK_DIV (see Figure 30.3-2).

The clock divider is a fractional divider. Thus, the divisor LEDC_CLK_DIV can be non-integer values. LEDC_CLK_DIV is configured according to the following equation.

$$\text{LEDC\_CLK\_DIV} = A + \frac{B}{256} \quad (30.1)$$

- The integer part A corresponds to the most significant 10 bits of `LEDC_CLK_DIV_TIMERx` (i.e., `LEDC_TIMERx_CONF_REG[22:13]`)
- The fractional part B corresponds to the least significant 8 bits of `LEDC_CLK_DIV_TIMERx` (i.e., `LEDC_TIMERx_CONF_REG[12:5]`)

When the fractional part B is 0, LEDC_CLK_DIV is equivalent to an integer divisor (i.e., an integer prescaler). In other words, a `ref_pulsex` clock pulse is generated after every A number of LEDC_CLK clock pulses.

However, when B is not 0, LEDC_CLK_DIV becomes a non-integer divisor. The clock divider implements non-integer frequency division by alternating between A and $(A+1)$ LEDC_CLK clock pulses per `ref_pulsex` clock pulse. In this way, the average frequency of `ref_pulsex` clock pulse will be the desired frequency (i.e. the non-integer divided frequency). For every 256 `ref_pulsex` clock pulses:

- A number of B `ref_pulsex` clock pulses are generated every $(A+1)$ LEDC_CLK clock pulses
- A number of $(256-B)$ `ref_pulsex` clock pulses are generated every A LEDC_CLK clock pulses
- The `ref_pulsex` clock pulses generated every $(A+1)$ pulses are evenly distributed amongst those generated every A pulses

Figure 30.4-1 illustrates the relation between LEDC_CLK clock pulses and ref_pulse_x clock pulses when LEDC_CLK_DIV is a non-integer value.

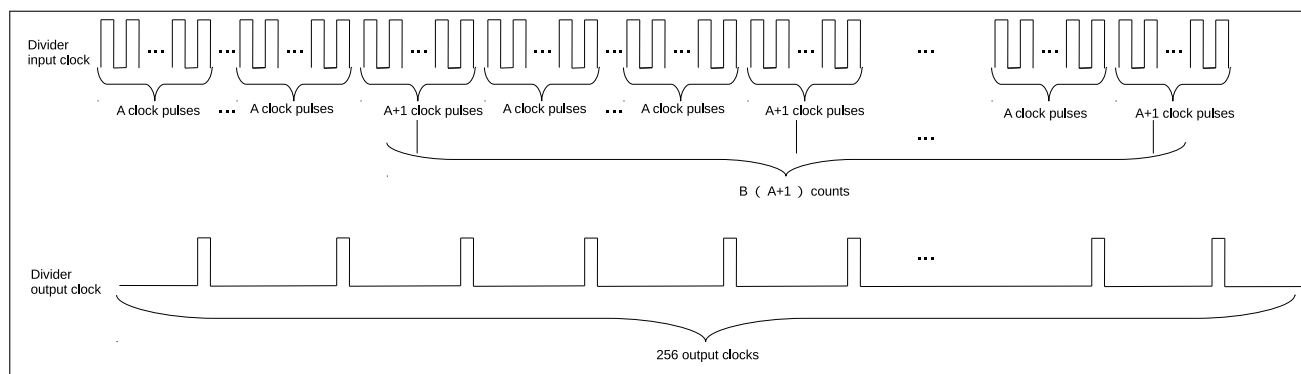


Figure 30.4-1. Frequency Division When LEDC_CLK_DIV is a Non-Integer Value

To change the timer's clock divisor at runtime, first configure the `LEDC_CLK_DIV_TIMERx` field, and then set the `LEDC_TIMERx_PARA_UP` field to apply the new configuration. This will cause the newly configured values to take effect upon the next overflow of the counter. The `LEDC_TIMERx_PARA_UP` field will be automatically cleared by hardware.

30.4.1.3 20-Bit Counter

Each timer contains a 20-bit timebase counter that uses ref_pulse_x as its reference clock (see Figure 30.3-2). The `LEDC_TIMERx_DUTY_RES` field configures the actual used bit width of this 20-bit counter. Hence, the maximum resolution of the PWM signal is 20 bits. The counter counts up to $2^{\text{LEDC\_TIMER}_x\text{\_DUTY\_RES}} - 1$, overflows and begins counting from 0 again. The counter's value can be read, reset, and suspended by software. Figure 30.4-2 shows the relationship between the counter and PWM resolution.

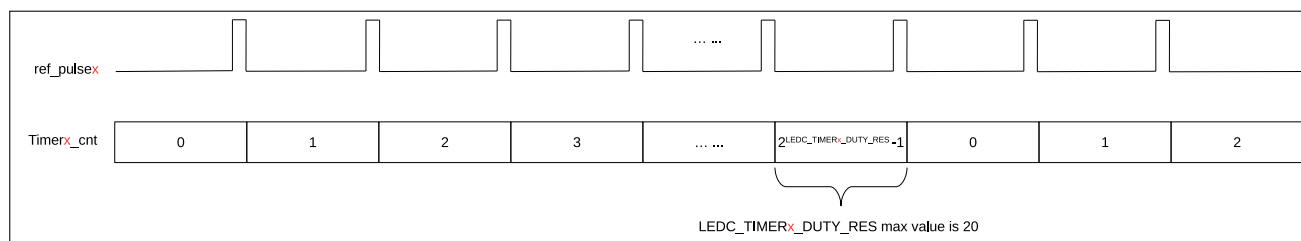


Figure 30.4-2. Relationship Between Counter And Resolution

Every time the counter overflows, it can trigger the `LEDC_TIMERx_OVF_INT` interrupt (generated automatically by hardware without configuration). It can also be configured to trigger `LEDC_OVF_CNT_CHn_INT` interrupt after overflowing `LEDC_OVF_NUM_CHn + 1` times. To configure `LEDC_OVF_CNT_CHn_INT` interrupt, please:

1. Configure `LEDC_TIMER_SEL_CHn` to select the timer for the PWM generator
2. Enable the overflow counter by setting `LEDC_OVF_CNT_EN_CHn`
3. Configure `LEDC_OVF_NUM_CHn` with the number of counter overflows (that triggers an interrupt) minus 1
4. Enable the overflow interrupt by setting `LEDC_OVF_CNT_CHn_INT_ENA`
5. Configure `LEDC_TIMERx_DUTY_RES` to specify the counter bit width for the selected Timer_x and wait for a `LEDC_OVF_CNT_CHn_INT` interrupt

To change the overflow value at runtime, first set the `LEDC_TIMERx_DUTY_RES` field, and then set the `LEDC_TIMERx_PARA_UP` field. This will cause the newly configured values to take effect upon the next overflow of the counter. If `LEDC_OVF_CNT_EN_CHn` field is reconfigured, `LEDC_PARA_UP_CHn` should be set to apply the new configuration. In summary, these configuration values need to be updated by setting `LEDC_TIMERx_PARA_UP` or `LEDC_PARA_UP_CHn`. `LEDC_TIMERx_PARA_UP` and `LEDC_PARA_UP_CHn` will be automatically cleared by hardware.

Referring to Figure 30.3-2, the frequency of a PWM generator output signal (`sig_outn`) is dependent on the frequency of the timer's clock source `LEDC_CLK`, the clock divisor `LEDC_CLK_DIV`, and the duty resolution (counter width) `LEDC_TIMERx_DUTY_RES`:

$$f_{\text{PWM}} = \frac{f_{\text{LEDC\_CLK}}}{\text{LEDC\_CLK\_DIV} \cdot 2^{\text{LEDC\_TIMERx\_DUTY\_RES}}} \quad (30.2)$$

Based on the formula above, the desired duty resolution can be calculated as follows:

$$\text{LEDC\_TIMERx\_DUTY\_RES} = \log_2 \left(\frac{f_{\text{LEDC\_CLK}}}{f_{\text{PWM}} \cdot \text{LEDC\_CLK\_DIV}} \right) \quad (30.3)$$

Table 30.4-1 lists the commonly-used frequencies and their corresponding resolutions.

Table 30.4-1. Commonly-used Frequencies and Resolutions

LEDC_CLK	PWM Frequency	Highest Resolution (bit) ¹	Lowest Resolution (bit) ²
REF_80M_CLK (80 MHz)	1 kHz	16	6
REF_80M_CLK (80 MHz)	5 kHz	13	3
REF_80M_CLK (80 MHz)	10 kHz	12	2
XTAL_40M_CLK (40 MHz)	1 kHz	15	5
XTAL_40M_CLK (40 MHz)	4 kHz	13	3
FOSC_20M_CLK (20 MHz)	1 kHz	14	4
FOSC_20M_CLK (20 MHz)	2 kHz	13	3

¹ The highest resolution is calculated when the clock divisor `LEDC_CLK_DIV` is 1. If the highest resolution calculated by the formula is higher than the counter's width 20 bits, then the highest resolution should be 20 bits.

² The lowest resolution is calculated when the clock divisor `LEDC_CLK_DIV` is $1023 + \frac{255}{256}$. If the lowest resolution calculated by the formula is lower than 0, then the lowest resolution should be 1.

30.4.2 PWM Generators

To generate a PWM signal, a PWM generator (`PWMn`) needs a timer (`Timerx`). Each PWM generator can be configured separately by setting `LEDC_TIMER_SEL_CHn` to use one of four timers to generate the PWM output.

As shown in Figure 30.3-2, each PWM generator has a comparator and two multiplexers. A PWM generator compares the timer's 20-bit counter value (`Timerx_cnt`) to two trigger values `Hpointn` and `Lpointn`. When the timer's counter value is equal to `Hpointn` or `Lpointn`, the PWM signal is high or low, respectively, as described below:

- If `Timer $x$ _cnt == Hpoint $n$` , `sig_out $n$` is 1.
- If `Timer $x$ _cnt == Lpoint $n$` , `sig_out $n$` is 0.

Figure 30.4-3 illustrates how `Hpoint $n$` and `Lpoint $n$` are used to generate a fixed duty cycle PWM output signal.

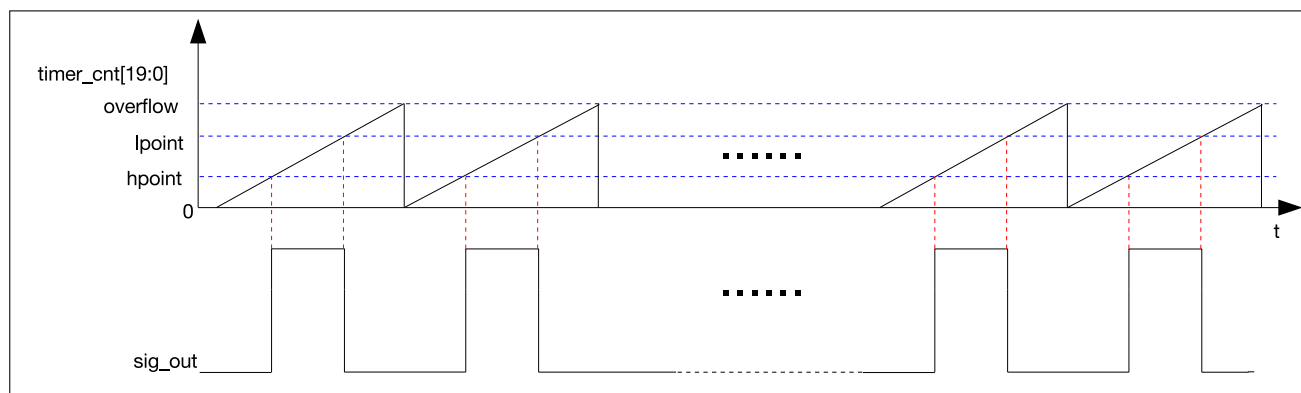


Figure 30.4-3. LED PWM Output Signal Diagram

For a particular PWM generator (PWM n), its `Hpoint $n$` is sampled from the `LEDC_HPOINT_CH $n$` field each time the selected timer's counter overflows. Likewise, `Lpoint $n$` is also sampled on every counter overflow and is calculated from the sum of the `LEDC_DUTY_CH $n$ [24:4]` and `LEDC_HPOINT_CH $n$` fields. By setting `Hpoint $n$` and `Lpoint $n$` via the `LEDC_HPOINT_CH $n$` and `LEDC_DUTY_CH $n$ [24:4]` fields, the relative phase and duty cycle of the PWM output can be set.

The PWM output signal (`sig_out $n$`) is enabled by setting `LEDC_SIG_OUT_EN_CH $n$` . When `LEDC_SIG_OUT_EN_CH $n$` is cleared, PWM signal output is disabled, and the output signal (`sig_out $n$`) will output a constant level specified by `LEDC_IDLE_LV_CH $n$` .

The bits `LEDC_DUTY_CH $n$ [3:0]` are used to dither the duty cycles of the PWM output signal (`sig_out $n$`) by periodically altering the duty cycle of `sig_out $n$` . When `LEDC_DUTY_CH $n$ [3:0]` is not 0, then for every 16 cycles of `sig_out $n$` , `LEDC_DUTY_CH $n$ [3:0]` of those cycles will have PWM pulses that are one timer tick longer than the other ($16 - \text{LEDC\_DUTY\_CH}[3:0]$) cycles. For instance, if `LEDC_DUTY_CH $n$ [24:4]` is set to 10 and `LEDC_DUTY_CH $n$ [3:0]` is set to 5, then 5 of 16 cycles will have a PWM pulse with a duty value of 11 and the rest of the 16 cycles will have a PWM pulse with a duty value of 10. The average duty cycle after 16 cycles is 10.3125.

If fields `LEDC_TIMER_SEL_CH $n$` , `LEDC_HPOINT_CH $n$` , `LEDC_DUTY_CH $n$ [24:4]`, and `LEDC_SIG_OUT_EN_CH $n$` are reconfigured, `LEDC_PARA_UP_CH $n$` must be set to apply the new configuration. This will cause the newly configured values to take effect upon the next overflow of the counter. `LEDC_PARA_UP_CH $n$` field will be automatically cleared by hardware.

30.4.3 Duty Cycle Fading

The PWM generators can fade the duty cycle of a PWM output signal (i.e. gradually change the duty cycle from one value to another). Each PWM generator can have up to 16 duty cycle ranges, which can be independently configured in terms of fading direction (increase or decrease), fading amount, the number of fades, and fading frequency. If Duty Cycle Fading is enabled, every range's `Lpoint $n$` value will change according to its fading configuration.

Please note that during the duty cycle fading process, if the duty cycle of a PWM output signal (PWM_n) can reach 100%, its LEDC_HPOINT_CH_n needs to be set to 0 to ensure the correctness of Lpoint_n.

30.4.3.1 Linear Duty Cycle Fading

Linear fading PWM signals can be generated by configuring the direction, fading amount, the number of fades, and fading frequency of the first duty cycle range.

The programming procedures will be described in detail in the section 30.7.

After the procedures, the PWM generator can fade the duty cycle of a PWM signal once per LEDC_CH_n_GAMMA_DUTY_CYCLE times of counter overflows. Every time when the PWM signal is faded, Lpoint_n increases or decreases (configured by LEDC_CH_n_GAMMA_DUTY_INC) by LEDC_CH_n_GAMMA_SCALE, and the duty cycle increases or decreases (configured by LEDC_CH_n_GAMMA_DUTY_INC) by

$$\frac{\text{LEDC\_CH}_n\_\text{GAMMA\_SCALE}}{\text{LEDC\_TIMER}_x\_\text{DUTY\_RES}} \quad (30.4)$$

The duty cycle is faded for LEDC_CH_n_GAMMA_DUTY_NUM times. After that, the PWM generator stops fading and keeps outputting signals at this duty cycle. Upon each fading the duty cycle increases or decreases by the same amount, and therefore the PWM signal is a linear fading signal.

Figure 30.4-4 shows a linear fading PWM signal.

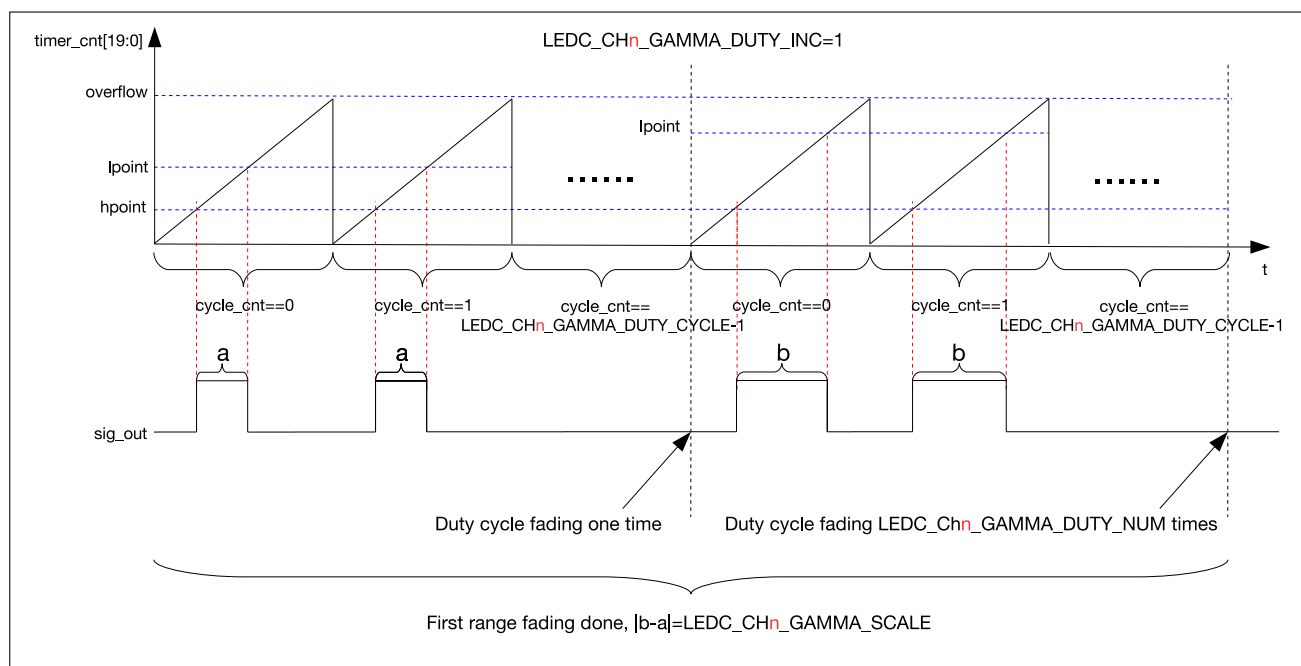


Figure 30.4-4. Output Signal of Linear Duty Cycle Fading

30.4.3.2 Gamma Curve Fading

Gamma curve fading PWM signals can be generated by configuring the fading direction, fading amount, the number of fades and fading frequency of multiple duty cycle fading ranges.

The programming procedures will be described in detail in the section 30.7.

After the procedures, the PWM generator can generate a PWM signal with `LEDC_CH $n$ _GAMMA_ENTRY_NUM` ranges. The duty cycle of the PWM signal fades according to the configurations of range 0 first, and then range 1, till range (`LEDC_CH $n$ _GAMMA_ENTRY_NUM` – 1) (the last range) where Duty Cycle Fading ends. The PWM signal fades independently in each range. According to the configuration of each range, every time when the counter overflows for `LEDC_CH $n$ _GAMMA_DUTY_CYCLE` times, `Lpoint $n$` increases or decreases (configured by `LEDC_CH $n$ _GAMMA_DUTY_INC`) by `LEDC_CH $n$ _GAMMA_SCALE`, and accordingly the duty cycle increases or decreases (configured by `LEDC_CH $n$ _GAMMA_DUTY_INC`) by

$$\frac{\text{LEDC\_CH}_n\text{\_GAMMA\_SCALE}}{\text{LEDC\_TIMER}_x\text{\_DUTY\_RES}} \quad (30.5)$$

After the duty cycle fades for `LEDC_CH $n$ _GAMMA_DUTY_NUM` times in a range, Duty Cycle Fading in this range finishes.

When Duty Cycle Fading finishes in all ranges (the number of ranges is specified by `LEDC_CH $n$ _GAMMA_ENTRY_NUM`), the PWM signal stops fading and keeps the duty cycle of the last fade. Given that the duty cycle fades differently and linearly in each range, several linear fading ranges would be fitted to a gamma curve.

Figure 30.4-5 illustrates a gamma curve fading PWM signal.

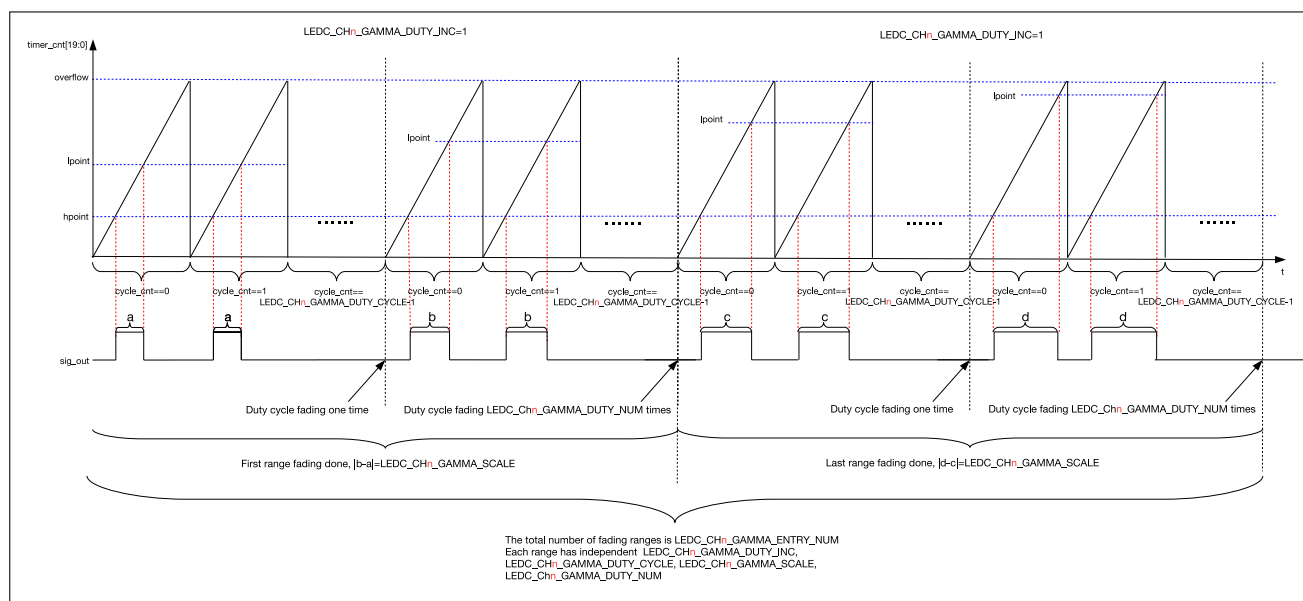


Figure 30.4-5. Output Signal of Gamma Curve Fading

30.4.3.3 Suspend and Resume Duty Cycle Fading

To suspend Duty Cycle Fading that has already been started, write 1 to the `LEDC_CH $n$ _GAMMA_PAUSE` field of the `LEDC_CH $n$ _GAMMA_CONF_REG` register. Once `LEDC_CH $n$ _GAMMA_PAUSE` is set to 1, the PWM signal keeps the duty cycle of the most recent fade.

To resume Duty Cycle Fading, write 1 to the `LEDC_CH $n$ _GAMMA_RESUME` field of the `LEDC_CH $n$ _GAMMA_CONF_REG` register. Once `LEDC_CH $n$ _GAMMA_RESUME` is set to 1, the PWM signal resumes fading from the range where the suspension occurs, until fading in the last range finishes. The fading will continue from the state when it was paused until all the ranges complete duty cycle fading (when

`LEDC_CH $n$ _GAMMA_RESUME` is set to 1, `LEDC_CH $n$ _GAMMA_PAUSE` is cleared automatically by hardware.

30.5 Event Task Matrix Feature

The LEDC on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows LEDC's ETM tasks to be triggered by any peripherals' ETM events, or LEDC's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to LEDC. For more information, please refer to [Chapter 8 Event Task Matrix \(ETM\)](#).

ETM-related events and tasks are enabled by configuring corresponding fields of `LEDC_EVT_TASK_ENO_REG`, `LEDC_EVT_TASK_EN1_REG` and `LEDC_EVT_TASK_EN2_REG` registers. For the correspondence between events, tasks, and fields, Please refer to [Section 30.9](#).

LEDC can receive the following ETM tasks:

- `LEDC_TASK_DUTY_SCALE_UPDATE_CH $n$` : If the `LEDC_TASK_DUTY_SCALE_UPDATE_CH $n$ _EN` field is enabled, upon receiving the `LEDC_TASK_DUTY_SCALE_UPDATE_CH $n$` task, PWM n generates fading PWM signals according to the newly configured `LEDC_CH $n$ _GAMMA_SCALE` field.
- `LEDC_TASK_TIMER $x$ _RES_UPDATE`: If the `LEDC_TASK_TIMER $x$ _RES_UPDATE_EN` field is enabled, upon receiving the `LEDC_TASK_TIMER $x$ _RES_UPDATE` task, Timer x updates its counter's overflow value to the value configured in the `LEDC_TIMER $x$ _DUTY_RES` field at the next overflow of the counter.
- `LEDC_TASK_TIMER $x$ _CAP`: If the `LEDC_TASK_TIMER $x$ _CAP_EN` field is enabled, upon receiving the `LEDC_TASK_TIMER $x$ _CAP` task, Timer x captures its counter's value, and stores the value into the `LEDC_TIMER $x$ _CNT_CAP` field of register `LEDC_TIMER $x$ _CNT_CAP_REG`.
- `LEDC_TASK_SIG_OUT_DIS_CH $n$` : If the `LEDC_TASK_SIG_OUT_DIS_CH $n$ _EN` field is enabled, upon receiving the `LEDC_TASK_SIG_OUT_DIS_CH $n$` task, PWM n 's signal output is disabled, and the output signal (`sig_out $n$`) outputs a constant level as specified by field `LEDC_IDLE_LV_CH $n$` , as shown in [Figure 30.3-2](#).
- `LEDC_TASK_OVF_CNT_RST_CH $n$` : If the `LEDC_TASK_OVF_CNT_RST_CH $n$ _EN` field is enabled, upon receiving the `LEDC_TASK_OVF_CNT_RST_CH $n$` task, PWM n timer's overflow counter is reset to 0.
- `LEDC_TASK_TIMER $x$ _RST`: If the `LEDC_TASK_TIMER $x$ _RST_EN` field is enabled, upon receiving the `LEDC_TASK_TIMER $x$ _RST` task, Timer x 's counter is reset to 0.
- `LEDC_TASK_TIMER $x$ _RESUME` and `LEDC_TASK_TIMER $x$ _PAUSE`: If the `LEDC_TASK_TIMER $x$ _PAUSE_RESUME_EN` field is enabled, upon receiving the `LEDC_TASK_TIMER $x$ _RESUME` and `LEDC_TASK_TIMER $x$ _PAUSE` task, Timer x is suspended and resumed alternately. That is, when the task is received, Timer x is paused; and when the task is received again, Timer x is resumed.
- `LEDC_TASK_GAMMA_RESTART_CH $n$` : If the `LEDC_TASK_GAMMA_RESTART_CH $n$ _EN` field is enabled, upon receiving the `LEDC_TASK_GAMMA_RESTART_CH $n$` task, the PWM n restarts to generate the fading PWM signal.
- `LEDC_TASK_GAMMA_PAUSE_CH $n$` : If the `LEDC_TASK_GAMMA_PAUSE_CH $n$ _EN` field is enabled, upon receiving the `LEDC_TASK_GAMMA_PAUSE_CH $n$` task, PWM n suspends Duty Cycle Fading at the next timer overflow. That is, after the task has been received, PWM n keeps the duty cycle of the last fade.

- LEDC_TASK_GAMMA_RESUME_CH n : If the [LEDC_TASK_GAMMA_RESUME_CH \$n\$ _EN](#) field is enabled, upon receiving the LEDC_TASK_GAMMA_RESUME_CH n task, PWM n resumes Duty Cycle Fading at the next counter overflow. That is, after the task has been received, PWM n resumes fading from the range where the suspension occurs.

LEDC can generate the following ETM events:

- LEDC_EVT_DUTY_CHNG_END_CH n : Generated when the [LEDC_EVT_DUTY_CHNG_END_CH \$n\$ _EN](#) field is enabled, and PWM n has finished Duty Cycle Fading.
- LEDC_EVT_OVF_CNT_PLS_CH n : Generated when the [LEDC_EVT_OVF_CNT_PLS_CH \$n\$ _EN](#) field is enabled and when PWM n timer's counter overflows for [LEDC_OVF_NUM_CH \$n\$](#) + 1 times.
- LEDC_EVT_TIME_OVF_TIMER x : Generated when the [LEDC_EVT_TIME_OVF_TIMER \$x\$ _EN](#) field is enabled and Timer x 's counter overflows.
- LEDC_EVT_TIMER x _CMP: Generated when the [LEDC_EVT_TIMER \$x\$ _CMP_EN](#) field is enabled and the value of Timer x 's counter reaches that of the [LEDC_TIMER \$x\$ _CMP](#) field of register [LEDC_TIMER \$x\$ _CMP_REG](#).

In practical applications, LEDC's ETM events can trigger its own ETM tasks. For example, LEDC_EVT_DUTY_CHNG_END_CH n event can trigger the LEDC_TASK_GAMMA_RESTART_CH n task, thus starting the next fading directly after the current fading is completed.

30.6 Interrupts

ESP32-C5's LEDC can generate the LEDC_INT interrupt signal, which will be sent to the [Interrupt Matrix](#).

Interrupt signal LEDC_INT can be generated by the following internal interrupt sources:

- LEDC_OVF_CNT_CH n _INT: Triggered when the timer counter overflows for [LEDC_OVF_NUM_CH \$n\$](#) + 1 times and the register [LEDC_OVF_CNT_EN_CH \$n\$](#) is set to 1.
- LEDC_DUTY_CHNG_END_CH n _INT: Triggered when a fade on PWM n has finished.
- LEDC_TIMER x _OVF_INT: Triggered when Timer x has reached its maximum counter value.

The above interrupt sources can only be triggered when the interrupt enable register [XXX_ENA](#) is set to 1. Write 1 to [XXX_CLR](#) will clear the interrupt ([XXX](#) is the interrupt source name).

30.7 Programming Procedures

The configuration process of the LED PWM controller to generate PWM signals is as follows:

1. Configure the Timer x according to section [30.4.1](#). After that, the clock source, divider, and counter are ready.
2. Configure the PWM n generator according to section [30.4.2](#). After that, PWM n selects one of the four timers and the phase and duty cycle of the PWM n output is set.
3. Choose the corresponding configuration process based on the different duty cycle fading types:
 - (a) Linear duty cycle fading:
 - i. Configure the [LEDC_DUTY_CH \$n\$](#) field with the initial value of [Lpoint \$n\$](#) .

- ii. Set the `LEDC_DUTY_START_CHn` field to enable Duty Cycle Fading. When this field is cleared, Duty Cycle Fading will be disabled.
 - iii. Write the linear duty cycle fading configuration to (`LEDC_CHn_MEM` starting address + duty cycle range number * 4), the `LEDC_CHn_MEM` starting address is provided in Table 30.8-1. Please note that the following bit names are just for easier description, and there are no such register fields.
 - A. Bit 0 of the configuration data is used to configure the direction (hereafter referred to as `LEDC_CHn_GAMMA_DUTY_INC`). When it is set or cleared, the `Lpointn` will increase or decrease in the current configured range.
 - B. Bit 1 to 10 of the configuration data is used to configure the number of times the counter overflows per an increment or decrement of `Lpointn` (hereafter referred to as `LEDC_CHn_GAMMA_DUTY_CYCLE`). In other words, `Lpointn` will increase or decrease after the counter overflows for the configured number of times.
 - C. Bit 11 to 20 of the configuration data is used to configure the amount by which `Lpointn` increase or decrease in the current configured range (hereafter referred to as `LEDC_CHn_GAMMA_SCALE`).
 - D. Bit 21 to 30 of the configuration data is used to configure the number of fades in the current configured range (hereafter referred to as `LEDC_CHn_GAMMA_DUTY_NUM`).
 - E. The duty cycle range number (from 0 to 15) specifies to which range the above configuration data apply. For linear duty cycle fading only the first range needs to be configured, so the duty cycle range number should be configured as 0.
 - iv. Configure the number of ranges per each fading (1 in this case) via the `LEDC_CHn_GAMMA_ENTRY_NUM` field of the `LEDC_CHn_GAMMA_CONF_REG`. Once the specified number of ranges have been faded, Duty Cycle Fading stops and the PWM generator triggers the `LEDC_DUTY_CHNG_END_CHn_INT` interrupt. For linear duty cycle fading there is only one duty cycle range (i.e. the first one), so configure `LEDC_CHn_GAMMA_ENTRY_NUM` as 1.
 - v. Set the `LEDC_PARA_UP_CHn` field to apply the above configurations. After this field is set, the configurations for Duty Cycle Fading will take effect upon the next overflow of the counter, and the PWM generator will output a linear fading PWM signal following configurations. `LEDC_PARA_UP_CHn` field will be automatically cleared by hardware.
- (b) Gamma curve fading:
- i. The same as Step 1 in Section 30.4.3.1.
 - ii. The same as Step 2 in Section 30.4.3.1.
 - iii. Configure multiple duty cycle ranges with the following steps. Write the gamma curve fading configuration to (`LEDC_CHn_MEM` starting address + duty cycle range number * 4), the `LEDC_CHn_MEM` starting address is provided in Table 30.8-1.
 - A. Bit 0 of the configuration data is used to configure the direction (hereafter referred to as `LEDC_CHn_GAMMA_DUTY_INC`). When it is set or cleared, the `Lpointn` will increment or decrement in the current configured range.

- B. Bit 1 to 10 of the configuration data is used to configure the number of times the counter overflows per an increase or decrease of Lpoint n (hereafter referred to as LEDC_CH n _GAMMA_DUTY_CYCLE). In other words, Lpoint n will increase or decrease after the counter overflows for the configured number of times.
 - C. Bit 11 to 20 of the configuration data is used to configure the amount by which Lpoint n increase or decrease in the current configured range (hereafter referred to as LEDC_CH n _GAMMA_SCALE).
 - D. Bit 21 to 30 of the configuration data is used to configure the number of fades in the current configured range (hereafter referred to as LEDC_CH n _GAMMA_DUTY_NUM).
 - E. The duty cycle range number (from 0 to 15) specifies to which range the above configuration data apply. For gamma curve fading, it must start from 0 and increase by 1 for the next range to be configured.
 - F. Once the above procedures are finished, the configuration for one range is complete. Other ranges are configured by repeating the same set of procedures. You can configure any number of ranges from 0 to 15, and each can be configured independently.
- iv. After all required ranges are configured, write the total number of ranges configured in Step 3 to the LEDC_CH n _GAMMA_ENTRY_NUM field of the LEDC_CH n _GAMMA_CONF_REG register.
 - v. Set the LEDC_PARA_UP_CH n field to apply the above configuration. After this field is set, the configurations for duty cycle fading will take effect upon the next overflow of the counter, and the PWM generator will output a gamma curve fading PWM signal following the configurations. LEDC_PARA_UP_CH n field will be automatically cleared by hardware.

After the above procedures, the LED PWM controller will generate the desired PWM signals.

At any time, duty cycle fading can be suspended or resumed, more details can be found in section 30.4.3.3. If the duty cycle fading configurations need to be changed before the PWM signals stop fading (that is, the LEDC_DUTY_CHNG_END_CH n _INT has not been triggered), the duty cycle fading needs to be suspended before changing the configurations. However, if the duty cycle fading configurations need to be changed after the PWM signals stop fading (that is, the LEDC_DUTY_CHNG_END_CH n _INT has been triggered), the configurations can be changed directly.

At any time, PWM signal output can be enabled or disabled by software, and the output signal level can be changed, more details can be found in section 30.4.2.

30.8 Memory Blocks

Each LEDC channel has a memory to store duty cycle fading configuration data. Each memory can store configuration data for up to 16 ranges. The addresses in this section are relative to LED PWM Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Table 30.8-1. LEDC Memory Blocks

Name	Description	Size	Starting Address	Ending Address	Access
LEDC_CH n _MEM	Memory of PWM n	64 B	0x400 + PWM n * 64	0x400 + (PWM n + 1) * 64	R/W

30.9 Register Summary

The addresses in this section are relative to LED PWM Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Register			
LEDC_CHO_CONFO_REG	Configuration register 0 for channel 0	0x0000	varies
LEDC_CHO_HPOINT_REG	High point register for channel 0	0x0004	R/W
LEDC_CHO_DUTY_REG	Initial duty cycle register for channel 0	0x0008	R/W
LEDC_CHO_CONF1_REG	Configuration register 1 for channel 0	0x000C	R/W/SC
LEDC_CH1_CONFO_REG	Configuration register 0 for channel 1	0x0014	varies
LEDC_CH1_HPOINT_REG	High point register for channel 1	0x0018	R/W
LEDC_CH1_DUTY_REG	Initial duty cycle register for channel 1	0x001C	R/W
LEDC_CH1_CONF1_REG	Configuration register 1 for channel 1	0x0020	R/W/SC
LEDC_CH2_CONFO_REG	Configuration register 0 for channel 2	0x0028	varies
LEDC_CH2_HPOINT_REG	High point register for channel 2	0x002C	R/W
LEDC_CH2_DUTY_REG	Initial duty cycle register for channel 2	0x0030	R/W
LEDC_CH2_CONF1_REG	Configuration register 1 for channel 2	0x0034	R/W/SC
LEDC_CH3_CONFO_REG	Configuration register 0 for channel 3	0x003C	varies
LEDC_CH3_HPOINT_REG	High point register for channel 3	0x0040	R/W
LEDC_CH3_DUTY_REG	Initial duty cycle register for channel 3	0x0044	R/W
LEDC_CH3_CONF1_REG	Configuration register 1 for channel 3	0x0048	R/W/SC
LEDC_CH4_CONFO_REG	Configuration register 0 for channel 4	0x0050	varies
LEDC_CH4_HPOINT_REG	High point register for channel 4	0x0054	R/W
LEDC_CH4_DUTY_REG	Initial duty cycle register for channel 4	0x0058	R/W
LEDC_CH4_CONF1_REG	Configuration register 1 for channel 4	0x005C	R/W/SC
LEDC_CH5_CONFO_REG	Configuration register 0 for channel 5	0x0064	varies
LEDC_CH5_HPOINT_REG	High point register for channel 5	0x0068	R/W
LEDC_CH5_DUTY_REG	Initial duty cycle register for channel 5	0x006C	R/W
LEDC_CH5_CONF1_REG	Configuration register 1 for channel 5	0x0070	R/W/SC
LEDC_TIMER0_CONF_REG	Timer 0 configuration register	0x00A0	varies
LEDC_TIMER1_CONF_REG	Timer 1 configuration register	0x00A8	varies
LEDC_TIMER2_CONF_REG	Timer 2 configuration register	0x00B0	varies
LEDC_TIMER3_CONF_REG	Timer 3 configuration register	0x00B8	varies
LEDC_CHO_GAMMA_CONF_REG	LEDC channel 0 gamma config register	0x0100	varies
LEDC_CH1_GAMMA_CONF_REG	LEDC channel 1 gamma config register	0x0104	varies
LEDC_CH2_GAMMA_CONF_REG	LEDC channel 2 gamma config register	0x0108	varies
LEDC_CH3_GAMMA_CONF_REG	LEDC channel 3 gamma config register	0x010C	varies
LEDC_CH4_GAMMA_CONF_REG	LEDC channel 4 gamma config register	0x0110	varies
LEDC_CH5_GAMMA_CONF_REG	LEDC channel 5 gamma config register.	0x0114	varies
LEDC_EVT_TASK_ENO_REG	LEDC event task enable bit register 0	0x0120	R/W
LEDC_EVT_TASK_EN1_REG	LEDC event task enable bit register 1	0x0124	R/W

Name	Description	Address	Access
LEDC_EVT_TASK_EN2_REG	LEDC event task enable bit register 2	0x0128	R/W
LEDC_TIMER0_CMP_REG	LEDC timer 0 compare value register	0x0140	R/W
LEDC_TIMER1_CMP_REG	LEDC timer 1 compare value register	0x0144	R/W
LEDC_TIMER2_CMP_REG	LEDC timer 2 compare value register	0x0148	R/W
LEDC_TIMER3_CMP_REG	LEDC timer 3 compare value register	0x014C	R/W
LEDC_CONF_REG	LEDC global configuration register	0x0170	R/W
Status Register			
LEDC_CHO_DUTY_R_REG	Current duty cycle register for channel 0	0x0010	RO
LEDC_CH1_DUTY_R_REG	Current duty cycle register for channel 1	0x0024	RO
LEDC_CH2_DUTY_R_REG	Current duty cycle register for channel 2	0x0038	RO
LEDC_CH3_DUTY_R_REG	Current duty cycle register for channel 3	0x004C	RO
LEDC_CH4_DUTY_R_REG	Current duty cycle register for channel 4	0x0060	RO
LEDC_CH5_DUTY_R_REG	Current duty cycle register for channel 5	0x0074	RO
LEDC_TIMER0_VALUE_REG	Timer 0 current counter value register	0x00A4	RO
LEDC_TIMER1_VALUE_REG	Timer 1 current counter value register	0x00AC	RO
LEDC_TIMER2_VALUE_REG	Timer 2 current counter value register	0x00B4	RO
LEDC_TIMER3_VALUE_REG	Timer 3 current counter value register	0x00BC	RO
LEDC_TIMER0_CNT_CAP_REG	LEDC timer 0 captured count value register	0x0150	RO
LEDC_TIMER1_CNT_CAP_REG	LEDC timer 1 captured count value register	0x0154	RO
LEDC_TIMER2_CNT_CAP_REG	LEDC timer 2 captured count value register	0x0158	RO
LEDC_TIMER3_CNT_CAP_REG	LEDC timer 3 captured count value register	0x015C	RO
Interrupt Register			
LEDC_INT_RAW_REG	Interrupt raw status register	0x00C0	R/WTC/SS
LEDC_INT_ST_REG	Interrupt masked status register	0x00C4	RO
LEDC_INT_ENA_REG	Interrupt enable register	0x00C8	R/W
LEDC_INT_CLR_REG	Interrupt clear register	0x00CC	WT
Version Register			
LEDC_DATE_REG	Version control register	0x0174	R/W

30.10 Registers

The addresses in this section are relative to LED PWM Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 30.1. LEDC_CH n _CONF0_REG (n : 0-5) (0x0000+0x14* n)

(reserved)																LEDC_OVF_CNT_RESET_CH ⁿ LEDC_OVF_CNT_EN_CH ⁿ				LEDC_OVF_NUM_CH ⁿ				LEDC_PARA_UP_CH ⁿ LEDC_IDLE_LV_CH ⁿ LEDC_SIG_OUT_EN_CH ⁿ LEDC_TIMER_SEL_CH ⁿ			
31													17	16	15	14					5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0	0	0	0	Reset			

LEDC_TIMER_SEL_CH n Configures which timer is channel n selected.

- 0: Select Timer 0
 - 1: Select Timer 1
 - 2: Select Timer 2
 - 3: Select Timer 3
- (R/W)

LEDC_SIG_OUT_EN_CH n Configures whether to enable signal output on channel n .

- 0: Signal output disable
 - 1: Signal output enable
- (R/W)

LEDC_IDLE_LV_CH n Configures the output value when channel n is inactive. Valid only when [LEDC_SIG_OUT_EN_CH \$n\$](#) is 0.

- 0: Output level is low
 - 1: Output level is high
- (R/W)

LEDC_PARA_UP_CH n Configures whether to update [LEDC_HPOINT_CH \$n\$](#) , [LEDC_DUTY_START_CH \$n\$](#) , [LEDC_SIG_OUT_EN_CH \$n\$](#) , [LEDC_TIMER_SEL_CH \$n\$](#) , [LEDC_OVF_CNT_EN_CH \$n\$](#) fields and duty cycle range configurations for channel n , and will be automatically cleared by hardware.

- 0: Invalid. No effect
 - 1: Update
- This bit is cleared by hardware
- (WT)

LEDC_OVF_NUM_CH n Configures the maximum times of overflow minus 1. The [LEDC_OVF_CNT_CH \$n\$ _INT](#) interrupt will be triggered when channel n overflows for ([LEDC_OVF_NUM_CH \$n\$](#) + 1) times. (R/W)

LEDC_OVF_CNT_EN_CH n Configures whether to enable the ovf_cnt of channel n .

- 0: Disable
 - 1: Enable
- (R/W)

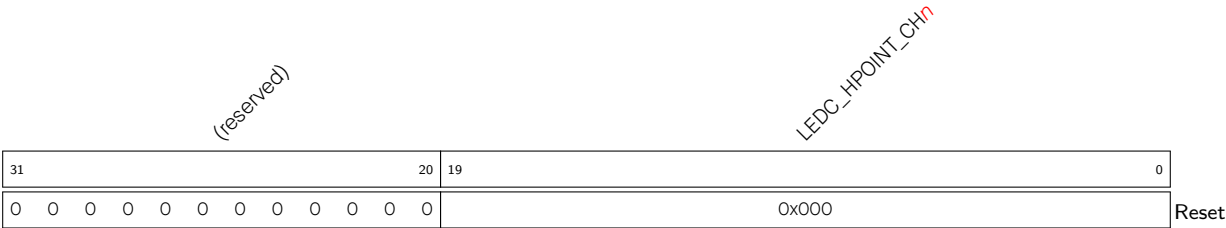
Continued on the next page...

Register 30.1. LEDC_CHn_CONFO_REG (n: 0-5) (0x0000+0x14*n)

Continued from the previous page...

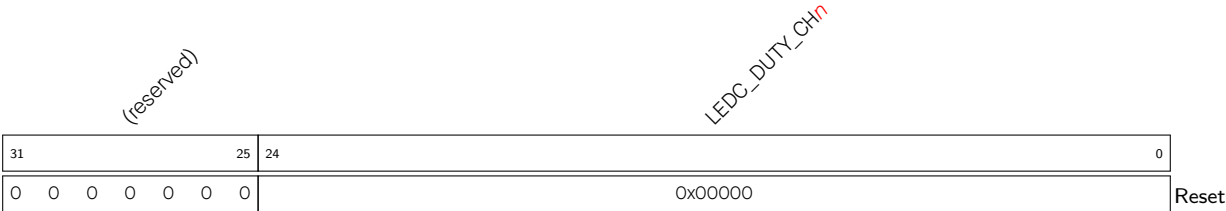
LEDC_OVF_CNT_RESET_CHn Configures whether to reset the ovf_cnt of channel *n*.
0: Invalid. No effect
1: Reset the ovf_cnt
(WT)

Register 30.2. LEDC_CHn_HPOINT_REG (n: 0-5) (0x0004+0x14*n)



LEDC_HPOINT_CHn Configures high point of signal output on channel *n*. The output value changes to high when the selected timers have reached the value specified by this register. (R/W)

Register 30.3. LEDC_CHn_DUTY_REG (n: 0-5) (0x0008+0x14*n)



LEDC_DUTY_CHn Configures the duty cycle of signal output on channel *n*. (R/W)

Register 30.4. LEDC_CH n _CONF1_REG (n : 0-5) (0x000C+0x14* n)

LEDC_DUTY_START_CHn

(reserved)

Reset

LEDC_DUTY_START_CH_n Configures whether the duty cycle fading configurations take effect.

0: Not take effect

1: Take effect

(R/W/SC)

Register 30.5. LEDC_TIMER_x_CONF_REG (x: 0-3) (0x00A0+0x8*x)

(reserved)										LEDC_TIMER ~~X~~ _PARA_UP										(reserved)										LEDC_TIMER ~~X~~ _RST										LEDC_TIMER ~~X~~ _PAUSE																				LEDC_CLK_DIV_TIMER ~~X~~																				LEDC_TIMER ~~X~~ _DUTY_RES									
31					27					26	25	24	23	22																	5					4	0																																																				
0	0	0	0	0	0	0	0	0	1	0	0x000																	0x0															Reset																																														

LEDC_TIMERx_DUTY_RES Configures the bit width of the counter in timer **x**. Valid values are 1 to 20. (R/W)

LEDC_CLK_DIV_TIMER x Configures the divisor for the divider in timer x .

The least significant eight bits represent the fractional part. The most significant ten bits represent the integer part. (R/W)

LEDC_TIMER_x_PAUSE Configures whether to pause the counter in timer *x*.

0: Normal

1: Pause

(R/W)

LEDC_TIMER x _RST Configures whether to reset timer x . The counter will show 0 after reset.

0: Not reset

1: Reset

(R/W)

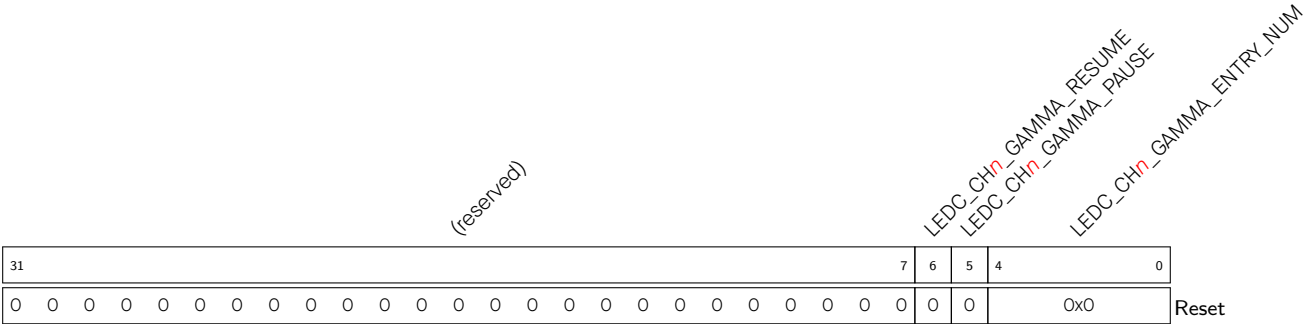
LEDC_TIMERx_PARA_UP Configures whether to update **LEDC_CLK_DIV_TIMERx** and **LEDC_TIMERx_DUTY_RES**.

0: Invalid. No effect

1: Update

(WT)

Register 30.6. LEDC_CHn_GAMMA_CONF_REG (n: 0-5) (0x0100+0x4*n)



LEDC_CHn_GAMMA_ENTRY_NUM Configures the number of duty cycle fading ranges for LEDC channel n. (R/W)

LEDC_CHn_GAMMA_PAUSE Configures whether to pause duty cycle fading of LEDC channel n.

0: Invalid. No effect

1: Pause

(WT)

LEDC_CHn_GAMMA_RESUME Configures whether to resume duty cycle fading of LEDC channel n.

0: Invalid. No effect

1: Resume

(WT)

Register 30.7. LEDC_EVT_TASK_ENO_REG (0x0120)

(reserved) LEDC_TASK_DUTY_SCALE_UPDATE_CH5_EN LEDC_TASK_DUTY_SCALE_UPDATE_CH4_EN LEDC_TASK_DUTY_SCALE_UPDATE_CH3_EN LEDC_TASK_DUTY_SCALE_UPDATE_CH2_EN LEDC_TASK_DUTY_SCALE_UPDATE_CH1_EN LEDC_EVT_TIMER3_CMP_EN LEDC_EVT_TIMER2_CMP_EN LEDC_EVT_TIMER1_CMP_EN LEDC_EVT_TIMER0_CMP_EN LEDC_EVT_TIMER3_OVF_TIMER3_EN LEDC_EVT_TIMER2_OVF_TIMER2_EN LEDC_EVT_TIMER1_OVF_TIMER1_EN LEDC_EVT_TIMER0_OVF_TIMER0_EN (reserved) LEDC_EVT_OVF_CNT_PLS_CH5_EN LEDC_EVT_OVF_CNT_PLS_CH4_EN LEDC_EVT_OVF_CNT_PLS_CH3_EN LEDC_EVT_OVF_CNT_PLS_CH2_EN LEDC_EVT_OVF_CNT_PLS_CH1_EN LEDC_EVT_OVF_CNT_PLS_CH0_EN (reserved) LEDC_EVT_DUTY_CHNG_END_CH5_EN LEDC_EVT_DUTY_CHNG_END_CH4_EN LEDC_EVT_DUTY_CHNG_END_CH3_EN LEDC_EVT_DUTY_CHNG_END_CH2_EN LEDC_EVT_DUTY_CHNG_END_CH1_EN LEDC_EVT_DUTY_CHNG_END_CH0_EN																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

LEDC_EVT_DUTY_CHNG_END_CH n _EN (n : 0-5) Configures whether to enable the LEDC_EVT_DUTY_CHNG_END_CH n event.

0: Disable

1: Enable

(R/W)

LEDC_EVT_OVF_CNT_PLS_CH n _EN (n : 0-5) Configures whether to enable the LEDC_EVT_OVF_CNT_PLS_CH n event.

0: Disable

1: Enable

(R/W)

LEDC_EVT_TIME_OVF_TIMER x _EN (x : 0-3) Configures whether to enable the LEDC_EVT_TIME_OVF_TIMER x event.

0: Disable

1: Enable

(R/W)

LEDC_EVT_TIMER x _CMP_EN (x : 0-3) Configures whether to enable the LEDC_EVT_TIMER x _CMP event.

0: Disable

1: Enable

(R/W)

LEDC_TASK_DUTY_SCALE_UPDATE_CH n _EN (n : 0-5) Configures whether to enable the LEDC_TASK_DUTY_SCALE_UPDATE_CH n task.

0: Disable

1: Enable

(R/W)

Register 30.8. LEDC_EVT_TASK_EN1_REG (0x0124)

LEDC_TASK_TIMER3_PAUSE_RESUME_EN																																LEDC_TASK_TIMER2_PAUSE_RESUME_EN																																LEDC_TASK_TIMER1_PAUSE_RESUME_EN																																LEDC_TASK_TIMER0_PAUSE_RESUME_EN																																LEDC_TASK_TIMER3_RST_EN																																LEDC_TASK_TIMER2_RST_EN																																LEDC_TASK_TIMER1_RST_EN																																(reserved)																																LEDC_TASK_OVF_CNT_RST_CH5_EN																																LEDC_TASK_OVF_CNT_RST_CH4_EN																																LEDC_TASK_OVF_CNT_RST_CH3_EN																																LEDC_TASK_OVF_CNT_RST_CH2_EN																																LEDC_TASK_OVF_CNT_RST_CH1_EN																																(reserved)																																LEDC_TASK_SIG_OUT_DIS_CH5_EN																																LEDC_TASK_SIG_OUT_DIS_CH4_EN																																LEDC_TASK_SIG_OUT_DIS_CH3_EN																																LEDC_TASK_SIG_OUT_DIS_CH2_EN																																LEDC_TASK_TIMER3_CAP_EN																																LEDC_TASK_TIMER2_CAP_EN																																LEDC_TASK_TIMER1_CAP_EN																																LEDC_TASK_TIMER0_CAP_EN																																LEDC_TASK_TIMER3_RES_UPDATE_EN																																LEDC_TASK_TIMER2_RES_UPDATE_EN																																LEDC_TASK_TIMER1_RES_UPDATE_EN																																LEDC_TASK_TIMER0_RES_UPDATE_EN																																																																																																																																																																																																																																																																								
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

LEDC_TASK_TIMER x _RES_UPDATE_EN (x : 0-3) Configures whether to enable LEDC_TASK_TIMER x _RES_UPDATE task.

0: Disable

1: Enable

(R/W)

LEDC_TASK_TIMER x _CAP_EN (x : 0-3) Configures whether to enable LEDC_TASK_TIMER x _CAP task.

0: Disable

1: Enable

(R/W)

LEDC_TASK_SIG_OUT_DIS_CH n _EN (n : 0-5) Configures whether to enable LEDC_TASK_SIG_OUT_DIS_CH n task.

0: Disable

1: Enable

(R/W)

LEDC_TASK_OVF_CNT_RST_CH n _EN (n : 0-5) Configures whether to enable LEDC_TASK_OVF_CNT_RST_CH n task.

0: Disable

1: Enable

(R/W)

LEDC_TASK_TIMER x _RST_EN (x : 0-3) Configures whether to enable LEDC_TASK_TIMER x _RST task.

0: Disable

1: Enable

(R/W)

LEDC_TASK_TIMER x _PAUSE_RESUME_EN (x : 0-3) Configures whether to enable LEDC_TASK_TIMER x _PAUSE and LEDC_TASK_TIMER x _RESUME task.

0: Disable

1: Enable

(R/W)

Register 30.9. LEDC_EVT_TASK_EN2_REG (0x0128)

(reserved)												LEDC_TASK_GAMMA_RESUME_CH5_EN LEDC_TASK_GAMMA_RESUME_CH4_EN LEDC_TASK_GAMMA_RESUME_CH3_EN LEDC_TASK_GAMMA_RESUME_CH2_EN LEDC_TASK_GAMMA_RESUME_CH1_EN (reserved) LEDC_TASK_GAMMA_PAUSE_CH5_EN LEDC_TASK_GAMMA_PAUSE_CH4_EN LEDC_TASK_GAMMA_PAUSE_CH3_EN LEDC_TASK_GAMMA_PAUSE_CH2_EN LEDC_TASK_GAMMA_PAUSE_CH1_EN (reserved) LEDC_TASK_GAMMA_RESTART_CH5_EN LEDC_TASK_GAMMA_RESTART_CH4_EN LEDC_TASK_GAMMA_RESTART_CH3_EN LEDC_TASK_GAMMA_RESTART_CH2_EN LEDC_TASK_GAMMA_RESTART_CH1_EN LEDC_TASK_GAMMA_RESTART_CH0_EN																						
31												22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset		

LEDC_TASK_GAMMA_RESTART_CH <i>n</i> _EN (<i>n</i> : 0-5)	Configures whether to enable LEDC_TASK_GAMMA_RESTART_CH <i>n</i> task.
0: Disable	
1: Enable	
(R/W)	

LEDC_TASK_GAMMA_PAUSE_CH <i>n</i> _EN (<i>n</i> : 0-5)	Configures	whether	to	enable
LEDC_TASK_GAMMA_PAUSE_CH <i>n</i> task.				
0: Disable				
1: Enable				
(R/W)				

LEDC_TASK_GAMMA_RESUME_CH n _EN (n : 0-5)	Configures whether to enable LEDC_TASK_GAMMA_RESUME_CH n task.
0: Disable	
1: Enable	
(R/W)	

Register 30.10. LEDC_TIMER_x_CMP_REG (x: 0-3) (0x0140+0x4*x)

Diagram of the 32-bit LEDC_TIMER register structure:

- Bits 31 to 20: (reserved)
- Bits 19 to 0: LEDC_TIMER (Timer value, shown as 0x000, with a Reset label)

LEDC_TIMERx_CMP Configures the comparison value for LEDC timer *x*. (R/W)

Register 30.11. LEDC_CONF_REG (0x0170)

[illegible]

LEDC_GAMMA_RAM_CLK_EN_CH n (n : 0-5) Configures whether to open LEDC channel n gamma RAM clock gate.

0: Open the clock gate only when an application writes or reads LEDC channel *n* gamma RAM

1: Force open the clock gate for LEDC channel *n* gamma RAM

(R/W)

LEDC_CLK_EN Configures whether to open the register clock gate.

0: Open the clock gate only when an application writes registers

1: Force open the clock gate for register

(R/W)

Register 30.12. LEDC_CH n _DUTY_R_REG (n : 0-5) (0x0010+0x14* n)

Register structure diagram for LEDC_DUTY_CHn_R:

- Bit 31 to 25: (reserved)
- Bit 24 to 0: LEDC_DUTY_CHn_R (value: 0x000000)
- Reset

LEDC_DUTY_CH n _R Represents the current duty cycle of output signal on channel n . (RO)

Register 30.13. LEDC_TIMER_x_VALUE_REG (x: 0-3) (0x00A4+0x8*x)

(reserved)																LEDC_TIMER_CNT															
31																0															
0 0 0 0 0 0 0 0 0 0 0 0 0																0															

Reset

LEDC_TIMER x _CNT Represents the current counter value of timer x . (RO)

Register 30.14. LEDC_TIMER x _CNT_CAP_REG (x : 0-3) (0x0150+0x4* x)

(reserved)																LEDC_TIMER ~~x~~ _CNT_CAP																																			
31																	20																	19																	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x000																Reset																			

LEDC_TIMER x _CNT_CAP Represents the captured LEDC timer x count value. (RO)

Register 30.15. LEDC_INT_RAW_REG (0x00C0)

(reserved)																LEDC_OVF_CNT_CH5_INT_RAW																					
																LEDC_OVF_CNT_CH4_INT_RAW																					
																LEDC_OVF_CNT_CH3_INT_RAW																					
																LEDC_OVF_CNT_CH2_INT_RAW																					
																LEDC_OVF_CNT_CH1_INT_RAW																					
																(reserved)																					
																LEDC_DUTY_CHNG_END_CH5_INT_RAW																					
																LEDC_DUTY_CHNG_END_CH4_INT_RAW																					
																LEDC_DUTY_CHNG_END_CH3_INT_RAW																					
																LEDC_DUTY_CHNG_END_CH2_INT_RAW																					
																LEDC_DUTY_CHNG_END_CH1_INT_RAW																					
																LEDC_TIMER3_OVF_INT_RAW																					
																LEDC_TIMER2_OVF_INT_RAW																					
																LEDC_TIMER1_OVF_INT_RAW																					
																LEDC_TIMER0_OVF_INT_RAW																					
31																	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset					

LEDC_TIMER x _OVF_INT_RAW (x : 0-3) Raw status bit: The raw interrupt status of LEDC_TIMER x _OVF_INT. Triggered when the timer x has reached its maximum counter value. (R/WTC/SS)

LEDC_DUTY_CHNG_END_CH n _INT_RAW (n : 0-5) Raw status bit: The raw interrupt status of LEDC_DUTY_CHNG_END_CH n _INT. Triggered when the fading of duty has finished. (R/WTC/SS)

LEDC_OVF_CNT_CH n _INT_RAW (n : 0-5) Raw status bit: The raw interrupt status of LEDC_OVF_CNT_CH n _INT. Triggered when the ovf_cnt has reached the value specified by LEDC_OVF_NUM_CH n . (R/WTC/SS)

Register 30.16. LEDC_INT_ST_REG (0x00C4)

(reserved) LEDC_OVF_CNT_CH5_INT_ST LEDC_OVF_CNT_CH4_INT_ST LEDC_OVF_CNT_CH3_INT_ST LEDC_OVF_CNT_CH2_INT_ST LEDC_OVF_CNT_CH1_INT_ST LEDC_OVF_CNT_CH0_INT_ST (reserved) LEDC_DUTY_CHNG_END_CH5_INT_ST LEDC_DUTY_CHNG_END_CH4_INT_ST LEDC_DUTY_CHNG_END_CH3_INT_ST LEDC_DUTY_CHNG_END_CH2_INT_ST LEDC_DUTY_CHNG_END_CH1_INT_ST LEDC_TIMER3_OVF_INT_ST LEDC_TIMER2_OVF_INT_ST LEDC_TIMER1_OVF_INT_ST LEDC_TIMER0_OVF_INT_ST																																						
31																		18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset								

LEDC_TIMER x _OVF_INT_ST (x : 0-3) Masked status bit: The masked interrupt status of LEDC_TIMER x _OVF_INT. Valid only when LEDC_TIMER x _OVF_INT_ENA is set to 1. (RO)

LEDC_DUTY_CHNG_END_CH n _INT_ST (n : 0-5) Masked status bit: The masked interrupt status of LEDC_DUTY_CHNG_END_CH n _INT. Valid only when LEDC_DUTY_CHNG_END_CH n _INT_ENA is set to 1. (RO)

LEDC_OVF_CNT_CH n _INT_ST (n : 0-5) Masked status bit: The masked interrupt status of LEDC_OVF_CNT_CH n _INT. Valid only when LEDC_OVF_CNT_CH n _INT_ENA is set to 1. (RO)

Register 30.17. LEDC_INT_ENA_REG (0x00C8)

(reserved)																		LEDC_OVF_CNT_CH5_INT_ENA LEDC_OVF_CNT_CH4_INT_ENA LEDC_OVF_CNT_CH3_INT_ENA LEDC_OVF_CNT_CH2_INT_ENA LEDC_OVF_CNT_CH1_INT_ENA LEDC_OVF_CNT_CH0_INT_ENA (reserved) LEDC_DUTY_CHNG_END_CH5_INT_ENA LEDC_DUTY_CHNG_END_CH4_INT_ENA LEDC_DUTY_CHNG_END_CH3_INT_ENA LEDC_DUTY_CHNG_END_CH2_INT_ENA LEDC_DUTY_CHNG_END_CH1_INT_ENA LEDC_DUTY_CHNG_END_CH0_INT_ENA LEDC_TIMER3_OVF_INT_ENA LEDC_TIMER2_OVF_INT_ENA LEDC_TIMER1_OVF_INT_ENA LEDC_TIMER0_OVF_INT_ENA																				
31																		18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset									

LEDC_TIMER x _OVF_INT_ENA (x : 0-3) Enable bit: Write 1 to enable LEDC_TIMER x _OVF_INT. (R/W)

LEDC_DUTY_CHNG_END_CH n _INT_ENA (n : 0-5) Enable bit: Write 1 to enable LEDC_DUTY_CHNG_END_CH n _INT. (R/W)

LEDC_OVF_CNT_CH n _INT_ENA (n : 0-5) Enable bit: Write 1 to enable LEDC_OVF_CNT_CH n _INT. (R/W)

Register 30.18. LEDC_INT_CLR_REG (0x00CC)

[illegible]

LEDC_TIMER_x_OVF_INT_CLR (x: 0-3) Clear bit: Write 1 to clear LEDC_TIMER_x_OVF_INT. (WT)

LEDC_DUTY_CHNG_END_CH_{*n*}_INT_CLR (*n*: 0-5) Clear bit: Write 1 to clear
LEDC_DUTY_CHNG_END_CH_{*n*}_INT. (WT)

LEDC_OVF_CNT_CH n _INT_CLR (n : 0-5) Clear bit: Write 1 to clear LEDC_OVF_CNT_CH n _INT. (WT)

Register 30.19. LEDC_DATE_REG (0x0174)

(reserved)				LEDC_LEDC_DATE																
31	28	27																	0	
0	0	0	0	0x2311220																Reset

LEDC_LEDC_DATE Configures the version. (R/W)

Chapter 31

Motor Control PWM (MCPWM)

31.1 Overview

The **M**otor **C**ontrol **P**ulse **W**idth **M**odulator (MCPWM) peripheral is intended for motor and power control. It provides six PWM outputs that can be set up to operate in several topologies. One common topology uses a pair of PWM outputs driving an H-bridge to control motor rotation speed and rotation direction.

The MCPWM can be divided into five main modules: PWM timers, PWM operators, Capture module, Event Task Matrix (ETM) module, and Fault Detection module. Each PWM timer provides timing references that can either run freely or be synced to other timers or external sources. Each PWM operator has all the necessary control resources to generate waveform pairs for one PWM channel. The Capture module is used for systems that need to accurately time external events. The ETM module responds to tasks received by the MCPWM, generating corresponding events depending on the state of motion. The Fault Detection module is used to capture external faults, allowing the system to respond by choice.

ESP32-C5 has one MCPWM peripheral, which is MCPWM0.

31.2 Features

An MCPWM peripheral has one clock divider (prescaler), three PWM timers, three PWM operators, a Capture module, an ETM module, and a Fault Detection module. MCPWM's core clock can be selected from three clock sources: PLL_F160M_CLK, XTAL_CLK, and RC_FAST_CLK (configured by the PCR_PWM_CLKM_SEL field of the [PCR_PWM_CLK_CONF_REG](#) register). Figure 31.2-1 shows the submodules inside MCPWM and the signals on the interface. PWM timers are used for generating timing references. The PWM operators generate the desired waveform based on the timing references. Any PWM operator can be configured to use the timing references of any PWM timers. Different PWM operators can use the same PWM timer's timing reference to generate PWM signals, or different PWM timers' values to generate separate PWM signals. Different PWM timers can also be synchronized together.

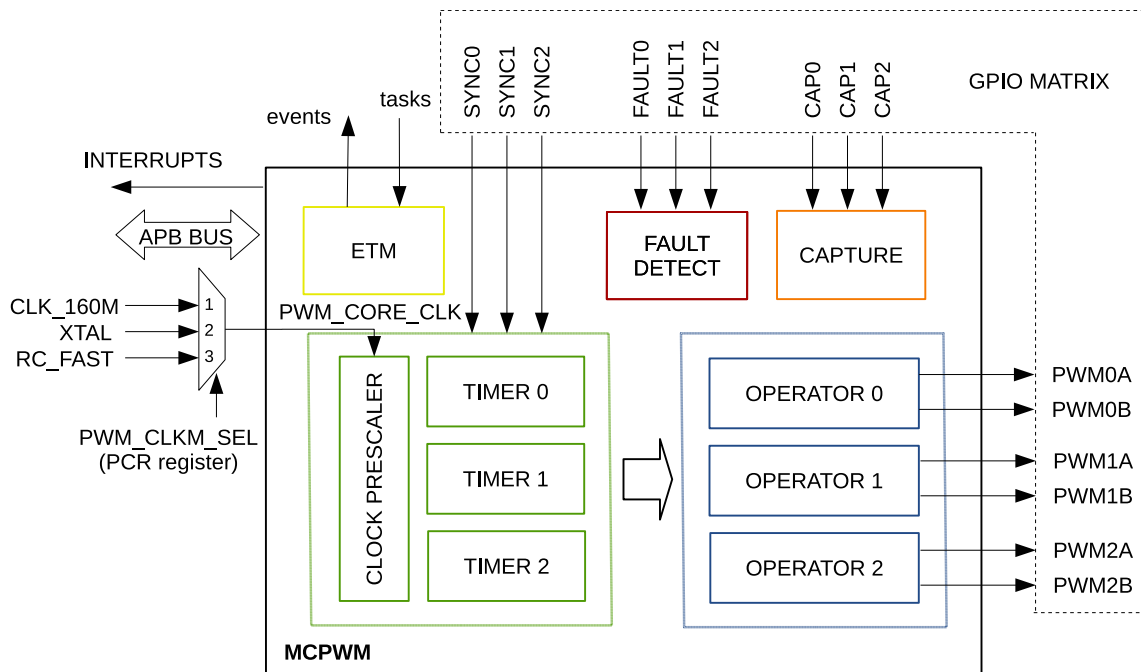


Figure 31.2-1. MCPWM Module Overview

Below is an overview of the submodules' functionality in Figure 31.2-1:

- PWM Timers 0, 1, and 2:
 - Every PWM timer has a dedicated 8-bit clock prescaler.
 - The 16-bit counter in the PWM timer can work in Count-Up Mode, Count-Down Mode, or Count-Up-Down Mode.
 - A hardware sync or software sync can trigger a reload on the PWM timer with a phase register. It will also trigger the prescaler's restart, so that the timer's clock can also be synced. The source of the hard sync can come from any GPIO or any other PWM timer's sync_out. The source of the soft sync comes from writing toggle value to the `MCPWM_TIMERn_SYNC_SW` bit.
- PWM Operators 0, 1, and 2:
 - Every PWM operator has two PWM outputs: PWMxA and PWMxB. They can work independently, in symmetric or asymmetric configurations.
 - The control of the PWM signal can be updated asynchronously.
 - Configurable dead time on rising and falling edges; each set up independently.
 - All events can trigger CPU interrupts.
 - Modulating of PWM output by high-frequency carrier signals, useful when gate drivers are insulated with a transformer.
 - Period, time-stamp, and important control registers have shadow registers with flexible updating methods.
- Fault Detection Module:
 - Programmable fault handling in both cycle-by-cycle mode and one-shot mode.

- A fault condition can force the PWM output to either high or low logic levels.
- Capture Module:
 - Clock of the capture module is the same as MCPWM's core clock.
 - Speed measurement of rotating machinery.
 - Measurement of elapsed time between position sensor pulses
 - Period and duty cycle measurement of pulse train signals
 - Decoding current or voltage amplitude derived from duty-cycle-encoded signals of current/voltage sensors
 - Three individual capture channels, each of which with a time-stamp register (32-bit)
 - Selection of edge polarity and prescaling of input capture signals
 - The capture timer can sync with a PWM timer or external signals.
 - Interrupt on each of the three capture channels
- ETM Module:
 - Generation of different events depending on the different running states of each timer and operator.
 - Each timer and operator responds to its corresponding task and automatically performs the corresponding operation.
 - Each event and task can be enabled independently. When an event is not enabled, the corresponding event will not be generated. When a task is not enabled, the corresponding task will not be responded to.

31.3 Modules

31.3.1 Overview

The key modules in each MCPWM are timer module, operator module, fault detection module, capture module, and ETM module. See their functions in the following sections.

31.3.1.1 Timer Module

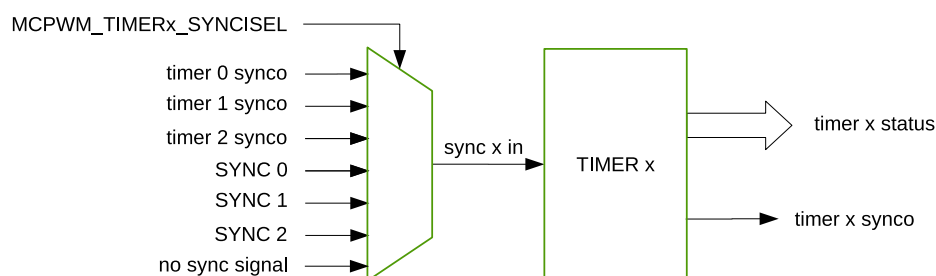


Figure 31.3-1. Timer Module

This module contains a timer, which can count at a specified period. It can work in Count-Up Mode, Count-Down Mode, or Count-Up-Down Mode. It supports different synchronization input sources (a total of seven optional input sources) for reloading the count value and direction, allowing synchronization among multiple timers. Furthermore, it provides synchronization output options (a total of four optional output sources) for use by other timers.

The timer x status in the figure represents the timer status output by the timer module, such as the counting mode, and count value equal to zero or period.

31.3.1.2 Operator Module

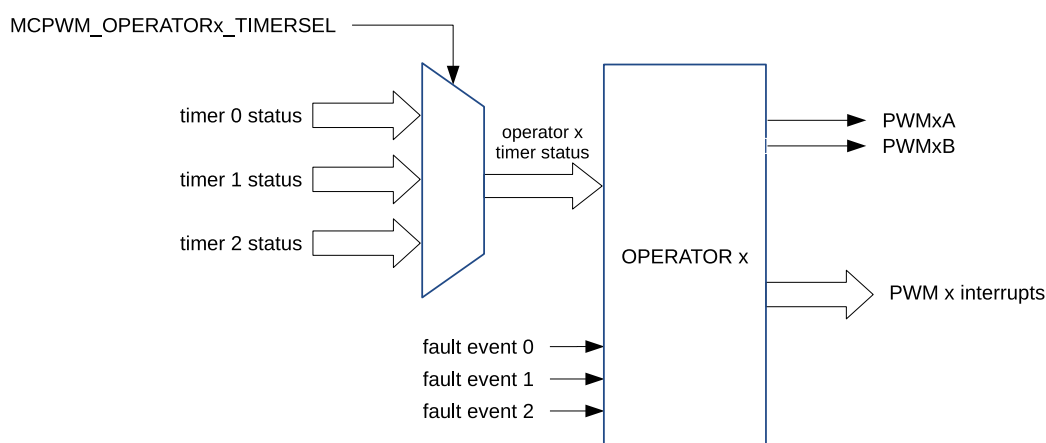


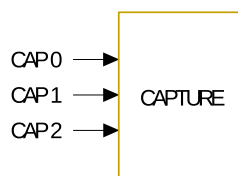
Figure 31.3-2. Operator Module

A PWM operator contains a PWM generator, a dead time generator, and a PWM carrier module. Their functions are shown in Table 31.3-1.

Table 31.3-1. Functions of Each Submodule in the Operator Module

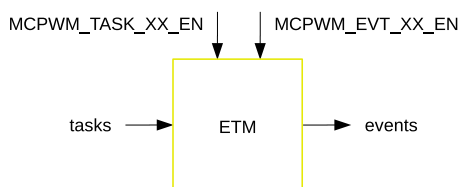
Submodule	Functions
PWM Generator	Generate PWM signals according to the configured duty cycle and waveform.
Dead Time Generator	Choose to add dead time to the rising and falling edges of the PWM signal generated by the PWM generator (add delays to the rising and falling edges), and perform operations such as inversion.
PWM Carrier	Choose to add a carrier to the PWM signal generated by the dead time generator. The carrier frequency and the first pulse duration of the PWM waveform after adding the carrier can be configured.
Fault Detector	Detect faults and generate corresponding fault events, respond to fault events based on the specified interval mode (one-shot mode or cycle-by-cycle mode), and when a fault event occurs, take action on the PWM signal generated by the PWM carrier in the specified way.

31.3.1.3 Capture Module

**Figure 31.3-3. Capture Module**

The module contains a timer that can be synchronized. When the input capture signal is valid, the count value of the timer is captured.

31.3.1.4 ETM Module

**Figure 31.3-4. ETM Module**

The module responds to received tasks, or generate and output events. Each event and task can be enabled independently.

31.3.2 PWM Timer Module

MCPWM has three PWM timer modules. Any of them can determine the necessary event timing for any of the three PWM operator modules. By using the synchronization signals from the GPIO matrix, built-in synchronization logic allows multiple PWM timer modules in one or more MCPWM peripherals to work together as a system.

31.3.2.1 Configurations of the PWM Timer Module

Users can configure the following functions of the PWM timer module:

- Control how often events occur by specifying the PWM timer frequency or period.
- Configure a particular PWM timer to synchronize phases with other PWM timers or modules.
- Configure the following timer counting modes: count-up, count-down, count-up-down.
- Change the rate of the PWM timer clock (PT_CLK) with a prescaler. Each timer has its own prescaler configured with `MCPWM_TIMER $n$ _PRESCALE` of the register `MCPWM_TIMER0_CFGO_REG`. The PWM timer increments or decrements at a slower pace, depending on the setting of this field. The new `MCPWM_TIMER $n$ _PRESCALE` configuration value will take effect when the timer stops and starts counting again.

31.3.2.2 PWM Timer's Working Modes and Timing Event Generation

The PWM timer has three working modes, selected by the PWM x timer mode field:

- Count-Up Mode:
The PWM timer increments from zero until reaching the value configured in the period field. Once done, the PWM timer returns to zero and starts increasing again. PWM period = the value of the period field + 1.
Note: The period field is `MCPWM_TIMER $n$ _PERIOD` ($x = 0, 1, 2$), i.e., `MCPWM_TIMER0_PERIOD`, `MCPWM_TIMER1_PERIOD`, `MCPWM_TIMER2_PERIOD`.
- Count-Down Mode:
The PWM timer decrements to zero, starting from the value configured in the period field. Once done, the PWM timer returns to the period value and starts decrementing again. In this case, the PWM period = the value of period field + 1.
- Count-Up-Down Mode:
This is a combination of the two modes mentioned above. The PWM timer starts increasing from zero until the period value is reached. Then, the timer decreases back to zero. The PWM timer cycles incrementally and decrementally in this mode. The PWM period = the value of the period field $\times$ 2.

Figures 31.3-5 to 31.3-8 show PWM timer waveforms in different modes, including timer behavior during synchronization events. In Count-Up mode, the counting direction after synchronization is always counting up. In Count-Down mode, the counting direction after synchronization is always counting down. In Count-Up-Down Mode, the counting direction after synchronization can be chosen by setting the `MCPWM_TIMER $n$ _PHASE_DIRECTION`.

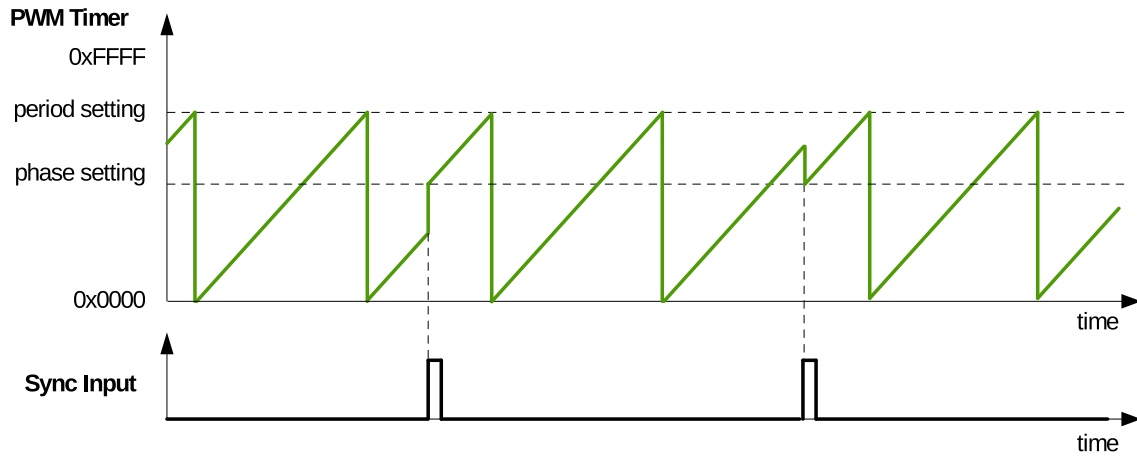


Figure 31.3-5. Count-Up Mode Waveform

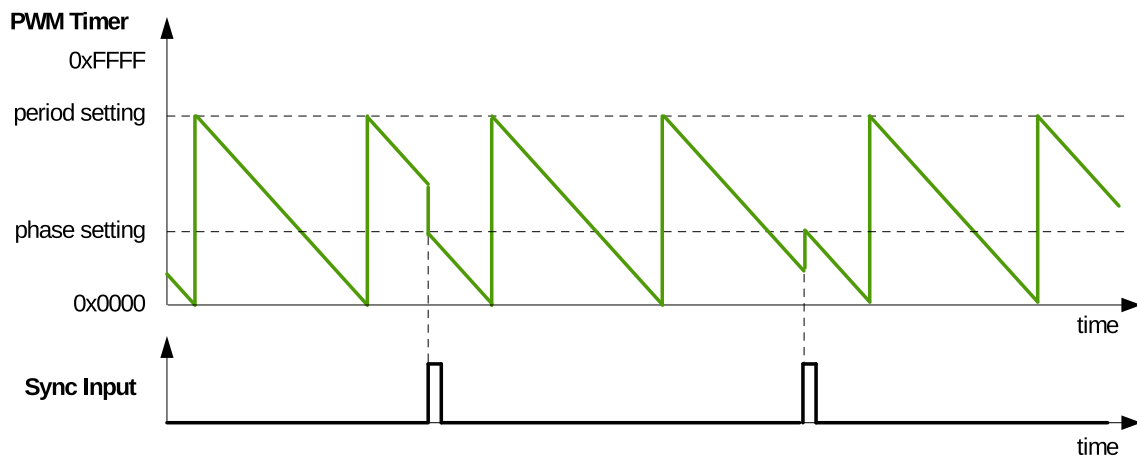


Figure 31.3-6. Count-Down Mode Waveforms

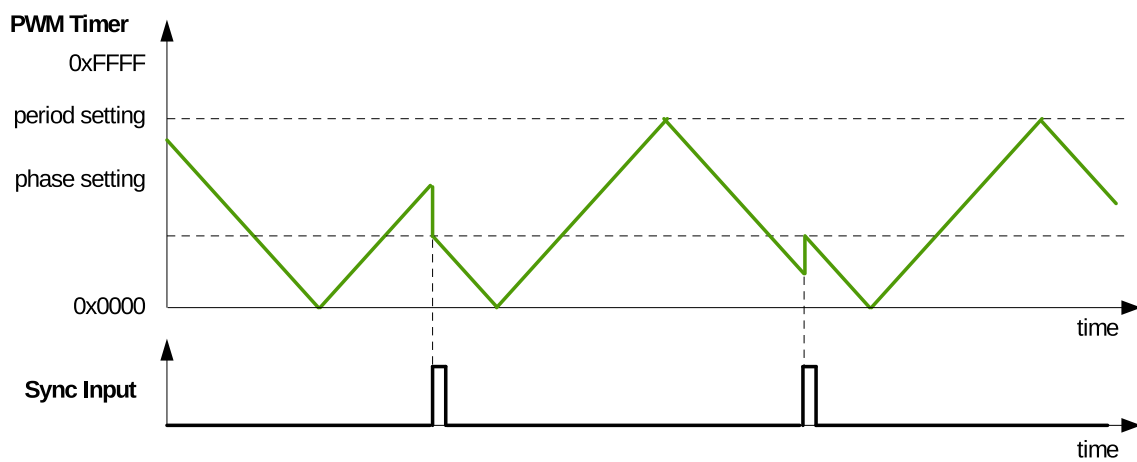


Figure 31.3-7. Count-Up-Down Mode Waveforms, Count-Down at Synchronization Event

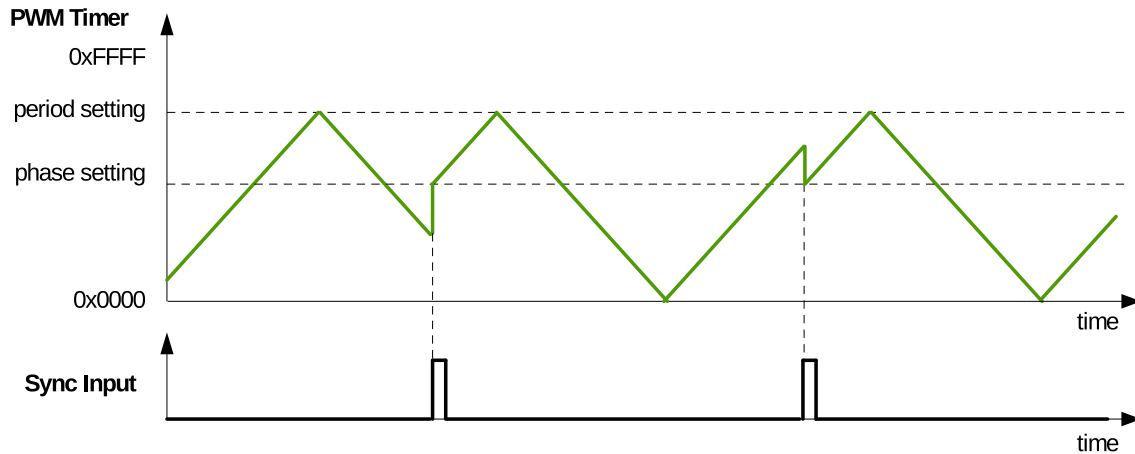


Figure 31.3-8. Count-Up-Down Mode Waveforms, Count-Up at Synchronization Event

When the PWM timer is running, it generates the following timing events periodically and automatically:

- UTEP: The timing event generated when the PWM timer's value is equal to the value of the period field (`MCPWM_TIMER $n$ _PERIOD`) and when the PWM timer is increasing.
- UTEZ: The timing event generated when the PWM timer's value equals zero and when the PWM timer is increasing.
- DTEP: The timing event generated when the PWM timer's value equals the value of the period field (`MCPWM_TIMER $n$ _PERIOD`) and when the PWM timer is decreasing.
- DTEZ: The timing event generated when the PWM timer's value equals zero and when the PWM timer is decreasing.

Figures 31.3-9 to 31.3-11 show the timing waveforms of U/DTEP and U/DTEZ.

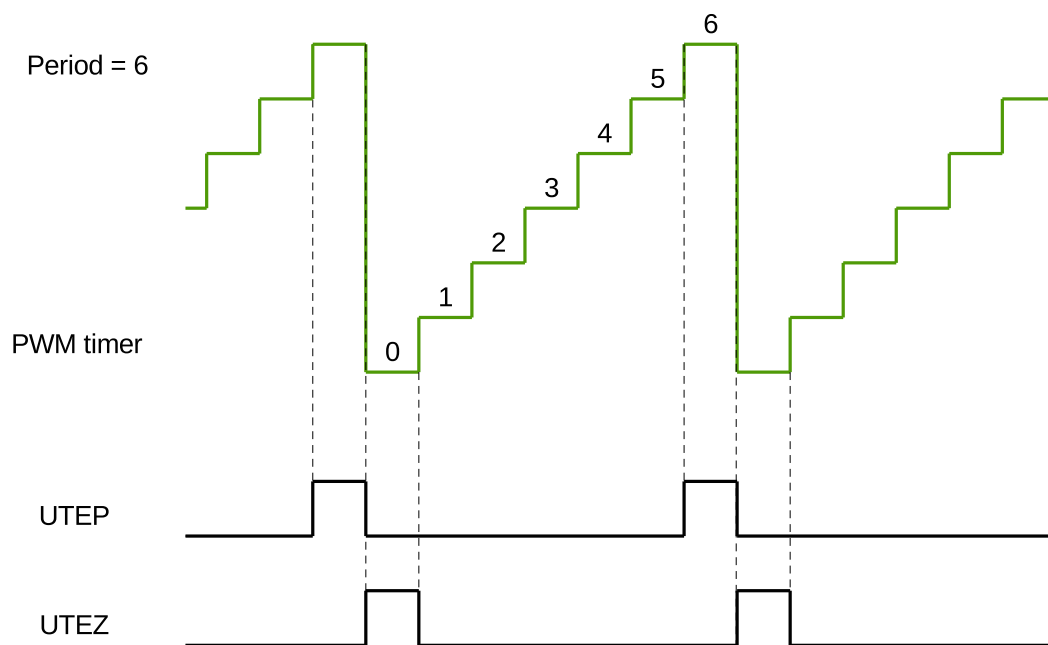


Figure 31.3-9. UTEP and UTEZ Generation in Count-Up Mode

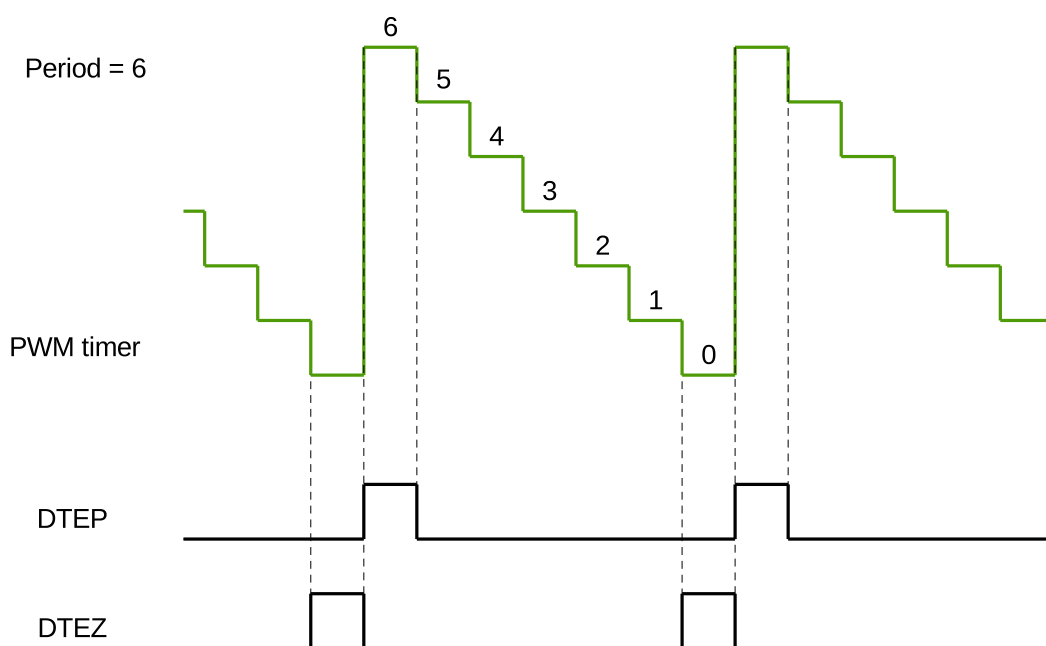


Figure 31.3-10. DTEP and DTEZ Generation in Count-Down Mode

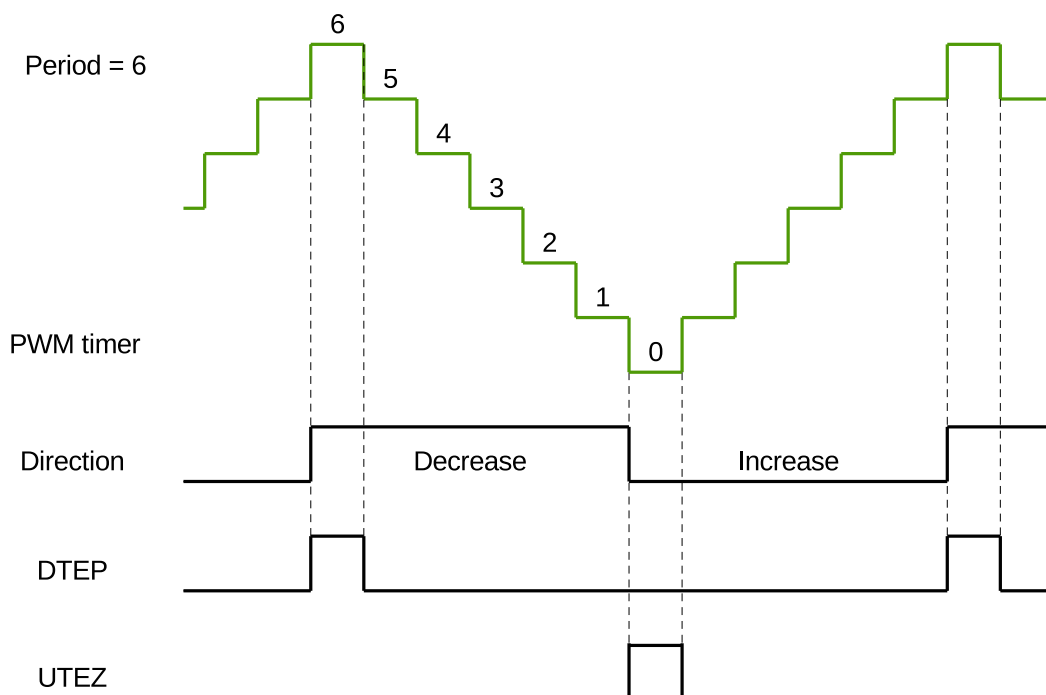


Figure 31.3-11. DTEP and UTEZ Generation in Count-Up-Down Mode

Please note that in the Count-Up-Down Mode, when the counting direction is increasing, the timer range is [0, period value - 1], and when the counting direction is decreasing, the timer range is [period value, 1]. That is, in this mode, when synchronizing the timer to 0, decreasing counting direction will be illegal, namely, `MCPWM_TIMER $n$ _PHASE_DIRECTION` cannot be set to 1. Similarly, when synchronizing the timer to period value, increasing counting direction will be illegal, namely, `MCPWM_TIMER $n$ _PHASE_DIRECTION` cannot be set to 0. Therefore, when the timer is synchronized to 0, the counting direction can only be increasing, and `MCPWM_TIMER $n$ _PHASE_DIRECTION` will be 0. When the timer is synchronized to the period value, the counting direction can only be decreasing, and `MCPWM_TIMER $n$ _PHASE_DIRECTION` will be 1.

31.3.2.3 Shadow Register of PWM Timer

The PWM timer's period register and the PWM timer's clock prescaler register have shadow registers. The shadow registers can back up the values that are about to be written to the valid registers. It also supports writing the values saved into the active register at a specific moment of hardware synchronization. The functionality of both register types is as follows:

- Active Register: Directly responsible for controlling all actions performed by hardware.
- Shadow Register: Acts as a temporary buffer for a value to be written to the active register. At a specific, user-configured point in time, the value saved in the shadow register is copied to the active register. Before this happens, the content of the shadow register has no direct effect on the controlled hardware. This helps to prevent erroneous operation of the hardware, which may happen when a register is asynchronously modified by software. Both the shadow register and the active register have the same memory address. The software always writes into, or reads from the shadow register.

The moment of updating the clock prescaler's active register is at the time when the timer starts operating. When `MCPWM_GLOBAL_UP_EN` is set to 1, the moment of updating the period active register can be selected by the following ways:

- By configuring the update method register `MCPWM_TIMERn_PERIOD_UPMETHOD` to 0, the update will start immediately.
- By configuring the update method register `MCPWM_TIMERn_PERIOD_UPMETHOD` to 1, the update can start when the PWM timer is equal to zero.
- By configuring the update method register `MCPWM_TIMERn_PERIOD_UPMETHOD` to 2, the update can start when the PWM timer is synchronized.
- By configuring the update method register `MCPWM_TIMERn_PERIOD_UPMETHOD` to 3, the update can start when the PWM timer is equal to zero or is synchronized.
- Software can also trigger a globally forced update bit `MCPWM_GLOBAL_FORCE_UP` which will prompt all registers in the module to be updated according to shadow registers.

31.3.2.4 PWM Timer Synchronization and Phase Locking

The PWM modules adopt a flexible synchronization method. Each PWM timer has a synchronization input and a synchronization output. The synchronization input can be selected from three synchronization outputs of the three PWM timers and three synchronization signals from the GPIO matrix. The synchronization output can be generated when the PWM timer's value equals to period or zero, the synchronization input signal is active, or software synchronizes. Thus, each PWM timer can flexibly select the synchronization input for different phase synchronization methods. During synchronization, the PWM timer clock prescaler will reset its counter and restart dividing the clock in order to ensure the correctness of the timer.

31.3.3 PWM Operator Module

The PWM Operator module has the following functions:

- Generates a PWM signal pair, based on timing references obtained from the corresponding PWM timer.

- Each signal out of the PWM signal pair includes a specific pattern of dead time (i.e., adding delays to the rising and falling edges of the PWM signal).
- Superimposes a carrier on the PWM signal, if configured to do so.
- Handles response under fault conditions.

Figure 31.3-12 shows the block diagram of a PWM operator.

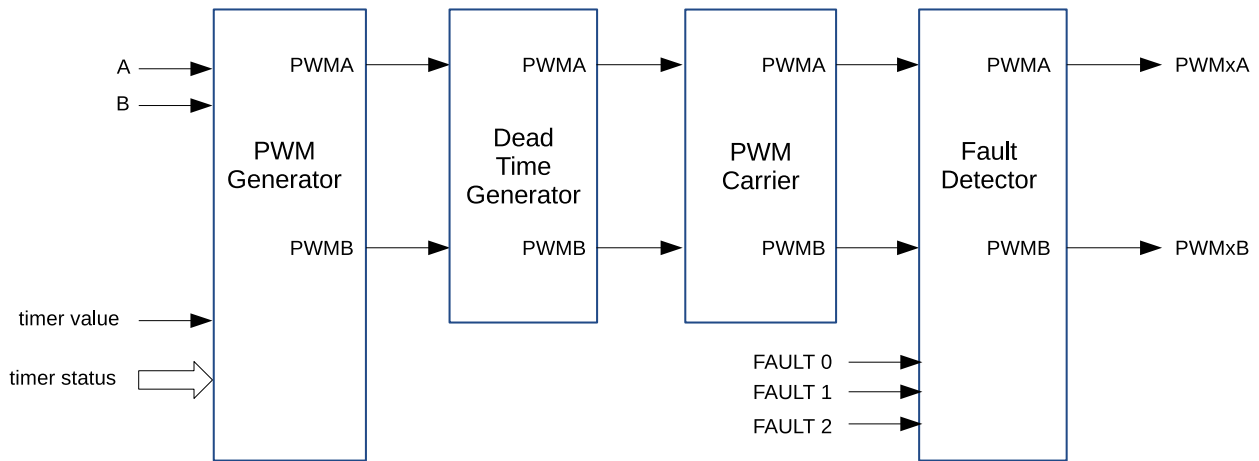


Figure 31.3-12. Block Diagram of A PWM Operator

31.3.3.1 PWM Generator Module

Purpose of the PWM Generator Module

In this module, important timing events are generated or imported. The events are then converted into specific actions to generate the desired waveforms at the PWMxA and PWMxB outputs.

The PWM generator module performs the following actions:

- Generation of timing events based on time stamps configured using the A and B registers. Events happen when the following conditions are met (for the configuration of registers A and B, see section [31.3.3.1](#)):
 - UTEA: the PWM timer is counting up and its value is equal to register A.
 - UTEB: the PWM timer is counting up and its value is equal to register B.
 - DTEA: the PWM timer is counting down and its value is equal to register A.
 - DTEB: the PWM timer is counting down and its value is equal to register B.
- Generation of U/DTO, U/DT1 timing events based on fault or synchronization events.
 - UTO: the PWM timer is counting up and FAULT0 detected (field [MCPWM_GENn_TO_SEL](#) is set to 0) or FAULT1 detected (field [MCPWM_GENn_TO_SEL](#) is set to 1) or FAULT2 detected (field [MCPWM_GENn_TO_SEL](#) is set to 2) or synchronized (field [MCPWM_GENn_TO_SEL](#) is set to 3).
 - UT1: the PWM timer is counting up and FAULT0 detected (field [MCPWM_GENn_T1_SEL](#) is set to 0) or FAULT1 detected (field [MCPWM_GENn_T1_SEL](#) is set to 1) or FAULT2 detected (field [MCPWM_GENn_T1_SEL](#) is set to 2) or synchronized (field [MCPWM_GENn_T1_SEL](#) is set to 3).
 - DTO: the PWM timer is counting down and FAULT0 detected (field [MCPWM_GENn_TO_SEL](#) is set to 0) or FAULT1 detected (field [MCPWM_GENn_TO_SEL](#) is set to 1) or FAULT2 detected (field [MCPWM_GENn_TO_SEL](#) is set to 2) or synchronized (field [MCPWM_GENn_TO_SEL](#) is set to 3).
 - DT1: the PWM timer is counting down and FAULT0 detected (field [MCPWM_GENn_T1_SEL](#) is set to 0) or FAULT1 detected (field [MCPWM_GENn_T1_SEL](#) is set to 1) or FAULT2 detected (field [MCPWM_GENn_T1_SEL](#) is set to 2) or synchronized (field [MCPWM_GENn_T1_SEL](#) is set to 3).
- Management of priority when these timing events occur concurrently.
- Generation of set, clear, and toggle actions, based on the timing events.
- Controlling of the PWM duty cycle, depending on the configuration of the PWM generator module.
- Handling of new time stamp values, using shadow registers to prevent glitches in the PWM cycle.

Shadow Register of PWM Generator

The time stamp registers A and B used by the hardware have shadow registers, which are registers [MCPWM_GENn_A_REG](#) and [MCPWM_GENn_B_REG](#). Shadowing provides a way of updating registers in sync with the hardware.

When [MCPWM_GLOBAL_UP_EN](#) is set to 1, the shadow registers can be written to the active register at a specified time. The update method field for [MCPWM_GENn_A_REG](#) and [MCPWM_GENn_B_REG](#) is [MCPWM_GENn_CFG_UPMETHOD](#). Software can also trigger a globally forced update bit

[MCPWM_GLOBAL_FORCE_UP](#) which will prompt all registers in the module to be updated according to shadow registers. For a description of the shadow registers, please see Section 31.3.2.3.

Timing Events

For convenience, all timing signals and events are summarized in Table 31.3-2.

Table 31.3-2. Timing Events Used in PWM Generator

Signal	Event Description	PWM Timer Operation
DTEP	PWM timer value is equal to the period register value	PWM timer counts down
DTEZ	PWM timer value is equal to zero	
DTEA	PWM timer value is equal to register A	
DTEB	PWM timer value is equal to register B	
DTO event	Based on fault or synchronization events	
DT1 event	Based on fault or synchronization events	
UTEP	PWM timer value is equal to the period register value	PWM timer counts up
UTEZ	PWM timer value is equal to zero	
UTEA	PWM timer value is equal to register A	
UTEB	PWM timer value is equal to register B	
UTO event	Based on fault or synchronization events	
UT1 event	Based on fault or synchronization events	
Software-force event	Software-initiated asynchronous event	N/A

The purpose of a software-force event is to impose non-continuous or continuous changes on the PWM_xA and PWM_xB outputs. The change is done asynchronously. Software-force control is handled by the [MCPWM_GEN_n_FORCE_REG](#) registers.

The selection and configuration of T0/T1 in the PWM generator module is independent of the configuration of fault events in the fault handler module. A particular trip event may or may not be configured to cause trip action in the fault handler submodule, but the same event can be used by the PWM generator to trigger T0/T1 for controlling PWM waveforms.

It is important to know that when the PWM timer is in Count-Up-Down Mode. It will always decrement after a TEP event, and increment after a TEZ event. So, when the PWM timer is in Count-Up-Down Mode, DTEP and UTEZ events will occur, while UTEP and DTEZ events never occurs.

The PWM generator can handle multiple events at the same time. Events are prioritized by the hardware and relevant details are provided in Table 31.3-3 and Table 31.3-4. Priority levels range from 1 (the highest) to 7 (the lowest). Please note that the priority of TEP and TEZ events depends on the PWM timer's counting mode.

If the value of A or B is set to be greater than the period, then U/DTEA and U/DTEB will never occur.

Table 31.3-3. Timing Events Priority When PWM Timer Increments

Priority Level	Event
1 (highest)	Software-forced event
2	UTEP

Priority Level	Event
3	UTO
4	UT1
5	UTEB
6	UTEA
7 (lowest)	UTEZ

Table 31.3-4. Timing Events Priority when PWM Timer Decrements

Priority level	Event
1 (highest)	Software-forced event
2	DTEZ
3	DT0
4	DT1
5	DTEB
6	DTEA
7 (lowest)	DTEP

Notes:

1. UTEP and UTEZ do not happen simultaneously. When the PWM timer is in Count-Up Mode, UTEP will always happen one cycle earlier than UTEZ, as demonstrated in Figure 31.3-9, so their action on PWM signals will not interrupt each other. When the PWM timer is in Count-Up-Down Mode, UTEP will not occur.
2. DTEP and DTEZ do not happen simultaneously. When the PWM timer is in Count-Down Mode, DTEZ will always happen one cycle earlier than DTEP, as demonstrated in Figure 31.3-10, so their action on PWM signals will not interrupt each other. When the PWM timer is in Count-Up-Down Mode, DTEZ will not occur.

PWM Signal Generation

The PWM generator module controls the behavior of outputting PWMxA and PWMxB when a particular timing event occurs. The timing events are further qualified by the PWM timer's counting mode (increment or decrement). Knowing the counting mode, the module may then perform an independent action at each stage of the PWM timer counting up or down.

The following actions may be configured on PWMxA and PWMxB outputs:

- Set High: Set the output of PWMxA or PWMxB to a high level.
- Clear Low: Clear the output of PWMxA or PWMxB by setting it to a low level.
- Toggle: Change the current output level of PWMxA or PWMxB to the opposite value. If it is currently pulled up, then pull it down, or vice versa.
- Do Nothing: Keep both outputs PWMxA and PWMxB unchanged. In this state, interrupts can still be triggered.

Actions on outputs is configured by using registers `MCPWM_GENn_A_REG` and

[MCPWM_GENn_B_REG](#). So, the action to be taken on each output is set independently. Also, there is great flexibility in selecting actions to be taken on a given output based on events. More specifically, any event listed in Table 31.3-2 can operate on either output of PWMx_A or PWMx_B. To check out registers for particular generator 0, 1, or 2, please refer to the register description in Section 31.5.

Waveforms for Common Configurations

In the configuration of various waveforms below, period refers to the configuration value stored in the period register of the PWM timer selected by the PWM operator (i.e. [MCPWM_TIMERn_PERIOD](#) ($x = 0, 1, 2$)). For configuration methods, please refer to the section 31.3.2. A and B denote the time stamp registers A and B used by the hardware. For register configuration, please refer to the section 31.3.3.1. UTEA/B, UTEZ/P, DTEA/B, DTEZ/P, DTO/1, and UTO/1 represent different timing events of the PWM generator. For event definition, please refer to the section 31.3.3.1. The upward arrow, downward arrow, and T mark signify setting the PWM waveform high, setting it low, toggling it when the timing event occurs. See section 31.3.3.1 for configuration. PWMx_A and PWMx_B are the output signals of PWM generator x .

Figure 31.3-13 presents the symmetrical PWM waveform generated when the PWM timer is in Count-Up-Down mode. DC 0%–100% modulation can be calculated via the formula below:

$$Duty = (Period - A) \div Period$$

If A matches the PWM timer value and the PWM timer is incrementing, then the PWM output is pulled up. If A matches the PWM timer value while the PWM timer is decrementing, then the PWM output is pulled low.

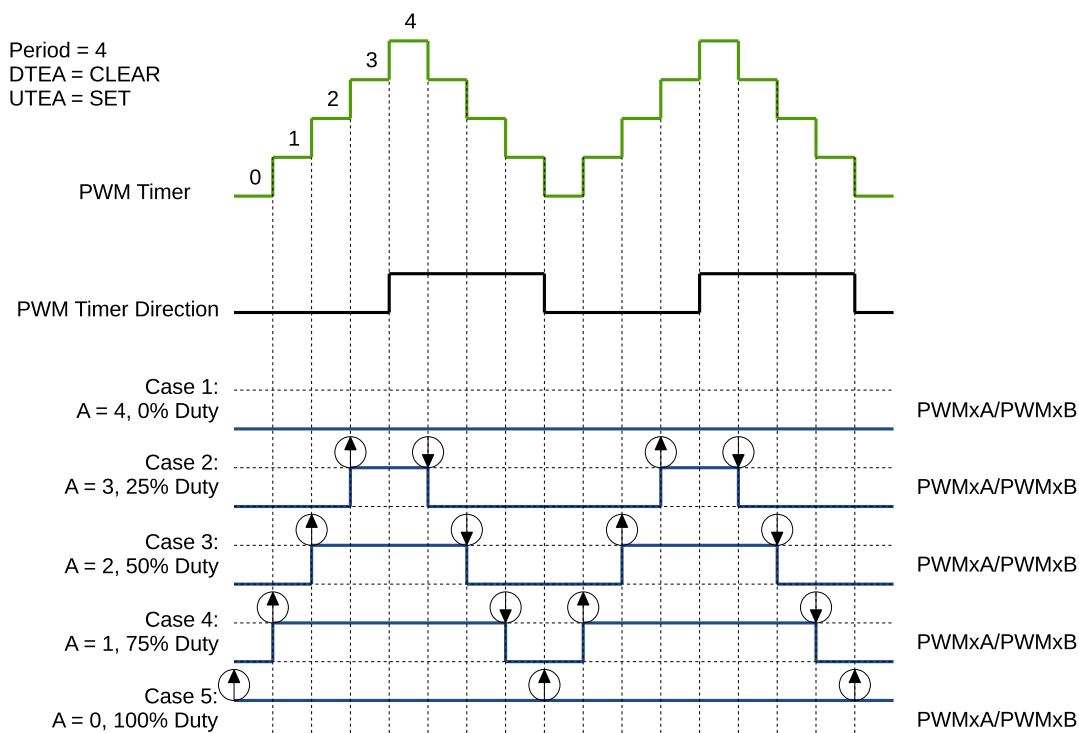


Figure 31.3-13. Symmetrical Waveform in Count-Up-Down Mode

The PWM waveforms in Figures 31.3-14 to 31.3-17 show some common PWM generator configurations.

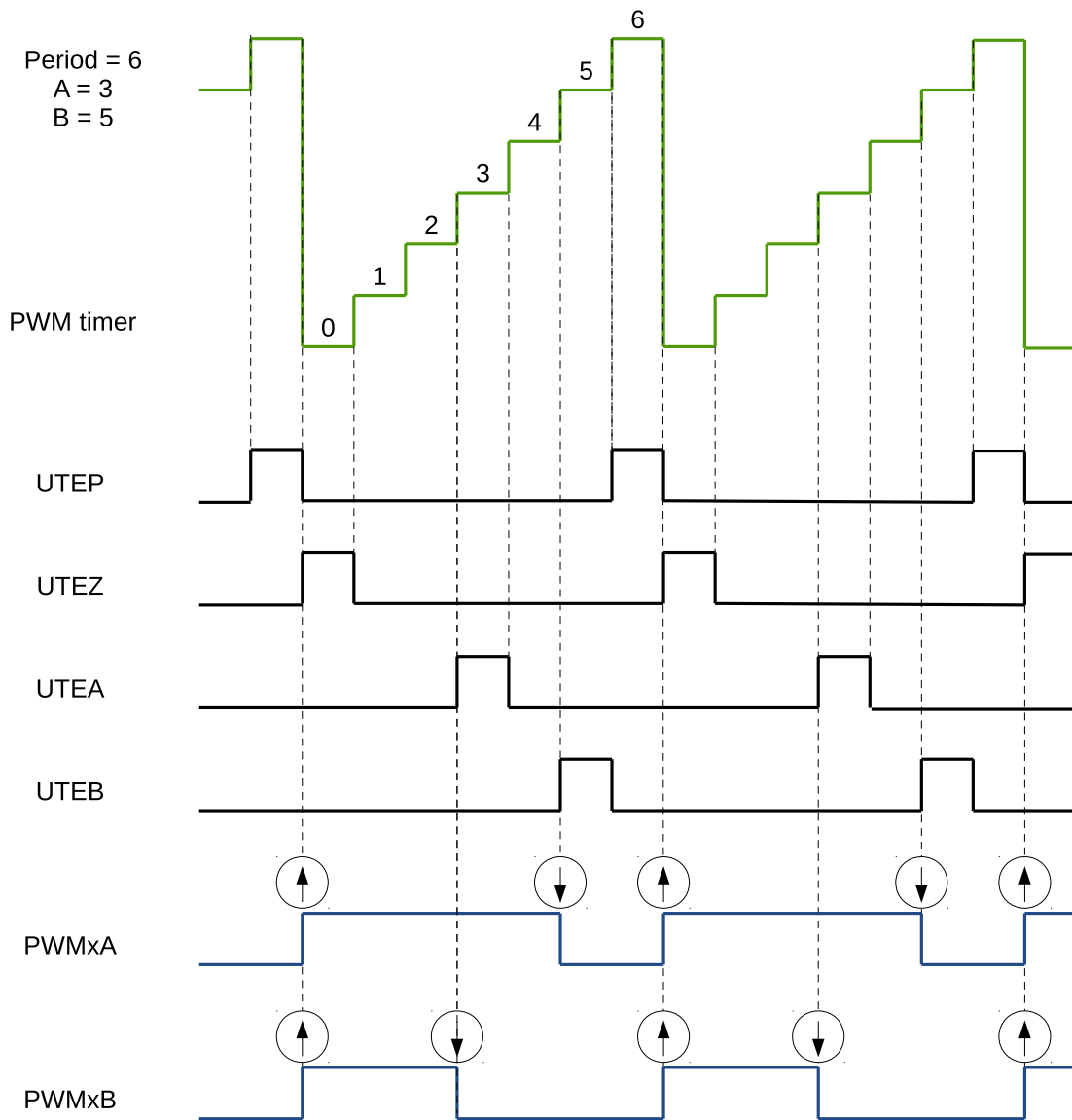


Figure 31.3-14. Count-Up, Single Edge Asymmetric Waveform, with Independent Modulation on PWMxA and PWMxB — Active High

The duty modulation for PWMxA is set by B, active high and proportional to B.

The duty modulation for PWMxB is set by A, active high and proportional to A.

$$Period = (MCPWM\_TIMER\_PERIOD + 1) \times T_{PT\_clk}$$

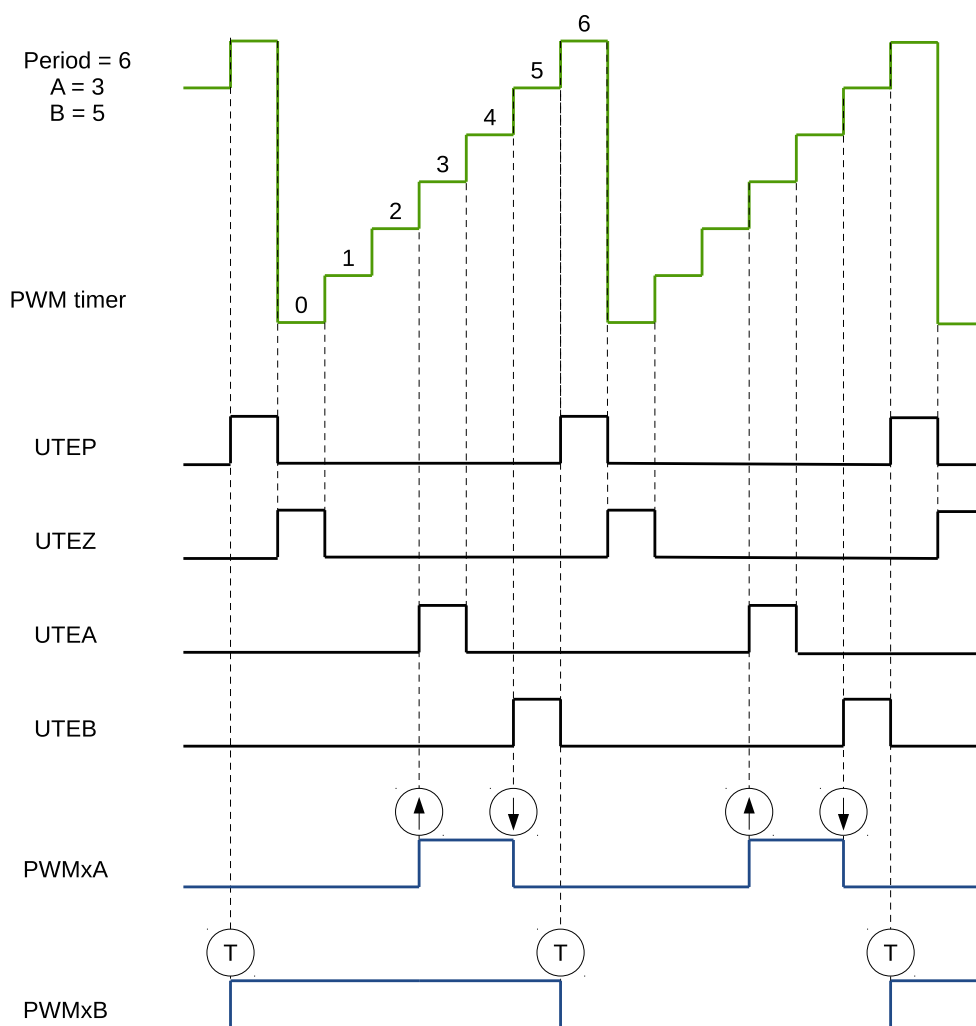


Figure 31.3-15. Count-Up, Pulse Placement Asymmetric Waveform with Independent Modulation on PWMxA

Pulses may be generated anywhere within the PWM cycle (zero to period).

PWMxA's high time duty is proportional to (B - A).

$$Period = (MCPWM\_TIMERn\_PERIOD + 1) \times T_{PT\_clk}$$

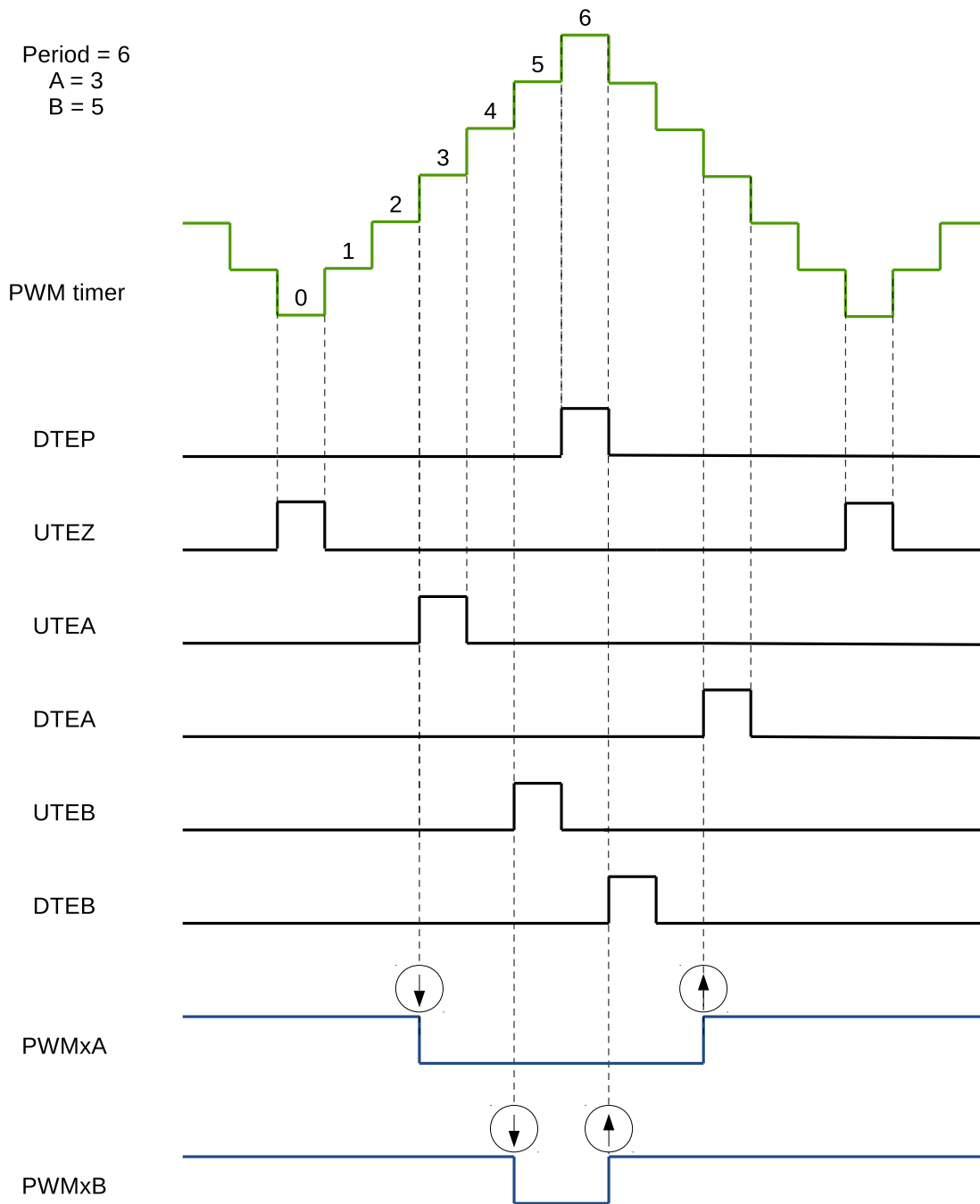


Figure 31.3-16. Count-Up-Down, Dual Edge Symmetric Waveform, with Independent Modulation on PWMxA and PWMxB — Active High

The duty modulation for PWMxA is set by A, active high and proportional to A.
The duty modulation for PWMxB is set by B, active high and proportional to B.
Outputting PWMxA and PWMxB can drive separate switches.

$$Period = (2 \times MCPWM\_TIMER\_PERIOD) \times T_{PT\_clk}$$

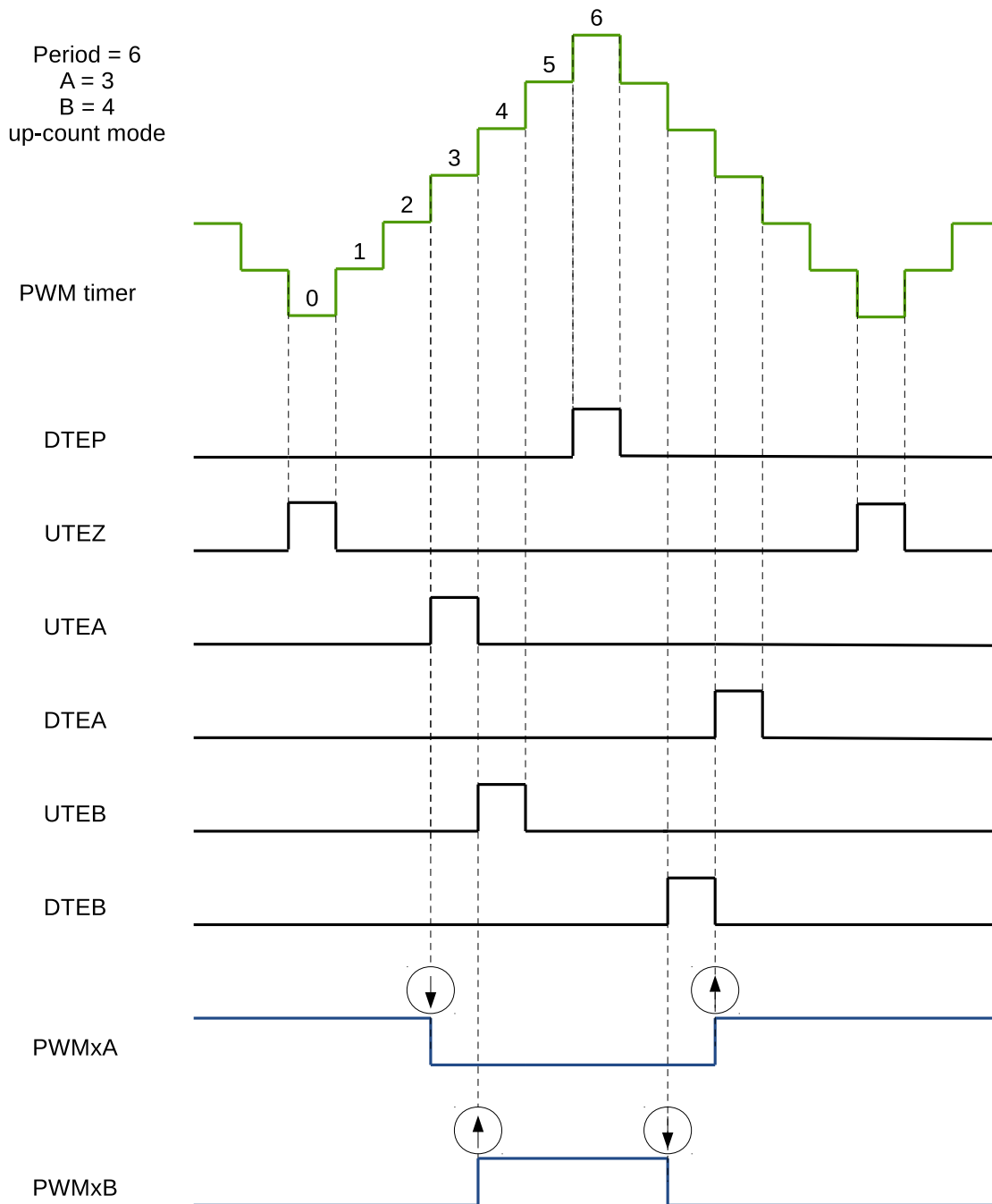


Figure 31.3-17. Count-Up-Down, Dual Edge Symmetric Waveform, with Independent Modulation on PWMxA and PWMxB – Complementary

The duty modulation of PWMxA is set by A, is active high and proportional to A.

The duty modulation of PWMxB is set by B, is active low and proportional to B.

Outputs PWMx can drive upper/lower (complementary) switches.

Dead time = B – A. Edge placement is configurable by software. Dead time generator module supports configuring edge delay methods when required.

$$Period = (2 \times MCPWM\_TIMERn\_PERIOD) \times T_{PT\_clk}$$

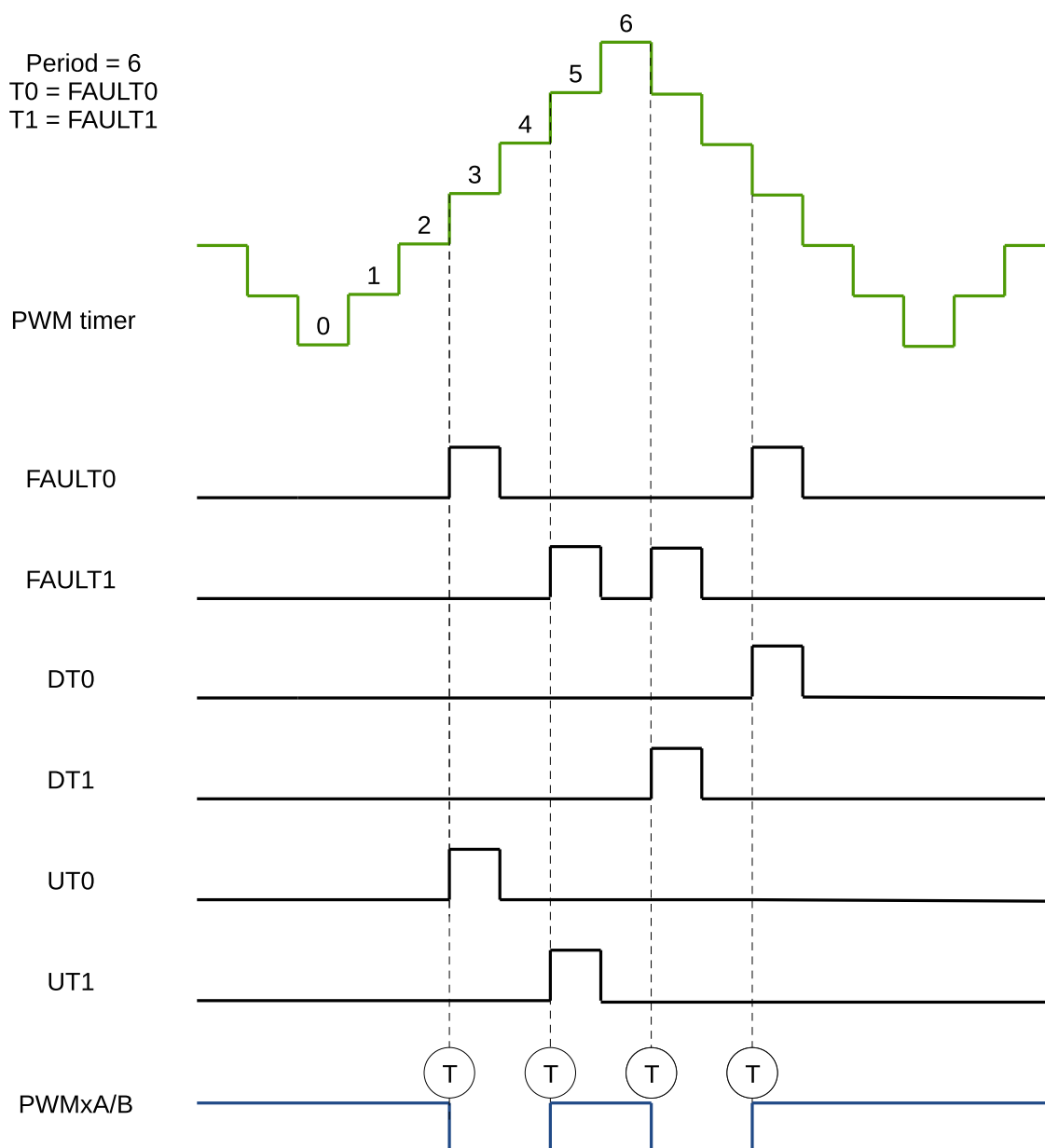


Figure 31.3-18. Count-Up-Down, Fault or Synchronization Events, with Same Modulation on PWMxA and PWMxB

Figure 31.3-18 shows a waveform when UT0/1 and DT0/1 events are generated. In this example, T0 selects FAULT0 and T1 selects FAULT1. The events selected by T0 and T1 can be configured independently, these events can be FAULT0, FAULT1, FAULT2 or synchronous. For detailed configuration, see section 31.3.3.1.

Software-Force Events

There are two types of software-force events inside the PWM generator:

- Non-continuous-immediate (NCI) software-force events: Such types of events are immediately effective on PWM outputs when triggered by software. The forcing is non-continuous, which means the next active timing events will be able to alter the PWM outputs.
- Continuous (CNTU) software-force events: Such types of events are continuous. The forced PWM outputs will continue until they are released by software. The events' triggers are configurable. They can

be configured to be timing events or immediate events.

For NCI software-force events, set register [MCPWM_GENn_A_NCIFORCE](#) or [MCPWM_GENn_B_NCIFORCE](#) to 1 and then to 0 to trigger the NCI software-forced events of PWMxA or PWMxB. Configure register [MCPWM_GENn_A_NCIFORCE_MODE](#) or [MCPWM_GENn_B_NCIFORCE_MODE](#) to specify the operation on the PWMxA or PWMxB waveform when the NCI software-force event occurs. Configuring the registers to 0 or 3 means no operation, 1 means low level, and 2 means high level.

For CNTU software-force events, set register [MCPWM_GENn_CNTUFORCE_UPMETHOD](#) to specify the trigger method of CNTU software-force events. Setting all bits to 0 means trigger CNTU software-force events immediately, setting bit 0 to 1 means the TEZ event triggers CNTU software-force events, setting bit 1 to 1 means the TEP event triggers CNTU software-force events, setting bit 2 to 1 means the TEA event triggers CNTU software-force events, setting bit 3 to 1 means the TEB event triggers CNTU software-force events, setting bit 4 to 1 means CNTU software-force events triggered when the PWM timer selected by the PWM generator is synchronized, and setting bit 5 to 1 means the CNTU software-forced events will not be triggered. Configure register [MCPWM_GENn_A_CNTUFORCE_MODE](#) or [MCPWM_GENn_B_CNTUFORCE_MODE](#) to specify the operation on the PWMxA or PWMxB waveform when the CNTU software-force events occur. Configuring the registers to 0 or 3 means no operation, 1 means low level, and 2 means high level.

Figure 31.3-19 shows a waveform of NCI software-force events. NCI events are used to force PWMxA output low. Forcing on PWMxB is disabled in this case.

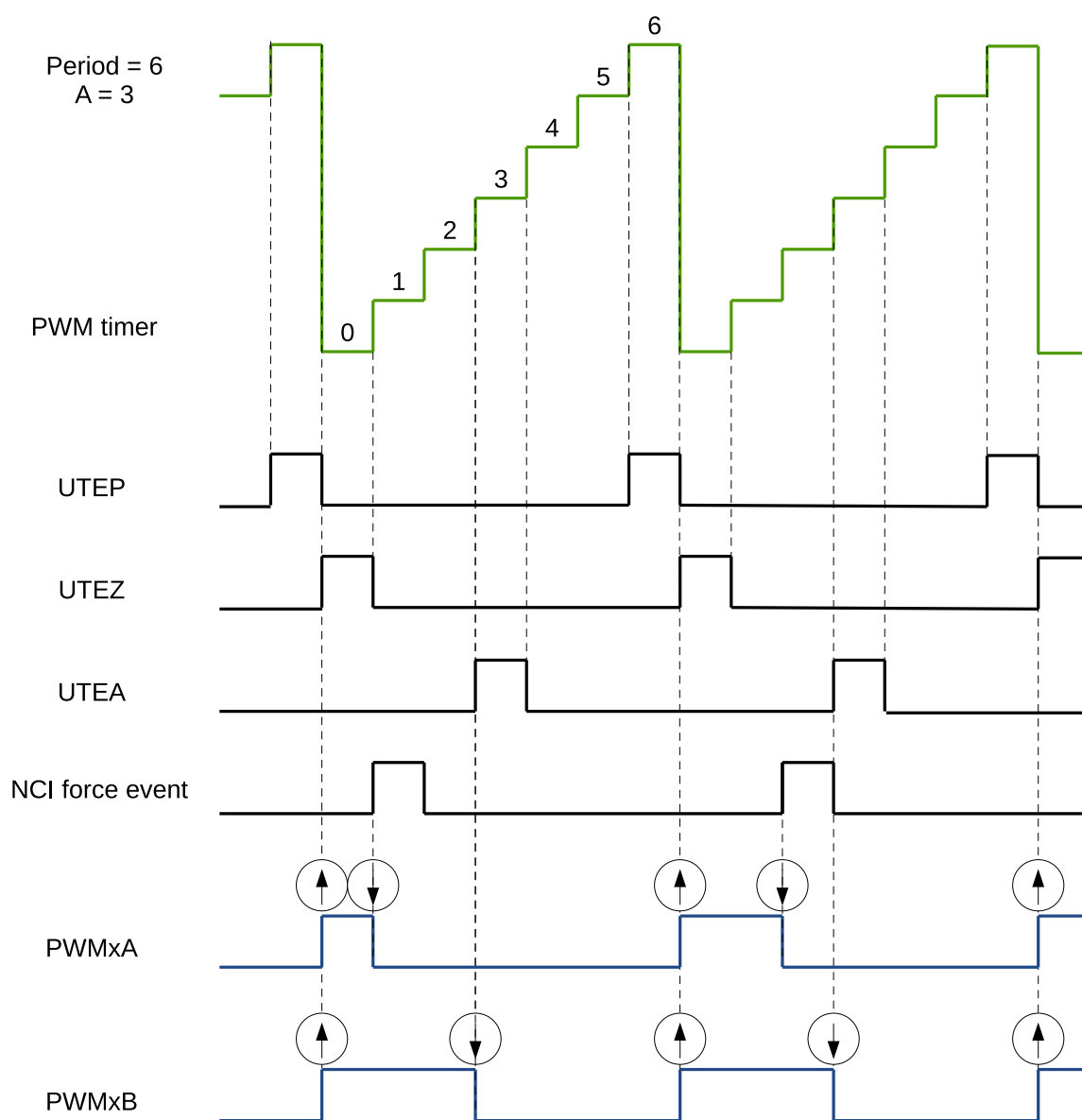


Figure 31.3-19. Example of an NCI Software-Force Event on PWMxA

Figure 31.3-20 shows a waveform of CNTU software-force events. TEZ events are selected as triggers for CNTU software-force events. CNTU is used to force the PWMxB output low. Forcing on PWMxA is disabled.

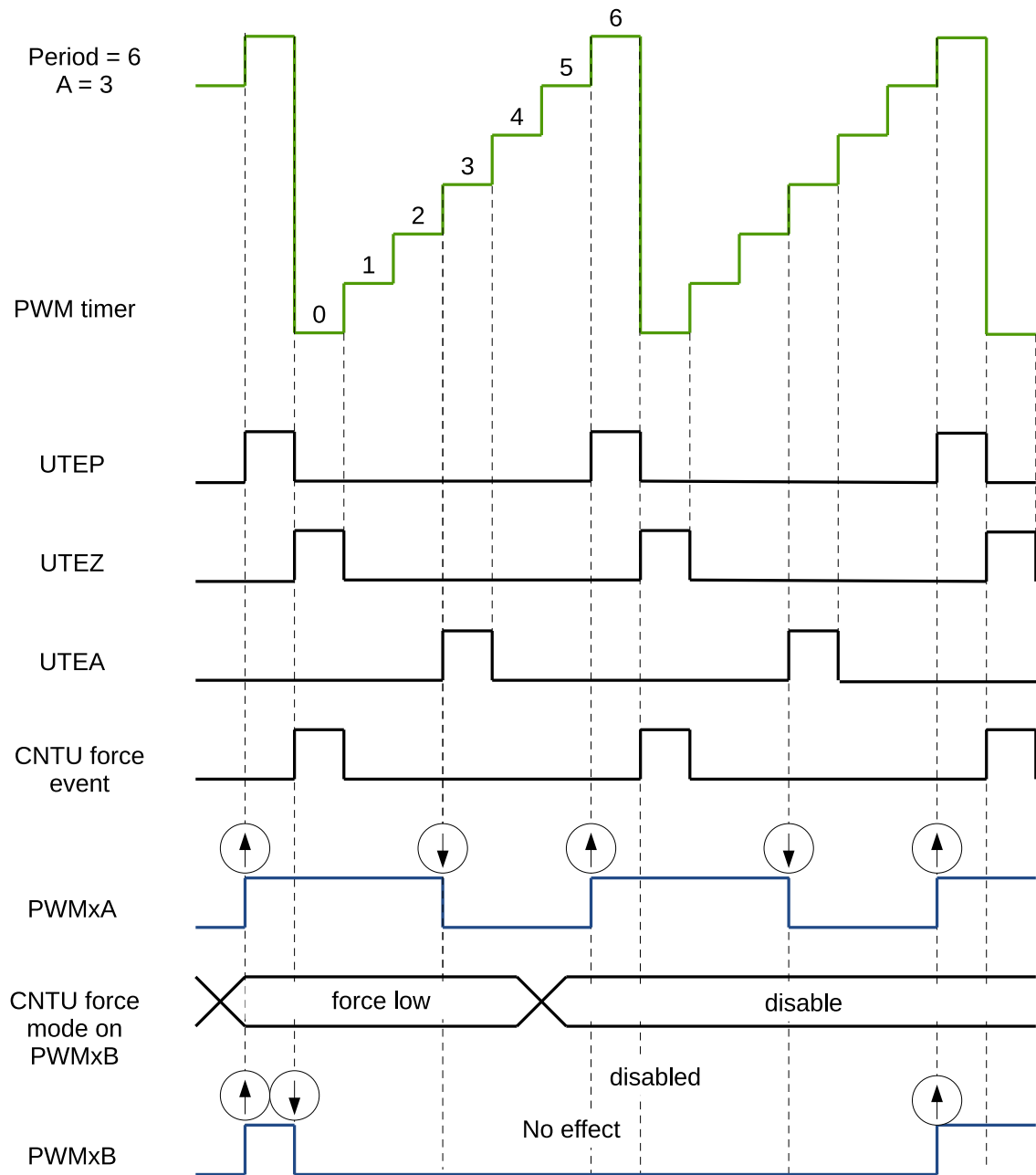


Figure 31.3-20. Example of a CNTU Software-Force Event on PWMxB

31.3.3.2 Dead Time Generator Module

Purpose of the Dead Time Generator Module

Section 31.3.3.1 introduced several options to generate signals on PWMxA and PWMxB outputs, with a specific placement of signal edges. The required dead time is obtained by altering the edge placement between signals and by setting the signal's duty cycle. Another option to control the dead time is to use a specialized module – Dead Time Generator.

The key functions of the Dead Time Generator module are as follows:

- Generating output signal pairs (PWMxA and PWMxB) with a dead time from a common source, which can be either the input signal PWMxA or PWMxB.
- Creating a dead time by adding delay to signal edges:
 - Rising edge delay (RED)
 - Falling edge delay (FED)
- Can generate PWM signal in various format. The typical dead time configurations are:
 - Active high complementary (AHC)
 - Active low complementary (ALC)
 - Active high (AH)
 - Active low (AL)
- This module may also be bypassed, if the dead time is configured directly in the generator module.

Shadow Register of Dead Time Generator

Delay registers RED and FED are shadowed with registers `MCPWM_DT $n$ _RED_CFG_REG` and `MCPWM_DT $n$ _FED_CFG_REG`. When `MCPWM_GLOBAL_UP_EN` is set to 1, the values saved in the shadow registers can be written to the active register at the specified time. The update method register for `MCPWM_DT $n$ _RED_CFG_REG` is `MCPWM_DB $n$ _RED_UPMETHOD`. The update method register for `MCPWM_DT $n$ _FED_CFG_REG` is `MCPWM_DB $n$ _FED_UPMETHOD`. The Software can also trigger a globally forced update bit `MCPWM_GLOBAL_FORCE_UP` which will prompt all registers in the module to be updated according to shadow registers. For the description of shadow registers, please see section 31.3.2.3.

Highlights for Operation of the Dead Time Generator

Options for setting up the dead time module are shown in Figure 31.3-21.

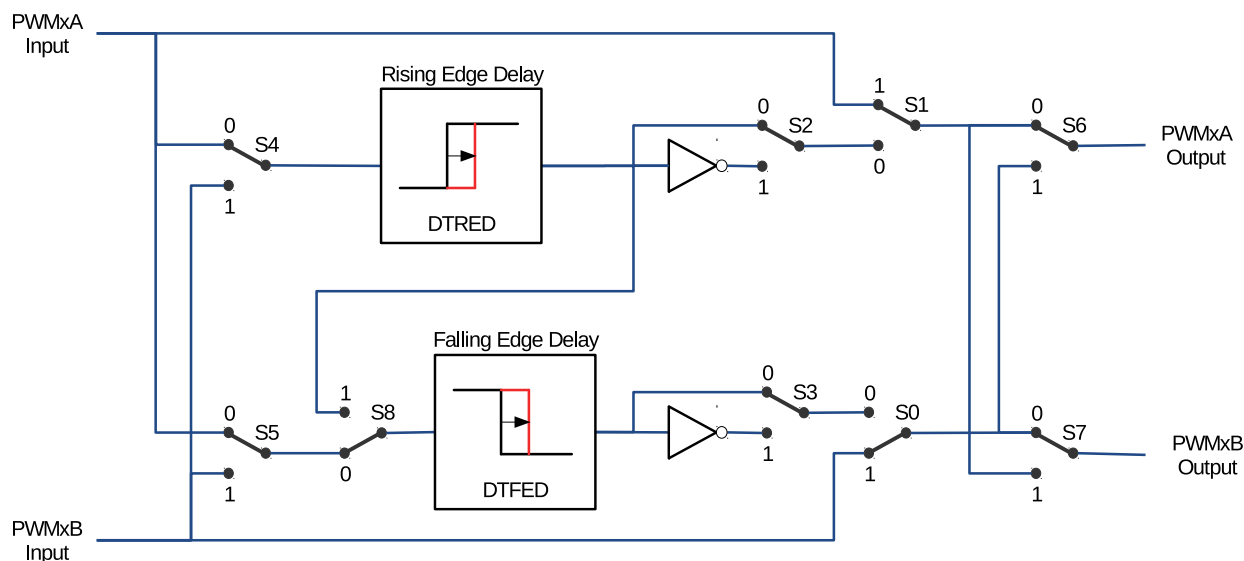


Figure 31.3-21. Options for Setting up the Dead Time Generator Module

S0-S8 in the figure above are switches controlled by fields in register `MCPWM_DTn_CFG_REG` shown in Table 31.3-5.

Table 31.3-5. Dead Time Generator Switches Control Fields

Switch	Field
S0	<code>MCPWM_DBn_B_OUTBYPASS</code>
S1	<code>MCPWM_DBn_A_OUTBYPASS</code>
S2	<code>MCPWM_DBn_RED_OUTINVERT</code>
S3	<code>MCPWM_DBn_FED_OUTINVERT</code>
S4	<code>MCPWM_DBn_RED_INSEL</code>
S5	<code>MCPWM_DBn_FED_INSEL</code>
S6	<code>MCPWM_DBn_A_OUTSWAP</code>
S7	<code>MCPWM_DBn_B_OUTSWAP</code>
S8	<code>MCPWM_DBn_DEB_MODE</code>

All switch combinations are supported, but not all of them represent the typical modes of use. Table 31.3-6 documents some typical dead time configurations. In these configurations, the position of S4 and S5 sets PWMxA as the common source of both falling edge delay (FED) and rising edge delay (RED). The modes presented in table 31.3-6 may be categorized as follows:

Table 31.3-6. Typical Dead Time Generator Operating Modes

Mode	Mode Description	S0	S1	S2	S3
1	PWMxA and PWMxB Pass Through/No Delay	1	1	X	X

Mode	Mode Description	S0	S1	S2	S3
2	Active High Complementary (AHC), see Figure 31.3-22	0	0	0	1
3	Active Low Complementary (ALC), see Figure 31.3-23	0	0	1	0
4	Active High (AH), see Figure 31.3-24	0	0	0	0
5	Active Low (AL), see Figure 31.3-25	0	0	1	1
6	PWMxA Output = PWMxA In (No Delay) PWMxB Output = PWMxA Input with Falling Edge Delay	0	1	0 or 1	0 or 1
7	PWMxA Output = PWMxA Input with Rising Edge Delay PWMxB Output = PWMxB Input with No Delay	1	0	0 or 1	0 or 1

Note:

For all the modes above, the position of the binary switches S4 to S8 is set to 0.

- **Mode 1: Bypass delays on both FED and RED**

In this mode, the dead time module is disabled. Signals of PWMxA and PWMxB pass through without any modifications.

- **Mode 2-5: Classical Dead Time Polarity Settings**

These four modes represent typical configurations of polarity and should cover the active-high/low modes in available industry power switch gate drivers. The typical waveforms are shown in Figures 31.3-22 to 31.3-25.

- **Modes 6 and 7: Bypass delay on the falling edge (FED) or rising edge (RED)**

In these two modes, either RED or FED is bypassed. As a result, the corresponding delay is not applied.

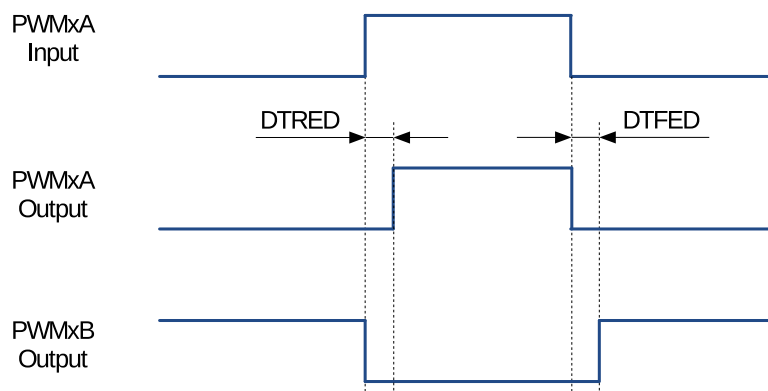


Figure 31.3-22. Active High Complementary (AHC) Dead Time Waveforms

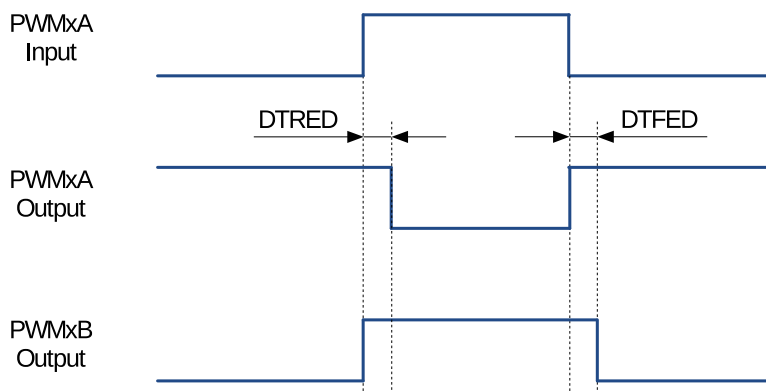


Figure 31.3-23. Active Low Complementary (ALC) Dead Time Waveforms

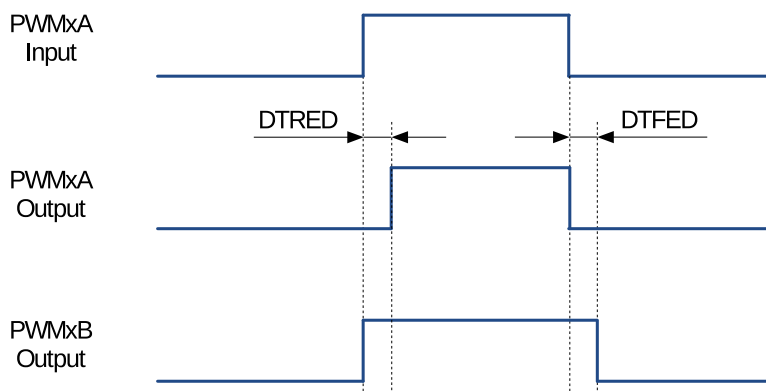


Figure 31.3-24. Active High (AH) Dead Time Waveforms

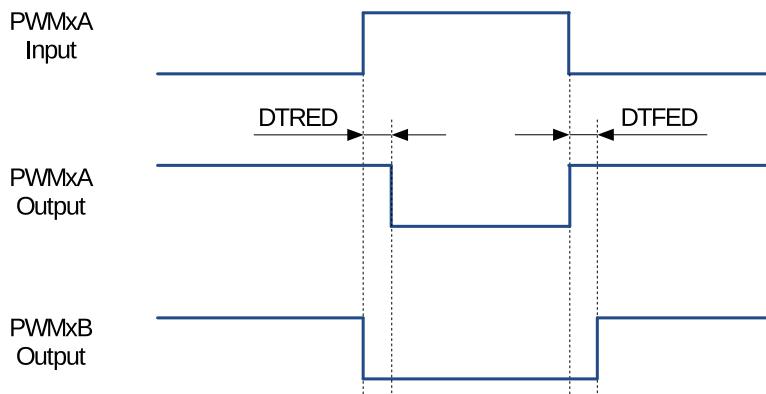


Figure 31.3-25. Active Low (AL) Dead Time Waveforms

RED and FED delays may be set up independently. The delay value is programmed using the 16-bit field `MCPWM_DB $n$ _RED` and `MCPWM_DB $n$ _FED`. The register value represents the number of clock (DT_clk) periods by which a signal edge is delayed. DT_clk can be selected from `PWM_clk` or `PT_clk` through register `MCPWM_DB $n$ _CLK_SEL`.

To calculate the delay on the falling edge (FED) and rising edge (RED), use the following formulas:

$$FED = MCPWM\_DTn\_FED \times T_{DT\_clk}$$

$$RED = MCPWM\_DTn\_RED \times T_{DT\_clk}$$

31.3.3.3 PWM Carrier Module

The coupling of PWM output to a motor driver may need isolation with a transformer. Transformers deliver only AC signals, while the duty cycle of a PWM signal may range anywhere from 0% to 100%. The PWM carrier module passes such a PWM signal through a transformer by using a high frequency carrier to modulate the signal.

Function Overview

The following key characteristics of this module are configurable:

- Carrier frequency
- Pulse width of the first pulse
- Duty cycle of the second and the subsequent pulses
- Enabling/disabling the carrier function

Operational Highlights

The PWM carrier clock (PC_clk) is derived from PWM_CLK. The frequency and duty cycle are configured by the `MCPWM_CHOPPERn_PRESCALE` and `MCPWM_CHOPPERn_DUTY` bits in the `MCPWM_CARRIERn_CFG_REG` register. The purpose of one-shot pulses is to provide the high-energy impulse to reliably turn on the power switch. Subsequent pulses sustain the power-on status. The width of a one-shot pulse is configurable with the `MCPWM_CHOPPERn_OSHTWTH` field. Enabling/disabling of the carrier module is done with the `MCPWM_CHOPPERn_EN` bit.

Waveform Examples

Figure 31.3-26 shows an example of waveforms, where a carrier is superimposed on original PWM pulses. This figure does not show the first one-shot pulse and the duty-cycle control. Related details are covered in the following two sections.

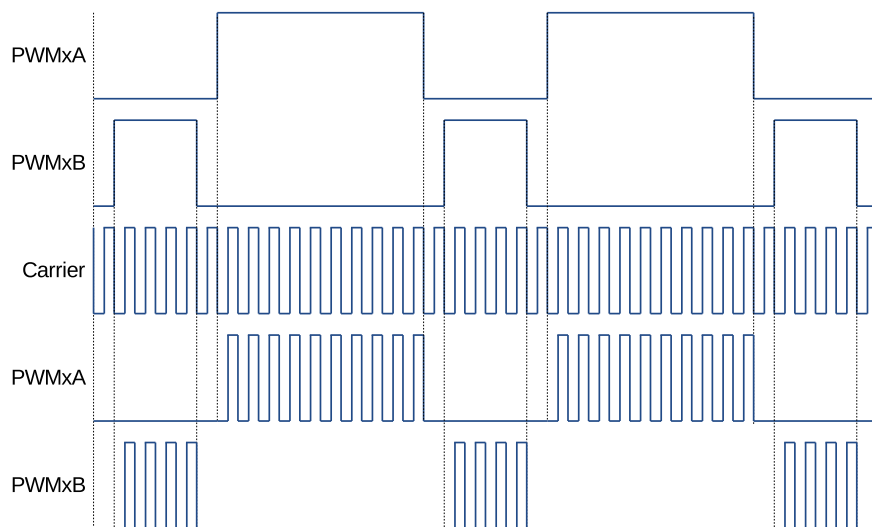


Figure 31.3-26. Example of Waveforms Showing PWM Carrier Action

One-Shot Pulse

The width of the first pulse can be configured to 16 different values, which can be calculated by the following equation:

$$T_{1stpulse} = T_{PWM\_clk} \times 8 \times (MCPWM\_CARRIERn\_PRESCALE + 1) \times (MCPWM\_CARRIERn\_OSHTWTH + 1)$$

Where:

- T_{PWM_clk} is the period of the PWM clock (PWM_clk).
- $(MCPWM\_CARRIERn\_OSHTWTH + 1)$ is the width of the first pulse (whose value ranges from 1 to 16).
- $(MCPWM\_CARRIERn\_PRESCALE + 1)$ is the PWM carrier clock's (PC_clk) prescaler value.

The first one-shot pulse and subsequent sustaining pulses are shown in Figure 31.3-27.

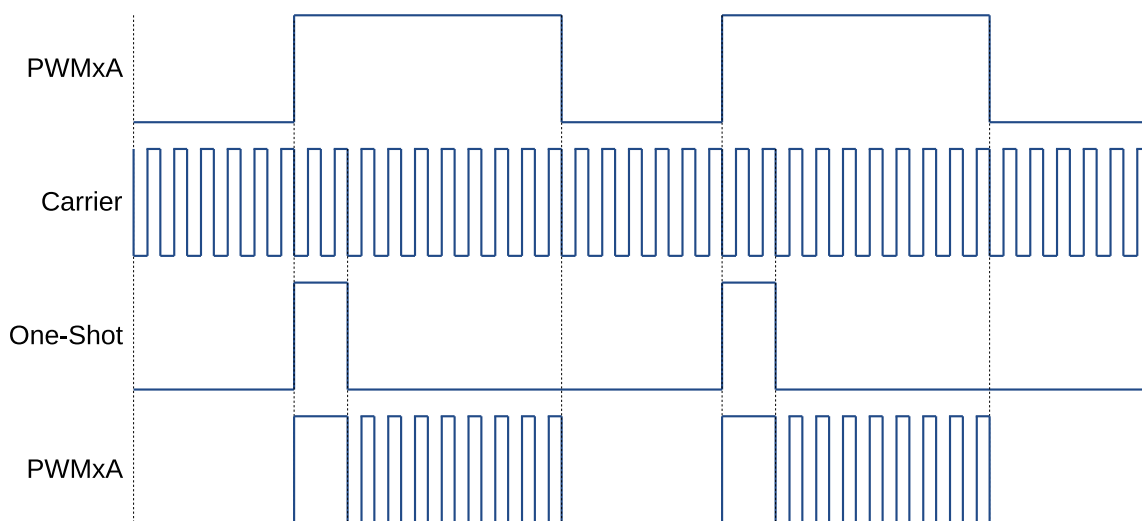


Figure 31.3-27. Example of the First Pulse and the Subsequent Sustaining Pulses of the PWM Carrier Sub-module

Duty Cycle Control

After issuing the first one-shot pulse, the remaining PWM signal is modulated according to the carrier frequency. Users can configure the duty cycle of this signal. Tuning of duty may be required, so that the signal passes through the isolating transformer and can still operate (turn on/off) the motor drive, changing rotation speed and direction.

The duty cycle may be set to one of seven values, using `MCPWM_CHOPPERn_DUTY`, or bits [7:5] of register `MCPWM_CARRIERn_CFG_REG`.

Below is the formula for calculating the duty cycle:

$$Duty = MCPWM\_CARRIERn\_DUTY \div 8$$

All seven settings of the duty cycle are shown in Figure 31.3-28.

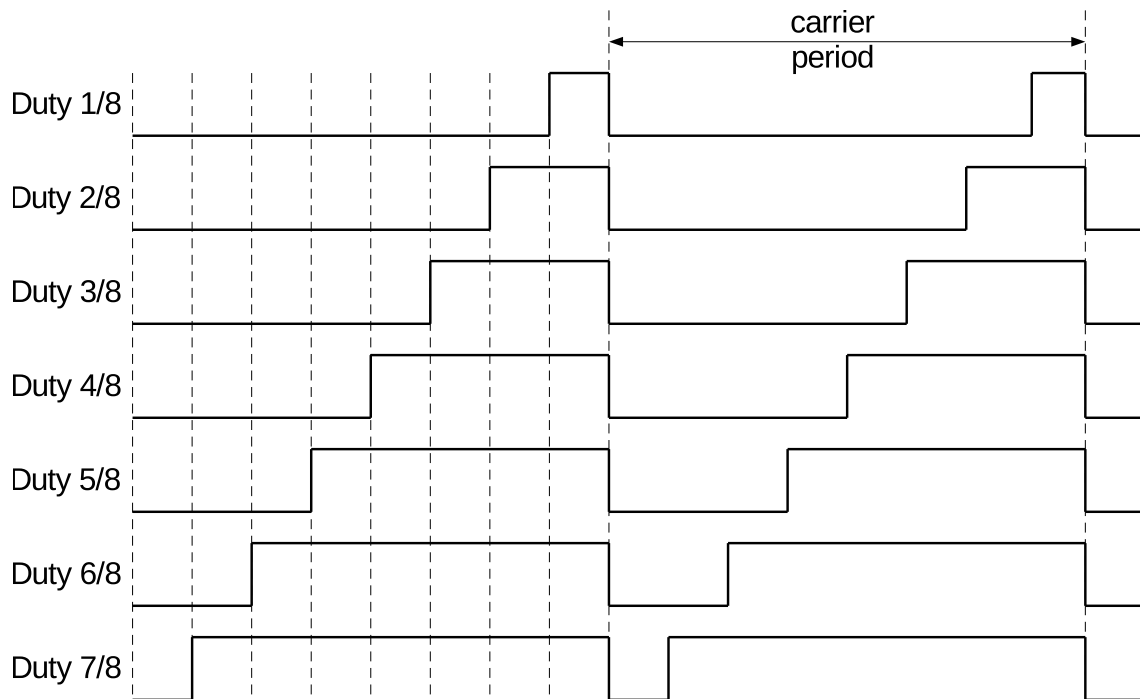


Figure 31.3-28. Possible Duty Cycle Settings for Sustaining Pulses in the PWM Carrier Submodule

31.3.3.4 Fault Detection Module

Each MCPWM peripheral is connected to three fault signals (FAULT0, FAULT1, and FAULT2) which are sourced from the GPIO matrix. These signals are intended to indicate external fault conditions, and may be preprocessed by the Fault Detection module to generate fault events. Fault events can then execute the user code to control MCPWM outputs in response to specific faults.

Function of Fault Detection Module

The key actions performed by the fault detection module are:

- Forcing outputs PWMxA and PWMxB, upon detected fault, to one of the following states:
 - High
 - Low
 - Toggle
 - No action taken
- Execution of one-shot trip (OST) upon detection of over-current conditions/short circuits.
- Cycle-by-cycle trip (CBC) to provide current-limiting operation.
- Allocation of either one-shot or cycle-by-cycle operation for each fault signal.
- Generation of interrupts for each fault input.
- Support for software-force tripping.
- Enabling or disabling of module function as required.

Operation and Configuration Tips

This section provides the operational tips and set-up options for the Fault Detection module.

Fault signals coming from pins are sampled and synced in the GPIO matrix. In order to guarantee the successful sampling of fault pulses, each pulse duration must be at least two APB clock cycles. The Fault Detection module will then sample fault signals by using PWM_clk. So, the duration of fault pulses coming from GPIO matrix must be at least one PWM_clk cycle. Differently put, regardless of the period relation between APB clock and PWM_clk, the width of fault signal pulses on pins must be at least equal to the sum of two APB clock cycles and one PWM_clk cycle.

Each level of fault signals, FAULT0 to FAULT2, can be used by the Fault Detection module to generate fault events (fault_event0 to fault_event2). Every fault event can be configured individually to provide CBC action, OST action, or none.

- **Cycle-by-Cycle (CBC) action:**

When CBC action is triggered, the state of PWMxA and PWMxB will be changed immediately according to the configuration of fields [MCPWM_TZn_A_CBC_U/D](#) and [MCPWM_TZn_B_CBC_U/D](#). Different actions can be indicated when the PWM timer is incrementing or decrementing. Different CBC action interrupts can be triggered for different fault events. Status field [MCPWM_TZn_CBC_ON](#) indicates whether a CBC action is on or off. When the fault event is no longer present, CBC actions on PWMxA/B will be cleared at a specified point, which is either a D/UTEP or D/UTEZ event. Field [MCPWM_TZn_CBCPULSE](#) determines at which event PWMxA and PWMxB will be able to resume normal actions. Therefore, in this mode, the CBC action is cleared or refreshed upon every PWM cycle.

- **One-Shot (OST) action:**

When OST action is triggered, the state of PWMxA and PWMxB will be changed immediately, depending on the setting of fields [MCPWM_TZn_A_OST_U/D](#) and [MCPWM_TZn_B_OST_U/D](#). Different actions can be configured when PWM timer is incrementing or decrementing. Different OST action interrupts can be triggered from different fault events. Status field [MCPWM_TZn_OST_ON](#) indicates whether an OST action is on or off. The OST actions on PWMxA/B are not automatically cleared when the fault event is no longer present. One-shot actions must be cleared manually by setting the [MCPWM_TZn_CLR_OST](#) bit.

31.3.4 Capture Module

31.3.4.1 Introduction

The capture module contains three complete capture channels. Channel inputs CAPO, CAP1, and CAP2 are sourced from the GPIO matrix. Thanks to the flexibility of the GPIO matrix, CAPO, CAP1, and CAP2 can be configured from any pin input. Multiple capture channels can be sourced from the same pin input, while prescaling for each channel can be set differently. Also, capture channels are sourced from different pins. This provides several options for handling capture signals by hardware in the background, instead of having them processed directly by the CPU. A capture module has the following independent key resources:

- One 32-bit timer (counter) which can be synchronized with the PWM timer, another module, or software.
- Three capture channels, each equipped with a 32-bit time-stamp and a capture prescaler.
- Independent edge polarity (rising/falling edge) selection for any capture channel.
- Input capture signal prescaling (from 1 to 256).

- Interrupt capabilities on any of the three capture events.

31.3.4.2 Capture Timer

The capture timer is a 32-bit counter incrementing continuously. It is enabled by setting `MCPWM_CAP_TIMER_EN` to 1. Its operating clock source is MCPWM core clock. When `MCPWM_CAP_SYNCI_EN` is configured, the counter will be loaded with phase stored in register `MCPWM_CAP_TIMER_PHASE_REG` at the time of a sync event. Sync events can select from PWM timers sync-out, or PWM module sync-in by configuring `MCPWM_CAP_SYNCI_SEL`. Sync events can also be generated by setting `MCPWM_CAP_SYNC_SW`. The capture timer provides timing references for all three capture channels.

31.3.4.3 Capture Channel

The capture signal coming to a capture channel will be inverted first, if needed, and then prescaled. Each capture channel has a prescaler register of `MCPWM_CAP $n$ _PRESCALE`. Finally, specified edges of preprocessed capture signal will trigger capture events. Setting `MCPWM_CAP $n$ _EN` to enable a capture channel. The capture event occurs at the time selected by the `MCPWM_CAP $n$ _MODE`. When a capture event occurs, the capture timer's value is stored in time-stamp register `MCPWM_CAP_CH $n$ _REG`. Different interrupts can be generated for different capture channels at capture events. The edge that triggers a capture event is recorded in register `MCPWM_CAP $n$ _EDGE`. The capture event can be also forced by software setting `MCPWM_CAP $n$ _SW`.

31.3.5 ETM Module

31.3.5.1 Overview

The MCPWM peripheral on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows MCPWM's ETM tasks to be triggered by any peripherals' ETM events, or MCPWM's ETM events to trigger any peripherals' ETM tasks. The capture module, the fault detection module, three timers, and three operators can generate events and respond to tasks independently. This section introduces the ETM tasks and events related to MCPWM. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

31.3.5.2 MCPWM-Related ETM Events

When setting enable field to 1, after the generation condition is met, the corresponding event would be generated. For details, please refer to Table 31.3-7 below:

Table 31.3-7. MCPWM-Related ETM Events

Enable Field	Generation Condition	Event Generated
<code>MCPWM_EVT_CAPn_EN</code>	CAP n capture event occurs	<code>MCPWMO_EVT_CAPn</code>
<code>MCPWM_EVT_TZn_OST_EN</code>	PWM operator n performs a One-Shot trip (OST) operation	<code>MCPWMO_EVT_TZn_OST</code>
<code>MCPWM_EVT_TZn_CBC_EN</code>	PWM operator n performs a cycle-by-cycle trip (CBC) operation	<code>MCPWMO_EVT_TZn_CBC</code>
<code>MCPWM_EVT_Fn_CLR_EN</code>	Fault event fault_event n is cleared	<code>MCPWMO_EVT_Fn_CLR</code>

¹ See Section 31.3.3.1 for a detailed description of timer stamp A and B.

Enable Field	Generation Condition	Event Generated
<code>MCPWM_EVT_Fn_EN</code>	Fault event <code>fault_eventn</code> is generated	<code>MCPWMO_EVT_Fn</code>
<code>MCPWM_EVT_OPn_TEB_EN</code>	The count value of the timer that PWM operator <code>n</code> selects is equal to the value of timer stamp B ¹	<code>MCPWMO_EVT_OPn_TEB</code>
<code>MCPWM_EVT_OPn_TEA_EN</code>	The count value of the timer that PWM operator <code>n</code> selects is equal to the value of timer stamp A ¹	<code>MCPWMO_EVT_OPn_TEA</code>
<code>MCPWM_EVT_OPn_TEE1_EN</code>	The count value of the timer that PWM operator <code>n</code> selects is equal to the value of register <code>MCPWM_OPn_TSTMP_E1_REG</code>	<code>MCPWMO_EVT_OPn_TEE1</code>
<code>MCPWM_EVT_OPn_TEE2_EN</code>	The count value of the timer that PWM operator <code>n</code> selects is equal to the value of register <code>MCPWM_OPn_TSTMP_E2_REG</code>	<code>MCPWMO_EVT_OPn_TEE2</code>
<code>MCPWM_EVT_TIMERn_TEP_EN</code>	The count value of timer <code>n</code> is equal to the period value <code>MCPWM_TIMERn_PERIOD</code>	<code>MCPWMO_EVT_TIMERn_TEP</code>
<code>MCPWM_EVT_TIMERn_TEZ_EN</code>	The count value of timer <code>n</code> is equal to 0	<code>MCPWMO_EVT_TIMERn_TEZ</code>
<code>MCPWM_EVT_TIMERn_STOP_EN</code>	Timer <code>n</code> 's count stops	<code>MCPWMO_EVT_TIMERn_STOP</code>

¹ See Section 31.3.3.1 for a detailed description of timer stamp A and B.

31.3.5.3 MCPWM-Related ETM Tasks

When setting the enable field to 1, after inputting valid tasks, the corresponding response operation would be generated. For details, please refer to Table 31.3-8 below:

Table 31.3-8. ETM Related Tasks

Enable Field	Valid Task Input	Response Operation
<code>MCPWM_TASK_CAPn_EN</code>	<code>MCPWMO_TASK_CAPn</code>	<code>CAPn</code> channel performs a capture operation
<code>MCPWM_TASK_CLRn_OST_EN</code>	<code>MCPWMO_TASK_CLRn_OST</code>	PWM operator <code>n</code> clears the One-Shot Trip operation
<code>MCPWM_TASK_TZn_OST_EN</code>	<code>MCPWMO_TASK_TZn_OST</code>	PWM operator <code>n</code> performs a One-Shot Trip (OST) operation
<code>MCPWM_TASK_TIMERn_PERIOD_UP_EN</code>	<code>MCPWMO_TASK_TIMERn_PERIOD_UP</code>	The period of timer <code>n</code> is updated to the value configured in the period register <code>MCPWM_TIMERn_PERIOD</code>
<code>MCPWM_TASK_TIMERn_SYNC_EN</code>	<code>MCPWMO_TASK_TIMERn_SYNC</code>	Timer <code>n</code> performs a sync operation
<code>MCPWM_TASK_GEN_STOP_EN</code>	<code>MCPWMO_TASK_GEN_STOP</code>	All the timers stop counting and the PWM signals output by all PWM operators remain unchanged

Enable Field	Valid Task Input	Response Operation
MCPWM_TASK_CMPR n _B_UP_EN	MCPWMO_TASK_CMPR n _B_UP	Timer stamp B of the PWM operator n is updated to the value of the shadow register MCPWM_GEN n _B
MCPWM_TASK_CMPR n _A_UP_EN	MCPWMO_TASK_CMPR n _A_UP	Timer stamp A of the PWM operator n is updated to the value of the shadow register MCPWM_GEN n _A

31.4 Interrupts

ESP32-C5's MCPWMO can generate the PWM0_INTR interrupt signal. The PWM0_INTR will be sent to the [Interrupt Matrix](#).

Interrupt signal PWM0_INTR can be generated by the following internal interrupt sources:

- MCPWM_TIMER x _STOP_INT: Triggered when timer x stops.
- MCPWM_TIMER x _TEZ_INT: Triggered by the TEZ event of PWM timer x .
- MCPWM_TIMER x _TEP_INT: Triggered by the TEP event of PWM timer x .
- MCPWM_FAULT x _INT: Triggered when fault_event x starts.
- MCPWM_FAULT x _CLR_INT: Triggered after fault_event x ends.
- MCPWM_CMPR x _TEA_INT: Triggered by the TEA event of PWM operator x .
- MCPWM_CMPR x _TEB_INT: Triggered by the TEB event of PWM operator x .
- MCPWM_TZ x _CBC_INT: Triggered by the CBC action of PWM x .
- MCPWM_TZ x _OST_INT: Triggered by the OST action of PWM x .
- MCPWM_CAP x _INT: Triggered by the capture event on channel x .

The above interrupt sources can only be triggered when the interrupt enable register `interrupt_source_name_ENA` is set to 1. Write 1 to `interrupt_source_name_CLR` will clear the interrupt.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [31.5 Register Summary](#).

31.5 Register Summary

The addresses in this section are relative to Motor Control PWM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Prescaler Configuration			
MCPWM_CLK_CFG_REG	PWM clock prescaler register	0x0000	R/W
MCPWM Timer 0 Configuration and Status			
MCPWM_TIMER0_CFG0_REG	PWM timer0 period and update method configuration register	0x0004	R/W
MCPWM_TIMER0_CFG1_REG	PWM timer0 working mode and start/stop control configuration register	0x0008	varies
MCPWM_TIMER0_SYNC_REG	PWM timer0 sync function configuration register	0x000C	R/W
MCPWM_TIMER0_STATUS_REG	PWM timer0 status register	0x0010	RO
MCPWM Timer 1 Configuration and Status			
MCPWM_TIMER1_CFG0_REG	PWM timer1 period and update method configuration register	0x0014	R/W
MCPWM_TIMER1_CFG1_REG	PWM timer1 working mode and start/stop control configuration register	0x0018	varies
MCPWM_TIMER1_SYNC_REG	PWM timer1 sync function configuration register	0x001C	R/W
MCPWM_TIMER1_STATUS_REG	PWM timer1 status register	0x0020	RO
MCPWM Timer 2 Configuration and status			
MCPWM_TIMER2_CFG0_REG	PWM timer2 period and update method configuration register	0x0024	R/W
MCPWM_TIMER2_CFG1_REG	PWM timer2 working mode and start/stop control configuration register	0x0028	varies
MCPWM_TIMER2_SYNC_REG	PWM timer2 sync function configuration register	0x002C	R/W
MCPWM_TIMER2_STATUS_REG	PWM timer2 status register	0x0030	RO
Common Configuration for MCPWM Timers			
MCPWM_TIMER_SYNCI_CFG_REG	Synchronization input selection for three PWM timers	0x0034	R/W
MCPWM_OPERATOR_TIMERSEL_REG	Select specific timer for PWM operators	0x0038	R/W
MCPWM Operator 0 Configuration and Status			
MCPWM_GEN0_STMP_CFG_REG	Transfer status and update method for time stamp registers A and B	0x003C	varies
MCPWM_GEN0_TSTMP_A_REG	Shadow register for register A	0x0040	R/W
MCPWM_GEN0_TSTMP_B_REG	Shadow register for register B	0x0044	R/W
MCPWM_GEN0_CFG0_REG	Fault event T0 and T1 handling	0x0048	R/W
MCPWM_GEN0_FORCE_REG	Permissives to force PWMOA and PWMOB outputs by software	0x004C	R/W

Name	Description	Address	Access
MCPWM_GEN0_A_REG	Actions triggered by events on PWM0A	0x0050	R/W
MCPWM_GEN0_B_REG	Actions triggered by events on PWM0B	0x0054	R/W
MCPWM_DTO_CFG_REG	Dead time type selection and configuration	0x0058	R/W
MCPWM_DTO_FED_CFG_REG	Shadow register for falling edge delay (FED)	0x005C	R/W
MCPWM_DTO_RED_CFG_REG	Shadow register for rising edge delay (RED)	0x0060	R/W
MCPWM_CARRIER0_CFG_REG	Carrier enable and configuratoin	0x0064	R/W
MCPWM_FH0_CFG0_REG	Actions on PWM0A and PWM0B trip events	0x0068	R/W
MCPWM_FH0_CFG1_REG	Software triggers for fault handler actions	0x006C	R/W
MCPWM_FH0_STATUS_REG	Status of fault events	0x0070	RO
MCPWM Operator 1 Configuration and Status			
MCPWM_GEN1_STMP_CFG_REG	Transfer status and update method for time stamp registers A and B	0x0074	varies
MCPWM_GEN1_TSTMP_A_REG	Shadow register for register A	0x0078	R/W
MCPWM_GEN1_TSTMP_B_REG	Shadow register for register B	0x007C	R/W
MCPWM_GEN1_CFG0_REG	Fault event T0 and T1 handling	0x0080	R/W
MCPWM_GEN1_FORCE_REG	Permissives to force PWM1A and PWM1B outputs by software	0x0084	R/W
MCPWM_GEN1_A_REG	Actions triggered by events on PWM1A	0x0088	R/W
MCPWM_GEN1_B_REG	Actions triggered by events on PWM1B	0x008C	R/W
MCPWM_DT1_CFG_REG	Dead time type selection and configuration	0x0090	R/W
MCPWM_DT1_FED_CFG_REG	Shadow register for falling edge delay (FED)	0x0094	R/W
MCPWM_DT1_RED_CFG_REG	Shadow register for rising edge delay (RED)	0x0098	R/W
MCPWM_CARRIER1_CFG_REG	Carrier enable and configuratoin	0x009C	R/W
MCPWM_FH1_CFG0_REG	Actions on PWM1A and PWM1B trip events	0x00A0	R/W
MCPWM_FH1_CFG1_REG	Software triggers for fault handler actions	0x00A4	R/W
MCPWM_FH1_STATUS_REG	Status of fault events	0x00A8	RO
MCPWM Operator 2 Configuration and Status			
MCPWM_GEN2_STMP_CFG_REG	Transfer status and update method for time stamp registers A and B	0x00AC	varies
MCPWM_GEN2_TSTMP_A_REG	Shadow register for register A	0x00B0	R/W
MCPWM_GEN2_TSTMP_B_REG	Shadow register for register B	0x00B4	R/W
MCPWM_GEN2_CFG0_REG	Fault event T0 and T1 handling	0x00B8	R/W
MCPWM_GEN2_FORCE_REG	Permissives to force PWM2A and PWM2B outputs by software	0x00BC	R/W
MCPWM_GEN2_A_REG	Actions triggered by events on PWM2A	0x00C0	R/W
MCPWM_GEN2_B_REG	Actions triggered by events on PWM2B	0x00C4	R/W
MCPWM_DT2_CFG_REG	Dead time type selection and configuration	0x00C8	R/W
MCPWM_DT2_FED_CFG_REG	Shadow register for falling edge delay (FED)	0x00CC	R/W
MCPWM_DT2_RED_CFG_REG	Shadow register for rising edge delay (RED)	0x00D0	R/W
MCPWM_CARRIER2_CFG_REG	Carrier enable and configuratoin	0x00D4	R/W
MCPWM_FH2_CFG0_REG	Actions on PWM2A and PWM2B trip events	0x00D8	R/W
MCPWM_FH2_CFG1_REG	Software triggers for fault handler actions	0x00DC	R/W
MCPWM_FH2_STATUS_REG	Status of fault events	0x00E0	RO

Name	Description	Address	Access
Fault Detection Configuration and Status			
MCPWM_FAULT_DETECT_REG	Fault detection configuration and status	0x00E4	varies
Capture Configuration and Status			
MCPWM_CAP_TIMER_CFG_REG	Configure capture timer	0x00E8	varies
MCPWM_CAP_TIMER_PHASE_REG	Phase for capture timer sync	0x00EC	R/W
MCPWM_CAP_CHO_CFG_REG	Capture channel 0 configuration and enable	0x00F0	varies
MCPWM_CAP_CH1_CFG_REG	Capture channel 1 configuration and enable	0x00F4	varies
MCPWM_CAP_CH2_CFG_REG	Capture channel 2 configuration and enable	0x00F8	varies
MCPWM_CAP_CHO_REG	ch0 capture value status register	0x00FC	RO
MCPWM_CAP_CH1_REG	ch1 capture value status register	0x0100	RO
MCPWM_CAP_CH2_REG	ch2 capture value status register	0x0104	RO
MCPWM_CAP_STATUS_REG	Edge of last capture trigger	0x0108	RO
Enable Update of Active Registers			
MCPWM_UPDATE_CFG_REG	Enable update	0x010C	R/W
Manage Interrupts			
MCPWM_INT_ENA_REG	Interrupt enable bits	0x0110	R/W
MCPWM_INT_RAW_REG	Raw interrupt status	0x0114	R/WTC /SS
MCPWM_INT_ST_REG	Masked interrupt status	0x0118	RO
MCPWM_INT_CLR_REG	Interrupt clear bits	0x011C	WT
MCPWM Event Enable Registers			
MCPWM_EVT_EN_REG	MCPWM event enable register	0x0120	R/W
MCPWM_EVT_EN2_REG	MCWM event enable register2	0x0128	R/W
MCPWM Task Enable Register			
MCPWM_TASK_EN_REG	MCPWM task enable register	0x0124	R/W
MCPWM Generator Configuration Registers			
MCPWM_OPO_TSTMP_E1_REG	Generator0 time stamp E1 value configuration register	0x012C	R/W
MCPWM_OPO_TSTMP_E2_REG	Generator0 time stamp E2 value configuration register	0x0130	R/W
MCPWM_OP1_TSTMP_E1_REG	Generator1 time stamp E1 value configuration register	0x0134	R/W
MCPWM_OP1_TSTMP_E2_REG	Generator1 time stamp E2 value configuration register	0x0138	R/W
MCPWM_OP2_TSTMP_E1_REG	Generator2 time stamp E1 value configuration register	0x013C	R/W
MCPWM_OP2_TSTMP_E2_REG	Generator2 time stamp E2 value configuration register	0x0140	R/W
MCPWM APB Configuration Register			
MCPWM_CLK_REG	MCPWM APB configuration register	0x0144	R/W
Version Register			
MCPWM_VERSION_REG	Version control register	0x0148	R/W

31.6 Registers

The addresses in this section are relative to Motor Control PWM base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

Register 31.1. MCPWM_CLK_CFG_REG (0x0000)

(reserved)																								MCPWM_CLK_PRESCALE									
31																								8	7	0							
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0x0								Reset	

MCPWM_CLK_PRESCALE Configures the prescaler value of the clock, so that the period of $PWM\_clk = 6.25\text{ ns} * (PWM\_CLK\_PRESCALE + 1)$. (R/W)

Register 31.2. MCPWM_TIMER n _CFG0_REG (n : 0-2) (0x0004+0x10* n)

31	26	25	24	23	8	7	0	
(reserved)			MCPWM_TIMER0_PERIOD			MCPWM_TIMER0_PRESCALE		
0 0 0 0 0 0			0			0xff		
						0x0		
Reset								

MCPWM_TIMER n _PRESCALE Configures the prescaler value of timer n , so that the period of $PTO\_clk = \text{Period of PWM\_clk} * (\text{PWM\_TIMER}n\_PRESCALE + 1)$. (R/W)

MCPWM_TIMER n _PERIOD Configures the period shadow of PWM timer n .
(R/W)

MCPWM_TIMER n _PERIOD_UPMETHOD Configures the update method for active register of PWM timer n period.

0: Immediate

1: TEZ

2: Sync

3: TEZ or sync

TEZ here and below means timer equals zero event.

(R/W)

Register 31.3. MCPWM_TIMER n _CFG1_REG(n : 0-2) (0x0008+0x10* n)

(reserved)																										MCPWM_TIMER _n _MOD		MCPWM_TIMER _n _START		
31																										5	4	3	2	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																										0x0		0x0		Reset

MCPWM_TIMER n _START Configures conditions to start/stop PWM timer n .

- 0: If PWM timer0 starts, then stops at TEZ
- 1: If timer0 starts, then stops at TEP
- 2: PWM timer0 starts and runs on
- 3: Timer0 starts and stops at the next TEZ
- 4: Timer0 starts and stops at the next TEP
- 5: Invalid. No effect
- 6: Invalid. No effect
- 7: Invalid. No effect

TEP here and below means the event that happens when the timer equals to period.

(R/W/SC)

MCPWM_TIMER n _MOD Configures the working mode of PWM timer n .

- 0: Freeze
- 1: Increase mode
- 2: Decrease mode
- 3: Up-down mode

(R/W)

Register 31.4. MCPWM_TIMER n _SYNC_REG (n : 0-2) (0x000C+0x10* n)

31	21	20	19	4	3	2	1	0
(reserved) MCPWM_TIMER_n_PHASE_DIRECTION MCPWM_TIMER_n_PHASE								
 MCPWM_TIMER_n_SYNCO_SEL MCPWM_TIMER_n_SYNC_SW MCPWM_TIMER_n_SYNCO_EN								
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0								
Reset								

MCPWM_TIMERn_SYNCI_EN Configures whether to enable timer n reloading with phase on sync input event.

0: Disable

1: Enable

(R/W)

MCPWM_TIMER n _SYNC_SW Configures whether to trigger a software sync.

0: No effect

1: Trigger a software sync

(R/W)

MCPWM_TIMER n _SYNCO_SEL Configures PWM timer n sync out selection.

0: sync_in. The sync out will always generate when toggling the MCPWM_TIMER_n_SYNC_SW bit.

1: TEZ

2: TEP

3: No effect

(R/W)

MCPWM_TIMER n _PHASE Configures the phase for timer n reload on sync event.

(R/W)

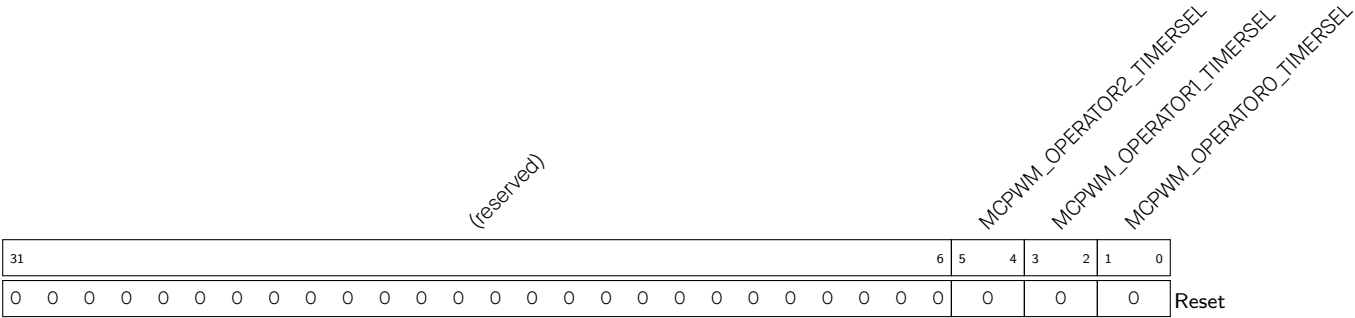
MCPWM_TIMER n _PHASE_DIRECTION Configures the PWM timer n 's direction when timer n mode is up-down mode.

0: Increase

1: Decrease

(R/W)

Register 31.7. MCPWM_OPERATOR_TIMERSEL_REG (0x0038)



MCPWM_OPERATOR_n_TIMERSEL Configures which PWM timer will be the timing reference for PWM operator_n.

0: timer0

1: timer1

2: timer2

3: Invalid, will select timer2

(R/W)

Register 31.8. MCPWM_GEN n _STMP_CFG_REG (n : 0-2) (0x003C+0x38* n)

(reserved)																								MCPWM_CMPR _n _B_SHDW_FULL				MCPWM_CMPR _n _A_SHDW_FULL				MCPWM_CMPR _n _B_UPMETHOD				MCPWM_CMPR _n _A_UPMETHOD			
31																								10		9	8	7	4		3	0		Reset					
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								0	0	0		0											

Reset

MCPWM_CMPR n _A_UPMETHOD Configures the update method for PWM generator n time stamp A's active register.

When all bits are set to 0: Immediately

When bit0 is set to 1: TEZ

When bit1 is set to 1: TEP

When bit2 is set to 1: Sync

When bit3 is set to 1: Disable the update

(R/W)

MCPWM_CMPR n _B_UPMETHOD Configures the update method for PWM generator n time stamp B's active register.

When all bits are set to 0: Immediately

When bit0 is set to 1: TEZ

When bit1 is set to 1: TEP

When bit2 is set to 1: Sync

When bit3 is set to 1: Disable the update

(R/W)

MCPWM_CMPR n _A_SHDW_FULL Configures whether the value in the shadow register is written to the corresponding active register at a specific time. This field is set and reset by hardware.

0: Write the latest value in the shadow register to A's active register

1: Write the value to the PWM generator n time stamp A's shadow register, and wait to be transferred to A's active register

(R/SC/WTC)

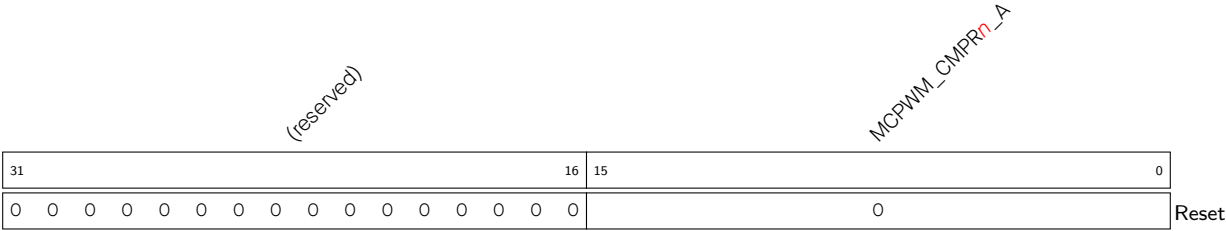
MCPWM_CMPR n _B_SHDW_FULL Configures whether the value in the shadow register is written to the corresponding active register at a specific time. This field is set and reset by hardware.

0: Write the latest value in the shadow register to B's active register

1: Write the value to the PWM generator n time stamp B's shadow register, and wait to be transferred to B's active register

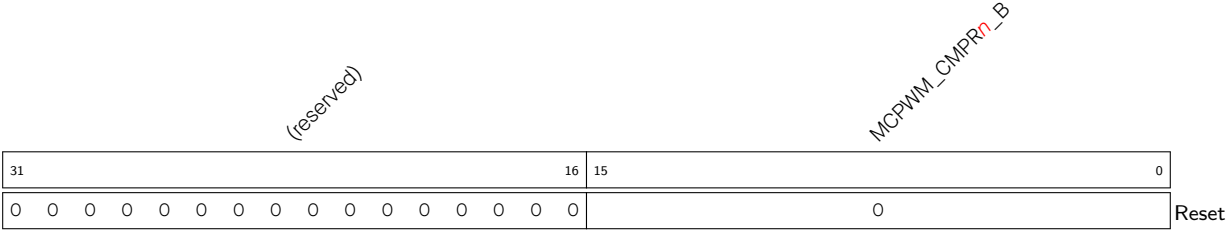
(R/SC/WTC)

Register 31.9. MCPWM_GENn_TSTMP_A_REG(*n*: 0-2) (0x0040+0x38**n*)



MCPWM_CMPRn_A Configures the value of PWM generator *n* time stamp A's shadow register.
(R/W)

Register 31.10. MCPWM_GENn_TSTMP_B_REG(*n*: 0-2) (0x0044+0x38**n*)



MCPWM_CMPRn_B Configures the value of PWM generator *n* time stamp B's shadow register.
(R/W)

Register 31.11. MCPWM_GEN n _CFG0_REG(n : 0-2) (0x0048+0x38* n)

[illegible]

MCPWM_GEN n _CFG_UPMETHOD Configures the update method for PWM generator n 's active register.

When all bits are set to 0: Immediately

When bit0 is set to 1: TEZ

When bit1 is set to 1: TEP

When bit2 is set to 1: Sync

When bit3 is set to 1: Disable the update

(R/W)

MCPWM_GEN n _TO_SEL Configures source selection for PWM generator n event_t0, take effect immediately.

0: fault_event0

1: fault_event1

2: fault_event2

3: sync_taken

4: Invalid, selects nothing

(R/W)

MCPWM_GEN*n*_T1_SEL Configures source selection for PWM generator*n* event_t1, take effect immediately.

0: fault_event0

1: fault_event1

2: fault event2

3: sync_taken

4: Invalid, selects nothing

(R/W)

Register 31.12. MCPWM_GEN n _FORCE_REG(n : 0-2) (0x004C+0x38* n)

(reserved)																MCPWM_GEN _n _B_NCIFORCE_MODE					MCPWM_GEN _n _B_CNTUFORCE_MODE					MCPWM_GEN _n _A_NCIFORCE_MODE					MCPWM_GEN _n _A_CNTUFORCE_MODE					MCPWM_GEN _n _UPMETHOD				
31																16	15	14	13	12	11	10	9	8	7	6	5	0												
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0	0	0	0	0	0	0	0	0	0x20					Reset										

MCPWM_GEN n _CNTUFORCE_UPMETHOD Configures the update method for continuous software force of PWM generator n .

When all bits are set to 0: Immediately

When bit0 is set to 1: TEZ

When bit1 is set to 1: TEP

When bit2 is set to 1: TEA

When bit3 is set to 1: TEB

When bit4 is set to 1: Sync

When bit5 is set to 1: Disable update

TEA/B means an event generated when the timer's value equals to that of register A/B.

(R/W)

MCPWM_GEN n _A_CNTUFORCE_MODE Configures the continuous software force mode for PWM n

A.

0: Disabled

1: Low

2: High

3: Disabled

(R/W)

MCPWM_GEN n _B_CNTUFORCE_MODE Configures the continuous software force mode for PWM n

B.

0: Disabled

1: Low

2: High

3: Disabled

(R/W)

MCPWM_GEN n _A_NCIFORCE Configures whether to trigger the non-continuous immediate software-force event for PWM n A.

0: Invalid

1: Trigger a force event

(R/W)

Continued on the next page...

Register 31.12. MCPWM_GEN n _FORCE_REG(n : 0-2) (0x004C+0x38* n)

Continued from the previous page...

MCPWM_GEN n _A_NCIFORCE_MODE Configures non-continuous immediate software force mode for PWM n A.

0: Disabled

1: Low

2: High

3: Disabled

(R/W)

MCPWM_GEN n _B_NCIFORCE Configures whether to trigger the non-continuous immediate software-force event for PWM n B.

0: Invalid

1: Trigger a force event

(R/W)

MCPWM_GEN n _B_NCIFORCE_MODE Configures non-continuous immediate software force mode for PWM n B.

0: Disabled

1: Low

2: High

3: Disabled

(R/W)

Register 31.13. MCPWM_GEN n _A_REG(n : 0-2) (0x0050+0x38* n)

(reserved)								MCPWM_GEN n _A_DT1		MCPWM_GEN n _A_DTO		MCPWM_GEN n _A_DTEB		MCPWM_GEN n _A_DTEA		MCPWM_GEN n _A_DTEP		MCPWM_GEN n _A_DTEZ		MCPWM_GEN n _A_UT1		MCPWM_GEN n _A_UTO		MCPWM_GEN n _A_UTEB		MCPWM_GEN n _A_UTEA		MCPWM_GEN n _A_UTEZ				
31							24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0		0		0		0		0		0		0		0		0		0		0		0		Reset

Reset

MCPWM_GEN n _A_UTEZ Configures action on PWM n A triggered by event TEZ when timer increasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _A_UTEZ Configures action on PWM n A triggered by event TEP when timer increasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _A_UTEA Configures action on PWM n A triggered by event TEA when timer increasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _A_UTEB Configures action on PWM n A triggered by event TEB when timer increasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

Continued on the next page...

Register 31.13. MCPWM_GEN n _A_REG(n : 0-2) (0x0050+0x38* n)

Continued from the previous page...

MCPWM_GEN n _A_UT0 Configures action on PWM n A triggered by event_t0 when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _A_UT1 Configures action on PWM n A triggered by event_t1 when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _A_DTEZ Configures action on PWM n A triggered by event TEZ when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _A_DTEP Configures action on PWM n A triggered by event TEP when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _A_DTEA Configures action on PWM n A triggered by event TEA when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

Continued on the next page...

Register 31.13. MCPWM_GEN n _A_REG(n : 0-2) (0x0050+0x38* n)

Continued from the previous page...

MCPWM_GEN n _A_DTEB Configures action on PWM n A triggered by event TEB when timer decreasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _A_DTO Configures action on PWM n A triggered by event_t0 when timer decreasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _A_DT1 Configures action on PWM n A triggered by event_t1 when timer decreasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

Register 31.14. MCPWM_GEN n _B_REG(n : 0-2) (0x0054+0x38* n)

(reserved)								MCPWM_GEN _n _B_DT1		MCPWM_GEN _n _B_DTO		MCPWM_GEN _n _B_DTEB		MCPWM_GEN _n _B_DTEA		MCPWM_GEN _n _B_DTEP		MCPWM_GEN _n _B_DTEZ		MCPWM_GEN _n _B_UT1		MCPWM_GEN _n _B_UTO		MCPWM_GEN _n _B_UTEA		MCPWM_GEN _n _B_UTEZ						
31								24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

MCPWM_GEN n _B_UTEZ Configures action on PWM n B triggered by event TEZ when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_UTEZ Configures action on PWM n B triggered by event TEP when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_UTEA Configures action on PWM n B triggered by event TEA when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_UTEB Configures action on PWM n B triggered by event TEB when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_UTO Configures action on PWM n B triggered by event_t0 when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

Continued on the next page...

Register 31.14. MCPWM_GEN n _B_REG(n : 0-2) (0x0054+0x38* n)

Continued from the previous page...

MCPWM_GEN n _B_UT1 Configures action on PWM n B triggered by event_t1 when timer increasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_DTEZ Configures action on PWM n B triggered by event TEZ when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_DTEP Configures action on PWM n B triggered by event TEP when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_DTEA Configures action on PWM n B triggered by event TEA when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

MCPWM_GEN n _B_DTEB Configures action on PWM n B triggered by event TEB when timer decreasing.

- 0: No change
 - 1: Low
 - 2: High
 - 3: Toggle
- (R/W)

Continued on the next page...

Register 31.14. MCPWM_GEN n _B_REG(n : 0-2) (0x0054+0x38* n)

Continued from the previous page...

MCPWM_GEN n _B_DTO Configures action on PWM n B triggered by event_t0 when timer decreasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

MCPWM_GEN n _B_DT1 Configures action on PWM n B triggered by event_t1 when timer decreasing.

0: No change

1: Low

2: High

3: Toggle

(R/W)

Register 31.15. MCPWM_DT n _CFG_REG(n : 0-2) (0x0058+0x38* n)

(reserved)																MCPWM_DBn_CLK_SEL MCPWM_DBn_B_OUTBYPASS MCPWM_DBn_A_OUTBYPASS MCPWM_DBn_FED_OUTINVERT MCPWM_DBn_FED_OUTINSEL MCPWM_DBn_FED_OUTSWAP MCPWM_DBn_DEB_MODE MCPWM_DBn_RED_UPMETHOD MCPWM_DBn_FED_UPMETHOD																	
31																18	17	16	15	14	13	12	11	10	9	8	7	4		3			0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0		Reset										

MCPWM_DBn_FED_UPMETHOD Configures the update method for FED (Falling edge delay) active register.

0: Immediate

Bit0 is set to 1: TEZ

Bit1 is set to 1: TEP

Bit2 is set to 1: Sync

Bit3 is set to 1: Disable the update

(R/W)

MCPWM_DBn_RED_UPMETHOD Configures the update method for RED (Rising edge delay) active register.

0: Immediate

Bit0 is set to 1: TEZ

Bit1 is set to 1: TEP

Bit2 is set to 1: Sync

Bit3 is set to 1: Disable the update

(R/W)

MCPWM_DBn_DEB_MODE Configures the S8 switch in Table 31.3-5.

0: FED/RED take effect on different paths separately

1: FED/RED take effect on B path, A out is in bypass or dulpB mode

(R/W)

MCPWM_DBn_A_OUTSWAP Configures the S6 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DBn_B_OUTSWAP Configures the S7 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DBn_RED_INSEL Configures the S4 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DBn_FED_INSEL Configures the S5 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DBn_RED_OUTINVERT Configures the S2 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

Continued on the next page...

Register 31.15. MCPWM_DT n _CFG_REG(n : 0-2) (0x0058+0x38* n)

Continued from the previous page...

MCPWM_DB n _FED_OUTINVERT Configures the S3 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DB n _A_OUTBYPASS Configures the S1 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DB n _B_OUTBYPASS Configures the S0 switch in Table 31.3-5. For typical configurations, please refer to Table 31.3-6. (R/W)

MCPWM_DB n _CLK_SEL Configures dead time generator n clock selection.

0: PWM_CLK

1: PT_CLK

(R/W)

Register 31.16. MCPWM_DT n _FED_CFG_REG(n : 0-2) (0x005C+0x38* n)

(reserved)																MCPWM_DB _n _FED															
31															16	15															0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														Reset	

MCPWM_DB n _FED Configures the shadow register for FED. (R/W)

Register 31.17. MCPWM_DT n _RED_CFG_REG(n : 0-2) (0x0060+0x38* n)

(reserved)																MCPWM_DB _n _RED																
31																0																
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0																Reset

MCPWM_DB n _RED Configures the shadow register for RED. (R/W)

Register 31.18. MCPWM_CARRIER n _CFG_REG(n : 0-2) (0x0064+0x38* n)

(reserved)																MCPWM_CHOPPER _n _IN_INVERT		MCPWM_CHOPPER _n _OUT_INVERT		MCPWM_CHOPPER _n _OSHTWTH		MCPWM_CHOPPER _n _DUTY		MCPWM_CHOPPER _n _PRESCALE		MCPWM_CHOPPER _n _EN		
31																14	13	12	11	8		7	5		4	1		0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0	0	0		0		0		0		Reset		

MCPWM_CHOPPER n _EN Configures whether to enable carrier n .

0: Bypassed

1: Enabled

(R/W)

MCPWM_CHOPPER n _PRESCALE Configures the prescale value of PWM carrier n clock (PC_clk), so that period of PC_clk = period of PWM_clk * (PWM_CARRIER n _PRESCALE + 1). (R/W)

MCPWM_CHOPPER n _DUTY Configures carrier duty. Duty = PWM_CARRIER n _DUTY / 8. (R/W)

MCPWM_CHOPPER n _OSHTWTH Configures width of the first pulse. Measurement unit: Periods of the carrier. (R/W)

MCPWM_CHOPPER n _OUT_INVERT Configures whether to invert the output of PWM n A and PWM n B for this submodule.

0: Normal

1: Invert

(R/W)

MCPWM_CHOPPER n _IN_INVERT Configures whether to invert the input of PWM n A and PWM n B for this submodule.

0: Normal

1: Invert

(R/W)

Register 31.19. MCPWM_FH_n_CFG0_REG(*n*: 0-2) (0x0068+0x38**n*)

[illegible]

MCPWM_TZ_n_SW_CBC Configures whether to enable software force cycle-by-cycle mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_F2_CBC Configures whether event_f2 will trigger cycle-by-cycle mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_F1_CBC Configures whether event_f1 will trigger cycle-by-cycle mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_FO_CBC Configures whether event_f0 will trigger cycle-by-cycle mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_SW_OST Configures whether to enable software force one-shot mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_F2_OST Configures whether event_f2 will trigger one-shot mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZ_n_F1_OST Configures whether event_f1 will trigger one-shot mode action.

0: Disable

1: Enable

(R/W)

MCPWM_TZn_Fn_OST Configures whether event_f0 will trigger one-shot mode action.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 31.19. MCPWM_FH n _CFG0_REG(n : 0-2) (0x0068+0x38* n)

Continued from the previous page...

MCPWM_TZ n _A_CBC_D Configures cycle-by-cycle mode action on PWM n A when fault event occurs and timer is decreasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ n _A_CBC_U Configures cycle-by-cycle mode action on PWM n A when fault event occurs and timer is increasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ n _A_OST_D Configures one-shot mode action on PWM n A when fault event occurs and timer is decreasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ n _A_OST_U Configures one-shot mode action on PWM n A when fault event occurs and timer is increasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ n _B_CBC_D Configures cycle-by-cycle mode action on PWM n B when fault event occurs and timer is decreasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

Continued on the next page...

Register 31.19. MCPWM_FH_{*n*}_CFG0_REG(*n*: 0-2) (0x0068+0x38n*)**

Continued from the previous page...

MCPWM_TZ_{*n*}_B_CBC_U Configures cycle-by-cycle mode action on PWM_{*n*} B when fault event occurs and timer is increasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ_{*n*}_B_OST_D Configures one-shot mode action on PWM_{*n*} B when fault event occurs and timer is decreasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

MCPWM_TZ_{*n*}_B_OST_U Configures one-shot mode action on PWM_{*n*} B when fault event occurs and timer is increasing.

- 0: Do nothing
 - 1: Force low
 - 2: Force high
 - 3: Toggle
- (R/W)

Register 31.20. MCPWM_FH_n_CFG1_REG(*n*: 0-2) (0x006C+0x38**n*)

(reserved)																												MCPWM_TZ _n _FORCE_OST				MCPWM_TZ _n _FORCE_CBC				MCPWM_TZ _n _CBCPULSE				MCPWM_TZ _n _CLR_OST																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																												
31																												5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

MCPWM_TZ_n_CLR_OST Configures whether to generate a one-shot mode action clear by software.

0: No effect

1: Triggers a clear for ongoing one-shot mode action by software

(R/W)

MCPWM_TZ_n_CBCPULSE Configures the refresh moment selection of cycle-by-cycle mode action.

0: Select nothing, will not refresh

Bit0 is set to 1: TEZ

Bit1 is set to 1: TEP

(R/W)

MCPWM_TZ_n_FORCE_CBC Configures whether to generate a software cycle-by-cycle mode action.

0: No effect

1: Triggers a cycle-by-cycle mode action by software

(R/W)

MCPWM_TZ_n_FORCE_OST Configures whether to generate a software one-shot mode action.

0: No effect

1: Triggers a one-shot mode action by software

(R/W)

Register 31.21. MCPWM_FH_{*n*}_STATUS_REG(*n*: 0-2) (0x0070+0x38**n*)

(reserved)																										MCPWM_TZ _{<i>n</i>} _OST_ON		MCPWM_TZ _{<i>n</i>} _CBC_ON	
31																									2	1	0		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

MCPWM_TZ_{*n*}_CBC_ON Represents whether an cycle-by-cycle mode action is on going.

0: No action

1: Ongoing

(RO)

MCPWM_TZ_{*n*}_OST_ON Represents whether a one-shot mode action is ongoing.

0: No action

1: Ongoing

(RO)

Register 31.22. MCPWM_FAULT_DETECT_REG (0x00E4)

(reserved)																								MCPWM_EVENT_F2 MCPWM_EVENT_F1 MCPWM_EVENT_F0 MCPWM_F2_POLE MCPWM_F1_POLE MCPWM_F0_POLE MCPWM_F2_EN MCPWM_F1_EN MCPWM_F0_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																								9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCPWM_F_{*n*}_EN Configures whether to enable event_*fn* generation.

0: Disable

1: Enable

(R/W)

MCPWM_F_{*n*}_POLE Configures event_*fn* trigger polarity on FAULT_{*n*} source from GPIO matrix.

0: Level low

1: Level high

(R/W)

MCPWM_EVENT_F_{*n*} Represents whether an event_*fn* is ongoing. This field is set and reset by hardware.

0: No action

1: Ongoing

(RO)

Register 31.23. MCPWM_CAP_TIMER_CFG_REG (0x00E8)

(reserved)																										MCPWM_CAP_SYNC_SW		MCPWM_CAP_SYNCI_SEL		MCPWM_CAP_SYNCI_EN		MCPWM_CAP_TIMER_EN	
31																										6	5	4		2		1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																										0	0		0		0	Reset	

Reset

MCPWM_CAP_TIMER_EN Configures whether to enable capture timer increment incrementing under APB_CLK.

0: Disable

1: Enable

(R/W)

MCPWM_CAP_SYNCI_EN Configures whether to enable capture timer sync.

0: Disable

1: Enable

(R/W)

MCPWM_CAP_SYNCI_SEL Configures the selection of capture module sync input.

0: None

1: Timer0 sync_out

2: Timer1 sync_out

3: Timer2 sync_out

4: SYNC0 from GPIO matrix

5: SYNC1 from GPIO matrix

6: SYNC2 from GPIO matrix

7: None

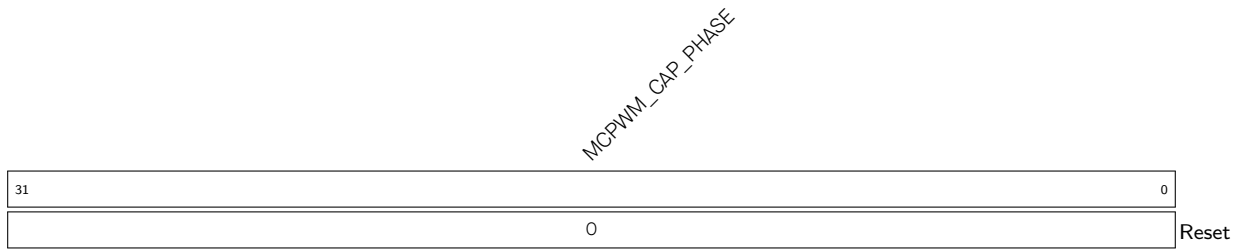
(R/W)

MCPWM_CAP_SYNC_SW When [MCPWM_CAP_SYNCI_EN](#) is set to 1, configures whether to trigger a capture timer sync so that capture timer is loaded with value in phase register.

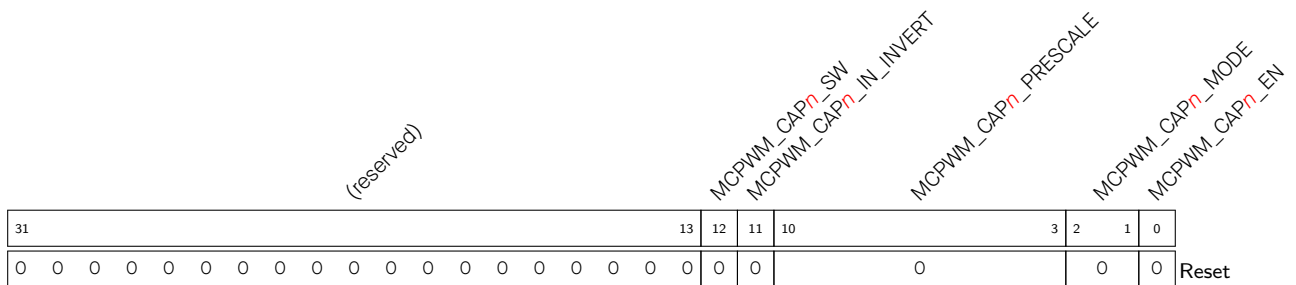
0: No effect

1: Trigger a capture timer sync

(WT)

Register 31.24. MCPWM_CAP_TIMER_PHASE_REG (0x00EC)

MCPWM_CAP_PHASE Configures phase value for capture timer sync operation.
(R/W)

Register 31.25. MCPWM_CAP_CH n _CFG_REG (n : 0-2) (0x00F0+0x4* n)

MCPWM_CAP n _EN Configures whether to enable capture on channel n .
0: Disable
1: Enable
(R/W)

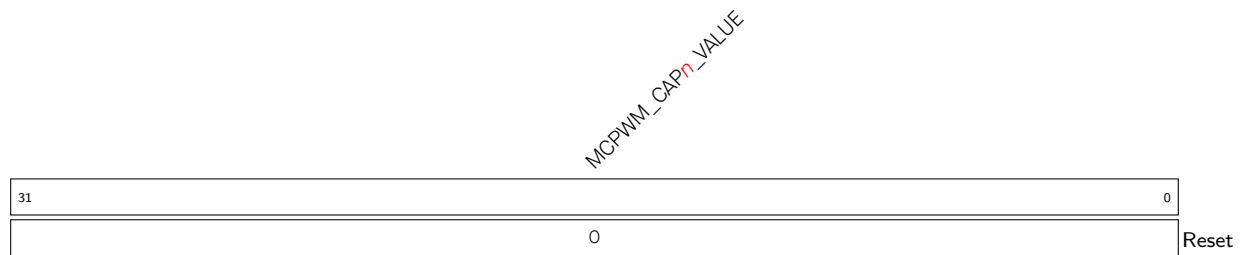
MCPWM_CAP n _MODE Configures the edge of capture on channel 0 after prescaling.
When bit0 is set to 1: enable capture on the falling edge.
When bit1 is set to 1: enable capture on the rising edge.
(R/W)

MCPWM_CAP n _PRESCALE Configures the prescale value on the rising edge of CAP0. Prescale value = PWM_CAP0_PRESCALE + 1. (R/W)

MCPWM_CAP n _IN_INVERT Configures whether to invert CAP n from GPIO matrix before prescale.
0: Normal
1: Invert
(R/W)

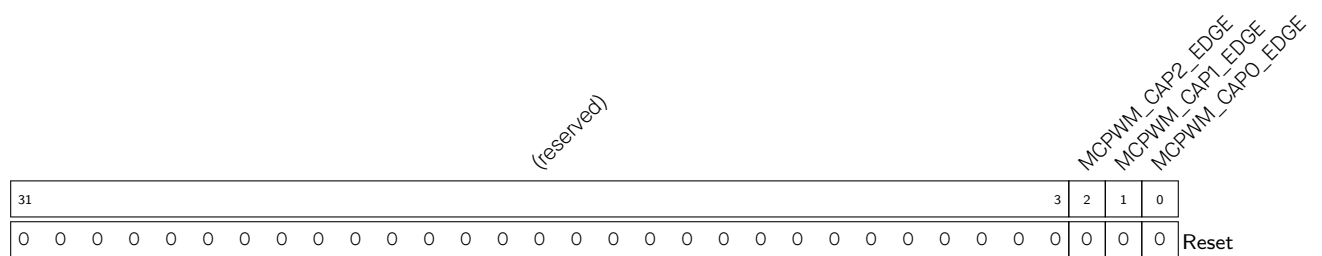
MCPWM_CAP n _SW Configures whether to trigger a software-forced capture on channel 0.
0: Not trigger
1: Trigger
(WT)

Register 31.26. MCPWM_CAP_CH n _REG (n : 0-2) (0x00FC+0x4* n)



MCPWM_CAP n _VALUE Represents value of last capture on CAP n . (RO)

Register 31.27. MCPWM_CAP_STATUS_REG (0x0108)



MCPWM_CAP n _EDGE Represents the edge of the last capture trigger on channel n .

0: Rising edge

1: Falling edge

(RO)

Register 31.28. MCPWM_UPDATE_CFG_REG (0x010C)

(reserved)																								MCPWM_OP2_FORCE_UP MCPWM_OP2_UP_EN MCPWM_OP1_FORCE_UP MCPWM_OP1_UP_EN MCPWM_OPO_FORCE_UP MCPWM_OPO_UP_EN MCPWM_GLOBAL_FORCE_UP MCPWM_GLOBAL_UP_EN															
31																								8	7	6	5	4	3	2	1	0							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1					
																																Reset							

MCPWM_GLOBAL_UP_EN Configures whether to globally update all active registers.

0: No effect

1: Update all active registers globally

(R/W)

MCPWM_GLOBAL_FORCE_UP Configures whether to trigger a forced update of all active registers globally.

0: No effect

1: Trigger a forced update

(R/W)

MCPWM_OPO_UP_EN Configures whether to update active registers in PWM operator 0 when [MCPWM_GLOBAL_UP_EN](#) is set to 1.

0: No effect

1: Update active registers in PWM operator 0

(R/W)

MCPWM_OPO_FORCE_UP Configures whether to trigger a forced update of active registers in PWM operator 0.

0: No effect

1: Trigger a forced update

(R/W)

MCPWM_OP1_UP_EN Configures whether to update active registers in PWM operator 1 when **MCPWM_GLOBAL_UP_EN** is set to 1.

0: No effect

1: Update active registers in PWM operator 1

(R/W)

Continued on the next page...

Register 31.28. MCPWM_UPDATE_CFG_REG (0x010C)

Continued from the previous page...

MCPWM_OP1_FORCE_UP Configures whether to trigger a forced update of active registers in PWM operator 1.

0: No effect

1: Trigger a forced update

(R/W)

MCPWM_OP2_UP_EN Configures whether to update active registers in PWM operator 2 when [MCPWM_GLOBAL_UP_EN](#) is set to 1.

0: No effect

1: Update active registers in PWM operator 2

(R/W)

MCPWM_OP2_FORCE_UP Configures whether to trigger a forced update of active registers in PWM operator 2.

0: No effect

1: Trigger a forced update

(R/W)

Register 31.29. MCPWM_INT_ENA_REG (0x0110)

(reserved) MCPWM_CAP2_INT_ENA MCPWM_CAP1_INT_ENA MCPWM_CAP0_INT_ENA MCPWM_TZ2_INT_ENA MCPWM_TZ1_OST_INT_ENA MCPWM_TZ0_OST_INT_ENA MCPWM_TZ1_OST_INT_ENA MCPWM_TZ0_OST_INT_ENA MCPWM_TZ1_CBC_INT_ENA MCPWM_TZ0_CBC_INT_ENA MCPWM_CMPR2_INT_ENA MCPWM_CMPR1_TEB_INT_ENA MCPWM_CMPR0_TEB_INT_ENA MCPWM_CMPR2_TEA_INT_ENA MCPWM_CMPR1_TEA_INT_ENA MCPWM_FAULT2_CLR_INT_ENA MCPWM_FAULT1_CLR_INT_ENA MCPWM_FAULT0_CLR_INT_ENA MCPWM_FAULT2_INT_ENA MCPWM_FAULT1_INT_ENA MCPWM_FAULT0_INT_ENA MCPWM_TIMER2_TEP_INT_ENA MCPWM_TIMER1_TEP_INT_ENA MCPWM_TIMER0_TEP_INT_ENA MCPWM_TIMER2_TEZ_INT_ENA MCPWM_TIMER1_TEZ_INT_ENA MCPWM_TIMER0_TEZ_INT_ENA MCPWM_TIMER2_STOP_INT_ENA MCPWM_TIMER1_STOP_INT_ENA MCPWM_TIMER0_STOP_INT_ENA																																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

MCPWM_TIMER0_STOP_INT_ENA Write 1 to enable [MCPWM_TIMER0_STOP_INT](#). (R/W)

MCPWM_TIMER1_STOP_INT_ENA Write 1 to enable [MCPWM_TIMER1_STOP_INT](#). (R/W)

MCPWM_TIMER2_STOP_INT_ENA Write 1 to enable [MCPWM_TIMER2_STOP_INT](#). (R/W)

MCPWM_TIMER0_TEZ_INT_ENA Write 1 to enable [MCPWM_TIMER0_TEZ_INT](#). (R/W)

MCPWM_TIMER1_TEZ_INT_ENA Write 1 to enable [MCPWM_TIMER1_TEZ_INT](#). (R/W)

MCPWM_TIMER2_TEZ_INT_ENA Write 1 to enable [MCPWM_TIMER2_TEZ_INT](#). (R/W)

MCPWM_TIMER0_TEP_INT_ENA Write 1 to enable [MCPWM_TIMER0_TEP_INT](#). (R/W)

MCPWM_TIMER1_TEP_INT_ENA Write 1 to enable [MCPWM_TIMER1_TEP_INT](#). (R/W)

MCPWM_TIMER2_TEP_INT_ENA Write 1 to enable [MCPWM_TIMER2_TEP_INT](#). (R/W)

MCPWM_FAULT0_INT_ENA Write 1 to enable [MCPWM_FAULT0_INT](#). (R/W)

MCPWM_FAULT1_INT_ENA Write 1 to enable [MCPWM_FAULT1_INT](#). (R/W)

MCPWM_FAULT2_INT_ENA Write 1 to enable [MCPWM_FAULT2_INT](#). (R/W)

MCPWM_FAULT0_CLR_INT_ENA Write 1 to enable [MCPWM_FAULT0_CLR_INT](#). (R/W)

MCPWM_FAULT1_CLR_INT_ENA Write 1 to enable [MCPWM_FAULT1_CLR_INT](#). (R/W)

MCPWM_FAULT2_CLR_INT_ENA Write 1 to enable [MCPWM_FAULT2_CLR_INT](#). (R/W)

MCPWM_CMPRO_TEA_INT_ENA Write 1 to enable [MCPWM_CMPRO_TEA_INT](#). (R/W)

MCPWM_CMPR1_TEA_INT_ENA Write 1 to enable [MCPWM_CMPR1_TEA_INT](#). (R/W)

MCPWM_CMPR2_TEA_INT_ENA Write 1 to enable [MCPWM_CMPR2_TEA_INT](#). (R/W)

MCPWM_CMPRO_TEB_INT_ENA Write 1 to enable [MCPWM_CMPRO_TEB_INT](#). (R/W)

MCPWM_CMPR1_TEB_INT_ENA Write 1 to enable [MCPWM_CMPR1_TEB_INT](#). (R/W)

MCPWM_CMPR2_TEB_INT_ENA Write 1 to enable [MCPWM_CMPR2_TEB_INT](#). (R/W)

MCPWM_TZO_CBC_INT_ENA Write 1 to enable [MCPWM_TZO_CBC_INT](#). (R/W)

Continued on the next page...

Register 31.29. MCPWM_INT_ENA_REG (0x0110)

Continued from the previous page...

MCPWM_TZ1_CBC_INT_ENA Write 1 to enable [MCPWM_TZ1_CBC_INT](#). (R/W)

MCPWM_TZ2_CBC_INT_ENA Write 1 to enable [MCPWM_TZ2_CBC_INT](#). (R/W)

MCPWM_TZ0_OST_INT_ENA Write 1 to enable [MCPWM_TZ0_OST_INT](#). (R/W)

MCPWM_TZ1_OST_INT_ENA Write 1 to enable [MCPWM_TZ1_OST_INT](#). (R/W)

MCPWM_TZ2_OST_INT_ENA Write 1 to enable [MCPWM_TZ2_OST_INT](#). (R/W)

MCPWM_CAPO_INT_ENA Write 1 to enable [MCPWM_CAPO_INT](#). (R/W)

MCPWM_CAP1_INT_ENA Write 1 to enable [MCPWM_CAP1_INT](#). (R/W)

MCPWM_CAP2_INT_ENA Write 1 to enable [MCPWM_CAP2_INT](#). (R/W)

Register 31.30. MCPWM_INT_RAW_REG (0x0114)

(reserved) MCPWM_CAP2_INT_RAW MCPWM_CAP1_INT_RAW MCPWM_CAP0_INT_RAW MCPWM_TZ2_OST_INT_RAW MCPWM_TZ1_OST_INT_RAW MCPWM_TZ0_OST_INT_RAW MCPWM_TZ2_CBC_INT_RAW MCPWM_TZ1_CBC_INT_RAW MCPWM_TZ0_CBC_INT_RAW MCPWM_CMPR2_INT_RAW MCPWM_CMPR1_TEB_INT_RAW MCPWM_CMPR0_TEB_INT_RAW MCPWM_CMPR2_TEA_INT_RAW MCPWM_CMPR1_TEA_INT_RAW MCPWM_FAULT2_CLR_INT_RAW MCPWM_FAULT1_CLR_INT_RAW MCPWM_FAULT0_CLR_INT_RAW MCPWM_TIMER2_TEP_INT_RAW MCPWM_TIMER1_TEP_INT_RAW MCPWM_TIMER0_TEP_INT_RAW MCPWM_TIMER2_TEZ_INT_RAW MCPWM_TIMER1_TEZ_INT_RAW MCPWM_TIMER0_TEZ_INT_RAW MCPWM_TIMER2_STOP_INT_RAW MCPWM_TIMER1_STOP_INT_RAW MCPWM_TIMER0_STOP_INT_RAW																																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

MCPWM_TIMER0_STOP_INT_RAW Represents the raw status of [MCPWM_TIMER0_STOP_INT](#).
(R/WTC/SS)

MCPWM_TIMER1_STOP_INT_RAW Represents the raw status of [MCPWM_TIMER1_STOP_INT](#).
(R/WTC/SS)

MCPWM_TIMER2_STOP_INT_RAW Represents the raw status of [MCPWM_TIMER2_STOP_INT](#).
(R/WTC/SS)

MCPWM_TIMER0_TEZ_INT_RAW Represents the raw status of [MCPWM_TIMER0_TEZ_INT](#).
(R/WTC/SS)

MCPWM_TIMER1_TEZ_INT_RAW Represents the raw status of [MCPWM_TIMER1_TEZ_INT](#).
(R/WTC/SS)

MCPWM_TIMER2_TEZ_INT_RAW Represents the raw status of [MCPWM_TIMER2_TEZ_INT](#).
(R/WTC/SS)

MCPWM_TIMER0_TEP_INT_RAW Represents the raw status of [MCPWM_TIMER0_TEP_INT](#).
(R/WTC/SS)

MCPWM_TIMER1_TEP_INT_RAW Represents the raw status of [MCPWM_TIMER1_TEP_INT](#).
(R/WTC/SS)

MCPWM_TIMER2_TEP_INT_RAW Represents the raw status of [MCPWM_TIMER2_TEP_INT](#).
(R/WTC/SS)

MCPWM_FAULT0_INT_RAW Represents the raw status of [MCPWM_FAULT0_INT](#). (R/WTC/SS)

MCPWM_FAULT1_INT_RAW Represents the raw status of [MCPWM_FAULT1_INT](#). (R/WTC/SS)

MCPWM_FAULT2_INT_RAW Represents the raw status of [MCPWM_FAULT2_INT](#). (R/WTC/SS)

MCPWM_FAULT0_CLR_INT_RAW Represents the raw status of [MCPWM_FAULT0_CLR_INT](#).
(R/WTC/SS)

MCPWM_FAULT1_CLR_INT_RAW Represents the raw status of [MCPWM_FAULT1_CLR_INT](#).
(R/WTC/SS)

Continued on the next page...

Register 31.30. MCPWM_INT_RAW_REG (0x0114)

Continued from the previous page...

MCPWM_FAULT2_CLR_INT_RAW Represents the raw status of [MCPWM_FAULT2_CLR_INT](#).
(R/WTC/SS)

MCPWM_CMPRO_TEA_INT_RAW Represents the raw status of [MCPWM_CMPRO_TEA_INT](#).
(R/WTC/SS)

MCPWM_CMPR1_TEA_INT_RAW Represents the raw status of [MCPWM_CMPR1_TEA_INT](#).
(R/WTC/SS)

MCPWM_CMPR2_TEA_INT_RAW Represents the raw status of [MCPWM_CMPR2_TEA_INT](#).
(R/WTC/SS)

MCPWM_CMPRO_TEB_INT_RAW Represents the raw status of [MCPWM_CMPRO_TEB_INT](#).
(R/WTC/SS)

MCPWM_CMPR1_TEB_INT_RAW Represents the raw status of [MCPWM_CMPR1_TEB_INT](#).
(R/WTC/SS)

MCPWM_CMPR2_TEB_INT_RAW Represents the raw status of [MCPWM_CMPR2_TEB_INT](#).
(R/WTC/SS)

MCPWM_TZO_CBC_INT_RAW Represents the raw status of [MCPWM_TZO_CBC_INT](#). (R/WTC/SS)

MCPWM_TZ1_CBC_INT_RAW Represents the raw status of [MCPWM_TZ1_CBC_INT](#). (R/WTC/SS)

MCPWM_TZ2_CBC_INT_RAW Represents the raw status of [MCPWM_TZ2_CBC_INT](#). (R/WTC/SS)

MCPWM_TZO_OST_INT_RAW Represents the raw status of [MCPWM_TZO_OST_INT](#). (R/WTC/SS)

MCPWM_TZ1_OST_INT_RAW Represents the raw status of [MCPWM_TZ1_OST_INT](#). (R/WTC/SS)

MCPWM_TZ2_OST_INT_RAW Represents the raw status of [MCPWM_TZ2_OST_INT](#). (R/WTC/SS)

MCPWM_CAPO_INT_RAW Represents the raw status of [MCPWM_CAPO_INT](#). (R/WTC/SS)

MCPWM_CAP1_INT_RAW Represents the raw status of [MCPWM_CAP1_INT](#). (R/WTC/SS)

MCPWM_CAP2_INT_RAW Represents the raw status of [MCPWM_CAP2_INT](#). (R/WTC/SS)

Register 31.31. MCPWM_INT_ST_REG (0x0118)

(reserved) MCPWM_CAP2_INT_ST MCPWM_CAP1_INT_ST MCPWM_CAO_INT_ST MCPWM_TZ2_OST_INT_ST MCPWM_TZ1_OST_INT_ST MCPWM_TZ0_OST_INT_ST MCPWM_TZ2_CBC_INT_ST MCPWM_TZ1_CBC_INT_ST MCPWM_TZ0_CBC_INT_ST MCPWM_CMPR2_TEB_INT_ST MCPWM_CMPR1_TEB_INT_ST MCPWM_CMPR2_TEA_INT_ST MCPWM_CMPR1_TEA_INT_ST MCPWM_FAULT2_CLR_INT_ST MCPWM_FAULT1_CLR_INT_ST MCPWM_FAULT2_INT_ST MCPWM_FAULT1_INT_ST MCPWM_TIMER2_TEP_INT_ST MCPWM_TIMER1_TEP_INT_ST MCPWM_TIMER2_TEZ_INT_ST MCPWM_TIMER1_TEZ_INT_ST MCPWM_TIMER2_STOP_INT_ST MCPWM_TIMER1_STOP_INT_ST MCPWM_TIMER0_STOP_INT_ST																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

MCPWM_TIMER0_STOP_INT_ST Represents the masked status of [MCPWM_TIMER0_STOP_INT](#). (RO)

MCPWM_TIMER1_STOP_INT_ST Represents the masked status of [MCPWM_TIMER1_STOP_INT](#). (RO)

MCPWM_TIMER2_STOP_INT_ST Represents the masked status of [MCPWM_TIMER2_STOP_INT](#). (RO)

MCPWM_TIMER0_TZ_INT_ST Represents the masked status of [MCPWM_TIMER0_TZ_INT](#). (RO)

MCPWM_TIMER1_TZ_INT_ST Represents the masked status of [MCPWM_TIMER1_TZ_INT](#). (RO)

MCPWM_TIMER2_TZ_INT_ST Represents the masked status of [MCPWM_TIMER2_TZ_INT](#). (RO)

MCPWM_TIMER0_TEP_INT_ST Represents the masked status of [MCPWM_TIMER0_TEP_INT](#). (RO)

MCPWM_TIMER1_TEP_INT_ST Represents the masked status of [MCPWM_TIMER1_TEP_INT](#). (RO)

MCPWM_TIMER2_TEP_INT_ST Represents the masked status of [MCPWM_TIMER2_TEP_INT](#). (RO)

MCPWM_FAULT0_INT_ST Represents the masked status of [MCPWM_FAULT0_INT](#). (RO)

MCPWM_FAULT1_INT_ST Represents the masked status of [MCPWM_FAULT1_INT](#). (RO)

MCPWM_FAULT2_INT_ST Represents the masked status of [MCPWM_FAULT2_INT](#). (RO)

MCPWM_FAULT0_CLR_INT_ST Represents the masked status of [MCPWM_FAULT0_CLR_INT](#). (RO)

MCPWM_FAULT1_CLR_INT_ST Represents the masked status of [MCPWM_FAULT1_CLR_INT](#). (RO)

Continued on the next page...

Register 31.31. MCPWM_INT_ST_REG (0x0118)

Continued from the previous page...

MCPWM_FAULT2_CLR_INT_ST Represents the masked status of [MCPWM_FAULT2_CLR_INT](#). (RO)

MCPWM_CMPRO_TEA_INT_ST Represents the masked status of [MCPWM_CMPRO_TEA_INT](#). (RO)

MCPWM_CMPR1_TEA_INT_ST Represents the masked status of [MCPWM_CMPR1_TEA_INT](#). (RO)

MCPWM_CMPR2_TEA_INT_ST Represents the masked status of [MCPWM_CMPR2_TEA_INT](#). (RO)

MCPWM_CMPRO_TEB_INT_ST Represents the masked status of [MCPWM_CMPRO_TEB_INT](#). (RO)

MCPWM_CMPR1_TEB_INT_ST Represents the masked status of [MCPWM_CMPR1_TEB_INT](#). (RO)

MCPWM_CMPR2_TEB_INT_ST Represents the masked status of [MCPWM_CMPR2_TEB_INT](#). (RO)

MCPWM_TZO_CBC_INT_ST Represents the masked status of [MCPWM_TZO_CBC_INT_ST](#). (RO)

MCPWM_TZ1_CBC_INT_ST Represents the masked status of [MCPWM_TZ1_CBC_INT_ST](#). (RO)

MCPWM_TZ2_CBC_INT_ST Represents the masked status of [MCPWM_TZ2_CBC_INT_ST](#). (RO)

MCPWM_TZO_OST_INT_ST Represents the masked status of [MCPWM_TZO_OST_INT](#). (RO)

MCPWM_TZ1_OST_INT_ST Represents the masked status of [MCPWM_TZ1_OST_INT](#). (RO)

MCPWM_TZ2_OST_INT_ST Represents the masked status of [MCPWM_TZ2_OST_INT](#). (RO)

MCPWM_CAPO_INT_ST Represents the masked status of [MCPWM_CAPO_INT](#). (RO)

MCPWM_CAP1_INT_ST Represents the masked status of [MCPWM_CAP1_INT](#). (RO)

MCPWM_CAP2_INT_ST Represents the masked status of [MCPWM_CAP2_INT](#). (RO)

Register 31.32. MCPWM_INT_CLR_REG (0x011C)

(reserved) MCPWM_CAP2_INT_CLR MCPWM_CAP1_INT_CLR MCPWM_CAPO_INT_CLR MCPWM_TZ2_OST_INT_CLR MCPWM_TZ1_OST_INT_CLR MCPWM_TZO_OST_INT_CLR MCPWM_TZ2_CBC_INT_CLR MCPWM_TZ1_CBC_INT_CLR MCPWM_TZO_CBC_INT_CLR MCPWM_CMPR2_TEB_INT_CLR MCPWM_CMPR1_TEB_INT_CLR MCPWM_CMPRO_TEB_INT_CLR MCPWM_CMPR2_TEA_INT_CLR MCPWM_CMPR1_TEA_INT_CLR MCPWM_CMPRO_TEA_INT_CLR MCPWM_FAULT2_CLR_INT_CLR MCPWM_FAULT1_CLR_INT_CLR MCPWM_FAULTO_CLR_INT_CLR MCPWM_TIMER2_TEP_INT_CLR MCPWM_TIMER1_TEP_INT_CLR MCPWM_TIMER2_TEZ_INT_CLR MCPWM_TIMER1_TEZ_INT_CLR MCPWM_TIMER2_STOP_INT_CLR MCPWM_TIMER1_STOP_INT_CLR MCPWM_TIMERO_STOP_INT_CLR																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

MCPWM_TIMERO_STOP_INT_CLR Write 1 to clear [MCPWM_TIMERO_STOP_INT](#). (WT)

MCPWM_TIMER1_STOP_INT_CLR Write 1 to clear [MCPWM_TIMER1_STOP_INT](#). (WT)

MCPWM_TIMER2_STOP_INT_CLR Write 1 to clear [MCPWM_TIMER2_STOP_INT](#). (WT)

MCPWM_TIMERO_TEZ_INT_CLR Write 1 to clear [MCPWM_TIMERO_TEZ_INT](#). (WT)

MCPWM_TIMER1_TEZ_INT_CLR Write 1 to clear [MCPWM_TIMER1_TEZ_INT](#). (WT)

MCPWM_TIMER2_TEZ_INT_CLR Write 1 to clear [MCPWM_TIMER2_TEZ_INT](#). (WT)

MCPWM_TIMERO_TEP_INT_CLR Write 1 to clear [MCPWM_TIMERO_TEP_INT](#). (WT)

MCPWM_TIMER1_TEP_INT_CLR Write 1 to clear [MCPWM_TIMER1_TEP_INT](#). (WT)

MCPWM_TIMER2_TEP_INT_CLR Write 1 to clear [MCPWM_TIMER2_TEP_INT](#). (WT)

MCPWM_FAULTO_INT_CLR Write 1 to clear [MCPWM_FAULTO_INT](#). (WT)

MCPWM_FAULT1_INT_CLR Write 1 to clear [MCPWM_FAULT1_INT](#). (WT)

MCPWM_FAULT2_INT_CLR Write 1 to clear [MCPWM_FAULT2_INT](#). (WT)

MCPWM_FAULTO_CLR_INT_CLR Write 1 to clear [MCPWM_FAULTO_CLR_INT](#). (WT)

MCPWM_FAULT1_CLR_INT_CLR Write 1 to clear [MCPWM_FAULT1_CLR_INT](#). (WT)

MCPWM_FAULT2_CLR_INT_CLR Write 1 to clear [MCPWM_FAULT2_CLR_INT](#). (WT)

MCPWM_CMPRO_TEA_INT_CLR Write 1 to clear [MCPWM_CMPRO_TEA_INT](#). (WT)

MCPWM_CMPR1_TEA_INT_CLR Write 1 to clear [MCPWM_CMPR1_TEA_INT](#). (WT)

MCPWM_CMPR2_TEA_INT_CLR Write 1 to clear [MCPWM_CMPR2_TEA_INT](#). (WT)

MCPWM_CMPRO_TEB_INT_CLR Write 1 to clear [MCPWM_CMPRO_TEB_INT](#). (WT)

MCPWM_CMPR1_TEB_INT_CLR Write 1 to clear [MCPWM_CMPR1_TEB_INT](#). (WT)

MCPWM_CMPR2_TEB_INT_CLR Write 1 to clear [MCPWM_CMPR2_TEB_INT](#). (WT)

MCPWM_TZO_CBC_INT_CLR Write 1 to clear [MCPWM_TZO_CBC_INT](#). (WT)

Continued on the next page...

Register 31.32. MCPWM_INT_CLR_REG (0x011C)

Continued from the previous page...

MCPWM_TZ1_CBC_INT_CLR Write 1 to clear [MCPWM_TZ1_CBC_INT](#). (WT)

MCPWM_TZ2_CBC_INT_CLR Write 1 to clear [MCPWM_TZ2_CBC_INT](#). (WT)

MCPWM_TZ0_OST_INT_CLR Write 1 to clear [MCPWM_TZ0_OST_INT](#). (WT)

MCPWM_TZ1_OST_INT_CLR Write 1 to clear [MCPWM_TZ1_OST_INT](#). (WT)

MCPWM_TZ2_OST_INT_CLR Write 1 to clear [MCPWM_TZ2_OST_INT](#). (WT)

MCPWM_CAPO_INT_CLR Write 1 to clear [MCPWM_CAPO_INT](#). (WT)

MCPWM_CAP1_INT_CLR Write 1 to clear [MCPWM_CAP1_INT](#). (WT)

MCPWM_CAP2_INT_CLR Write 1 to clear [MCPWM_CAP2_INT](#). (WT)

Register 31.33. MCPWM_EVT_EN_REG (0x0120)

(reserved)																																MCPWM_EVT_CAP2_EN																																MCPWM_EVT_CAP1_EN																																MCPWM_EVT_CAP0_EN																																MCPWM_EVT_TZ2_EN																																MCPWM_EVT_TZ1_EN																																MCPWM_EVT_TZ0_EN																																MCPWM_EVT_TZ2_OST_EN																																MCPWM_EVT_TZ1_OST_EN																																MCPWM_EVT_TZ0_OST_EN																																MCPWM_EVT_TZ1_CBC_EN																																MCPWM_EVT_TZ0_CBC_EN																																MCPWM_EVT_F2_CLR_EN																																MCPWM_EVT_F1_CLR_EN																																MCPWM_EVT_F0_CLR_EN																																MCPWM_EVT_F2_EN																																MCPWM_EVT_F1_EN																																MCPWM_EVT_F0_EN																																MCPWM_EVT_OP2_TEB_EN																																MCPWM_EVT_OP1_TEB_EN																																MCPWM_EVT_OP0_TEB_EN																																MCPWM_EVT_OP2_TEA_EN																																MCPWM_EVT_OP1_TEA_EN																																MCPWM_EVT_OP0_TEA_EN																																MCPWM_EVT_TIMER2_TEP_EN																																MCPWM_EVT_TIMER1_TEP_EN																																MCPWM_EVT_TIMER0_TEP_EN																																MCPWM_EVT_TIMER2_TEZ_EN																																MCPWM_EVT_TIMER1_TEZ_EN																																MCPWM_EVT_TIMER0_TEZ_EN																																MCPWM_EVT_TIMER2_STOP_EN																																MCPWM_EVT_TIMER1_STOP_EN																																MCPWM_EVT_TIMER0_STOP_EN																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</			

MCPWM_EVT_TIMER0_STOP_EN Configures whether to enable timer0 stop event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER1_STOP_EN Configures whether to enable timer1 stop event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER2_STOP_EN Configures whether to enable timer2 stop event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER0_TEZ_EN Configures whether to enable timer0 equal zero event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER1_TEZ_EN Configures whether to enable timer1 equal zero event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER2_TEZ_EN Configures whether to enable timer2 equal zero event generation.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 31.33. MCPWM_EVT_EN_REG (0x0120)

Continued from the previous page...

MCPWM_EVT_TIMER0_TEP_EN Configures whether to enable timer0 equal period event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER1_TEP_EN Configures whether to enable timer1 equal period event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TIMER2_TEP_EN Configures whether to enable timer2 equal period event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_OPO_TEA_EN Configures whether to enable PWM generator0 timer equal A event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_OP1_TEA_EN Configures whether to enable PWM generator1 timer equal A event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_OP2_TEA_EN Configures whether to enable PWM generator2 timer equal A event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_OPO_TEB_EN Configures whether to enable PWM generator0 timer equal B event generation.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 31.33. MCPWM_EVT_EN_REG (0x0120)

Continued from the previous page...

MCPWM_EVT_OP1_TEB_EN Configures whether to enable PWM generator1 timer equal B event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_OP2_TEB_EN Configures whether to enable PWM generator2 timer equal B event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F0_EN Configures whether to enable FAULT0 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F1_EN Configures whether to enable FAULT1 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F2_EN Configures whether to enable FAULT2 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F0_CLR_EN Configures whether to enable FAULT0 clear event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F1_CLR_EN Configures whether to enable FAULT1 clear event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_F2_CLR_EN Configures whether to enable FAULT2 clear event generation.

0: Disable

1: Enable

(R/W)

Continued on the next page...

Register 31.33. MCPWM_EVT_EN_REG (0x0120)

Continued from the previous page...

MCPWM_EVT_TZ0_CBC_EN Configures whether to enable cycle by cycle trip0 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TZ1_CBC_EN Configures whether to enable cycle by cycle trip1 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TZ2_CBC_EN Configures whether to enable cycle by cycle trip2 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TZ0_OST_EN Configures whether to enable one shot trip0 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TZ1_OST_EN Configures whether to enable one shot trip1 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_TZ2_OST_EN Configures whether to enable one shot trip2 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_CAPO_EN Configures whether to enable capture0 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_CAP1_EN Configures whether to enable capture1 event generation.

0: Disable

1: Enable

(R/W)

MCPWM_EVT_CAP2_EN Configures whether to enable capture2 event generation.

0: Disable

1: Enable

(R/W)

Register 31.34. MCPWM_EVT_EN2_REG (0x0128)

(reserved)

31	6	5	4	3	2	1	0
Reset							

MCPWM_EVT_OP2_TEE2_EN
MCPWM_EVT_OP1_TEE2_EN
MCPWM_EVT_OP2_TEE2_EN
MCPWM_EVT_OP1_TEE2_EN
MCPWM_EVT_OP2_TEE1_EN
MCPWM_EVT_OP1_TEE1_EN
MCPWM_EVT_OP0_TEE1_EN

MCPWM_EVT_OPn_TEE1_EN Configures whether to generate the MCPWM_EVT_OPn_TEE1 event when the PWM generatorn timer equals OPn_TSTMP_E1_REG.

0: Not generate

1: Generate

(R/W)

MCPWM_EVT_OPn_TEE2_EN Configures whether to generate the MCPWM_EVT_OPn_TEE2 event when the PWM generatorn timer equals OPn_TSTMP_E2_REG.

0: Not generate

1: Generate

(R/W)

Register 31.35. MCPWM_TASK_EN_REG (0x0124)

(reserved)																																MCPWM_TASK_CAP2_EN MCPWM_TASK_CAP1_EN MCPWM_TASK_CAP0_EN MCPWM_TASK_CLR2_EN MCPWM_TASK_CLR1_EN MCPWM_TASK_CLR0_EN MCPWM_TASK_TZ2_EN MCPWM_TASK_TZ1_EN MCPWM_TASK_TZ0_EN MCPWM_TASK_TIMER2_EN MCPWM_TASK_TIMER1_EN MCPWM_TASK_TIMER0_EN MCPWM_TASK_SYNC2_EN MCPWM_TASK_SYNC1_EN MCPWM_TASK_SYNC0_EN MCPWM_TASK_STOP_EN MCPWM_TASK_CMPR2_B_UP_EN MCPWM_TASK_CMPR1_B_UP_EN MCPWM_TASK_CMPR2_A_UP_EN MCPWM_TASK_CMPR1_A_UP_EN MCPWM_TASK_CMPRO_A_UP_EN MCPWM_TASK_CMPRO_B_UP_EN																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																				22										21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0										0										0										0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCPWM_TASK_CMPRO_A_UP_EN Configures whether to receive update task of PWM generator0 timer stamp A's shadow register.
 0: No effect
 1: Receive
 (R/W)

MCPWM_TASK_CMPR1_A_UP_EN Configures whether to receive update task of PWM generator1 timer stamp A's shadow register.
 0: No effect
 1: Receive
 (R/W)

MCPWM_TASK_CMPR2_A_UP_EN Configures whether to receive update task of PWM generator2 timer stamp A's shadow register.
 0: No effect
 1: Receive
 (R/W)

MCPWM_TASK_CMPRO_B_UP_EN Configures whether to receive update task of PWM generator0 timer stamp B's shadow register.
 0: No effect
 1: Receive
 (R/W)

MCPWM_TASK_CMPR1_B_UP_EN Configures whether to receive update task of PWM generator1 timer stamp B's shadow register.
 0: No effect
 1: Receive
 (R/W)

MCPWM_TASK_CMPR2_B_UP_EN Configures whether to receive update task of PWM generator2 timer stamp B's shadow register.
 0: No effect
 1: Receive
 (R/W)

Continued on the next page...

Register 31.35. MCPWM_TASK_EN_REG (0x0124)

Continued from the previous page...

MCPWM_TASK_GEN_STOP_EN Configures whether to receive all PWM generate stop task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER0_SYNC_EN Configures whether to receive timer0 sync task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER1_SYNC_EN Configures whether to receive timer1 sync task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER2_SYNC_EN Configures whether to receive timer2 sync task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER0_PERIOD_UP_EN Configures whether to receive timer0 period update task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER1_PERIOD_UP_EN Configures whether to receive timer1 period update task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TIMER2_PERIOD_UP_EN Configures whether to receive timer2 period update task.

0: No effect

1: Receive

(R/W)

Continued on the next page...

Register 31.35. MCPWM_TASK_EN_REG (0x0124)

Continued from the previous page...

MCPWM_TASK_TZ0_OST_EN Configures whether to receive one shot trip0 task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TZ1_OST_EN Configures whether to receive one shot trip1 task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_TZ2_OST_EN Configures whether to receive one shot trip2 task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CLRO_OST_EN Configures whether to receive one shot trip0 clear task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CLR1_OST_EN Configures whether to receive one shot trip1 clear task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CLR2_OST_EN Configures whether to receive one shot trip2 clear task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CAPO_EN Configures whether to receive capture0 task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CAP1_EN Configures whether to receive capture1 task.

0: No effect

1: Receive

(R/W)

MCPWM_TASK_CAP2_EN Configures whether to receive capture2 task.

0: No effect

1: Receive

(R/W)

Register 31.36. MCPWM_OP n _TSTMP_E1_REG (n : 0-2) (0x012C+0x8* n)

(reserved)																MCPWM_OP n _TSTMP_E1															
31																16	15														0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													Reset

MCPWM_OP n _TSTMP_E1 Configures the time stamp E1 value of the generator n . (R/W)

Register 31.37. MCPWM_OP n _TSTMP_E2_REG(n : 0-2) (0x0130+0x8* n)

(reserved)																MCPWM_OP n _TSTMP_E2															
31																16	15														0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													Reset

MCPWM_OP n _TSTMP_E2 Configures the time stamp E2 value of the generator n . (R/W)

Register 31.38. MCPWM_CLK_REG (0x0144)

(reserved)																														MCPWM_CLK_EN	
31																														1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

MCPWM_CLK_EN Configures whether to force open register clock gate.

0: Open the clock gate only when application writes registers

1: Force open the clock gate for register

(R/W)

Register 31.39. MCPWM_VERSION_REG (0x0148)

(reserved)				MCPWM_DATE																										
31	28	27	0																											
0	0	0	0	0x2212290																										Reset

MCPWM_DATE Version control register. (R/W)

Chapter 32

Remote Control Peripheral (RMT)

32.1 Overview

The Remote Control (RMT) module is designed to transmit and receive infrared remote control signals. A variety of remote control protocols can be encoded/decoded via software based on the RMT module. The RMT module converts pulse codes stored in the module's built-in RAM into output signals, or converts input signals into pulse codes and stores them in RAM. In addition, the RMT module optionally modulates its output signals with a carrier wave, or optionally demodulates and filters its input signals.

The RMT module has four channels, numbered from zero to three. Each channel is able to independently transmit or receive signals.

- Channels 0 ~ 1 (TX channel) are dedicated to transmitting signals;
- Channels 2 ~ 3 (RX channel) are dedicated to receiving signals.

Each TX/RX channel has the same functionality controlled by a dedicated set of registers and is able to independently transmit or receive data. TX channels are indicated by *n* which is used as a placeholder for the channel number, and by *m* for RX channels.

32.2 Features

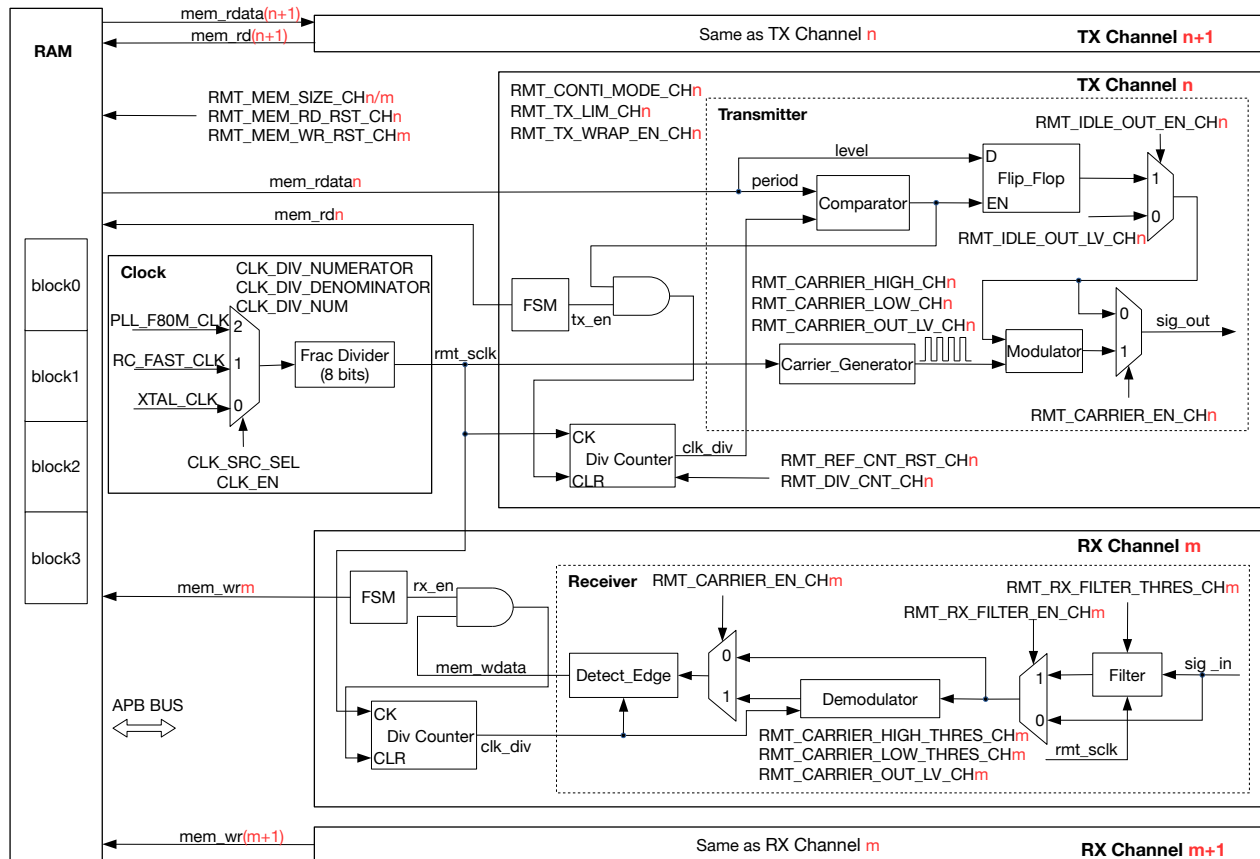
The RMT module has the following features:

- Four channels:
 - TX channels 0 ~ 1
 - RX channels 2 ~ 3
 - Four channels share a 192 x 32-bit RAM
- The transmitter supports:
 - Normal TX mode
 - Wrap TX mode
 - Modulation on TX pulses
 - Continuous TX mode
 - Multiple channels (programmable) transmitting data simultaneously
- The receiver supports:
 - Normal RX mode

- Wrap RX mode
- RX filtering
- Demodulation on RX pulses

32.3 Functional Description

32.3.1 RMT Architecture



Note: In the figure, $n = 0$ and $m = 2$.

Figure 32.3-1. RMT Architecture

As shown in Figure 32.3-1, each TX channel has:

- Clock divider counter (Div Counter)
- State machine (FSM)
- Transmitter

Each RX channel also has:

- Clock divider counter (Div Counter)
- State machine (FSM)
- Receiver

The four channels share a 192 x 32-bit RAM.

In the “Clock” frame, the clock-related registers that need to be configured for RMT are specified below:

- CLK_EN: [PCR_RMT_SCLK_EN](#) to enable the rmt_sclk clock
- CLK_SRC_SEL: [PCR_RMT_SCLK_SEL](#) to select the rmt_sclk clock
- CLK_DIV_NUM: [PCR_RMT_SCLK_DIV_NUM](#) to configure the integral part of the divisor for the rmt_sclk clock
- CLK_DIV_NUMERATOR: [PCR_RMT_SCLK_DIV_A](#) to configure the denominator of the divisor’s fractional part for the rmt_sclk clock
- CLK_DIV_DENOMINATOR: [PCR_RMT_SCLK_DIV_B](#) to configure the numerator of the divisor’s fractional part for the rmt_sclk clock

32.3.2 RAM

32.3.2.1 Structure of RAM

Figure 32.3-2 shows the format of pulse code in RAM. Each pulse code contains a 16-bit entry with two fields: “level” and “period”. “level” (0 or 1) indicates a low-/high-level value that has been received or is going to be sent, while “period” points out the number of clock cycles (see clk_div in Figure 32.3-1) that the level lasts for.

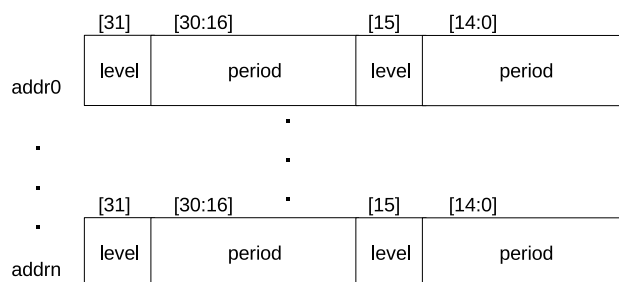


Figure 32.3-2. Format of Pulse Code in RAM

The minimum value for the period is zero (0) and is interpreted as a transmission end-marker. For a non-zero period (i.e., not an end-marker), its value is limited by APB clock and RMT clock according to the formula below:

$$3 \times T_{apb\_clk} + 5 \times T_{rmt\_sclk} < period \times T_{clk\_div} \quad (1)$$

32.3.2.2 Use of RAM

The RAM is divided into four 48 x 32-bit blocks. By default, each channel uses one block (block 0 for channel 0, block 1 for channel 1, and so on).

If the data size of one single transfer is larger than the block size of TX channel *n* or RX channel *m*, users can configure the channel:

- to enable [wrap mode](#) by setting [RMT_MEM_TX/RX_WRAP_EN_CHn/m](#);

- or to use more blocks by configuring `RMT_MEM_SIZE_CH $n/m$` .

Setting `RMT_MEM_SIZE_CH $n/m$` > 1 allows channel n/m to use the memory of the subsequent channels, i.e., block $(n/m) \sim \text{block } (n/m + \text{RMT\_MEM\_SIZE\_CH}_{n/m} - 1)$. In such case, the subsequent channels $n/m+1 \sim n/m + \text{RMT\_MEM\_SIZE\_CH}_{n/m} - 1$ can not be used since their RAM blocks are occupied. For example, if channel 0 is configured to use block 0 and block 1, then channel 1 will be unavailable since its block is occupied, while channel 2 and channel 3 are not affected and can be used normally.

Note that the RAM used by each channel is mapped from low address to high address. In such mode, channel 0 is able to use the RAM blocks of channels 1, 2, and 3 by setting `RMT_MEM_SIZE_CHO`, but channel 3 can not use the blocks of channels 0, 1, or 2. Therefore, the maximum value of `RMT_MEM_SIZE_CH $n$` should not exceed $(4-n)$ and the value of `RMT_MEM_SIZE_CH $m$` should not exceed $(2-m)$.

The RMT RAM can be accessed via APB bus, or read by the transmitter and written by the receiver. To avoid any possible access conflict between the receiver writing RAM and the APB bus reading RAM, RMT can be configured to designate the RAM block's owner, be it the receiver or the APB bus, by configuring `RMT_MEM_OWNER_CH $m$` . If this ownership is violated, a flag signal `RMT_MEM_OWNER_ERR_CH $m$` will be generated.

32.3.2.3 RAM Access

APB bus is able to access RAM in FIFO mode and in NONFIFO (Direct Address) mode, depending on the configuration of `RMT_APB_FIFO_MASK`:

- 0: use FIFO mode;
- 1: use NONFIFO mode.

FIFO Mode

In FIFO mode, the APB reads data from or writes data to RAM via a fixed address stored in `RMT_CH $n/m$ DATA_REG`.

NONFIFO Mode

In NONFIFO mode, the APB writes data to or reads data from a continuous address range.

- The write-starting address of TX channel n is (RMT base address + 0x400 + $(n-1) \times 48$). The access address for the second data and the following data are (RMT base address + 0x400 + $(n-1) \times 48 + 0x4$), and so on, incremented by 0x4.
- The read-starting address of RX channel m is (RMT base address + 0x460 + $(m-1) \times 48$). The access address for the second data and the following data are (RMT base address + 0x460 + $(m-1) \times 48 + 0x4$), and so on, incremented by 0x4.

32.3.3 Clock

The clock source of RMT can be `PLL_F80M_CLK`, `RC_FAST_CLK`, or `XTAL_CLK`, depending on the configuration of `PCR_RMT_SCLK_SEL`. RMT clock can be enabled by setting `PCR_RMT_SCLK_EN`. RMT working clock (see `rmt_sclk` in Figure 32.3-1) is obtained by dividing the selected clock source with a fractional divider. The divider is `PCR_RMT_SCLK_DIV_NUM + 1 + PCR_RMT_SCLK_DIV_A / PCR_RMT_SCLK_DIV_B`

For more information, see Chapter 6 *Reset and Clock*. `RMT_DIV_CNT_CH $n$ /m` is used to configure the divider coefficient of internal clock divider for RMT channels. The coefficient is normally equal to the value of `RMT_DIV_CNT_CH $n$ /m`, except for value 0 that represents divider 256. The clock divider can be reset by setting `RMT_REF_CNT_RST_CH $n$ /m`. The clock generated from the divider can be used by the counter (see Figure 32.3-1).

32.3.4 Transmitter

Note:

Updating the configuration described in this and subsequent sections requires to set `RMT_CONF_UPDATE_CH $n$ /m` first. See Section 32.4.

32.3.4.1 Normal TX Mode

When `RMT_TX_START_CH $n$` is set, the transmitter of channel n starts reading and transmitting pulse codes from the starting address of its RAM block. The codes are transmitted starting from low-address entry. When an end-marker (a zero period) is encountered, the transmitter stops the transmission, returns to idle state, and generates an `RMT_CH $n$ _TX_END_INT` interrupt. Setting `RMT_TX_STOP_CH $n$` to 1 also stops the transmission and immediately sets the transmitter back to idle. The output level of a transmitter in idle state is determined by the “level” field of the end-marker or by the content of `RMT_IDLE_OUT_LV_CH $n$` , depending on the configuration of `RMT_IDLE_OUT_EN_CH $n$` :

- 0: the level in idle state is determined by the “level” field of the end-marker;
- 1: the level is determined by `RMT_IDLE_OUT_LV_CH $n$` .

32.3.4.2 Wrap TX Mode

To transmit more pulse codes than can be fitted in the channel's RAM, users can enable wrap TX mode for channel n by setting `RMT_MEM_TX_WRAP_EN_CH $n$` . In this mode, the transmitter transmits the data from RAM in loops till an end-marker is encountered. For example, if `RMT_MEM_SIZE_CH $n$` = 1, the transmitter starts transmitting data from the address $48 * n$, and then the data from higher RAM address. Once the transmitter finishes transmitting the data from $(48 * (n+1) - 1)$, it continues transmitting data from $48 * n$ again till an end-marker is encountered. Wrap mode is also applicable for `RMT_MEM_SIZE_CH $n$` > 1.

When the size of transmitted pulse codes is larger than or equal to the value set by `RMT_TX_LIM_CH $n$` , an `RMT_CH $n$ _TX_THR_EVENT_INT` interrupt is triggered. In wrap mode, `RMT_TX_LIM_CH $n$` can be set to a half or a fraction of the size of the channel's RAM block. When an `RMT_CH $n$ _TX_THR_EVENT_INT` interrupt is detected by software, the already used RAM region can be updated by new pulse codes. In such way, the transmitter can seamlessly transmit unlimited pulse codes in wrap mode.

32.3.4.3 TX Modulation

Transmitter output can be modulated with a carrier wave by setting `RMT_CARRIER_EN_CH $n$` . The carrier waveform is configurable. In a carrier cycle, high level lasts for $(\text{RMT_CARRIER_HIGH_CH n } + 1) \text{ rmt_sclk}$ cycles, while low level lasts for $(\text{RMT_CARRIER_LOW_CH n } + 1) \text{ rmt_sclk}$ cycles. When `RMT_CARRIER_OUT_LV_CH $n$` is set, carrier wave is added on the high-level of output signals; while

`RMT_CARRIER_OUT_LV_CH $n$` is cleared, carrier wave is added on the low-level of output signals. Carrier wave can be added on all output signals during modulation, or just added on valid pulse codes (the data stored in RAM), which can be set by configuring `RMT_CARRIER_EFF_EN_CH $n$` :

- 0: add carrier wave on all output signals;
- 1: add carrier wave only on valid signals.

32.3.4.4 Continuous TX Mode

The continuous TX mode can be enabled by setting `RMT_TX_CONTI_MODE_CH $n$` . In this mode, the transmitter transmits the pulse codes from RAM in loops:

- If an end-marker is encountered, the transmitter starts transmitting the first data of the channel's RAM again.
- If no end-marker is encountered, there are two possible situations. In normal TX mode (`RMT_MEM_TX_WRAP_EN_CH $n$` = 0), an error interrupt occurs because the RAM is empty without any data to transmit. In wrap TX mode (`RMT_MEM_TX_WRAP_EN_CH $n$` = 1), the transmitter starts transmitting the first data again after the last data is transmitted.

If `RMT_TX_LOOP_CNT_EN_CH $n$` is set, the loop counting is incremented by 1 each time an end-marker is encountered. If the counting reaches the value set by `RMT_TX_LOOP_NUM_CH $n$` , an `RMT_CH $n$ _TX_LOOP_INT` interrupt is generated. If `RMT_LOOP_STOP_EN_CH $n$` is set, the transmission stops instantly after an `RMT_CH $n$ _TX_LOOP_INT` interrupt is generated. Otherwise, the transmission continues. In an end-marker, if its `period[14:0]` is 0, then the period of the previous data must satisfy:

$$6 \times T_{apb\_clk} + 12 \times T_{rmt\_sclk} < period \times T_{clk\_div} \quad (2)$$

The period of the other data only need to satisfy [relation \(1\)](#).

32.3.4.5 Simultaneous TX Mode

RMT module supports multiple channels transmitting data simultaneously. To use this function, follow the steps below:

1. Configure `RMT_TX_SIM_CH $n$` to choose which multiple channels are used to transmit data simultaneously;
2. Set `RMT_TX_SIM_EN` to enable this transmission mode;
3. Set `RMT_TX_START_CH $n$` for each selected channel to start data transmission.

The transmission starts once the final channel is configured. Due to hardware limitations, there is no guarantee that two channels can start transmitting data exactly at the same time. The interval between two channels starting transmitting data is within $3 \times T_{clk\_div}$.

32.3.5 Receiver

32.3.5.1 Normal RX Mode

The receiver of channel m is controlled by `RMT_RX_EN_CH $m$` :

- 0: the receiver stops receiving data;

- 1: the receiver starts working.

When the receiver becomes active, it starts counting from the first edge of the signal, detecting signal levels and counting clock cycles the level lasts for. Each cycle count (period) is then written back to RAM together with the level information (level). When the receiver detects no change in a signal level for a number of clock cycles more than the value set by `RMT_IDLE_THRES_CH $m$` , the receiver stops receiving data, returns to idle state, and generates an `RMT_CH $m$ _RX_END_INT` interrupt. Please note that `RMT_IDLE_THRES_CH $m$` should be configured to a maximum value according to your application, otherwise a valid received level may be mistaken as a level in idle state. If the RAM space of this RX channel is used up by the received data, the receiver stops receiving data, and an `RMT_CH $m$ _ERR_INT` interrupt is triggered by RAM FULL event.

32.3.5.2 Wrap RX Mode

To receive more pulse codes than can be fitted in the channel's RAM, users can enable wrap mode for channel m by configuring `RMT_MEM_RX_WRAP_EN_CH $m$` . In wrap mode, the receiver stores the received data to RAM space of this channel in loops. The receiving ends when the receiver detects no change in a signal level for a number of clock cycles more than the value set by `RMT_IDLE_THRES_CH $m$` . The receiver returns to idle state and generates an `RMT_CH $m$ _RX_END_INT` interrupt. For example, if `RMT_MEM_SIZE_CH $m$` is set to 1, the receiver starts receiving data and stores the data to address $48 * m$, and then to higher RAM address. When the receiver finishes storing the received data to $(48 * (m + 1) - 1)$, the receiver continues receiving data and storing data to the address $48 * m$ again, and the receiving ends when no change is detected on a signal level for more than `RMT_IDLE_THRES_CH $m$` clock cycles. Wrap mode is also applicable when `RMT_MEM_SIZE_CH $m$` > 1.

An `RMT_CH $m$ _RX_THR_EVENT_INT` interrupt is generated when the size of received pulse codes is larger than or equal to the value set by `RMT_RX_LIM_CH $m$` . In wrap mode, `RMT_RX_LIM_CH $m$` can be set to a half or a fraction of the size of the channel's RAM block. When an `RMT_CH $m$ _RX_THR_EVENT_INT` interrupt is detected, the already used RAM region can be updated by subsequent data.

32.3.5.3 RX Filtering

Users can enable the receiver to filter input signals by setting `RMT_RX_FILTER_EN_CH $m$` for channel m . The filter samples input signals continuously, and detects the signals which remain unchanged for a continuous `RMT_RX_FILTER_THRES_CH $m$` rmt_sclk cycles as valid. Otherwise, the signals will be detected as invalid. Only the valid signals can pass through the filter. The filter removes pulses with a length of less than `RMT_RX_FILTER_THRES_CH $m$` rmt_sclk cycles.

32.3.5.4 RX Demodulation

Users can enable RX demodulation on input signals or on filtered signals by setting `RMT_CARRIER_EN_CH $m$` . RX demodulation can be applied to high-level carrier wave or low-level carrier wave, depending on the configuration of `RMT_CARRIER_OUT_LV_CH $m$` :

- 0: demodulate low-level carrier wave;
- 1: demodulate high-level carrier wave.

Users can configure `RMT_CARRIER_HIGH_THRES_CH $m$` and `RMT_CARRIER_LOW_THRES_CH $m$` to set the thresholds to demodulate high-level carrier or low-level carrier. If the high-level of a signal lasts for less than `RMT_CARRIER_HIGH_THRES_CH $m$` clk_div cycles, or the low-level lasts for less than

`RMT_CARRIER_LOW_THRES_CH $m$` clk_div cycles, the signal is detected as a carrier and is then filtered out.

32.4 Configuration Update

To update RMT registers configuration, please set `RMT_CONF_UPDATE_CH $n/m$` for each channel first.

All the bits/fields listed in the second column of Table 32.4-1 should follow this rule.

Table 32.4-1. Configuration Update

Register	Bit/Field Configuration Update
TX Channel	
<code>RMT_CHnCONFO_REG</code>	<code>RMT_CARRIER_OUT_LV_CHn</code>
	<code>RMT_CARRIER_EN_CHn</code>
	<code>RMT_CARRIER_EFF_EN_CHn</code>
	<code>RMT_DIV_CNT_CHn</code>
	<code>RMT_IDLE_OUT_EN_CHn</code>
	<code>RMT_IDLE_OUT_LV_CHn</code>
	<code>RMT_TX_CONTI_MODE_CHn</code>
<code>RMT_CHnCARRIER_DUTY_REG</code>	<code>RMT_CARRIER_HIGH_CHn</code>
	<code>RMT_CARRIER_LOW_CHn</code>
<code>RMT_CHnTX_LIM_REG</code>	<code>RMT_TX_LOOP_CNT_EN_CHn</code>
	<code>RMT_TX_LOOP_NUM_CHn</code>
	<code>RMT_TX_LIM_CHn</code>
<code>RMT_TX_SIM_REG</code>	<code>RMT_TX_SIM_EN</code>
RX Channel	
<code>RMT_CHmCONFO_REG</code>	<code>RMT_CARRIER_OUT_LV_CHm</code>
	<code>RMT_CARRIER_EN_CHm</code>
	<code>RMT_IDLE_THRES_CHm</code>
	<code>RMT_DIV_CNT_CHm</code>
<code>RMT_CHmCONF1_REG</code>	<code>RMT_RX_FILTER_THRES_CHm</code>
	<code>RMT_RX_EN_CHm</code>
<code>RMT_CHmRX_CARRIER_RM_REG</code>	<code>RMT_CARRIER_HIGH_THRES_CHm</code>
	<code>RMT_CARRIER_LOW_THRES_CHm</code>
<code>RMT_CHmRX_LIM_REG</code>	<code>RMT_RX_LIM_CHm</code>
<code>RMT_REF_CNT_RST_REG</code>	<code>RMT_REF_CNT_RST_CHm</code>

32.5 Interrupts

ESP32-C5's RMT can generate the RMT_INTR interrupt signal that will be sent to the [Interrupt Matrix](#).

The following interrupt sources can generate the RMT_INTR interrupt signal:

- `RMT_CH $n/m$ _ERR_INT`: triggered when channel n/m does not read or write data correctly. For example, if the transmitter still tries to read data from RAM when the RAM is empty, or the receiver still tries to write data into RAM when the RAM is full, this interrupt will be triggered.

- RMT_CH n _TX_THR_EVENT_INT: triggered when the amount of data the transmitter has transmitted reaches the value set in [RMT_CH \$n\$ _TX_LIM_REG](#).
- RMT_CH m _RX_THR_EVENT_INT: triggered each time when the amount of data received by the receiver reaches the value set in [RMT_CH \$m\$ _RX_LIM_REG](#).
- RMT_CH n _TX_END_INT: triggered when the transmitter has finished transmitting signals.
- RMT_CH m _RX_END_INT: triggered when the receiver has finished receiving signals.
- RMT_CH n _TX_LOOP_INT: triggered when the loop counting reaches the value set by [RMT_TX_LOOP_NUM_CH \$n\$](#) .

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [32.6 Register Summary](#).

32.6 Register Summary

The addresses in this section are relative to Remote Control Peripheral base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
FIFO read and writer registers			
RMT_CH0DATA_REG	The read and write data register for channel 0 by APB FIFO access	0x0000	HRO
RMT_CH1DATA_REG	The read and write data register for channel 1 by APB FIFO access	0x0004	HRO
RMT_CH2DATA_REG	The read and write data register for channel 2 by APB FIFO access	0x0008	HRO
RMT_CH3DATA_REG	The read and write data register for channel 3 by APB FIFO access	0x000C	HRO
Configuration registers			
RMT_CHOCONFO_REG	Configuration register 0 for channel 0	0x0010	varies
RMT_CH1CONFO_REG	Configuration register 0 for channel 1	0x0014	varies
RMT_CH2CONFO_REG	Configuration register 0 for channel 2	0x0018	R/W
RMT_CH2CONF1_REG	Configuration register 1 for channel 2	0x001C	varies
RMT_CH3CONFO_REG	Configuration register 0 for channel 3	0x0020	R/W
RMT_CH3CONF1_REG	Configuration register 1 for channel 3	0x0024	varies
RMT_SYS_CONF_REG	Configuration register for RMT APB	0x0068	R/W
RMT_REF_CNT_RST_REG	RMT clock divider reset register	0x0070	WT
Status registers			
RMT_CH0STATUS_REG	Channel 0 status register	0x0028	RO
RMT_CH1STATUS_REG	Channel 1 status register	0x002C	RO
RMT_CH2STATUS_REG	Channel 2 status register	0x0030	RO
RMT_CH3STATUS_REG	Channel 3 status register	0x0034	RO
Interrupt registers			
RMT_INT_RAW_REG	Raw interrupt status	0x0038	R/WTC/SS
RMT_INT_ST_REG	Masked interrupt status	0x003C	RO
RMT_INT_ENA_REG	Interrupt enable bits	0x0040	R/W
RMT_INT_CLR_REG	Interrupt clear bits	0x0044	WT
Carrier wave duty cycle registers			
RMT_CH0CARRIER_DUTY_REG	Duty cycle configuration register for channel 0	0x0048	R/W
RMT_CH1CARRIER_DUTY_REG	Duty cycle configuration register for channel 1	0x004C	R/W
RMT_CH2_RX_CARRIER_RM_REG	Carrier remove register for channel 2	0x0050	R/W
RMT_CH3_RX_CARRIER_RM_REG	Carrier remove register for channel 3	0x0054	R/W
TX event configuration registers			
RMT_CHO_TX_LIM_REG	Configuration register for channel 0 TX event	0x0058	varies
RMT_CH1_TX_LIM_REG	Configuration register for channel 1 TX event	0x005C	varies
RMT_TX_SIM_REG	RMT TX synchronous register	0x006C	R/W

Name	Description	Address	Access
RX event configuration registers			
RMT_CH2_RX_LIM_REG	Configuration register for channel 2 RX event	0x0060	R/W
RMT_CH3_RX_LIM_REG	Configuration register for channel 3 RX event	0x0064	R/W
Version control register			
RMT_DATE_REG	Version control register	0x00CC	R/W

32.7 Registers

The addresses in this section are relative to Remote Control Peripheral base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 32.1. RMT_CH n DATA_REG (n : 0-3) (0x0000+0x4* n)

RMT_CHnDATA																															
31																															0
0x000000																															
																															Reset

RMT_CH n DATA Read and write data for channel n via APB FIFO. (HRO)

Register 32.2. RMT_CH n CONFO_REG (n : 0-1) (0x0010+0x4* n)

(reserved) RMT_CONF_UPDATE_CHⁿ (reserved) RMT_CARRIER_OUT_LV_CHⁿ RMT_CARRIER_EN_CHⁿ (reserved) RMT_MEM_SIZE_CHⁿ RMT_DIV_CNT_CHⁿ RMT_TX_STOP_CHⁿ RMT_IDLE_OUT_EN_CHⁿ RMT_IDLE_OUT_LV_CHⁿ RMT_MEM_TX_WRAP_EN_CHⁿ RMT_TX_CONTI_MODE_CHⁿ RMT_APB_MEM_RST_CHⁿ RMT_MEM_RD_RST_CHⁿ RMT_TX_START_CHⁿ																																							
31					25	24	23	22	21	20	19	18	16		15								8	7	6	5	4	3	2	1	0								
0	0	0	0	0	0	0	0	0	0	1	1	1	0	0x1		0x2							0							0	0	0	0	0	0	0	0	0	Reset

RMT_TX_START_CH n Configures whether to enable sending data in channel n .

0: No effect

1: Enable

(WT)

RMT_MEM_RD_RST_CH n Configures whether to reset RAM read address accessed by the transmitter for channel n .

0: No effect

1: Reset

(WT)

RMT_APB_MEM_RST_CH n Configures whether to reset RAM W/R address accessed by APB FIFO for channel n .

0: No effect

1: Reset

(WT)

Continued on the next page...

Register 32.2. RMT_CH n CONFO_REG (n : 0-1) (0x0010+0x4* n)

Continued from the previous page...

RMT_TX_CONTI_MODE_CH n Configures whether to enable continuous TX mode for channel n .

0: No Effect

1: Enable

In this mode, the transmitter starts transmission from the first data. If an end-marker is encountered, the transmitter starts transmitting data from the first data again. If no end-marker is encountered, the transmitter starts transmitting the first data again when the last data is transmitted.

(R/W)

RMT_MEM_TX_WRAP_EN_CH n Configures whether to enable wrap TX mode for channel n .

0: No effect

1: Enable

In this mode, if the TX data size is larger than the channel's RAM block size, the transmitter continues transmitting the first data to the last data in loops.

(R/W)

RMT_IDLE_OUT_LV_CH n Configures the level of output signal for channel n when the transmitter is in idle state. (R/W)

RMT_IDLE_OUT_EN_CH n Configures whether to enable the output for channel n in idle state.

0: No effect

1: Enable

(R/W)

RMT_TX_STOP_CH n Configures whether to stop the transmitter of channel n sending data out.

0: No effect

1: Stop

(R/W/SC)

RMT_DIV_CNT_CH n Configures the divider for clock of channel n .

Measurement unit: rmt_sclk

(R/W)

RMT_MEM_SIZE_CH n Configures the maximum number of memory blocks allocated to channel n .

(R/W)

RMT_CARRIER_EFF_EN_CH n Configures whether to add carrier modulation on the output signal only at data-sending state for channel n .

0: Add carrier modulation on the output signal at data-sending state and idle state for channel n

1: Add carrier modulation on the output signal only at data-sending state for channel n

Only valid when RMT_CARRIER_EN_CH n is 1.

(R/W)

Continued on the next page...

Register 32.2. RMT_CH n CONFO_REG (n : 0-1) (0x0010+0x4* n)

Continued from the previous page...

RMT_CARRIER_EN_CH n Configures whether to enable the carrier modulation on output signal for channel n .

0: Disable

1: Enable

(R/W)

RMT_CARRIER_OUT_LV_CH n Configures the position of carrier wave for channel n .

0: Add carrier wave on low level

1: Add carrier wave on high level

(R/W)

RMT_CONF_UPDATE_CH n Synchronization of RMT Channel n . (WT)

Register 32.3. RMT_CH m CONFO_REG (m : 2-3) (0x0018+0x8*(m -2))

(reserved)				RMT_CARRIER_OUT_LV_CH ^m				RMT_CARRIER_EN_CH ^m				(reserved)				RMT_MEM_SIZE_CH ^m				RMT_IDLE_THRES_CH ^m								RMT_DIV_CNT_CH ^m							
31	30	29	28	27	26	25	23	22											8	7											0				
0	0	1	1	0	0	0x1		0x7fff										0x2										Reset							

RMT_DIV_CNT_CH m Configures the clock divider of channel m .

Measurement unit: rmt_sclk

(R/W)

RMT_IDLE_THRES_CH m Configures RX threshold.

When no edge is detected on the input signal for continuous clock cycles longer than this field value, the receiver stops receiving data.

Measurement unit: clk_div

(R/W)

Continued on the next page...

Register 32.3. RMT_CH m CONFO_REG (m : 2-3) (0x0018+0x8*(m -2))

Continued from the previous page...

RMT_MEM_SIZE_CH m Configures the maximum number of memory blocks allocated to channel m .
(R/W)

RMT_CARRIER_EN_CH m Configures whether to enable carrier modulation on output signal for channel m .
0: Disable
1: Enable
(R/W)

RMT_CARRIER_OUT_LV_CH m Configures the position of carrier wave for channel m .
0: Add carrier wave on low level
1: Add carrier wave on high level
(R/W)

Register 32.4. RMT_CH m CONF1_REG (m : 2-3) (0x001C+0x8*(m -2))

(reserved)																RMT_CONF_UPDATE_CH _m (reserved) RMT_MEM_RX_WRAP_EN_CH _m RMT_RX_FILTER_THRES_CH _m RMT_RX_FILTER_EN_CH _m RMT_MEM_OWNER_CH _m RMT_APB_MEM_RST_CH _m RMT_MEM_WR_RST_CH _m RMT_RX_EN_CH _m													
31																16	15	14	13	12	5				4	3	2	1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0	0	0	0xf				0	1	0	0	0	Reset	

Reset

RMT_RX_EN_CH m Configures whether to enable the receiver to start receiving data in channel m .
0: Disable
1: Enable
(R/W)

RMT_MEM_WR_RST_CH m Configures whether to reset RAM write address accessed by the receiver for channel m .
0: No effect
1: Reset
(WT)

RMT_APB_MEM_RST_CH m Configures whether to reset RAM W/R address accessed by APB FIFO for channel m .
0: No effect
1: Reset
(WT)

Continued on the next page...

Register 32.4. RMT_CH m CONF1_REG (m : 2-3) (0x001C+0x8*(m -2))

Continued from the previous page...

RMT_MEM_OWNER_CH m Configures the ownership of channel m 's RAM block.

- 0: APB bus is using the RAM
 - 1: Receiver is using the RAM
- (R/W/SC)

RMT_RX_FILTER_EN_CH m Configures whether to enable the receiver's filter for channel m .

- 0: Disable
 - 1: Enable
- (R/W)

RMT_RX_FILTER_THRES_CH m Configures whether the receiver, when receiving data, ignores the input pulse when its width is shorter than this register value in units of rmt_sclk cycles.

- 0: No effect
 - 1: Reset
- (R/W)

RMT_MEM_RX_WRAP_EN_CH m Configures whether to enable wrap RX mode for channel m .

- 0: Disable
- 1: Enable

In this mode, if the RX data size is larger than channel m 's RAM block size, the receiver stores the RX data from the first address to the last address in loops.

(R/W)

RMT_CONF_UPDATE_CH m Synchronization of RMT Channel m . (WT)**Register 32.5. RMT_SYS_CONF_REG (0x0068)**

RMT_CLK_EN										(reserved)																				RMT_APB_FIFO_MASK									
31	30																													1	0								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset							

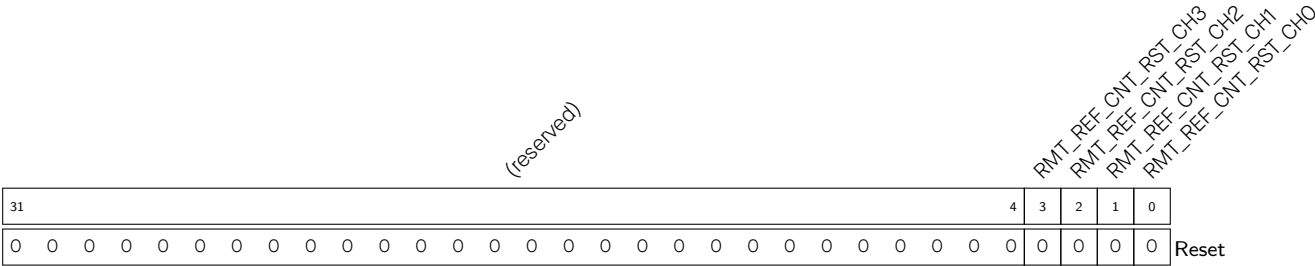
RMT_APB_FIFO_MASK Configures the memory access mode.

- 0: Access memory by FIFO
 - 1: Access memory directly
- (R/W)

RMT_CLK_EN Configures whether to enable signal of RMT register clock gate.

- 0: Power down the drive clock of registers
 - 1: Power up the drive clock of registers
- (R/W)

Register 32.6. RMT_REF_CNT_RST_REG (0x0070)



RMT_REF_CNT_RST_CH n (n : 0-1) Configures whether to reset the clock divider of channel n .
0: No effect
1: Reset
(WT)

RMT_REF_CNT_RST_CH m (m : 2-3) Configures whether to reset the clock divider of channel m .
0: No effect
1: Reset
(WT)

Register 32.7. RMT_CH n STATUS_REG (n : 0-1) (0x0028+0x4* n)

RMT_APB_MEM_RADDR_CH _n																RMT_APB_MEM_WR_ERR_CH _n																RMT_MEM_EMPTY_CH _n																RMT_APB_MEM_RD_ERR_CH _n																RMT_APB_MEM_WADDR_CH _n																RMT_STATE_CH _n																RMT_MEM_RADDR_EX_CH _n															
31								24								23				22				21				20				12								11				9				8				0																																																											
0x0								0								0				0				0								0				0				0								Reset																																																															

Reset

RMT_MEM_RADDR_EX_CH n Represents the memory address offset when transmitter of channel n is using the RAM. (RO)

RMT_STATE_CH n Represents the FSM status of channel n . (RO)

RMT_APB_MEM_WADDR_CH n Represents the memory address offset when writes RAM over APB bus. (RO)

RMT_APB_MEM_RD_ERR_CH n Represents whether the offset address exceeds memory size when reading via APB bus.

0: Not exceed

1: Exceed

(RO)

RMT_MEM_EMPTY_CH n Represents whether the TX data size exceeds the memory size and the wrap TX mode is disabled.

0: Not exceed

1: Exceed

(RO)

RMT_APB_MEM_WR_ERR_CH n Represents whether the offset address exceeds memory size (overflows) when writes via APB bus.

0: Not exceed

1: Exceed

(RO)

RMT_APB_MEM_RADDR_CH n Represents the memory address offset when reading RAM over APB bus. (RO)

Register 32.8. RMT_CH m STATUS_REG (m : 2-3) (0x0030+0x4*(m -2))

(reserved)				RMT_APB_MEM_RD_ERR_CH ^m				RMT_MEM_FULL_CH ^m				RMT_MEM_OWNER_ERR_CH ^m				RMT_STATE_CH ^m				(reserved)				RMT_APB_MEM_RADDR_CH ^m				(reserved)				RMT_MEM_WADDR_EX_CH ^m			
31	28	27	26	25	24	22	21	20	12				11	9	8	0																			
0	0	0	0	0	0	0	0	0	0				0	0	0	0				Reset															

RMT_MEM_WADDR_EX_CH m Represents the memory address offset when receiver of channel m is using the RAM. (RO)

RMT_APB_MEM_RADDR_CH m Represents the memory address offset when reads RAM over APB bus. (RO)

RMT_STATE_CH m Represents the FSM status of channel m . (RO)

RMT_MEM_OWNER_ERR_CH m Represents whether the ownership of memory block is wrong.
 0: The ownership of memory block is correct
 1: The ownership of memory block is wrong
 (RO)

RMT_MEM_FULL_CH m Represents whether the receiver receives more data than the memory can fit.
 0: The receiver does not receive more data than the memory can fit
 1: The receiver receives more data than the memory can fit
 (RO)

RMT_APB_MEM_RD_ERR_CH m Represents whether the offset address exceeds memory size (overflows) when reads RAM via APB bus.
 0: Not exceed
 1: Exceed
 (RO)

Register 32.9. RMT_INT_RAW_REG (0x0038)

(reserved)															RMT_CH1_TX_LOOP_INT_RAW RMT_CH0_TX_LOOP_INT_RAW RMT_CH3_RX_THR_EVENT_INT_RAW RMT_CH2_RX_THR_EVENT_INT_RAW RMT_CH1_TX_THR_EVENT_INT_RAW RMT_CH0_TX_THR_EVENT_INT_RAW RMT_CH3_ERR_INT_RAW RMT_CH2_ERR_INT_RAW RMT_CH1_ERR_INT_RAW RMT_CH0_ERR_INT_RAW RMT_CH3_RX_END_INT_RAW RMT_CH2_RX_END_INT_RAW RMT_CH1_TX_END_INT_RAW RMT_CH0_TX_END_INT_RAW														
31															14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset	

RMT_CH n _TX_END_INT_RAW (n : 0-1) The raw interrupt status of [RMT_CH \$n\$ _TX_END_INT](#). Triggered when the transmission is done. (R/WTC/SS)

RMT_CH m _RX_END_INT_RAW (m : 2-3) The raw interrupt status of [RMT_CH \$m\$ _RX_END_INT](#). Triggered when the reception is done. (R/WTC/SS)

RMT_CH n _ERR_INT_RAW (n : 0-3) The raw interrupt status of [RMT_CH \$n\$ / \$m\$ _ERR_INT](#). Triggered when error occurs. (R/WTC/SS)

RMT_CH n _TX_THR_EVENT_INT_RAW (n : 0-1) The raw interrupt status of [RMT_CH \$n\$ _TX_THR_EVENT_INT](#). Triggered when the transmitter sent more data than the configured value. (R/WTC/SS)

RMT_CH m _RX_THR_EVENT_INT_RAW (m : 2-3) The raw interrupt status of [RMT_CH \$m\$ _RX_THR_EVENT_INT](#). Triggered when the receiver receives more data than the configured value. (R/WTC/SS)

RMT_CH n _TX_LOOP_INT_RAW (n : 0-1) The raw interrupt status of [RMT_CH \$n\$ _TX_LOOP_INT](#). Triggered when the loop count reaches the configured threshold value. (R/WTC/SS)

Register 32.10. RMT_INT_ST_REG (0x003C)

(reserved)																RMT_CH1_TX_LOOP_INT_ST RMT_CH0_TX_LOOP_INT_ST RMT_CH3_RX_THR_EVENT_INT_ST RMT_CH2_RX_THR_EVENT_INT_ST RMT_CH1_TX_THR_EVENT_INT_ST RMT_CH0_TX_THR_EVENT_INT_ST RMT_CH3_ERR_INT_ST RMT_CH2_ERR_INT_ST RMT_CH1_ERR_INT_ST RMT_CH0_ERR_INT_ST RMT_CH3_RX_END_INT_ST RMT_CH2_RX_END_INT_ST RMT_CH1_TX_END_INT_ST RMT_CH0_TX_END_INT_ST															
31																14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset		

RMT_CH n _TX_END_INT_ST (n : 0-1) The masked interrupt status of RMT_CH n _TX_END_INT. (RO)

RMT_CH m _RX_END_INT_ST (m : 2-3) The masked interrupt status of RMT_CH m _RX_END_INT. (RO)

RMT_CH n _ERR_INT_ST (n : 0-3) The masked interrupt status of RMT_CH n / m _ERR_INT. (RO)

RMT_CH n _TX_THR_EVENT_INT_ST (n : 0-1) The masked interrupt status of RMT_CH n _TX_THR_EVENT_INT. (RO)

RMT_CH m _RX_THR_EVENT_INT_ST (m : 2-3) The masked interrupt status of RMT_CH m _RX_THR_EVENT_INT. (RO)

RMT_CH n _TX_LOOP_INT_ST (n : 0-1) The masked interrupt status of RMT_CH n _TX_LOOP_INT. (RO)

Register 32.11. RMT_INT_ENA_REG (0x0040)

(reserved)																RMT_CH1_TX_LOOP_INT_ENA RMT_CH0_TX_LOOP_INT_ENA RMT_CH3_RX_THR_EVENT_INT_ENA RMT_CH2_RX_THR_EVENT_INT_ENA RMT_CH1_TX_THR_EVENT_INT_ENA RMT_CH0_TX_THR_EVENT_INT_ENA RMT_CH3_ERR_INT_ENA RMT_CH2_ERR_INT_ENA RMT_CH1_ERR_INT_ENA RMT_CH0_ERR_INT_ENA RMT_CH3_RX_END_INT_ENA RMT_CH2_RX_END_INT_ENA RMT_CH1_TX_END_INT_ENA RMT_CH0_TX_END_INT_ENA															
31																14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset		

RMT_CH n _TX_END_INT_ENA (n : 0-1) Write 1 to enable RMT_CH n _TX_END_INT. (R/W)

RMT_CH m _RX_END_INT_ENA (m : 2-3) Write 1 to enable RMT_CH m _RX_END_INT. (R/W)

RMT_CH n _ERR_INT_ENA (n : 0-3) Write 1 to enable RMT_CH n/m _ERR_INT. (R/W)

RMT_CH n _TX_THR_EVENT_INT_ENA (n : 0-1) Write 1 to enable RMT_CH n _TX_THR_EVENT_INT.
(R/W)

RMT_CH m _RX_THR_EVENT_INT_ENA (m : 2-3) Write 1 to enable RMT_CH m _RX_THR_EVENT_INT.
(R/W)

RMT_CH n _TX_LOOP_INT_ENA (n : 0-1) Write 1 to enable RMT_CH n _TX_LOOP_INT. (R/W)

Register 32.12. RMT_INT_CLR_REG (0x0044)

(reserved)														RMT_CH1_TX_LOOP_INT_CLR RMT_CH0_TX_LOOP_INT_CLR RMT_CH3_RX_THR_EVENT_INT_CLR RMT_CH2_RX_THR_EVENT_INT_CLR RMT_CH1_TX_THR_EVENT_INT_CLR RMT_CH0_TX_THR_EVENT_INT_CLR RMT_CH3_ERR_INT_CLR RMT_CH2_ERR_INT_CLR RMT_CH1_ERR_INT_CLR RMT_CH0_ERR_INT_CLR RMT_CH3_RX_END_INT_CLR RMT_CH2_RX_END_INT_CLR RMT_CH1_TX_END_INT_CLR RMT_CH0_TX_END_INT_CLR															
31														14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reset
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0														0	0	0	0	0	0	0	0	0	0	0	0	0	0		

RMT_CH n _TX_END_INT_CLR (n : 0-1) Write 1 to clear the **RMT_CH n _TX_END_INT** interrupt. (WT)

RMT_CH m _RX_END_INT_CLR (m : 2-3) Write 1 to clear the **RMT_CH m _RX_END_INT** interrupt. (WT)

RMT_CH n _ERR_INT_CLR (n : 0-3) Write 1 to clear the **RMT_CH n / m _ERR_INT** interrupt. (WT)

RMT_CH n _TX_THR_EVENT_INT_CLR (n : 0-1) Write 1 to clear the **RMT_CH n _TX_THR_EVENT_INT** interrupt. (WT)

RMT_CH m _RX_THR_EVENT_INT_CLR (m : 2-3) Write 1 to clear the **RMT_CH m _RX_THR_EVENT_INT** interrupt. (WT)

RMT_CH n _TX_LOOP_INT_CLR (n : 0-1) Write 1 to clear the **RMT_CH n _TX_LOOP_INT** interrupt. (WT)

Register 32.13. RMT_CH n CARRIER_DUTY_REG (n : 0-1) (0x0048+0x4* n)

RMT_CARRIER_HIGH_CH ⁿ																RMT_CARRIER_LOW_CH ⁿ														
31															16	15														0
0x40																0x40														Reset

RMT_CARRIER_LOW_CH n Configures carrier wave's low level clock period for channel n .

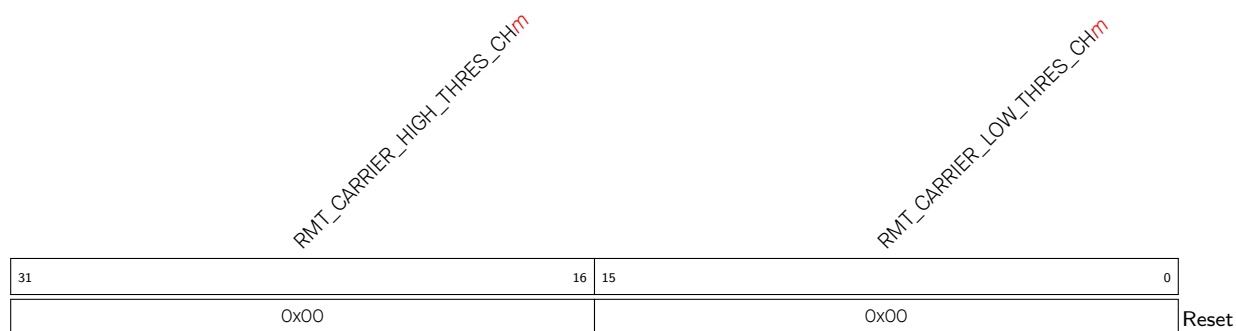
Measurement unit: rmt_sclk

(R/W)

RMT_CARRIER_HIGH_CH n Configures carrier wave's high level clock period for channel n .

Measurement unit: rmt_sclk

(R/W)

Register 32.14. RMT_CH m _RX_CARRIER_RM_REG (m : 2-3) (0x0050+0x4*(m -2))

RMT_CARRIER_LOW_THRES_CH m Configures the low level period in a carrier modulation mode for channel m .

The low level period in a carrier modulation mode is (RMT_CARRIER_LOW_THRES_CH m + 1) for channel m .

Measurement unit: clk_div

(R/W)

RMT_CARRIER_HIGH_THRES_CH m Configures the high level period in a carrier modulation mode for channel m .

The high level period in a carrier modulation mode is (RMT_CARRIER_HIGH_THRES_CH m + 1) for channel m .

Measurement unit: clk_div

(R/W)

Register 32.15. RMT_CH n _TX_LIM_REG (n : 0-1) (0x0058+0x4* n)

(reserved)										RMT_LOOP_STOP_EN_CH ⁿ RMT_LOOP_COUNT_RESET_CH ⁿ RMT_TX_LOOP_CNT_EN_CH ⁿ				RMT_TX_LOOP_NUM_CH ⁿ				RMT_TX_LIM_CH ⁿ								
31											22	21	20	19	18					9	8					0
0	0	0	0	0	0	0	0	0	0	0	0	0	0				0x80								Reset	

Reset

RMT_TX_LIM_CH n Configures the maximum entries that channel n can send out. (R/W)

RMT_TX_LOOP_NUM_CH n Configures the maximum loop count when Continuous TX mode is valid.
(R/W)

RMT_TX_LOOP_CNT_EN_CH n Configures whether to enable loop count.
0: No effect
1: Enable
(R/W)

RMT_LOOP_COUNT_RESET_CH n Configures whether to reset the loop count when tx_conti_mode is valid.
0: No effect
1: Reset
(WT)

RMT_LOOP_STOP_EN_CH n Configures whether to enable the loop send stop function after the loop counter counts to loop number for channel n .
0: No effect
1: Enable
(R/W)

Register 32.16. RMT_TX_SIM_REG (0x006C)

(reserved)																															RMT_TX_SIM_EN RMT_TX_SIM_CH1 RMT_TX_SIM_CH0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RMT_TX_SIM_CH n (n : 0-1) Configures whether to enable channel n to start sending data synchronously with other enabled channels.

0: No effect

1: Enable

(R/W)

RMT_TX_SIM_EN Configures whether to enable multiple of channels to start sending data synchronously.

0: No effect

1: Enable

(R/W)

Register 32.17. RMT_CH m _RX_LIM_REG (m : 2-3) (0x0060+0x4*(m -2))

(reserved)																RMT_RX_LIM_CH ^m																	
31																9	8	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x80																Reset	

RMT_RX_LIM_CH m Configures the maximum entries that Channel m can receive. (R/W)

Register 32.18. RMT_DATE_REG (0x00CC)

(reserved)				RMT_DATE																							
31	28	27																						0			
0	0	0	0	0x2108213																							Reset

RMT_DATE Version control register. (R/W)

Chapter 33

Parallel IO Controller (PARLIO)

33.1 Introduction

ESP32-C5 contains a Parallel IO controller (PARLIO) capable of transferring data between external devices and internal memory on a parallel bus through GDMA. It is composed of a TX unit and an RX unit, which are fixed as a transmitter and a receiver respectively. With the two units combined, PARLIO achieves full-duplex communication.

Due to its flexibility, PARLIO can function as a general interface to connect various peripherals. For example, with SPI as the master device and PARLIO as the slave device, a peer-to-peer transfer can be achieved. For detailed application examples, refer to Section [33.8](#).

33.2 Glossary

This section covers terminology used to describe the functionality of PARLIO.

RX unit	Module in PARLIO responsible for receiving data from external parallel bus and storing them into internal memory.
TX unit	Module in PARLIO responsible for transmitting data from internal memory to external parallel bus.
RXD	Parallel data received from the IO interface of the RX unit.
TXD	Parallel data sent from the IO interface of the TX unit.
Frame	Transferred data unit from the moment the START signal is set to the moment the End of Frame (EOF) signal is received.
Free-running clock	Clock that toggles continuously. Otherwise, the clock only toggles during the period when valid data is received and remains constant for the rest of the time.
GDMA SUC EOF	Signal that indicates GDMA successful end of frame. When GDMA receives this signal, a GDMA interrupt will be triggered, indicating that the current frame is correct and the receive is finished.
GDMA ERR EOF	Signal that indicates GDMA error end of frame. When GDMA receives this signal, a GDMA interrupt will be triggered, indicating that the current frame has error and the receive is finished.
CDC	Clock domain crossing.

33.3 Features

The PARLIO module has the following main features:

- Variety of clock sources:
 - Including external IO clock PAD_CLK_TX/RX and internal system clocks XTAL_CLK, PLL_F240M_CLK, and RC_FAST_CLK
 - Maximum clock frequency of 40 MHz
 - Integer clock frequency division
- 1/2/4/8-bit configurable data bus width
- Full-duplex communication with 8-bit data bus width
- Bit reversal when data bus width is 1/2/4-bit
- RX unit for receiving IO parallel data, which supports:
 - Output clock gating
 - RX unit input and output clock inverse
 - Various receive modes
 - Configurable GDMA SUC EOF generation
 - Configurable IO pin of external enable signal
- TX unit for sending IO parallel data, which supports:
 - Chip select function
 - Output clock gating
 - TX unit input and output clock inverse
 - Configurable bus idle value

33.4 Architectural Overview

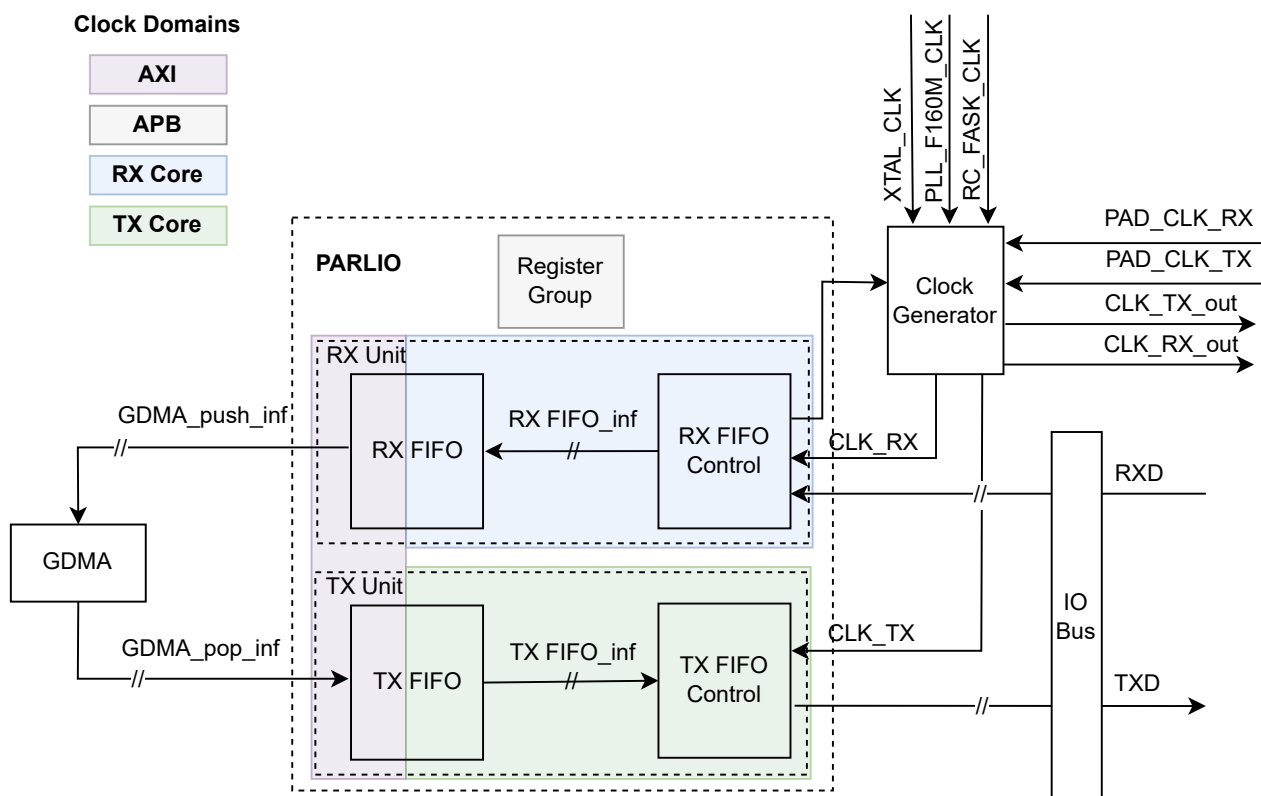


Figure 33.4-1. PARLIO Architecture

Figure 33.4-1 shows the architecture of PARLIO. In addition to the RX unit and the TX unit, a group of status configuration registers is also included.

The RX unit receives the RXD, stores it in an asynchronous FIFO after the serial-to-parallel conversion, and then stores the data to internal memory via GDMA.

The TX unit fetches data from the internal memory via GDMA, stores it in an asynchronous FIFO, and outputs the data from the IO bus via the TXD signal after the parallel-to-serial conversion.

33.5 Functional Description

33.5.1 Clock Generator

There are four input clock domains in PARLIO, namely, RX Core, TX Core, AHB, and APB, as shown in Figure 33.4-1.

The status configuration register group works in the APB clock domain.

The GDMA interface logic works in the AHB clock domain.

RX Core and TX Core clock domains each have four clock sources for selection, i.e., the internal system clock sources XTAL_CLK, RC_FAST_CLK, PLL_F240M_CLK, and the external clock source (PAD_CLK_TX/RX), as shown in Figure 33.5-1. Clock sources can be selected by configuring [PCR_PARL_CLK_RX_SEL](#) and

PCR_PARL_CLK_TX_SEL. The clock can be divided by configuring **PCR_PARL_CLK_RX_DIV_NUM** and **PCR_PARL_CLK_TX_DIV_NUM**. The clock division factor can be configured up to $(2^{16} - 1)$.

The input clock of the TX and RX units can be inverted. The operating clock of the TX and RX units can also be inverted before being output to IO. The TX and RX units also support clock gating of the output clock.

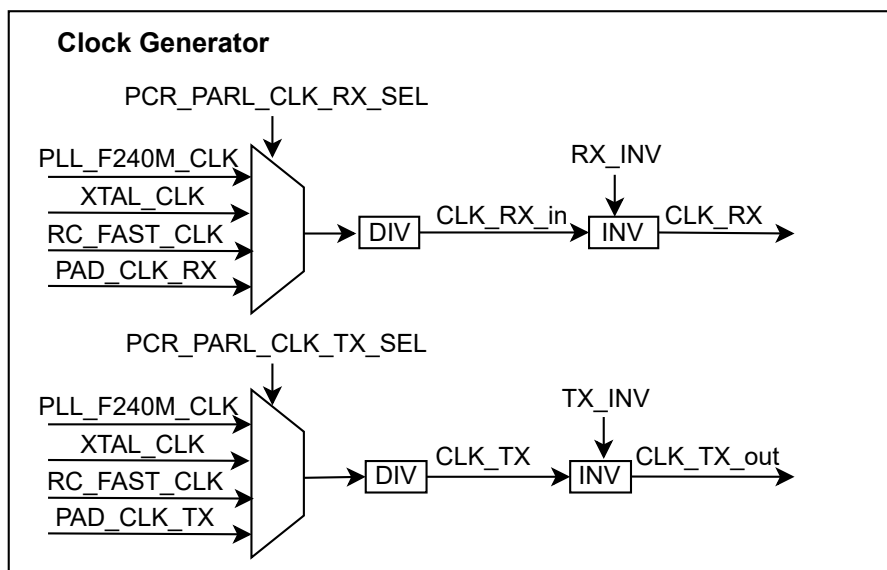


Figure 33.5-1. PARLIO Clock Generation

33.5.2 Clock and Reset Restriction

Due to the versatility of PARLIO, the PAD clocks of PARLIO (PAD_CLK_TX/RX) may come from different masters (external devices or internal clock sources). These clocks might be either free-running clock or not. If the clock is not free-running, some internal control signals of PARLIO cannot process CDC, so there are certain restrictions during the operation.

1. During the reset of the asynchronous FIFO, it takes two clock cycles to initialize FIFO. Therefore, if the reset of AHB clock domain is performed with a clock that is not free-running, the reset synchronization must be performed two clock cycles in advance. The specific operation is shown in the table below.

Table 33.5-1. Operations to Reset AHB Clock Domain with Clock Restrictions

Clock Restriction		Specific Operation
The Current Frame	The Next Frame	
Free-running clock	Not free-running clock	Users can reset the next frame transfer before switching to the clock that is not free-running. After the reset is completed, users can switch the clock.
Not free-running clock	Free-running clock	The next frame can be reset freely. Users only need to ensure that there is an interval of two clock cycles between the reset and the start of the transfer.
Not free-running clock	Not free-running clock	If the next frame transfer needs to be reset, users need to first switch to the internal free-running clock, and then switch to the actual clock after the reset is completed.

2. Due to the restrictions caused by a clock that is not free-running, [PARL_IO_RX_START](#) and [PARL_IO_TX_START](#) cannot perform CDC processing. Therefore, it is necessary to wait until [PARL_IO_RX_START](#) and [PARL_IO_TX_START](#) are stable before starting the data transfer, otherwise the transfer might enter a metastable state.

Here are the specific operation steps in the RX unit:

- (a) Clear [PCR_PARL_CLK_RX_EN](#) to turn off RX Core clock domain;
- (b) Write 1 to [PARL_IO_RX_START](#);
- (c) Set [PCR_PARL_CLK_RX_EN](#) to turn on RX Core clock domain;
- (d) Operate the external device to start sending data;
- (e) Clear [PCR_PARL_CLK_RX_EN](#) to turn off RX Core clock domain;
- (f) Write 0 to [PARL_IO_RX_START](#).

Here are the specific operation steps in the TX unit:

- (a) Clear [PCR_PARL_CLK_TX_EN](#) to turn off TX Core clock domain;
- (b) Write 1 to [PARL_IO_TX_START](#);
- (c) Set [PCR_PARL_CLK_TX_EN](#) to turn on TX Core clock domain;
- (d) Operate the external device to start receiving data;
- (e) Clear [PCR_PARL_CLK_TX_EN](#) to turn off TX Core clock domain;
- (f) Write 0 to [PARL_IO_TX_START](#).

3. Reset should follow the requirements below:

- The clock reset during the chip start-up should follow the sequence below:
 - (a) Reset APB clock domain;
 - (b) Reset AHB clock domain;
 - (c) Reset Core clock domain.
- Inter-frame transfer requires Core clock domain reset and async FIFO reset.

33.5.3 Master-Slave Mode

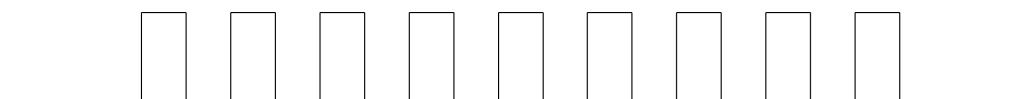
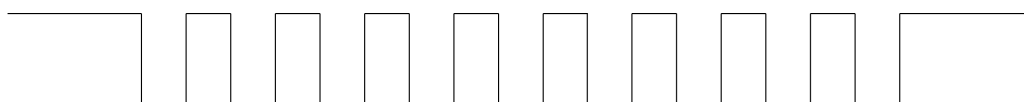
The TX and RX units can function as both master and slave.

When the TX unit serves as master, it is necessary to set the internal free-running clock as the clock source. The TX unit drives TXD on the rising edge of the clock.

When the TX unit functions as a slave device, there are three scenarios, as shown in the table below.

Table 33.5-2. Requirements for TX Unit Operating as Slave with Clock Restrictions

Clock Restriction		Requirement
Clock Sent by Master	Clock Waveform	
Free-running clock	Any waveform	There is no requirement for the sampling edge of the master clock.
Not free-running clock	Positive waveform (Figure 33.5-2)	The master clock should sample TXD at the falling edge.
Not free-running clock	Negative waveform (Figure 33.5-3)	The master device should invert the original clock and convert it to the waveform as Figure 33.5-2 shows before output.

**Figure 33.5-2. Positive Waveform****Figure 33.5-3. Negative Waveform**

When the RX unit serves as the master, it is required to set the clock source as the internal free-running clock. The RX unit drives RXD on the rising edge of the clock.

When the RX unit functions as the slave, there are three scenarios, as shown in the table below.

Table 33.5-3. Requirements for RX Unit Operating as Slave with Clock Restrictions

Clock Restriction		Requirement
Clock Sent by Master	Clock Waveform	
Free-running clock	Any waveform	There is no requirement for the sampling edge of the master clock, and the valid data is subject to the external enable signal.
Not free-running clock	Positive waveform (Figure 33.5-2)	It is required for the master device to drive the data at the rising edge and the RX unit to sample the data at the falling edge (i.e., to inverse the master clock).
Not free-running clock	Negative waveform (Figure 33.5-3)	It is required for the master device to drive the data at the falling edge and the RX unit to sample the data at the rising edge (i.e., to use the original master clock).

33.5.4 Receive Modes of the RX Unit

PARLIO supports 8 receive modes, which can be divided into three major categories according to the enable signal:

- Level Enable mode: data received is enabled by the external signal level;

- Pulse Enable mode: data received is enabled by the external signal pulse;
- Software Enable mode: the enable signal of data received can be configured by users directly.

The RX unit also supports inverse of the external enable signal. If the external enable signal is active-low, users can enable the function by setting [PARL_IO_RX_EXT_EN_INV](#) to switch to the corresponding receive mode introduced as follows.

33.5.4.1 Level Enable Mode

Figure 33.5-4 shows the Level Enable mode. In this mode, an active level on the external enable signal must be aligned with valid data. Since the external level enable signal occupies one IO pin, there are at most 7 IO pins left usable for RXD.

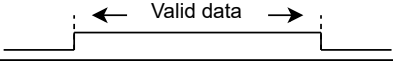
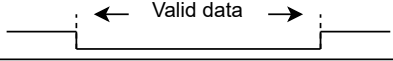
Mode	Sub-mode	Description
LEVEL_ENABLE	sub-mode 1	signal level high
		
	sub-mode 2	signal level low
		

Figure 33.5-4. Sub-Modes of Level Enable Mode for RX Unit

33.5.4.2 Pulse Enable Mode

Pulse Enable mode can be divided into 6 sub-modes depending on the pulse active level and its alignment with valid data. For detailed classification, see Figure 33.5-5.

Sub-modes 1 ~ 4 all contain start pulse and end pulse. The difference lies in whether start pulse and end pulse are aligned with valid data.

Sub-modes 5 ~ 6 only contain start pulse and the end of valid data is signaled by configuring [PARL_IO_RX_BITLEN](#).

Since the external pulse enable signal occupies one IO pin, there are at most 7 IO pins left usable for RXD. However, in sub-modes 4 and 6, as the data is considered valid before the pulse's first edge and after the pulse's last edge, the enable signal IO pin can serve as a data IO pin at the same time. Therefore, there are 8 IO pins usable for RXD in these two sub-modes.

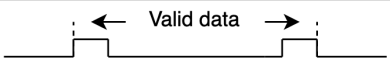
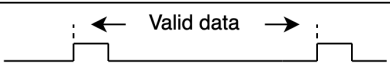
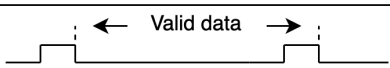
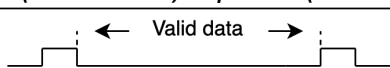
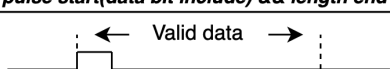
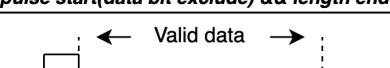
Mode	Sub-mode	Description
PULSE_ENABLE	sub-mode1	<i>pulse start(data bit include) && pulse end(data bit include)</i>
		
	sub-mode2	<i>pulse start(data bit include) && pulse end(data bit exclude)</i>
		
	sub-mode3	<i>pulse start(data bit exclude) && pulse end(data bit include)</i>
		
	sub-mode4	<i>pulse start(data bit exclude) && pulse end(data bit exclude)</i>
		
	sub-mode5	<i>pulse start(data bit include) && length end</i>
		
	sub-mode6	<i>pulse start(data bit exclude) && length end</i>
		

Figure 33.5-5. Sub-Modes of Pulse Enable Mode for RX Unit

33.5.4.3 Software Enable Mode

The enable signal in Software Enable mode is determined by the internal configuration register. If users switch to this mode, the receive will only be activated when both `PARL_IO_RX_SW_EN` and `PARL_IO_RX_START` are set to 1.

Since the enable signal does not occupy IO pins on the interface, there are at most 8 IO pins usable by the RXD. Due to the differences of clock domains, the enable signal cannot be aligned with valid data. Thus, the validity of data needs to be identified by the valid clock edge. In this case, the RX Core clock needs to be aligned with valid data.

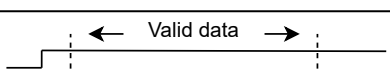
Mode	Sub-mode	Description
SW_ENABLE	/	/
		

Figure 33.5-6. Sub-Mode of Software Enable Mode for RX Unit

33.5.5 RX Unit GDMA SUC EOF Generation

The RX unit generates a GDMA SUC EOF signal to indicate the end of current frame transfer and send it to the GDMA interface. GDMA SUC EOF can be generated by an external enable signal or triggered by the internally configured bit length.

- When GDMA SUC EOF is generated by the internally configured bit length, there is no restriction on the receive mode selection, but `PARL_IO_RX_BITLEN` must be configured. If the configured value of `PARL_IO_RX_BITLEN` is less than the actual received data, the GDMA SUC EOF will be triggered in

advance. In this case, the RX unit stops reading data from the FIFO, but the FIFO continues to receive external data until an [RX_FIFO_WOVF_INT](#) interrupt is triggered.

- When GDMA SUC EOF is generated by the external enable signal, only sub-modes 1 and 3 of Pulse Enable mode can be selected. In this mode, the transfer is not affected by the value of [PARL_IO_RX_BITLEN](#), and the transferred data length of the frame is not limited.

33.5.6 RX Unit Timeout

The RX unit supports receive timeout. If the RX FIFO has not been receiving valid data for a long time, a timeout will be triggered. In such case, a GDMA ERR EOF signal will be generated and sent to the GDMA interface to indicate the end of the reception. Configure [PARL_IO_RX_TIMEOUT_THRES](#) to set the timeout threshold.

The timeout function is enabled by default and can be disabled by users. The upper threshold of the configurable timeout is $(2^{16} - 1)$ cycles of GDMA clock domain, and the lower threshold depends on the relative frequency relationship between GDMA clock domain and RX Core clock domain. It is recommended to set a relatively large value for [PARL_IO_RX_TIMEOUT_THRES](#) to avoid undesired GDMA ERR EOF signals.

33.5.7 Unlimited Length Data Transmission of TX Unit

By default, the end of transmission for the TX unit is determined by the configured value of [PARL_IO_TX_BITLEN](#). When the transmitted data reaches the configured value, the system generates a GDMA SUC EOF interrupt signal, indicating the completion of the current data frame transmission. This approach limits the maximum length of a single frame transmission to the bit width defined by [PARL_IO_TX_BITLEN](#), preventing it from achieving unlimited length data transmission.

To enable larger single-frame transmissions, set [PARL_IO_TX_EOF_GEN_SEL](#) to 1. In this mode, the hardware uses the EOF configured in the GDMA inlink list as the end-of-transmission signal for a single frame. As a result, data transmission is no longer constrained by the bit width of [PARL_IO_TX_BITLEN](#), allowing for significantly larger data frames.

33.5.8 TX Chip Select Function

To emulate the LCD I8080 protocol, the TX unit is equipped with an output chip select (CS) signal, which is active low. Configure [PARL_IO_TX_CS_START_DELAY](#) and [PARL_IO_TX_CS_STOP_DELAY](#) to control the phase relationship between the DMA output data and the CS signal. Note that when clock gating is enabled and the CS signal is used as the control source for clock gating, configuring these two registers not only implements the basic function, but also allows control over the phase relationship between the output clock and the CS signal.

33.5.9 Output Clock Gating of TX Unit

The TX unit supports output clock gating. The clock gating function is disabled by default, and can be enabled by software via setting [PARL_IO_TX_GATING_EN](#).

Currently, the most significant bit (MSB) of TXD can be configured to be controlled by one of two sources: DMA output data or the chip select signal (CS). Set [PARL_IO_TX_VALID_OUTPUT_EN](#) to 0 to select the DMA output data as the control source. In this case, the TX unit must be configured with the maximum bit width,

because when the control source is the DMA output data, the gating signal is fixed to the MSB of TXD. The clock signal can be toggled when this bit is at a high level. Note that when selecting DMA output data as the control source, if the peripheral is in an idle state, the clock gating will be controlled by the MSB of the user-configured bus idle value. For configuration guidelines, refer to [Section 33.5.10 Bus Idle Value of TX Unit](#). When the gating function is enabled while using DMA output data as the control source, there are at most 7 IO pins left usable for TXD as the gating signal occupies one IO.

Set [PARL_IO_TX_VALID_OUTPUT_EN](#) to 1 to select the chip select signal (CS) as the control source. In this case, there is no limit to the bit width.

33.5.10 Bus Idle Value of TX Unit

The TX unit is regarded as in idle state when it is not transmitting data. It supports a configurable bus idle value.

Note that the configured idle value should not conflict with other enabled functions. For example, when the MSB of TXD is used as the valid signal, users should avoid configuring the MSB of the idle value as 1.

33.5.11 Data Transfer in a Single Frame

The RX unit and the TX unit transfer data in the unit of bits, i.e., a single frame transfers 1 bit of data at least.

When the RX unit generates GDMA EOF signals through bit length, the maximum length of the single-frame transmission is $(2^{19} - 1)$ bits. When the RX unit generates GDMA EOF signals through the enable signal from an external device, there is no limit to the amount of bits of the single-frame transmission.

The TX unit only generates GDMA EOF signals through byte length, so the maximum length of a single frame transmission is $(2^{19} - 1)$ bytes.

Since PARLIO transfers data in the unit of bytes on the GDMA side, it will process the IO data when it is not aligned with the bytes.

- When receiving data, PARLIO will automatically pad 0 to the high bits of the data stored in memory via GDMA to make it aligned with bytes.
- When sending data, PARLIO will truncate the data retrieved from memory according to the configured value of [PARL_IO_TX_BITLEN](#). Data exceeding the value will not be sent.

When the configured data bus width is 2/4/8-bit, the bit length must be configured as a multiple of the corresponding bus width.

33.5.12 Bit Reversal in One Byte

The sequence of data within one byte can be reversed when data bus width is 1/2/4-bit. Taking the RX unit as an example, when the configured bus width is 2 bits, the data needs to be packed into one byte before being written into the RX FIFO.

Presume that the original bit sequence is:

$$\{\{b_0, b_1\}, \{b_2, b_3\}, \{b_4, b_5\}, \{b_6, b_7\}\}$$

If the bit reversal is enabled, the sequence will be reordered to:

$$\{\{b_6, b_7\}, \{b_4, b_5\}, \{b_2, b_3\}, \{b_0, b_1\}\}$$

33.6 Interrupts

ESP32-C5's PARLIO module can generate the following interrupt signals that will be sent to the [Interrupt Matrix](#).

- PARL_IO_TX_INTR
- PARL_IO_RX_INTR

There are several internal interrupt sources from PARLIO that can generate the above interrupt signals. The interrupt sources from PARLIO are listed with their trigger conditions and the resulted interrupt signals in Table 33.6-1.

Table 33.6-1. PARLIO's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
TX_FIFO_EMPTY_INT	TX FIFO is empty, indicating possible error in the data sent by TX	PARL_IO_TX_INTR
RX_FIFO_WOVF_INT	RX FIFO is full, indicating possible error in the data received by RX	PARL_IO_RX_INTR
TX_EOF_INT	TX finishes sending a complete frame of data	PARL_IO_TX_INTR

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix](#) [to be added later] > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section 33.9 [Register Summary](#).

33.7 Programming Procedures

33.7.1 Data Receiving Operation Process

This section introduces the programming procedure for receiving data in the RX unit. Perform the following procedure to receive parallel data from IO pins connected to external devices to be stored in the internal memory. For detailed description of the clock and reset operation restrictions in the RX unit, refer to Section 33.5.2.

1. Reset the RX unit. For specific reset scenarios and sequences, refer to Section 33.5.2.
2. Set [PARL_IO_RX_FIFO_WOVF_INT_CLR](#) and [PARL_IO_RX_FIFO_WOVF_INT_ENA](#).
3. Select the RXD IO pins. If a PAD clock is used, the clock IO pin also needs to be configured.
4. Select the clock source and divide the clock by configuring PCR registers.
5. Turn off the clock of RX Core clock domain.

6. Select the receive mode and enable functions required as described in Sections [33.3](#) and [33.5](#).
7. Configure GDMA inlink list.
8. Set [PARL_IO_RX_REG_UPDATE](#) to synchronize the register signals.
9. Set [PARL_IO_RX_START](#).
10. Turn on the clock of RX Core clock domain.
11. Operate the external device to start sending data.
12. Poll the GDMA SUC EOF interrupt.
13. Clear the GDMA SUC EOF interrupt.
14. Turn off the clock of RX Core clock domain.
15. Clear [PARL_IO_RX_START](#).

33.7.2 Data Transmitting Operation Process

This section introduces the programming procedure for transmitting data in the TX unit. Perform the following procedure to transmit parallel data from internal memory to the IO pins connected to external devices. For detailed description of the clock and reset operation restrictions in the TX unit, refer to Section [33.5.2](#).

1. Reset the TX unit. For specific reset scenarios and sequences, refer to Section [33.5.2](#).
2. Set [PARL_IO_TX_FIFO_EMPTY_INT_CLR](#), [PARL_IO_TX_EOF_INT_CLR](#), [PARL_IO_TX_FIFO_EMPTY_INT_ENA](#), and [PARL_IO_TX_EOF_INT_ENA](#) consecutively.
3. Select the TXD IO pins. If a PAD clock is used, the clock IO PAD also needs to be configured.
4. Select the clock source and divide the clock by configuring PCR registers.
5. Turn off the clock of TX Core clock domain.
6. Select the functions required as described in Section [33.5](#).
7. Configure GDMA outlink list.
8. Poll the [PARL_IO_TX_READY](#).
9. Set [PARL_IO_TX_START](#).
10. Turn on the clock of TX Core clock domain.
11. Operate the external device to start receiving data.
12. Poll the [PARL_IO_TX_EOF_INT_ST](#).
13. Set [PARL_IO_TX_EOF_INT_CLR](#).
14. Turn off the clock of TX Core clock domain.
15. Clear [PARL_IO_TX_START](#).

33.8 Application Examples

This section introduces some PARLIO application examples and their detailed operation process. All peripherals used in the examples are from ESP series chips and can work with PARLIO to form a complete data path.

Note:

The data paths constructed in the examples may not be the optimal. For example, users can use the SPI peripherals on two identical ESP chips to complete the peer-to-peer transfer in real case instead of using PARLIO to work with SPI. However, these examples demonstrate the flexibility of the PARLIO interface to a certain extent.

33.8.1 Co-working with SPI

In this example, external SPI sends data as a master device and PARLIO RX unit receives data as a slave device, and at the same time, PARLIO TX sends data as a master device and SPI receives data as a slave device, thus achieving a peer-to-peer serial data transfer.

- Follow the operation process below to achieve SPI transmit and PARLIO receive:
 1. Configure SPI clock.
 2. Configure SPI as the master device.
 3. Configure signal pins. Connect FSPICLK to PAD_CLK_RX, FSPICSO to RXD[7], and FSPID to RXD[0].
 4. Write the data sent into the SPI buffer and configure the bit length of the data sent.
 5. Set SPI_UPDATE to update the configured register value.
 6. Reset PARLIO RX unit.
 7. Configure PARLIO RX unit clock.
 8. Turn off the PARLIO RX Core clock domain.
 9. Configure PARLIO receive mode as sub-mode 1 of Level Enable mode. Configure RX unit data bus width as 1 bit. Configure [PARL_IO_RX_BITLEN](#) according to the sending length of SPI. Set [PARL_IO_RX_REG_UPDATE](#).
 10. Configure PARLIO GDMA inlink list.
 11. Set [PARL_IO_RX_START](#).
 12. Turn on the PARLIO RX Core clock domain.
 13. Set SPI_USR to start transmitting data of SPI.
 14. Poll GDMA SUC EOF interrupt.
 15. Clear [PARL_IO_RX_START](#).
- Follow the operation process below to achieve PARLIO transmit and SPI receive:
 1. Configure SPI clock.
 2. Configure SPI as the slave device.

3. Configure signal pins. Connect FSPICLK to PAD_CLK_TX, FSPICSO to TXD[7], and FSPID to TXD[0].
4. Set SPI_RD_BIT_ORDER to invert the bit order.
5. Set SPI_UPDATE to update the configured register value.
6. Reset PARLIO TX unit.
7. Set [PARL_IO_TX_EOF_INT_CLR](#) and [PARL_IO_TX_EOF_INT_ENA](#).
8. Configure PARLIO TX unit clock.
9. Turn off the clock of TX Core clock domain.
10. Configure data bus width as 1 bit. Write 1 to [PARL_IO_TX_VALID_OUTPUT_EN](#). Configure [PARL_IO_TX_BITLEN](#).
11. Configure GDMA outlink list.
12. Poll [PARL_IO_TX_READY](#).
13. Write 1 to [PARL_IO_TX_START](#).
14. Turn on the clock of TX Core clock domain.
15. Start data transfer.
16. Poll [PARL_IO_TX_EOF_INT_ST](#).
17. Set [PARL_IO_TX_EOF_INT_CLR](#).
18. Turn off the clock of TX Core clock domain.
19. Clear [PARL_IO_TX_START](#).

33.8.2 Co-working with I2S

In this example, external I2S sends data as a master device and PARLIO RX unit receives data as a slave device. PARLIO supports the transmission of the I2S TDM MSB alignment standard and the TDM PCM standard. When the I2S transfer protocol is the TDM MSB alignment standard, it is required to configure the receive mode of PARLIO as Level Enable mode. When the I2S transfer protocol is the TDM PCM standard, it is required to configure the receive mode of PARLIO as the sub-mode 10 of Pulse Enable mode and set [PARL_IO_RX_EXT_EN_INV](#).

This section takes the TDM PCM alignment standard as an example. The specific operation process is as follows:

1. Configure I2S clock.
2. Configure signal pins. Connect I2SO_BCK_out to PAD_CLK_RX, I2SO_WS_out to RXD[7], and I2SO_Data_out to RXD[0].
3. Configure I2S as the master device.
4. Configure the I2S TX data mode and channel mode required. Set [I2S_TX_UPDATE](#).
5. Reset I2S TX unit and TX FIFO.
6. Enable [I2S_TX_DONE_INT](#).
7. Configure I2S GDMA outlink list.

8. Set [I2S_TX_STOP_EN](#).
9. Reset PARLIO RX unit.
10. Configure PARLIO RX unit clock.
11. Turn off PARLIO RX Core clock domain.
12. Configure PARLIO receive mode as sub-mode 4 of Pulse Enable mode. Configure the RX unit data bus width as 1 bit. Configure [PARL_IO_RX_BITLEN](#) according to the length of the data sent by I2S. Set [PARL_IO_RX_REG_UPDATE](#).
13. Configure PARLIO GDMA inlink list.
14. Set [PARL_IO_RX_START](#).
15. Turn on PARLIO RX Core clock domain.
16. Set [I2S_TX_START](#) to start transmitting data.
17. Poll [I2S_TX_DONE_INT](#).
18. Poll GDMA SUC EOF interrupt.
19. Clear [I2S_TX_START](#).
20. Clear [PARL_IO_RX_START](#).

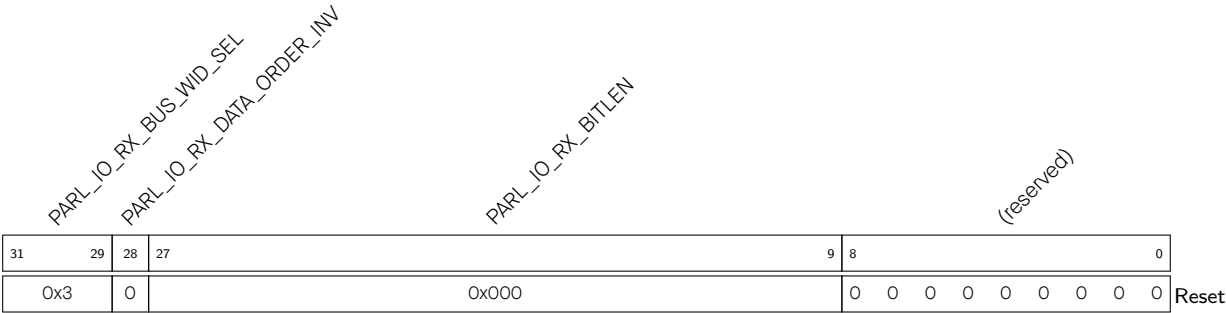
33.9 Register Summary

The addresses in this section are relative to Parallel IO Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
PARLIO RX Configuration Registers			
PARL_IO_RX_MODE_CFG_REG	PARLIO RX sampling mode configuration register	0x0000	R/W
PARL_IO_RX_DATA_CFG_REG	PARLIO RX data configuration register	0x0004	R/W
PARL_IO_RX_GENRL_CFG_REG	PARLIO RX general configuration register	0x0008	R/W
PARL_IO_RX_START_CFG_REG	PARLIO RX start configuration register	0x000C	R/W
PARLIO TX Configuration Registers			
PARL_IO_TX_DATA_CFG_REG	PARLIO TX data configuration register	0x0010	R/W
PARL_IO_TX_START_CFG_REG	PARLIO TX start configuration register	0x0014	R/W
PARL_IO_TX_GENRL_CFG_REG	PARLIO TX general configuration register	0x0018	R/W
PARLIO Configuration and Status Registers			
PARL_IO_FIFO_CFG_REG	PARLIO FIFO configuration register	0x001C	R/W
PARL_IO_REG_UPDATE_REG	PARLIO register update configuration register	0x0020	WT
PARL_IO_ST_REG	PARLIO module status register	0x0024	RO
PARLIO Interrupt Configuration and Status Registers			
PARL_IO_INT_ENA_REG	PARLIO interrupt enable signal configuration register	0x0028	R/W
PARL_IO_INT_RAW_REG	PARLIO interrupt raw signal status register	0x002C	R/SS/WTC
PARL_IO_INT_ST_REG	PARLIO interrupt signal status register	0x0030	RO
PARL_IO_INT_CLR_REG	PARLIO interrupt clear signal configuration register	0x0034	WT
PARLIO RX/TX Status Registers			
PARL_IO_RX_STO_REG	PARLIO RX status register 0	0x0038	RO
PARL_IO_RX_ST1_REG	PARLIO RX status register 1	0x003C	RO
PARL_IO_TX_STO_REG	PARLIO TX status register 0	0x0040	RO
PARLIO Clock Configuration Registers			
PARL_IO_RX_CLK_CFG_REG	PARLIO RX clock configuration register	0x0044	R/W
PARL_IO_TX_CLK_CFG_REG	PARLIO TX clock configuration register	0x0048	R/W
PARL_IO_CLK_REG	PARLIO clock configuration register	0x0120	R/W
PARLIO TX CS Configuration Register			
PARL_IO_TX_CS_CFG_REG	PARLIO TX CS configuration register	0x004C	R/W
PARLIO Version Register			
PARL_IO_VERSION_REG	Version control register	0x03FC	R/W

Register 33.2. PARL_IO_RX_DATA_CFG_REG (0x0004)



- PARL_IO_RX_BITLEN** Configures the expected bit number of RXD. (R/W)
- PARL_IO_RX_DATA_ORDER_INV** Configures whether to invert the bit order of one byte sent from RX FIFO to GDMA.
0: No effect
1: Invert
(R/W)
- PARL_IO_RX_BUS_WID_SEL** Configures the RXD bus width.
0: 1 bit
1: 2 bits
2: 4 bits
3: 8 bits
4 ~ 15: Invalid and will incur error
(R/W)

Register 33.3. PARL_IO_RX_GENRL_CFG_REG (0x0008)

(reserved)				PARL_IO_RX_EOF_GEN_SEL								PARL_IO_RX_TIMEOUT_THRES												PARL_IO_RX_GATING_EN								(reserved)			
31	30	29	28									13	12	11																	0				
0	0	1	0xfff								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset					

PARL_IO_RX_GATING_EN Configures whether to enable the clock gating of the RX output clock.

0: Disable

1: Enable

(R/W)

PARL_IO_RX_TIMEOUT_THRES Configures the threshold of the RX timeout counter. (R/W)

PARL_IO_RX_TIMEOUT_EN Configures whether to enable the timeout function to generate GDMA ERR EOF.

0: Disable

1: Enable

(R/W)

PARL_IO_RX_EOF_GEN_SEL Configures the generation mechanism of GDMA SUC EOF.

0: Generate GDMA SUC EOF by the configured data bit length

1: Generate GDMA SUC EOF by the external enable signal

(R/W)

Register 33.4. PARL_IO_RX_START_CFG_REG (0x000C)

PARL_IO_RX_START																																(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31	30																																																															0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

PARL_IO_RX_START Configures whether to start RX data sampling.

0: No effect

1: Start

(R/W)

Register 33.5. PARL_IO_TX_DATA_CFG_REG (0x0010)

[illegible]

PARL_IO_TX_BITLEN Configures the expected bit number of TXD. (R/W)

PARL_IO_TX_DATA_ORDER_INV Configures whether to invert the bit order of one byte sent from TX FIFO to IO data.

0: No effect

1: Invert

(R/W)

PARL_IO_TX_BUS_WID_SEL Configures the TXD bus width.

O: 1 bit

1: 2 bits

2: 4 bits

3: 8 bits

4 ~ 15: Invalid and will incur error

(R/W)

Register 33.6. PARL_IO_TX_START_CFG_REG (0x0014)

Diagram of the PARL_IO_TX_START register structure:

- Bit 31: PARL_IO_TX_START
- Bit 30: (reserved)
- Bits 29 down to 0: 0
- Reset: Indicated at the bottom right of the register block.

PARL_IO_TX_START Configures whether to start TX data transmission.

0: No effect

1: Start

(R/W)

Register 33.7. PARL_IO_TX_GENRL_CFG_REG (0x0018)

PARL_IO_RX_REG_UPDATE

(reserved)

(WT)

PARL_IO_TX_READY

(reserved)

(RO)

(reserved)

PARL_IO_TX_EOF_INT_ENA
PARL_IO_RX_FIFO_WOVF_INT_ENA
PARL_IO_TX_FIFO_EMPTY_INT_ENA

PARL_IO_TX_EOF_INT_ENA Write 1 to enable **TX_EOF_INT**. (R/W)

Register 33.12. PARL_IO_INT_RAW_REG (0x002C)

[illegible]

PARL_IO_TX_FIFO_EMPTY_INT_RAW The raw interrupt status of **TX_FIFO_EMPTY_INT**.
(R/WTC/SS)

PARL_IO_RX_FIFO_WOVF_INT_RAW The raw interrupt status of [RX_FIFO_WOVF_INT](#). (R/WTC/SS)

PARL_IO_TX_EOF_INT_RAW The raw interrupt status of [TX_EOF_INT](#). (R/WTC/SS)

Register 33.13. PARL_IO_INT_ST_REG (0x0030)

Register 0x00000000: PARL_IO_TX_EMPTY_INT_ST

31																														3	2	1	0
0																														0	0	0	0

Reset

PARL_IO_TX_FIFO_EMPTY_INT_ST The masked interrupt status of [TX_FIFO_EMPTY_INT](#). (RO)

PARL_IO_RX_FIFO_WOVF_INT_ST The masked interrupt status of [RX_FIFO_WOVF_INT](#). (RO)

PARL_IO_TX_EOF_INT_ST The masked interrupt status of **TX_EOF_INT**. (RO)

Register 33.14. PARL_IO_INT_CLR_REG (0x0034)

(reserved)																															PARL_IO_TX_EOF_INT_CLR PARL_IO_RX_FIFO_WOVF_INT_CLR PARL_IO_TX_FIFO_EMPTY_INT_CLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
31																															3	2	1	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PARL_IO_TX_FIFO_EMPTY_INT_CLR Write 1 to clear [TX_FIFO_EMPTY_INT](#). (WT)

PARL_IO_RX_FIFO_WOVF_INT_CLR Write 1 to clear [RX_FIFO_WOVF_INT](#). (WT)

PARL_IO_TX_EOF_INT_CLR Write 1 to clear [TX_EOF_INT](#). (WT)

Register 33.15. PARL_IO_RX_STO_REG (0x0038)

PARL_IO_RX_FIFO_WR_BIT_CNT														PARL_IO_RX_CNT							(reserved)						
31													13	12	8				7	0							
0x000													0x0		0 0 0 0 0 0 0 0 0							Reset					

PARL_IO_RX_CNT Represents the clock cycle number of reading the RX FIFO. (RO)

PARL_IO_RX_FIFO_WR_BIT_CNT Represents the bit number currently written into the RX FIFO. (RO)

Register 33.16. PARL_IO_RX_ST1_REG (0x003C)

PARL_IO_RX_FIFO_RD_BIT_CNT																(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																13																12																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0x000																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0															

PARL_IO_RX_FIFO_RD_BIT_CNT Represents the bit number currently read from the RX FIFO. (RO)

Register 33.17. PARL_IO_TX_STO_REG (0x0040)

PARL_IO_TX_FIFO_RD_BIT_CNT																																	
PARL_IO_TX_CNT										(reserved)																							
31													13			12			6			5			0								
0x000																0x0			0			0			0			0			Reset		

PARL_IO_TX_CNT Represents the cycle number of reading the TX FIFO. (RO)

PARL_IO_TX_FIFO_RD_BIT_CNT Represents the bit number currently read from the TX FIFO. (RO)

Register 33.18. PARL_IO_RX_CLK_CFG_REG (0x0044)

PARL_IO_RX_CLK_O_INV																																		
PARL_IO_RX_CLK_I_INV																																		
(reserved)																																		
31	30	29																													0			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PARL_IO_RX_CLK_I_INV Configures whether to invert the RX input Core clock.

0: No effect

1: Invert

(R/W)

PARL_IO_RX_CLK_O_INV Configures whether to invert the RX output Core clock.

0: No effect

1: Invert

(R/W)

Register 33.19. PARL_IO_TX_CLK_CFG_REG (0x0048)

Diagram of the 32-bit PARL register structure:

- Bit 31: PARL_IO_TX_CLK_O_INV
- Bit 30: PARL_IO_TX_CLK_O_INV
- Bit 29: PARL_IO_TX_CLK_I_INV
- Bits 28-0: (reserved)

PARL_IO_TX_CLK_I_INV Configures whether to invert the TX input Core clock.

0: No effect

1: Invert

(R/W)

PARL_IO_TX_CLK_O_INV Configures whether to invert the TX output Core clock.

0: No effect

1: Invert

(R/W)

Register 33.20. PARL_IO_TX_CS_CFG_REG (0x004C)

PARL_IO_TX_CS_START_DELAY	PARL_IO_TX_CS_STOP_DELAY
0x00	0x00
Reset	

PARL_IO_TX_CS_STOP_DELAY Configures the delay between the end of TX data transmission and the rising edge of TX_CS_0.

Unit: TX Core clock cycle

(R/W)

PARL_IO_TX_CS_START_DELAY Configures the delay between the falling edge of TX_CS_0 and the start of TX data transmission.

Unit: TX Core clock cycle

(R/W)

Register 33.21. PARL_IO_CLK_REG (0x0120)

PARL_IO_CLK_EN																																(reserved)																
31	30																																															0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset															

PARL_IO_CLK_EN Register clock gating.

0: Enable the register clock only when reading or writing registers

1: Always enable the register clock

(R/W)

Register 33.22. PARL_IO_VERSION_REG (0x03FC)

(reserved)				PARL_IO_DATE																									
31	28	27	0																										
0	0	0	0	0x2409230																									Reset

PARL_IO_DATE Version control register. (R/W)

Chapter 34

BitScrambler

The ESP32-C5 has an extensive amount of DMA-capable peripherals (see Chapter [3 GDMA Controller \(GDMA\)](#)). These can move data from memory to an external device, and vice versa, without any interference from the CPU. This only works if the external device needs or emits the data in question in the same format as the software expects it: if not, the CPU needs to rewrite the format of the data. Examples include a need to swap bytes, reverse bytes, and shift the data left or right.

As bitwise operations tend to be fairly CPU-expensive and the purpose of DMA is to not use the CPU in the transfer, ESP32-C5 includes a BitScrambler, which is a dedicated peripheral to change the format of data in between memory and the peripheral. A BitScrambler is capable of performing the aforementioned operations, but as a flexible programmable state machine, it is capable of more advanced things as well. The BitScrambler can be used for memory-to-peripheral, memory-to-memory, and peripheral-to-memory transfers.

34.1 Introduction

In ESP32-C5, a DMA-capable peripheral is one that can read data directly from memory and/or write data directly to memory using a DMA channel. The format this data is read/written in is dependent on the peripheral and generally has only a small amount of flexibility. For most DMA-capable peripherals in ESP32-C5, it is possible to attach the BitScrambler to any of these paths for additional flexibility.

When the BitScrambler is attached to a peripheral, it gets the chance to modify any data that flows down a DMA channel associated with that peripheral. Data written to the DMA channel will appear on the BitScrambler input, and what the BitScrambler writes to its output will flow down the DMA channel. The BitScrambler contains instruction memory where instructions on how to route the data between input and output can be stored. As such, by writing an appropriate BitScrambler program, the way the data is rewritten by the BitScrambler can be controlled.

As shown in Figure [34.1-1](#), the BitScrambler can either be placed in the DMA TX BitScrambler path, where it modifies data flowing from memory to a peripheral, or in the DMA RX BitScrambler path, where it modifies data flowing from a peripheral to memory. It can also be placed in loopback mode, modifying data from memory to memory.

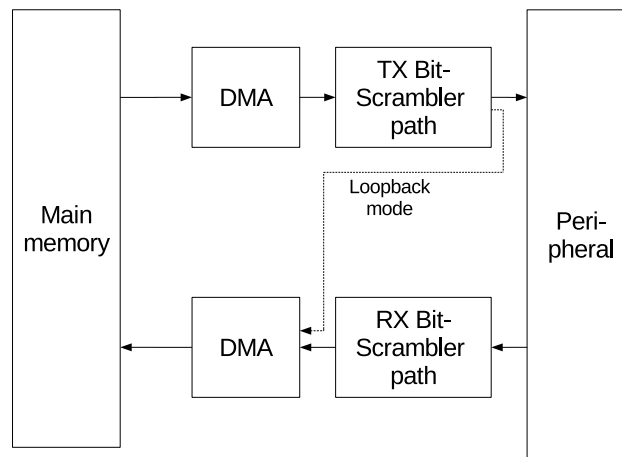


Figure 34.1-1. BitScrambler System Context Diagram

34.2 Feature List

The BitScrambler has the following features:

- One BitScrambler core, which can be used as either a RX (peripheral-to-memory), or TX (memory-to-peripheral) unit
- Support for memory-to-memory transfers
- Processing up to 32 bits per DMA clock period
- Data path controlled by a BitScrambler program stored in instruction memory
- Input registers able to read 0, 8, 16, or 32 bits per clock cycle
- Output registers:
 - Able to write 0, 8, 16, or 32 bits per clock cycle
 - Data sources for output register bits: 64 bits of input data, two counters, LUT RAM data, data output of last cycle, comparators
 - With some restrictions, each of the 32 output register bits can come from any bit on the data sources
- 8 x 257-bit instruction memory, for storing eight instructions, controlling control flow and the data path
- 2048 bytes of lookup table (LUT) memory, configurable as various word widths

34.3 Application Examples

- Swapping byte, nibble or bit orders in data
- Converting palette-based bitmaps to full RGB bitmaps
- Run-length encoding or decoding of data
- Generating complicated waveforms for e.g. WS2811 chips

34.4 Architectural Overview

The BitScrambler can be separated into a data path, a control path, and the registers that the CPU can use to program them. The data path moves bits between the incoming DMA stream, internal registers, and the outgoing DMA stream. The control path parses the instructions in the instruction memory to control the data path, as well as handle program flow.

34.4.1 Data Path

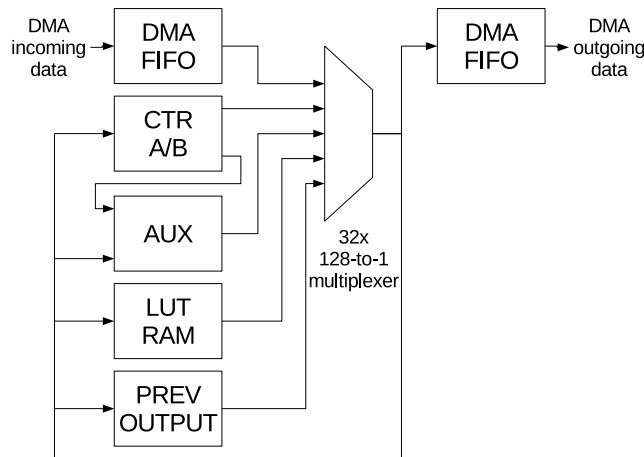


Figure 34.4-1. BitScrambler Data Path Diagram

The data path of the BitScrambler is intended to take data from the incoming DMA stream, process it according to the program code stored in instruction memory, and write it to the outgoing DMA stream.

1. Data is read from the incoming DMA stream into a 64-bit register. At the end of an instruction cycle, as specified in the instruction, N bits (with N being 0, 8, 16, or 32) are read from the DMA stream.
2. The data in the register is then shifted toward the LSB of the register with the least significant N bits disappearing.
3. The read data appears as the N most significant bits.

Optionally, on startup, the BitScrambler will automatically read the first 64 bits into this register.

On the other side of the BitScrambler, data is deposited into a 32-bit output register. At the end of the instruction cycle, depending on what the instruction specifies, the least significant 0, 8, 16, or 32 bits will be written to the outgoing DMA stream.

The BitScrambler derives its name from the fact that it can put input bits in any random position in the output, and this is achieved by 32 x 128-to-1 multiplexers. For each of the 32 output register bits, there is one multiplexer that selects the input signals from one of the data sources, dependent on the output of the BitScrambler control path. These data sources are:

- 64 bits, shifted in from the input FIFO (i.e., the left DMA FIFO block in Figure 34.4-1). As described before, data from the incoming DMA stream is deposited here. In order to facilitate iterating over input bits in a loop, an instruction can enable relative addressing. In this addressing mode, any mux sourcing a bit from the input FIFO register will actually get the bit offset by the counter A register (i.e., CTR A in Figure 34.4-1).

- 32 bits that were sent to the output in the last cycle (i.e., PREV OUTPUT block in Figure 34.4-1). Data sent to the output register will appear on this register one clock cycle later. This allows the BitScrambler to 'remember' data over multiple clock cycles: by outputting a bit to the output register, even if that bit subsequently is not sent to the output FIFO, the next clock cycle can still access it by selecting from this register. Bits can be remembered longer-term by selecting bits from this register for output again, which makes them appear here in the subsequent cycle.
- 8, 16, or 32 bits that are the output of LUT RAM. The address used to select the data is bit 16 to 24, 25 or 26 of the output data of the last cycle, assuming the LUT width is 32, 16 or 8 bits, respectively. The LUT RAM is a part of the BitScrambler that takes an address, looks it up in its memory, and outputs the data located at that specific address. The LUT RAM can be filled when the BitScrambler is initialized, but the BitScrambler itself does not have the ability to modify it. Depending on requirements, the LUT RAM can be initialized in the following modes:
 - 512 x 32-bit words
 - 1024 x 16-bit words
 - 2048 x 8-bit words

The LUT RAM can be used to do translations of data, e.g. using a calibration curve for a given sensor. It can also be used to implement mathematical functions, with the address input split into two or more input variables and the data output used as the output of the function; the memory should be loaded with the output of the function for each of the input values.

- 32 bits from 2 x 16-bit counters (i.e., CTR A/B block in Figure 34.4-1). This register contains two 16-bit counters, named 'A' and 'B'. These can be loaded, incremented and decremented via control path instructions.
- 30 bits from comparators between counter B and the previous data on the output (PREV OUT in Figure 34.4-1; the 30 bits described here are part of the AUX block there). These allow the result of comparisons being used for output or for program flow. This can be useful in e.g. decoding run-length encoded streams, or generating PWM signals.
- 2 bits, one fixed-high, one fixed-low (i.e., part of AUX in Figure 34.4-1). These are useful if a protocol needs an always-high or always-low bit in the output that does not depend on the input data.

Note that some sources cannot be accessed at the same time. Specifically, a given instruction cannot source data from both the high 32 bits of the input FIFO source register as well as the counter registers. When data is read from the LUT, for a LUT that is set to a width of N, the top N bits of both the input FIFO register as well as the top N bits of the counter register become unavailable.

34.4.2 Control Path

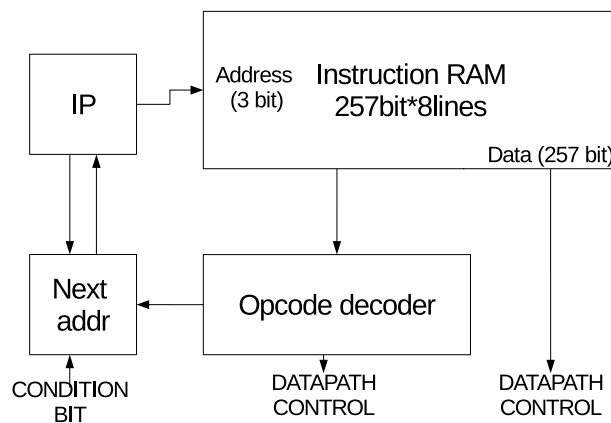


Figure 34.4-2. BitScrambler Control Path Diagram

The BitScrambler behavior is governed by a program stored in instruction RAM. The instruction RAM has space for up to eight instructions, each being 257 bits in size. An instruction configures the data path for the clock cycle the instruction runs in, as well as contains opcodes that control program flow.

A BitScrambler program is similar to a microprocessor program, in that each cycle the BitScrambler will execute the next instruction in instruction memory, unless the opcode tells it to specifically jump to another location. Concretely, this behaviour is implemented using an instruction pointer (IP, see Figure 34.4-2), which points to the currently executing instruction. Opcodes can also affect the counter registers.

The bits of the instruction that configure the data path consist of settings for the 32 muxes, as well as some bits to select the availability of the counter bits or LUT output. It also can put the muxes into relative mode, where if they get a bit from the DMA FIFO, the position of that bit is offset by counter B.

34.5 Functional Description

The BitScrambler is a flexible device, with its behavior controlled by the program loaded into it as well as the configuration registers that set several global BitScrambler behaviors as well. As such, a BitScrambler program consists of the contents of the 257-bit x 8 instruction RAM as well as the settings for various configuration registers. Both are described below.

34.5.1 Instructions

The 257 bits that make up an instruction are structured as shown in Table 34.5-1.

Table 34.5-1. Instruction Structure

Bit	Field	Bit	Field
0 - 6	CTL_MUX_SEL_0	133 - 139	CTL_MUX_SEL_19
7 - 13	CTL_MUX_SEL_1	140 - 146	CTL_MUX_SEL_20
14 - 20	CTL_MUX_SEL_2	147 - 153	CTL_MUX_SEL_21
21 - 27	CTL_MUX_SEL_3	154 - 160	CTL_MUX_SEL_22
28 - 34	CTL_MUX_SEL_4	161 - 167	CTL_MUX_SEL_23
35 - 41	CTL_MUX_SEL_5	168 - 174	CTL_MUX_SEL_24
42 - 48	CTL_MUX_SEL_6	175 - 181	CTL_MUX_SEL_25
49 - 55	CTL_MUX_SEL_7	182 - 188	CTL_MUX_SEL_26
56 - 62	CTL_MUX_SEL_8	189 - 195	CTL_MUX_SEL_27
63 - 69	CTL_MUX_SEL_9	196 - 202	CTL_MUX_SEL_28
70 - 76	CTL_MUX_SEL_10	203 - 209	CTL_MUX_SEL_29
77 - 83	CTL_MUX_SEL_11	210 - 216	CTL_MUX_SEL_30
84 - 90	CTL_MUX_SEL_12	217 - 223	CTL_MUX_SEL_31
91 - 97	CTL_MUX_SEL_13	224 - 249	OPCODE
98 - 104	CTL_MUX_SEL_14	250 - 251	CTL_READ_IN
105 - 111	CTL_MUX_SEL_15	252 - 253	CTL_WR_OUT
112 - 118	CTL_MUX_SEL_16	254	CTL_MUX_REL
119 - 125	CTL_MUX_SEL_17	255	CTL_CTR_SRC_SEL
126 - 132	CTL_MUX_SEL_18	256	CTL_LUT_SRC_SEL

The meanings of these fields are as follows:

- CTL_MUX_SEL_n: This 7-bit field selects the source of bit n in the output register. See Table 34.5-3 for details.
- OPCODE: These bits encode an opcode. For the bit pattern of this field, please refer to Table 34.5-4.
- CTL_READ_IN: This selects the number of bits that are read from the input FIFO at the end of the instruction:
 - 0: Do not read any data
 - 1: Read 8 bits
 - 2: Read 16 bits
 - 3: Read 32 bits
- CTL_WR_OUT: This selects the number of bits that are written from the output register to the output FIFO at the end of the instruction:
 - 0: Do not write any data
 - 1: Write 8 bits
 - 2: Write 16 bits
 - 3: Write 32 bits

- CTL_MUX_REL, CTL_CTR_SRC_SEL and CTR_LUT_SRC_SEL: Select which bits are available to select using the CTL_MUX_SEL bits.
 - CTRL_CTR_SRC_SEL (Control Counter Source Select) makes the bits from counter A and B available
 - CTL_LUT_SRC_SEL (Control LUT Source Select) makes the output value of the LUT RAM available. See Table 34.5-3 for details.
 - When CTL_MUX_REL (Control Mux Relative) is 1, the BitScrambler applies the value of counter A as an offset to any CTL_MUX_SEL_n value that is 63 or below. The exact behaviour depends on CTR_SRC_SEL, as detailed in Table 34.5-2.

Table 34.5-2. Effective CTL_MUX_SEL_n values when CTL_MUX_REL is 1

Condition	Effective value of CTL_MUX_SEL
CTL_MUX_SEL_n >= 64	CTL_MUX_SEL (unchanged)
CTL_MUX_SEL_n < 64 and CTL_CTR_SRC_SEL = 0	(CTL_MUX_SEL_n + CTRA) & 63
CTL_MUX_SEL_n < 31 and CTL_CTR_SRC_SEL = 1	(CTL_MUX_SEL_n + CTRA) & 31
32 <= CTL_MUX_SEL_n < 64 and CTL_CTR_SRC_SEL = 1	((CTL_MUX_SEL_n + CTRA) & 31) + 32

Of these fields, most are data path configuration items. The field that affects the control path is OPCODE, although CTL_MUX_REL, CTL_CTR_SRC_SEL, and CTR_LUT_SRC_SEL may influence how some of the opcodes work by adjusting the source selection for the IF/IFN opcodes.

CTL_MUX_SEL_n: These fields, one for each bit in the output register, select the source for that particular bit. The specific bit chosen is partially dependent on the settings of the CTL_MUX_REL, CTL_CTR_SRC_SEL, and CTR_LUT_SRC_SEL. Given CTL_MUX_SRC_n has value N, the bit is selected as described in Table 34.5-3. Note that the source selection for IF/IFN opcodes is affected in the same way.

Table 34.5-3. CTL_MUX_SEL Values

Bit	Description
0-31	The bit is sourced from bit N in the incoming FIFO register.
32-63	The source of the bit depends on the setting of CTL_CTR_SRC_SEL and CTR_LUT_SRC_SEL: If CTL_CTR_SRC_SEL = 0: the bit is sourced from bit N in the incoming FIFO register. If CTL_CTR_SRC_SEL = 1: the bit is sourced from bit (N-32) of the counter registers. Specifically, if N is 32 to 47, the bit is sourced from bit (N-32) of counter A and if N is 48 to 63, the bit is sourced from bit (N-48) of counter B. If CTL_LUT_SRC_SEL = 1: Depending on the width of the LUT (8, 16, or 32 bits), a read of the top 8, 16, or 32 bits will return data from the LUT rather than what CTL_CTR_SRC_SEL selects. Specifically, given a LUT width of N, 63 will return LUT output bit (N-1), 62 will return LUT output bit (N-2) and so on.
64	CounterB[7:0] =< last[7:0]
65	CounterB[7:0] > last[7:0]
66	CounterB[7:0] = last[7:0]
67	CounterB[7:0] =< last[15:8]
68	CounterB[7:0] > last[15:8]
69	CounterB[7:0] = last[15:8]

Bit	Description
70	CounterB[7:0] =< last[23:16]
71	CounterB[7:0] > last[23:16]
72	CounterB[7:0] = last[23:16]
73	CounterB[7:0] =< last[31:24]
74	CounterB[7:0] > last[31:24]
75	CounterB[7:0] = last[31:24]
76	CounterB[15:8] =< last[7:0]
77	CounterB[15:8] > last[7:0]
78	CounterB[15:8] = last[7:0]
79	CounterB[15:8] =< last[15:8]
80	CounterB[15:8] > last[15:8]
81	CounterB[15:8] = last[15:8]
82	CounterB[15:8] =< last[23:16]
83	CounterB[15:8] > last[23:16]
84	CounterB[15:8] = last[23:16]
85	CounterB[15:8] =< last[31:24]
86	CounterB[15:8] > last[31:24]
87	CounterB[15:8] = last[31:24]
88	CounterB[15:0] =< last[15:0]
89	CounterB[15:0] > last[15:0]
90	CounterB[15:0] = last[15:0]
91	CounterB[15:0] =< last[31:16]
92	CounterB[15:0] > last[31:16]
93	CounterB[15:0] = last[31:16]
94	Always 0
95	Always 1
96-127	The bits selected here are the same as the bits on the output register at the end of the previous cycle.

The BitScrambler recognizes seven opcodes, which are encoded in the opcode fields as shown in Table 34.5-4.

Table 34.5-4. Instruction Opcode

Opcode \ Bit	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOOP	1	c	tgt			end_val																ctr_add				
ADD	0	c	h	l	0	0	0	0	0	0	ctr_add															
IF	0	0	tgt			0	0	0	0	1	0	0	0	0	0	0	0	0	0	src						
IFN	0	0	tgt			0	0	0	1	0	0	0	0	0	0	0	0	0	0	src						
LDCTD	0	c	h	l	0	0	0	0	1	1	ctr_set															
LDCTI	0	c	h	l	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ADDCTI	0	c	h	l	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The fields are defined as follows:

- c: Configures which counter to use.
 - 0: counter A
 - 1: counter B
- end_val, ctr_add, ctr_set: 16-bit (or 5-bit) value. Please refer to the instruction descriptions below for the meanings of these fields.
- tgt: The location of an instruction. Range: 0 ~ 7.
- h: 1 if the upper 8 bits need to be written back to a counter (if the opcode ends with -H).
- l: 1 if the lower 8 bits need to be written back to a counter (if the opcode ends with -L).

Note that if an opcode does not specify H or L, it is assumed that the full 16 bits need to be written back and the h and l bits will both be 1.

These seven opcodes affect the state of the BitScrambler as follows:

Table 34.5-5. LOOP Opcode

LOOPc end_val, ctr_add, tgt	
Loop on counter indicated by c, add ctr_add to the counter, and loop back to tgt until the counter reaches end_val. When that happens, reset the counter and fall through.	
Argument	c: Either A or B, indicates counter to use tgt: 3-bit target instruction addresses end_val: 16-bit value to compare the counter to ctr_add: 5-bit signed value to add to the counter
Pseudo-code	<pre> if ((unsigned)ctrC < end_val) begin IP <= tgt ctr_c <= ctr_c + sign_extend(ctr_add) end else begin IP <= IP + 1 ctr_c <= 0 end </pre>

Table 34.5-6. ADD Opcode

ADDc[H L] ctr_add	
Add immediate value to counter indicated by c, and optionally write back only upper or lower 8 bits.	
Argument	c: Either A or B, indicate counter to use ctr_add: 16-bit value to add to counter If opcode is ADDcH, only most significant 8 bits are written back If opcode is ADDcL, only least-significant 8 bits are written back If opcode is ADDc, all 16 bits will be written back (h=1, l=1).
Pseudo-code	<pre> tmp <= ctr_c + ctr_add if (h) begin ctr_c[15:8] <= tmp[15:8] end if (l) begin ctr_c[7:0] <= tmp[7:0] end IP <= IP + 1 </pre>
Note	The NOP (no operation) pseudo-op is encoded as 'ADDA 0'.

Table 34.5-7. IF Opcode

IF ctl_cond_src, tgt	
Jump to target if the indicated mux source bit is set.	
Argument	ctl_cond_src: Mux source (see CTR_MUX_SRC sources) tgt: 3-bit target instruction addresses
Pseudo-code	<pre> if (ctl_cond_src) begin IP <= tgt end else begin IP <= IP + 1 end </pre>
Note	The JMP (always jump) pseudo-op is implemented as an 'IF 95,tgt' opcode, using the always-1 bit from the AUX bits.

Table 34.5-8. IFN Opcode

IFN ctl_cond_src, tgt	
Jump to target if the indicated mux source bit is NOT set (note: same as IF but condition is negated).	
Argument	ctl_cond_src: Mux source (see CTR_MUX_SRC sources) tgt: 3-bit target instruction addresses
Pseudo-code	<pre> if (ctl_cond_src) begin IP <= IP + 1 end else begin IP <= tgt end </pre>

Table 34.5-9. LDCTD Opcode

LDCTDc[H L] ctr_set	
Load counter direct (from immediate), and optionally only load high or low 8 bits of counter.	
Argument	c: Either A or B, indicate counter to use ctr_set: 16-bit value to set the counter to h: if opcode is LDCTDcH, only most significant 8 bits are set l: if opcode is LDCTDcL, only least-significant 8 bits are set If opcode is LDCTDc, all 16 bits are set (h=1, l=1)
Pseudo-code	<pre> if (h) begin ctr_c[15:8] <= ctr_set[15:8] end if (l) begin ctr_c[7:0] <= ctr_set[7:0] end IP <= IP + 1 </pre>

Table 34.5-10. LDCTI Opcode

LDCTIc[H L]	
Load counter indirect (from most significant 16 bits of mux output), and optionally only load high or low 8 bits of counter.	
Argument	c: Either A or B, indicate counter to use h: If opcode is LDCTIcH, only the most significant 8 bits are set l: If opcode is LDCTIcL, only the least significant 8 bits are set If opcode is LDCTIc, all 16 bits are set (h=1, l=1)
Pseudo-code	<pre> if (h) begin ctr_c[15:8] <= mux_out[31:24] end if (l) begin ctr_c[7:0] <= mux_out[23:16] end IP <= IP + 1 </pre>

Table 34.5-11. ADDCTI Opcode

ADDCTIc[H L]	
Add indirect (from most significant 16 bits of mux output) value to counter. Optionally only add to high or low 8 bits of counter.	
Argument	c: Either A or B, indicate counter to use h: If opcode is ADDCTIcH, bit 24 to 31 of the output mux are added to the most significant 8 bits of the counter l: If opcode is ADDCTIcL, bit 16 to 23 of the output mux are added to the least significant 8 bits of the counter If opcode is ADDCTIc, bit 16 to 31 of the output mux are added to all 16 bits of the counter (h=1, l=1)
Pseudo-code	<pre> if (h) begin ctr_c[15:8] <= ctr_c[15:8] + mux_out[31:24] end else if (l) begin ctr_c[7:0] <= ctr_c[7:0] + mux_out[23:16] end else begin ctr_c[15:0] <= ctr_c[15:0] + mux_out[31:16] end IP <= IP + 1 </pre>

34.5.2 Configuration Registers

The BitScrambler is configured using a set of registers. Specifically, these registers are used to write a program into BitScrambler, write its LUT RAM, configure its behaviour, and start and stop BitScrambler operation. They are also used to attach a BitScrambler to a peripheral DMA path.

The ESP32-C5 only has one BitScrambler core which can either be placed in the TX path (where data is sent from the memory to a peripheral) or in a RX path (where data is sent from the peripheral to memory). For compatibility with other ESP series chips, when the BitScrambler core is placed in the TX path, it is controlled with the registers prefixed by BITSCRAMBLER_TX_ while when it is placed in the RX path, the registers prefixed by BITSCRAMBLER_RX_ govern its behaviour. Which path is active is decided by [BITSCRAMBLER_RX_ENA](#) and [BITSCRAMBLER_TX_ENA](#), of which only one is allowed to be active at any given time.

Note:

For convenience, in this section we will refer to the TX path registers only; to get the RX path register descriptions, simply swap the prefixes.

When modifying the input or output of a peripheral, the BitScrambler needs to be attached to the DMA path of that peripheral. To do this, write the appropriate value for the peripheral into the HP_SYSTEM_BITSCRAMBLER_PERI_TX_SEL or HP_SYSTEM_BITSCRAMBLER_PERI_RX_SEL field of the HP_SYSTEM_BITSCRAMBLER_PERI_SEL_REG register.

The BitScrambler can also run in memory-to-memory mode. For this, the [BITSCRAMBLER_LOOP_MODE](#) bit in [BITSCRAMBLER_SYS_REG](#) needs to be set to one. In this mode, the TX BitScrambler is put into loopback mode while the RX BitScrambler is unused and unavailable. The BitScrambler still needs to be attached to a peripheral (although the peripheral does not need to be configured); the DMA path for this peripheral will be

unusable.

Neither the program memory nor the LUT RAM are accessible to the main processor directly. Rather, they need to be written indirectly, by setting an address in one register and then writing the data in a second register.

Specifically, for the LUT RAM, the address is written in the [BITSCRAMBLER_TX_LUT_IDX](#) field of [BITSCRAMBLER_TX_LUT_CFG0_REG](#) and the data is written to or read from [BITSCRAMBLER_TX_LUT_CFG1_REG](#). The width of the LUT, as visible to the BitScrambler, can be configured using the [BITSCRAMBLER_TX_LUT_MODE](#) field. Note that when the host processor writes to the LUT in the aforementioned fashion, the addressing and data size needs to adhere to the configured LUT width.

Similarly, the instruction memory is written by setting the address up in [BITSCRAMBLER_TX_INST_CFG0_REG](#) and writing the data to or reading it from [BITSCRAMBLER_TX_INST_CFG1_REG](#). As the BitScrambler instructions are 257 bits, one instruction needs to be written as 9 32-bit words (with the 257th bit being in the LSB of the 9th word). To achieve that purpose, the [BITSCRAMBLER_TX_INST_CFG0_REG](#) register is divided into two fields. The position of the instruction is written into the [BITSCRAMBLER_TX_INST_IDX](#) field while the word offset within the instruction is written to [BITSCRAMBLER_TX_INST_POS](#).

As the BitScrambler has the ability to generate more or less data than it gets on the input, a BitScrambler-controlled DMA stream can end in one or two ways. The first is that the receiver (either memory in case of the RX BitScrambler, or the peripheral in case of the TX BitScrambler) stops accepting data because it is configured to only receive a limited number of bytes. This case is pretty simple: the BitScrambler will simply halt as it cannot write any more data. The other way is that the transmitter (the peripheral for the RX BitScrambler and the memory for the TX BitScrambler) stops sending data. This is a more complicated situation, as the BitScrambler may or may not be processing some data that needs to be written to the output.

In order to allow the BitScrambler to send out the data it is still processing, two methods have been devised to handle a certain amount of data after the input data stream has ended. Both depend on setting [BITSCRAMBLER_RX_TAILING_BITS_REG](#) to a certain value N:

- EOF-on-N-reads: After the input data stream ends, the BitScrambler reads N dummy bytes from the input. The value of these bytes is zero. After the Nth byte is read, the BitScrambler halts and the DMA stream ends.
- EOF-on-N-writes: After the input data stream ends, the BitScrambler is allowed to write N more bytes to the output. During this time, any reads on the input stream return zero-value dummy bytes. After the N'th byte is written, the BitScrambler halts and the DMA stream ends.

To configure either mode, set the option bits as follows:

Table 34.5-12. Settings for EOF modes

Register	EOF-on-N-reads	EOF-on-N-writes
BITSCRAMBLER_TX_RD_DUMMY	1	1
BITSCRAMBLER_TX_FETCH_MODE	0	0
BITSCRAMBLER_TX_EOF_MODE	1	0
BITSCRAMBLER_TX_HALT_MODE	1	1

34.6 Programming Procedures

Note that these instructions are written for a BitScrambler with the TX path enabled, either in the memory-to-peripheral path, or memory-to-memory path if in loopback mode. For the peripheral-to-memory path, the RX path is used. Unless otherwise indicated, you can exchange "TX" for "RX" in these instructions to configure the RX path.

1. Attach the BitScrambler to a peripheral by configuring `HP_SYSTEM_BITSCRAMBLER_PERI_TX_SEL` field in `HP_SYSTEM_BITSCRAMBLER_PERI_SEL_REG` register. Note that this step is required even in loopback mode.
2. Enable the BitScrambler by setting the `BITSCRAMBLER_TX_ENA` field in the `BITSCRAMBLER_TX_CTRL_REG` register. Make sure only one of `BITSCRAMBLER_TX_ENA` and `BITSCRAMBLER_RX_ENA` is set at any time.
3. Configure loopback mode if required. If loopback mode is needed, set `BITSCRAMBLER_LOOP_MODE` in the `BITSCRAMBLER_SYS_REG` register. Otherwise, clear it.
4. Load program into BitScrambler. For each instruction and each 32-bit word within the instruction, set the `BITSCRAMBLER_TX_INST_IDX` and `BITSCRAMBLER_TX_INST_POS` fields in the `BITSCRAMBLER_TX_INST_CFG0_REG` register, then write the word in the `BITSCRAMBLER_TX_INST_CFG1_REG` register. Note that the 257th bit is written as bit 0 into `BITSCRAMBLER_TX_INST_CFG1_REG`.
5. Load LUT into BitScrambler, if needed:
 - (a) Configure `BITSCRAMBLER_TX_LUT_MODE` in `BITSCRAMBLER_TX_LUT_CFG0_REG` to set the LUT width
 - (b) Write the address to the `BITSCRAMBLER_TX_LUT_CFG0_REG`
 - (c) Write the data word or the byte itself to `BITSCRAMBLER_TX_LUT_CFG1_REG`

Note that it is acceptable to change the LUT width after writing the data and before starting the BitScrambler.

6. Configure BitScrambler. Dependent on what the BitScrambler program expects, you may need to set or clear the various bits in the `BITSCRAMBLER_TX_CTRL_REG` register. You may also need to set `BITSCRAMBLER_TX_TAILING_BITS_REG` to an applicable value.
7. Start the BitScrambler by clearing the `BITSCRAMBLER_TX_HALT` bit in the `BITSCRAMBLER_TX_CTRL_REG` register.
8. Start DMA transaction. Please refer to Chapter 3 *GDMA Controller (GDMA)* for more information about DMA subsystem and refer to the peripheral chapter if loopback mode is not used.
9. Wait until DMA transaction is done.
10. Halt the BitScrambler by setting the `BITSCRAMBLER_TX_HALT` in the `BITSCRAMBLER_TX_CTRL_REG` register.
11. Wait for halt state. The BitScrambler is halted when the `BITSCRAMBLER_TX_IN_IDLE` bit in the `BITSCRAMBLER_TX_STATE_REG` register is set.

12. Reset the FIFO by writing 1 and then 0 to the [BITSCRAMBLER_TX_FIFO_RST](#) bit in the [BITSCRAMBLER_TX_CTRL_REG](#) register.
13. The BitScrambler is ready for a new transaction with the current program and LUT configuration. Simply start from step 7 to do this.

34.7 Register Summary

The addresses in this section are relative to the Bit-scrambler base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

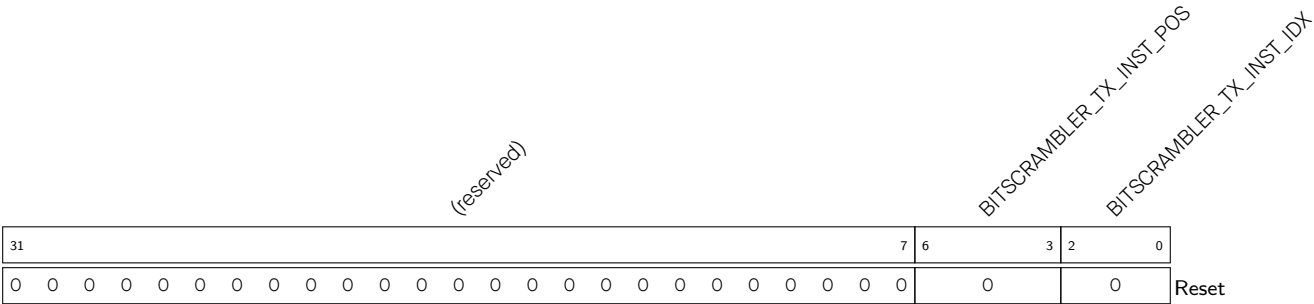
Name	Description	Address	Access
Control and configuration registers			
BITSRAMBLER_TX_INST_CFG0_REG	BitScrambler TX path instruction memory address selection register	0x0000	R/W
BITSRAMBLER_TX_INST_CFG1_REG	BitScrambler TX path instruction memory access register	0x0004	R/W
BITSRAMBLER_RX_INST_CFG0_REG	BitScrambler RX path instruction memory address selection register	0x0008	R/W
BITSRAMBLER_RX_INST_CFG1_REG	BitScrambler RX path instruction memory access register	0x000C	R/W
BITSRAMBLER_TX_LUT_CFG0_REG	BitScrambler TX path LUT memory address selection register	0x0010	R/W
BITSRAMBLER_TX_LUT_CFG1_REG	BitScrambler TX path LUT memory access register	0x0014	R/W
BITSRAMBLER_RX_LUT_CFG0_REG	BitScrambler RX path LUT memory address selection register	0x0018	R/W
BITSRAMBLER_RX_LUT_CFG1_REG	BitScrambler RX path LUT memory access register	0x001C	R/W
Configuration registers			
BITSRAMBLER_TX_TAILING_BITS_REG	BitScrambler TX path extra data length register	0x0020	R/W
BITSRAMBLER_RX_TAILING_BITS_REG	BitScrambler RX path extra data length register	0x0024	R/W
BITSRAMBLER_TX_CTRL_REG	BitScrambler TX path control register	0x0028	varies
BITSRAMBLER_RX_CTRL_REG	BitScrambler RX path control register	0x002C	varies
BITSRAMBLER_SYS_REG	Loopback control register	0x00F8	R/W
Status registers			
BITSRAMBLER_TX_STATE_REG	BitScrambler TX path status register	0x0030	varies
BITSRAMBLER_RX_STATE_REG	BitScrambler RX path status register	0x0034	varies
Version register			
BITSRAMBLER_VERSION_REG	Version control register	0x00FC	R/W

34.8 Registers

The addresses in this section are relative to the Bit-scrambler base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section [Programming Reserved Register Field](#).

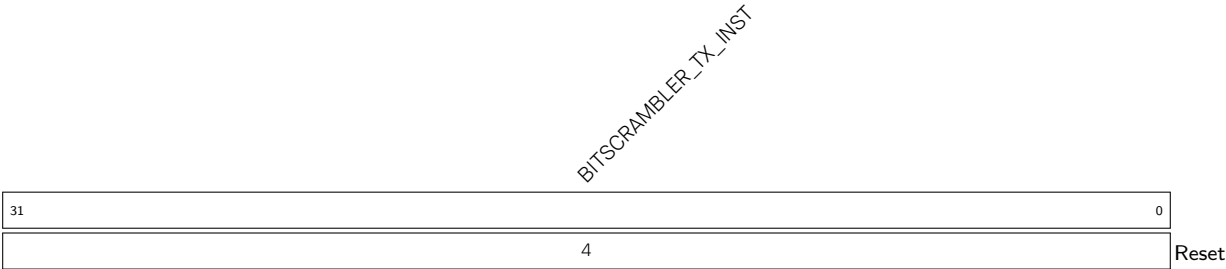
Register 34.1. BITSCRAMBLER_TX_INST_CFG0_REG (0x0000)



BITSCRAMBLER_TX_INST_IDX Configures the index of the BitScrambler instruction to be accessed via [BITSCRAMBLER_TX_INST_CFG1_REG](#). (R/W)

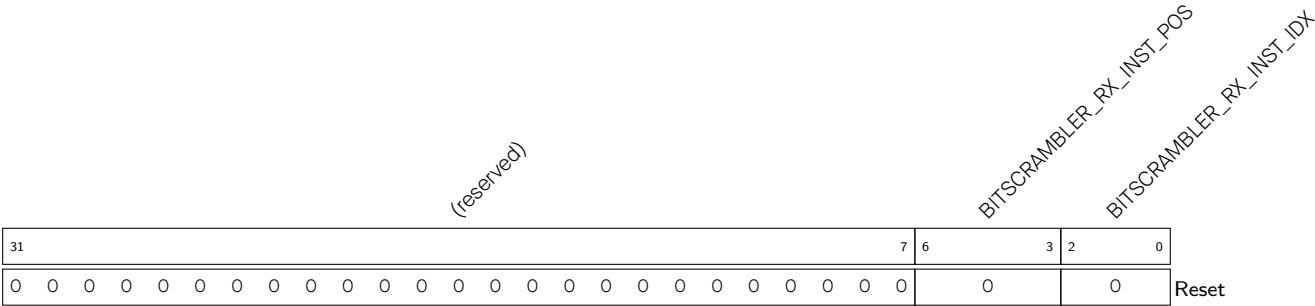
BITSCRAMBLER_TX_INST_POS Configures the offset into the 257-bit BitScrambler instruction (in 32-bit increments) to be accessed via [BITSCRAMBLER_TX_INST_CFG1_REG](#). (R/W)

Register 34.2. BITSCRAMBLER_TX_INST_CFG1_REG (0x0004)



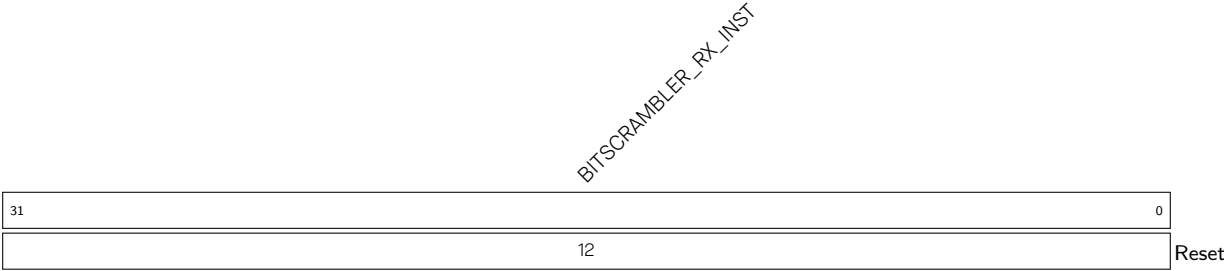
BITSCRAMBLER_TX_INST Configures the instruction to be accessed at the address specified by [BITSCRAMBLER_TX_INST_CFG0_REG](#). (R/W)

Register 34.3. BITSCRAMBLER_RX_INST_CFG0_REG (0x0008)



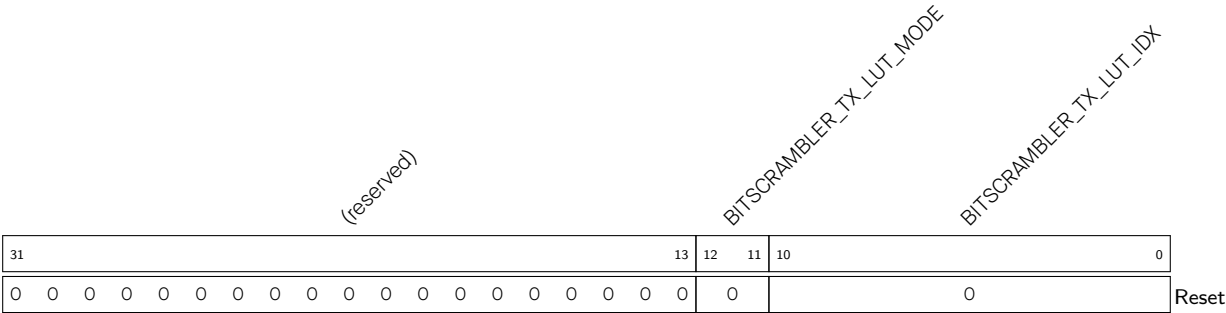
- BITSCRAMBLER_RX_INST_IDX** Configures the index of the BitScrambler instruction to be accessed via [BITSCRAMBLER_RX_INST_CFG1_REG](#). (R/W)
- BITSCRAMBLER_RX_INST_POS** Configures the offset into the 257-bit BitScrambler instruction (in 32-bit increments) to be accessed via [BITSCRAMBLER_RX_INST_CFG1_REG](#). (R/W)

Register 34.4. BITSCRAMBLER_RX_INST_CFG1_REG (0x000C)



- BITSCRAMBLER_RX_INST** Configures the instruction to be accessed at the address specified by [BITSCRAMBLER_RX_INST_CFG0_REG](#). (R/W)

Register 34.5. BITSCRAMBLER_TX_LUT_CFG0_REG (0x0010)



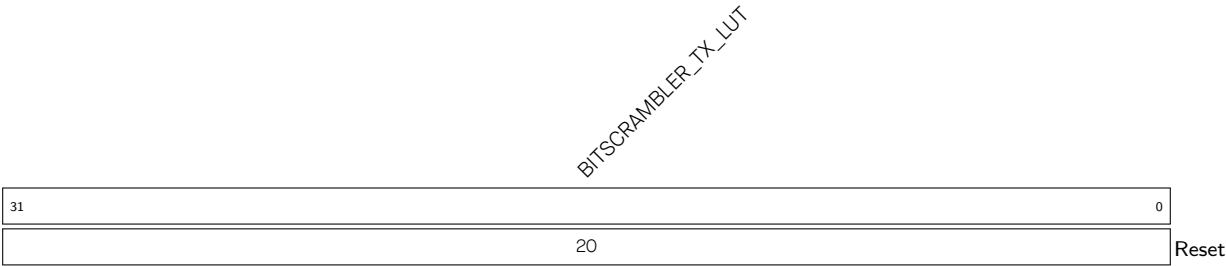
BITSCRAMBLER_TX_LUT_IDX Configures where in LUT RAM to access. Measurement unit: word size configured by [BITSCRAMBLER_TX_LUT_MODE](#). (R/W)

BITSCRAMBLER_TX_LUT_MODE Configures the word size of LUT RAM for BitScrambler programs.

- 0: 1 byte
- 1: 2 bytes
- 2: 4 bytes
- 3: Reserved

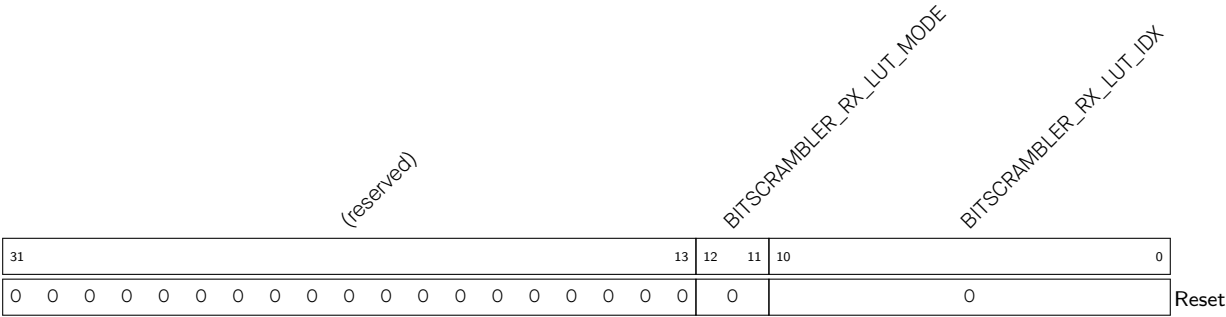
(R/W)

Register 34.6. BITSCRAMBLER_TX_LUT_CFG1_REG (0x0014)



BITSCRAMBLER_TX_LUT Configures the LUT entry to be accessed at the position specified by [BITSCRAMBLER_TX_LUT_CFG0_REG](#). (R/W)

Register 34.7. BITSCRAMBLER_RX_LUT_CFG0_REG (0x0018)



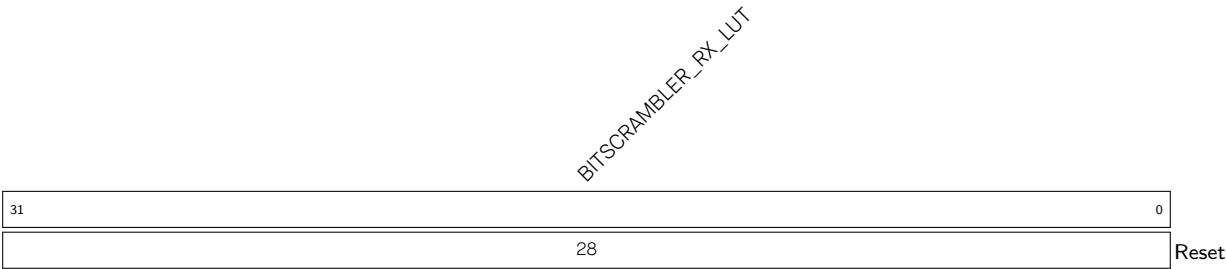
BITSCRAMBLER_RX_LUT_IDX Configures where in LUT RAM to access. Measurement unit: word size configured by [BITSCRAMBLER_RX_LUT_MODE](#). (R/W)

BITSCRAMBLER_RX_LUT_MODE Configures the word size of LUT RAM for BitScrambler programs.

- 0: 1 byte
- 1: 2 bytes
- 2: 4 bytes
- 3: Reserved

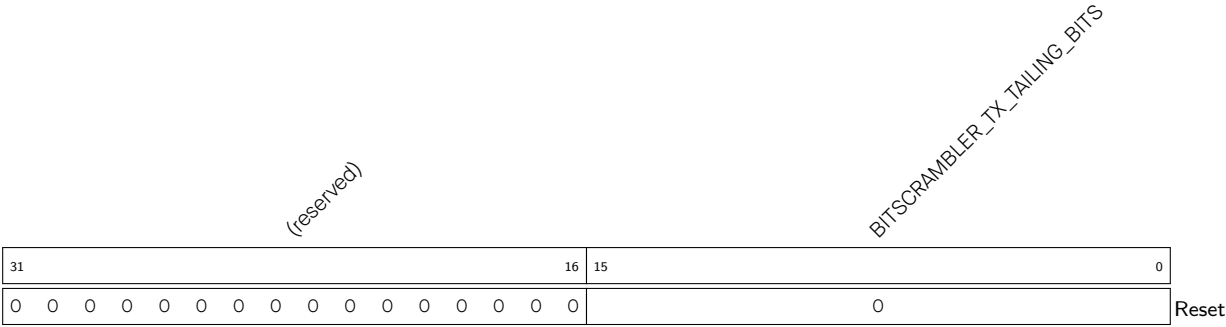
(R/W)

Register 34.8. BITSCRAMBLER_RX_LUT_CFG1_REG (0x001C)



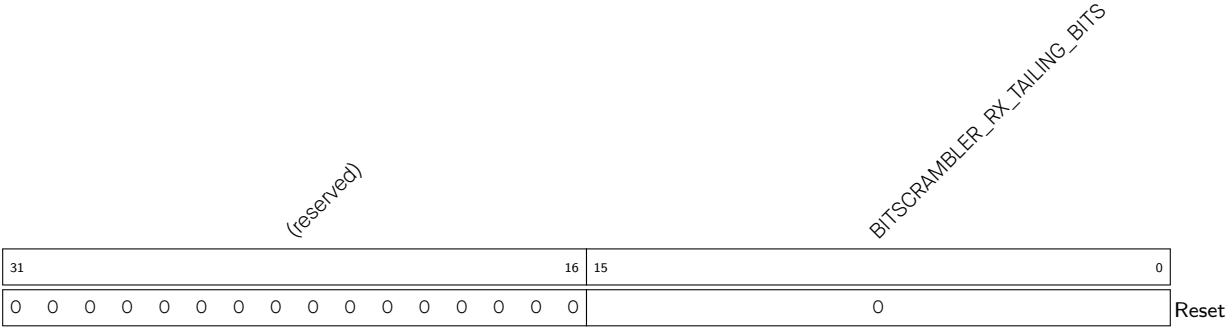
BITSCRAMBLER_RX_LUT Configures the LUT entry to be accessed at the position specified by [BITSCRAMBLER_RX_LUT_CFG0_REG](#). (R/W)

Register 34.9. BITSCRAMBLER_TX_TAILING_BITS_REG (0x0020)



BITSCRAMBLER_TX_TAILING_BITS Configures the length of extra data after getting EOF for the TX BitScrambler path. Measurement unit: bit. (R/W)

Register 34.10. BITSCRAMBLER_RX_TAILING_BITS_REG (0x0024)



BITSCRAMBLER_RX_TAILING_BITS Configures the length of extra data after getting EOF for the BitScrambler RX path. Measurement unit: bit. (R/W)

Register 34.11. BITSCRAMBLER_TX_CTRL_REG (0x0028)

(reserved)																BITSCRAMBLER_TX_FIFO_RST BITSCRAMBLER_TX_RD_DUMMY BITSCRAMBLER_TX_HALT_MODE BITSCRAMBLER_TX_FETCH_MODE BITSCRAMBLER_TX_COND_MODE BITSCRAMBLER_TX_EOF_MODE BITSCRAMBLER_TX_PAUSE BITSCRAMBLER_TX_ENA			
31									9	8	7	6	5	4	3	2	1	0	Reset
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	

BITSCRAMBLER_TX_ENA Configures whether to enable the BitScrambler core in the TX path.

0: Disable

1: Enable

Note that the BitScrambler core can only be enabled either in the RX or TX path, never both.

(R/W)

BITSCRAMBLER_TX_PAUSE Configures whether to pause BitScrambler TX path. A paused core can be un-paused to resume execution.

0: Not pause

1: Pause

(R/W)

BITSCRAMBLER_TX_HALT Configures whether to halt BitScrambler TX path. Halting the BitScrambler TX path will flush the FIFOs and cannot be resumed; a FIFO reset and a restart is needed.

0: Not halt

1: Halt (R/W)

BITSCRAMBLER_TX_EOF_MODE Configures BitScrambler TX path EOF signal generating mode.

This is combined with [BITSCRAMBLER_TX_TAILING_BITS](#) for use.

0: Data written to the output FIFO are counted to generate delayed EOF

1: Data read from the input FIFO are counted to generate delayed EOF

(R/W)

BITSCRAMBLER_TX_COND_MODE Configures BitScrambler TX LOOP instruction condition mode.

0: Use the less than operator to get the condition

1: Use not equal operator to get the condition

(R/W)

BITSCRAMBLER_TX_FETCH_MODE Configures BitScrambler TX path fetch instruction mode.

0: Prefetch by reset

1: Fetch by instructions

(R/W)

BITSCRAMBLER_TX_HALT_MODE Configures BitScrambler TX path halt mode when [BITSCRAMBLER_TX_HALT](#) is set to 1.

0: Wait for write data back done

1: Ignore write data back

(R/W)

Continued on the next page...

Register 34.11. BITSCRAMBLER_TX_CTRL_REG (0x0028)

Continued from the previous page...

BITSCRAMBLER_TX_RD_DUMMY Configures BitScrambler TX path read data mode when EOF received.

0: Wait read data

1: Ignore read data

(R/W)

BITSCRAMBLER_TX_FIFO_RST Configures whether to reset BitScrambler TX FIFO.

0: Not reset

1: Reset

(WT)

Register 34.12. BITSCRAMBLER_RX_CTRL_REG (0x002C)

(reserved)																								BITSCRAMBLER_RX_FIFO_RST BITSCRAMBLER_RX_RD_DUMMY BITSCRAMBLER_RX_HALT_MODE BITSCRAMBLER_RX_FETCH_MODE BITSCRAMBLER_RX_COND_MODE BITSCRAMBLER_RX_EOF_MODE BITSCRAMBLER_RX_PAUSE BITSCRAMBLER_RX_ENA																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																								9	8	7	6	5	4	3	2	1	0	Reset																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BITSCRAMBLER_RX_ENA Configures whether to enable the BitScrambler core in the RX path.

0: Disable

1: Enable

Note that the BitScrambler core can only be enabled either in the RX or TX path, never both.

(R/W)

BITSCRAMBLER_RX_PAUSE Configures whether to pause BitScrambler RX path. A paused core can be un-paused to resume execution.

0: Not pause

1: Pause

(R/W)

BITSCRAMBLER_RX_HALT Configures whether to halt BitScrambler RX path. Halting the BitScrambler RX path will flush the FIFOs and cannot be resumed; a FIFO reset and a restart is needed.

0: Not halt

1: Halt

(R/W)

BITSCRAMBLER_RX_EOF_MODE Configures BitScrambler RX path EOF signal generating mode.

This is combined with [BITSCRAMBLER_RX_TAILING_BITS](#) for use.

0: Data written to the output FIFO are counted to generate delayed EOF

1: Data read from the input FIFO are counted to generate delayed EOF

(R/W)

BITSCRAMBLER_RX_COND_MODE Configures BitScrambler RX LOOP instruction condition mode.

0: Use the less than operator to get the condition

1: Use not equal operator to get the condition

(R/W)

BITSCRAMBLER_RX_FETCH_MODE Configures BitScrambler RX path fetch instruction mode

0: Prefetch by reset

1: Fetch by instructions

(R/W)

Continued on the next page...

Register 34.12. BITSCRAMBLER_RX_CTRL_REG (0x002C)

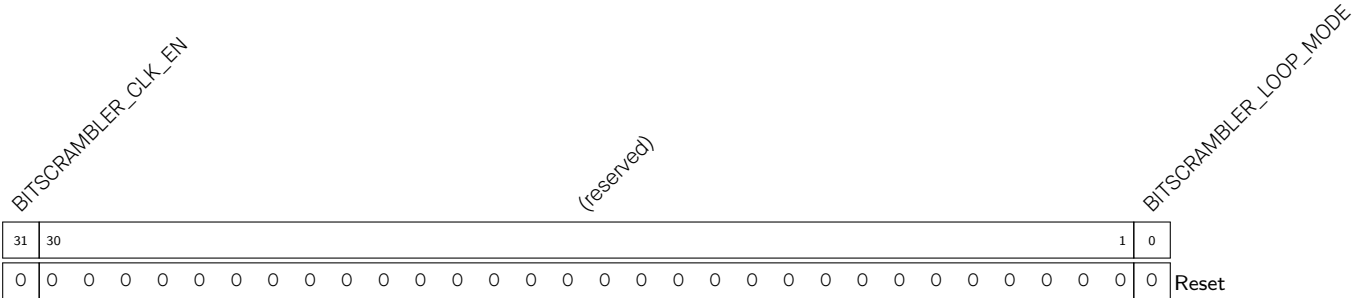
Continued from the previous page...

BITSCRAMBLER_RX_HALT_MODE Configures BitScrambler RX path halt mode when **BITSCRAMBLER_RX_HALT** is set to 1.
0: Wait for write data back done
1: Ignore write data back
(R/W)

BITSCRAMBLER_RX_RD_DUMMY Configures BitScrambler RX path read data mode when EOF received.
0: Wait read data
1: Ignore read data
(R/W)

BITSCRAMBLER_RX_FIFO_RST Configures whether to reset BitScrambler RX FIFO.
0: Not reset
1: Reset
(WT)

Register 34.13. BITSCRAMBLER_SYS_REG (0x00F8)



BITSCRAMBLER_LOOP_MODE Configures whether to enable BitScrambler TX to DMA RX loopback mode
0: Disable
1: Enable
(R/W)

BITSCRAMBLER_CLK_EN Configure whether to enable the configuration register clock to be always-on.
0: turn on during access
1: always-on
(R/W)

Register 34.14. BITSCRAMBLER_TX_STATE_REG (0x0030)

31	30	29	16	15	5	4	3	2	1	0	
0	0	0	0	0	0	1	0	0	0	1	Reset

BITSCRAMBLER_TX_IN_IDLE Represents whether BitScrambler TX path is halted.

0: Not halted

1: Halted

(RO)

BITSCRAMBLER_TX_IN_RUN Represents whether BitScrambler TX path is running.

0: Not running

1: Running

(RO)

BITSCRAMBLER_TX_IN_WAIT Represents whether BitScrambler TX path is waiting for write back done.

0: Not waiting

1: Waiting

(RO)

BITSCRAMBLER_TX_IN_PAUSE Represents whether BitScrambler TX path is paused.

0: Not paused

1: Paused

(RO)

BITSCRAMBLER_TX_FIFO_EMPTY Represents whether BitScrambler TX FIFO is empty

0: Not empty

1: Empty

(RO)

BITSCRAMBLER_TX_EOF_GET_CNT	Represents byte count of BitScrambler TX path after EOF is received. (RO)

BITSCRAMBLER_TX_EOF_OVERLOAD Represents whether BitScrambler TX path tries to process more than one EOF.

0: Not try to process more than one EOF

1: Try to process more than one EOF

(RO)

BITSCRAMBLER_TX_EOF_TRACE_CLR Configures whether to clear **BITSCRAMBLER_TX_EOF_OVERLOAD** and **BITSCRAMBLER_TX_EOF_GET_CNT**.

0: Not clear

1: Clear

(WT)

Register 34.15. BITSCRAMBLER_RX_STATE_REG (0x0034)

[illegible]

BITSCRAMBLER_RX_IN_IDLE Represents whether BitScrambler TX path is halted.

0: Not halted

1: Halted

(RO)

BITSCRAMBLER_RX_IN_RUN Represents whether BitScrambler TX path is running.

0: Not running

1: Running

(RO)

BITSCRAMBLER_RX_IN_WAIT Represents whether BitScrambler TX path is waiting for write back done.

0: Not waiting

1: Waiting

(RO)

BITSCRAMBLER_RX_IN_PAUSE Represents whether BitScrambler TX path is paused.

0: Not paused

1: Paused

(RO)

BITSCRAMBLER_RX_FIFO_FULL Represents whether BitScrambler RX FIFO is full.

0: Not full

1: Full

(RO)

BITSCRAMBLER_RX_EOF_GET_CNT	Represents byte count of BitScrambler TX path after EOF is received. (RO)

SCRAMBLER_RX_EOF_OVERLOAD Represents whether BitScrambler RX path tries to process more than one EOF.

0: Not try to process more than one EOF

1: Try to process more than one EOF

(RO)

BITSCRAMBLER_RX_EOF_TRACE_CLR Configures whether to clear **BITSCRAMBLER_RX_EOF_OVERFLOW** and **BITSCRAMBLER_RX_EOF_GET_CNT**.

0: Not clear

1: Clear

(WT)

Register 34.16. BITSCRAMBLER_VERSION_REG (0x00FC)



BITSCRAMBLER_BITSCRAMBLER_VER Version control register. (R/W)

Part VI

Analog Signal Processing

This part describes components related to analog-to-digital conversion, voltage comparison, on-chip sensors, and features such as temperature sensing, demonstrating the system's capabilities in handling analog signals.

Chapter 35

Temperature Sensor

35.1 Overview

ESP32-C5 provides a temperature sensor to monitor temperature changes inside the chip in real time. The sensor converts analog voltage to digital values and supports compensation for the temperature offset.

35.2 Features

The temperature sensor has the following features:

- Software-triggered temperature measurement. Once triggered, the sensor continuously measures temperature. Software can read the data any time.
- Hardware-triggered automatic temperature monitoring
- Two modes for automatic monitoring of temperature and support for triggering interrupts
- Configurable temperature offset based on the application scenario for improved accuracy
- Configurable temperature measurement range
- Support for several Event Task Matrix (ETM) related events and tasks

35.3 Architecture

Figure [35.3-1](#) shows the internal structure of the temperature sensor.

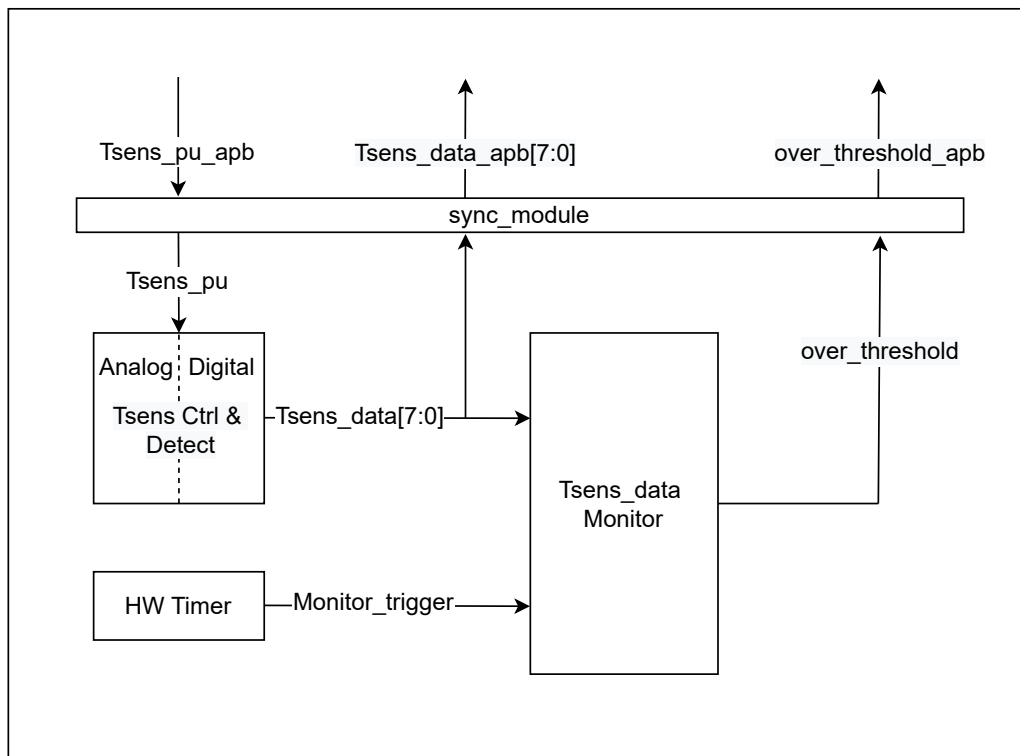


Figure 35.3-1. Temperature Sensor Architecture

As Figure 35.3-1 shows, the temperature sensor module contains the following major blocks:

- Tsens Ctrl & Detect: temperature sensor
- HW Timer: triggers automatic temperature monitoring
- Tsens_data Monitor: monitors whether the temperature is outside the threshold range
- sync_module: Synchronization block between APB clock domain and temperature sensor clock domain

35.4 Functional Description

35.4.1 Temperature Sensor Power Up

The temperature sensor can be powered up by setting the [APB_SARADC_TSENS_PU](#) register.

35.4.2 Temperature Sensor Clock

The temperature sensor has two clock sources: RC_FAST_CLK and XTAL_CLK, selected by [PCR_TSENS_CLK_SEL](#). The clock can be divided with [APB_SARADC_TSENS_CLK_DIV](#).

35.4.3 Automatic Temperature Monitoring Modes

Two modes for the automatic temperature monitoring are available for selection by configuring [APB_SARADC_WAKEUP_MODE](#):

- Absolute value mode:

- Monitors the absolute value of the current temperature. Configure [APB_SARADC_WAKEUP_TH_LOW](#) and [APB_SARADC_WAKEUP_TH_HIGH](#) to set the temperature thresholds. An interrupt will be triggered if the sampled value is above the high threshold or below the low threshold.
- Change value mode:
 - Monitors the temperature changes inside the chip. If the temperature increment of two consecutive samplings exceeds the high threshold configured in [APB_SARADC_WAKEUP_TH_HIGH](#), or the temperature decrement of two consecutive samplings exceeds the low threshold configured in [APB_SARADC_WAKEUP_TH_LOW](#), an interrupt will be triggered. For example, when [APB_SARADC_WAKEUP_TH_LOW](#) is configured as 8, if two consecutive sampling values are 28 and 19 respectively, i.e., the temperature decrement is 9, then an interrupt will be triggered.

35.4.4 Temperature Measurement Range and Offset

To improve the temperature measurement accuracy, the temperature sensor is designed with five measurement ranges as shown in Table 35.4-1. Each measurement range comes with specific measurement errors. Users do not need to worry about the errors, as they can choose specific offsets to get calibrated data.

Table 35.4-1. Temperature Measurement Range and Offset

Temperature Measurement Range (°C)	Temperature Offset
50 ~ 125	-2
20 ~ 100	-1
-10 ~ 80	0
-30 ~ 50	1
-40 ~ 20	2

35.4.5 Data Conversion

The sensor output value is stored in [APB_SARADC_TSENS_OUT](#). To calculate the actual temperature T (°C) based on $VALUE$, use the following equation:

$$T = 0.4386 * VALUE - 27.88 * offset - 20.52$$

where *offset* is the temperature offset shown in Table 35.4-1.

35.5 Programming Procedure

The temperature sensor can be started by software as follows:

1. Set [APB_SARADC_TSENS_PU](#) to power up the temperature sensor.
2. Set [PCR_TSENS_CLK_EN](#) to enable the temperature sensor clock.
3. Configure [APB_SARADC_TSENS_CLK_SEL](#) to select the temperature sensor clock.
4. Wait for [APB_SARADC_TSENS_XPD_WAIT](#) clock cycles till the temperature sensor releases from reset and starts measuring the temperature.

5. Wait for 100 μ s (so that the output value gradually approaches the actual temperature) and then read the data from [APB_SARADC_TSENS_OUT](#).

To enable hardware-triggered automatic temperature monitoring, add the following programming steps before powering up the sensor:

1. Configure [APB_SARADC_TSENS_SAMPLE_RATE](#) to set sampling rate.
2. Configure [APB_SARADC_WAKEUP_MODE](#) to select an automatic temperature monitoring mode.
3. Configure [APB_SARADC_WAKEUP_TH_HIGH/LOW](#) to set the high and low temperature monitoring thresholds.
4. Set [APB_SARADC_WAKEUP_EN](#) to start temperature monitoring.
5. Set [APB_SARADC_TSENS_SAMPLE_EN](#) to enable automatic temperature monitoring.

In automatic temperature monitoring mode, the output value will not be stored. However, users can get the value anytime from [APB_SARADC_TSENS_OUT](#).

35.6 Interrupts

ESP32-C5's Temperature Sensor can generate the following interrupt signal that will be sent to the [Interrupt Matrix](#).

- LP_TSENS_INTR

There is one internal interrupt source from the Temperature Sensor that can generate the above interrupt signal:

- APB_SARADC_TSENS_INT: Triggered when the temperature sample value exceeds the threshold.

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter 3 [Interrupt Matrix \[to be added later\]](#) > Section 3.1 [Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [35.8 Register Summary](#).

35.7 Event Task Matrix Feature

The temperature sensor on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows the temperature sensor's ETM tasks to be triggered by any peripherals' ETM events, or temperature sensor's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to the temperature sensor. For more information, please refer to Chapter 8 [Event Task Matrix \(ETM\)](#).

The temperature sensor can receive the following ETM tasks:

- TMPSNSR_TASK_START_SAMPLE: The temperature sensor starts sampling when this task is triggered.
- TMPSNSR_TASK_STOP_SAMPLE: The temperature sensor stops sampling when this task is triggered.

The temperature sensor can generate the following ETM events:

- `TMPSNSR_EVT_OVER_LIMIT`: Generated when the temperature is beyond the threshold.

In practical applications, temperature sensor's ETM events can trigger its own ETM tasks.

For example, the `TMPSNSR_EVT_OVER_LIMIT` event can trigger the `TMPSNSR_TASK_STOP_SAMPLE` task.

35.8 Register Summary

The addresses in this section are relative to Temperature Sensor base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
APB_SARADC_INT_ENA_REG	Enable register of SAR ADC/temperature sensor interrupts	0x0040	R/W
APB_SARADC_INT_RAW_REG	Raw register of SAR ADC/temperature sensor interrupts	0x0044	R/WTC/SS
APB_SARADC_INT_ST_REG	State register of SAR ADC/temperature sensor interrupts	0x0048	RO
APB_SARADC_INT_CLR_REG	Clear register of SAR ADC/temperature sensor interrupts	0x004C	WT
APB_SARADC_APB_TSENS_CTRL_REG	Temperature sensor control register 1	0x0058	varies
APB_SARADC_APB_TSENS_CTRL2_REG	Temperature sensor control register 2	0x005C	R/W
APB_TSENS_WAKE_REG	Temperature sensor automatic monitoring mode configuration register	0x0064	varies
APB_TSENS_SAMPLE_REG	Temperature sensor configuration register	0x0068	R/W
Version Control Register			
APB_SARADC_CTRL_DATE_REG	Version control register	0x03FC	R/W

35.9 Registers

The addresses in this section are relative to Temperature Sensor base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

For how to program reserved fields, please refer to Section *Programming Reserved Register Field*.

Register 35.1. APB_SARADC_APB_TSENS_CTRL_REG (0x0058)

(reserved)										APB_SARADC_TSENS_PU										APB_SARADC_TSENS_CLK_DIV										APB_SARADC_TSENS_IN_INV										(reserved)										APB_SARADC_TSENS_OUT																																																	
31										23										22										21										14										13										12										8										7										0									
0 0 0 0 0 0 0 0 0 0										0										6										0										0 0 0 0 0 0										0x80										Reset																																							

Reset

APB_SARADC_TSENS_OUT Stores the temperature sensor output value. (RO)

APB_SARADC_TSENS_IN_INV Configures whether to invert temperature sensor data.

0: No effect

1: Invert

(R/W)

APB_SARADC_TSENS_CLK_DIV Configures temperature sensor clock division. (R/W)

APB_SARADC_TSENS_PU Configures whether to power up temperature sensor.

0: No effect

1: Power up

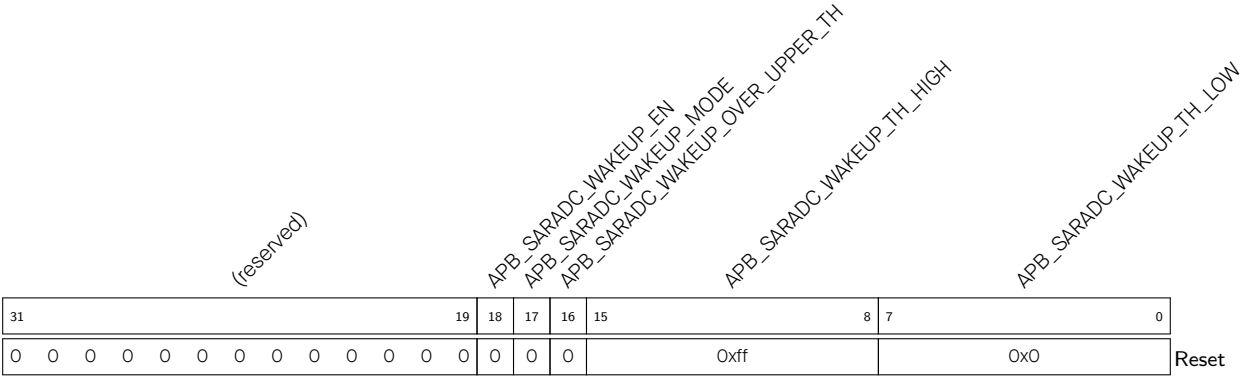
(R/W)

Register 35.2. APB_SARADC_APB_TSENS_CTRL2_REG (0x005C)

(reserved)																APB_SARADC_TSENS_CLK_SEL				APB_SARADC_TSENS_CLK_INV				APB_SARADC_TSENS_XPD_FORCE				APB_SARADC_TSENS_XPD_WAIT									
31																16	15	14	13	12	11	0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0	0x0	0x0	0x2																Reset		

- APB_SARADC_TSENS_CLK_SEL Configures the working clock for temperature sensor.
0: RC_FAST_CLK
1: XTAL_CLK
(R/W)
- APB_SARADC_TSENS_CLK_INV Configures the phase of the temperature sensor clock. (R/W)
- APB_SARADC_TSENS_XPD_FORCE Configures whether to enable force power up/down the temperature sensor.
0: Disable force power up function
1: Disable force power down function
2: Enable force power up temperature sensor
3: Enable force power down temperature sensor
(R/W)
- APB_SARADC_TSENS_XPD_WAIT Configure the wait time (in clock cycles) for the sensor to release from reset. (R/W)

Register 35.3. APB_TSENS_WAKE_REG (0x0064)



APB_SARADC_WAKEUP_TH_LOW Configures the low threshold for automatic temperature monitoring. (R/W)

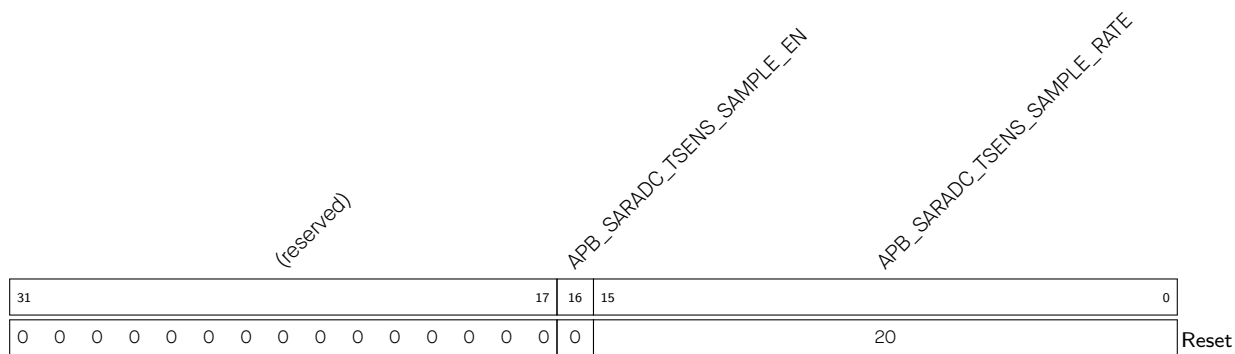
APB_SARADC_WAKEUP_TH_HIGH Configures the high threshold for automatic temperature monitoring. (R/W)

APB_SARADC_WAKEUP_OVER_UPPER_TH Represents whether the temperature output value exceeds the threshold. Valid only when APB_SARADC_WAKEUP_EN=1.
0: The temperature output value is below the low threshold
1: The temperature output value is above the high threshold
(RO)

APB_SARADC_WAKEUP_MODE Selects the temperature monitoring mode.
0: Absolute value mode
1: Change value mode
(R/W)

APB_SARADC_WAKEUP_EN Configures whether to enable the temperature monitoring function.
0: Disable
1: Enable
(R/W)

Register 35.4. APB_TSENS_SAMPLE_REG (0x0068)



APB_SARADC_TSENS_SAMPLE_RATE Configures the sampling rate for hardware-triggered automatic temperature monitoring. The sampling period = configured value × the sensor's working clock cycle. (R/W)

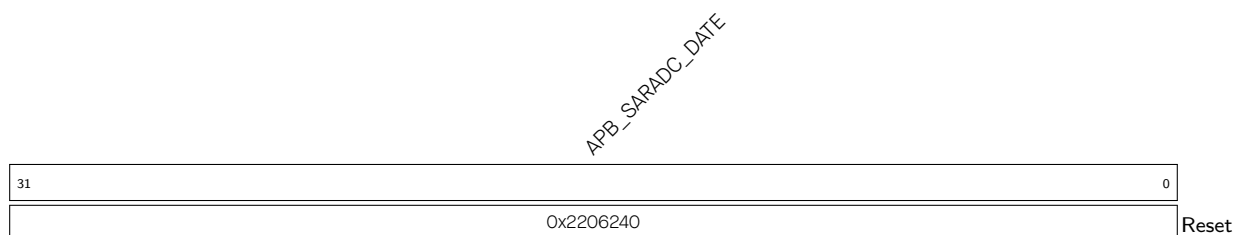
APB_SARADC_TSENS_SAMPLE_EN Configures whether to enable automatic temperature monitoring.

0: Disable

1: Enable

(R/W)

Register 35.5. APB_SARADC_CTRL_DATE_REG (0x03FC)



APB_SARADC_DATE Version control register for SAR ADC (please refer to Chapter [36 ADC Controller](#) for more information) and the temperature sensor. (R/W)

Chapter 36

ADC Controller

36.1 Overview

ESP32-C5 integrates one 12-bit successive approximation ADC (SAR ADC) for measuring analog signals from up to six channels.

The SAR ADC is managed by the DIG ADC controller that drives ADC sampling. The SAR ADC supports one-shot sampling and multi-channel sampling.

36.2 Terminology

The following terms related to SAR ADC are defined in the context of the *ESP32-C5 Technical Reference Manual* to help readers better understand this document:

SAR ADC:	Including analog ADC circuit and digital ADC controller.
One-shot sampling mode:	In this mode, ADC samples one channel at a time.
Multi-channel sampling mode:	In this mode, ADC sequentially samples a group of channels or continuously samples a single channel.
Conversion result:	The conversion result is the binary digital value obtained after the analog-to-digital conversion process. It is sometimes written as conversion data.
Filtered data:	Filtered data is the result of processing conversion data through a filter.
Sampling:	The term typically refers to the entire process of sampling analog inputs, converting the sampled data, and transferring the conversion results to memory. In certain contexts and stages of the process, sampling may also refer to the SAR ADC capturing data points from an analog signal.

36.3 Feature List

The SAR ADC has the following features:

- 12-bit resolution
- Analog inputs sampling from up to six pins
- One-shot sampling mode and multi-channel sampling mode
- Multi-channel sampling mode supports:

- configurable channel sampling sequence
- two filters whose filter coefficients are configurable
- two threshold monitors that can trigger an interrupt when the filtered value is below a low threshold or above a high threshold
- continuous transfer of converted data to memory via GDMA interface
- Support for several Event Task Matrix (ETM) related events and tasks

36.4 Architectural Overview

The major components of SAR ADC and their interconnections are shown in Figure 36.4-1.

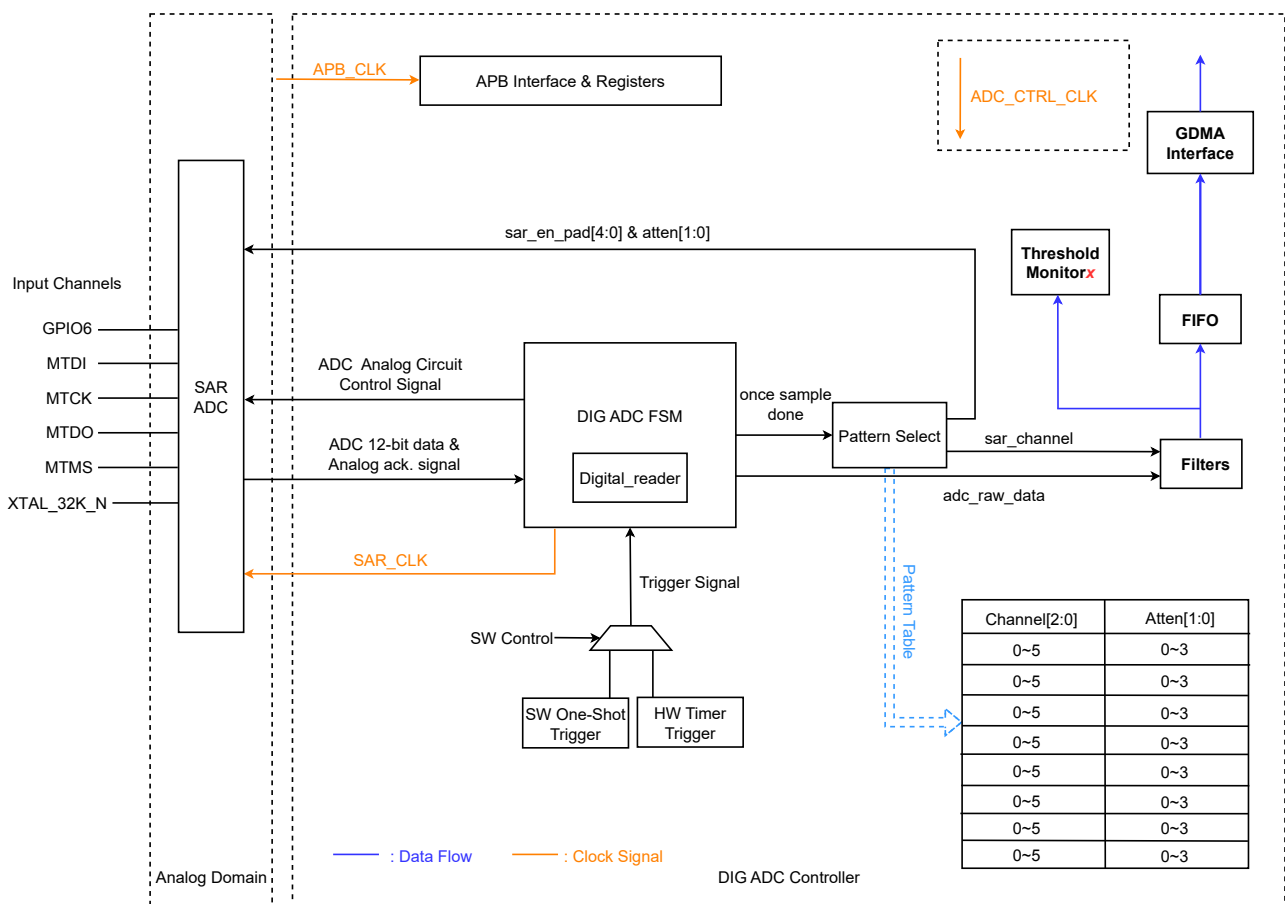


Figure 36.4-1. SAR ADC Architecture

As Figure 36.4-1 shows, the SAR ADC module contains the following major functional blocks:

- Six channels, connected to six pins on the chip
- SAR ADC: analog domain of the SAR ADC module
- DIG ADC Controller: digital domain of the SAR ADC module, mainly including:
 - Clock management module: selects clock source and division
 - DIG ADC FSM: generates the signals required throughout the ADC sampling process

- Digital_reader: reads data from SAR ADC, driven by DIG ADC FSM
- Filter: filters ADC converted data in multi-channel sampling mode
- Threshold monitor^x: threshold monitor 1 and threshold monitor 2. The monitor^x will trigger an interrupt when the converted value is below a low threshold or above a high threshold.

36.5 Functional Description

36.5.1 ADC Power Up

The ADC can be powered up by setting PMU_XPD_PERIF_I2C and PMU_PERIF_I2C_RST. ADC sampling can be performed right after power-up. Users do not need to worry about wait time, as it has been dealt with in hardware design.

36.5.2 ADC Channels

The SAR ADC has six channels that are connected to six pins on the chip. In order to sample an analog signal, the SAR ADC must first select the analog pin to measure via an internal multiplexer.

Table 36.5-1 shows the pins used as ADC channels and the corresponding GPIO numbers.

Table 36.5-1. SAR ADC Channels

Pin Name	GPIO Number	ADC Channel
XTAL_32K_N	GPIO1	0
MTMS	GPIO2	1
MTDI	GPIO3	2
MTCK	GPIO4	3
MTDO	GPIO5	4
GPIO6	GPIO6	5

36.5.3 ADC Clock

Figure 36.5-1 shows the clock structure of SAR ADC.

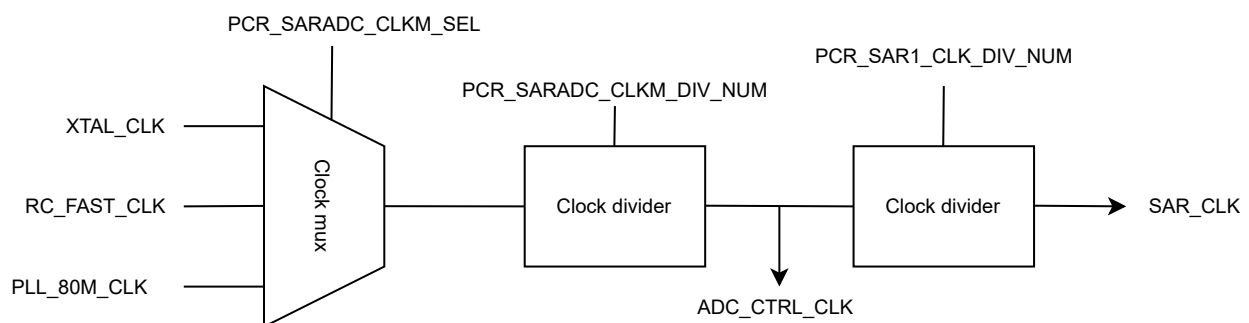


Figure 36.5-1. SAR ADC Clock Structure

The system clock for SAR ADC is `ADC_CTRL_CLK`, which is the operating clock for DIG ADC FSM and other control logic (except APB interface and `Digital_reader`).

`ADC_CTRL_CLK` has three possible sources: `XTAL_CLK`, `RC_FAST_CLK`, and `PLL_80M_CLK`, selected by `PCR_SARADC_CLKM_SEL`.

`SAR_CLK` is the operating clock for SAR ADC and `Digital_reader`. It is divided from `ADC_CTRL_CLK` and must not exceed 5 MHz.

For more information about clocks, please refer to Chapter 6 [Reset and Clock](#).

36.5.4 One-Shot Sampling Mode

In one-shot sampling mode, the ADC samples one channel once. This mode is started by software using `APB_SARADC_ONETIME_SAMPLE`. Once sampling is done, the conversion result is stored in `APB_SARADC_DATA`. To switch channels, configure `APB_SARADC_ONETIME_SAMPLE` once again.

36.5.5 Multi-Channel Sampling Mode

Multi-channel sampling mode is triggered by a timer that is specifically designed for SAR ADC. In this mode the ADC samples a group of channels according to the sequence defined in the pattern table. The multi-channel sampling mode can also be used for continuous sampling on one channel.

The timer is enabled by setting `APB_SARADC_TIMER_EN`. A trigger target for the timer needs to be configured with `APB_SARADC_TIMER_TARGET`. When the timer counts up to two times of `APB_SARADC_TIMER_TARGET`, a sampling operation is triggered. The timer is clocked from `ADC_CTRL_CLK`.

When sampling is complete, the timer resets to 0 and starts counting again. The conversion result is transferred to memory continuously via the GDMA interface.

Note:

The SAR ADC can only work under one operating mode at one time, either one-shot sampling mode or multi-channel sampling mode.

36.5.6 ADC Conversion and Attenuation

The SAR ADC can measure analog voltages from 0 mV to V_{ref} . V_{ref} is the SAR ADC's internal reference voltage (1100 mV by design). The conversion result (*data*) is a 12-bit digital value, the raw data. To calculate the voltage V_{data} based on the raw data, please use the formula below:

$$V_{data} = \frac{V_{ref}}{k} \times \frac{data}{4095}$$

k is the coefficient corresponding to the configured attenuation.

To convert voltages larger than V_{ref} , apply attenuation to the input signals using `APB_SARADC_ONETIME_ATTEN`. The attenuation can be 0 dB, 2.5 dB, 6 dB, or 12 dB.

36.5.7 DIG ADC FSM

In multi-channel sampling mode DIG ADC FSM (hereinafter referred to as FSM) generates all types of signals used in the sampling process. Figure 36.5-2 illustrates how DIG ADC FSM works.

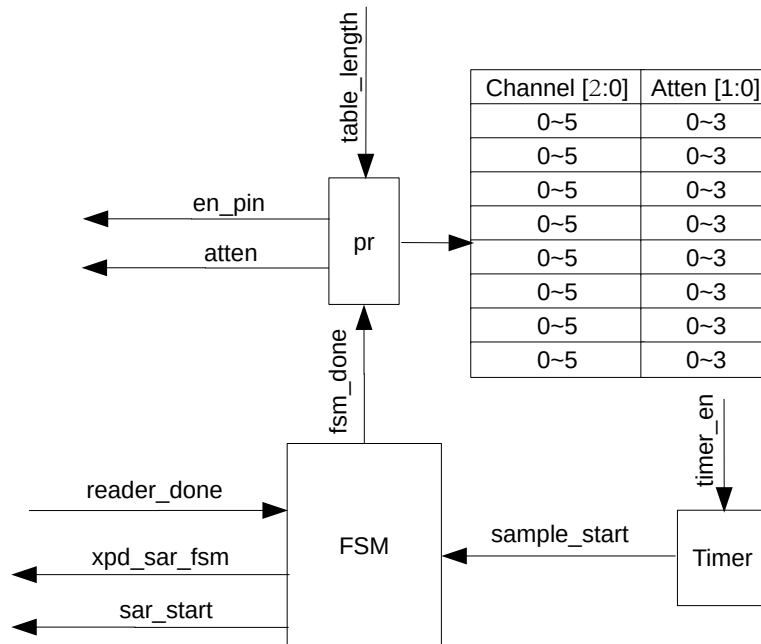


Figure 36.5-2. DIG ADC FSM Block Diagram

Wherein:

- Timer: a dedicated timer for the DIG ADC controller to generate **sample_start** signal.
- pr: a pointer to the pattern table that defines the conversion rules for ADC. FSM sends out corresponding signals based on the conversion rules.

Set [APB_SARADC_TIMER_EN](#) to enable the timer. The timeout event of this timer triggers a **sample_start** signal. This signal drives the FSM module to start sampling. When the FSM module receives the **sample_start** signal, it starts the following operations:

- Powers up SAR ADC.
- Determines the conversion rules (selected sample channels and attenuations) defined in the patterns that the current pr points to.
- Outputs the **en_pin** and **atten** signals corresponding to the conversion rules to the analog side.
- Initiates the **sar_start** signal and starts sampling.

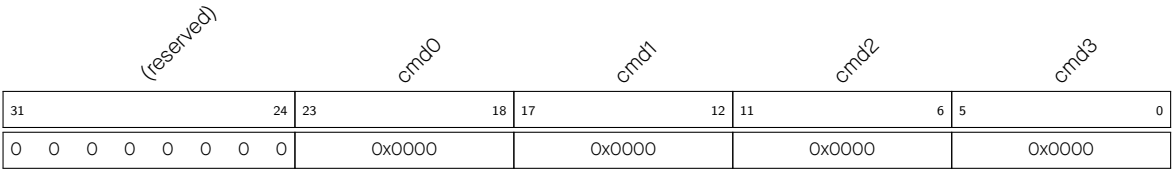
When FSM receives the **reader_done** signal from Digital_reader, it starts the following operations:

- Stops sampling.
- Transfers the conversion result to the filter. Then the threshold monitor transfers the filtered result to memory via GDMA (see Figure 36.4-1).

- Updates pr and waits for the next sampling. The pointer pr counts cyclically between 0 and `APB_SARADC_SAR_PATT_LEN` (`table_length`).

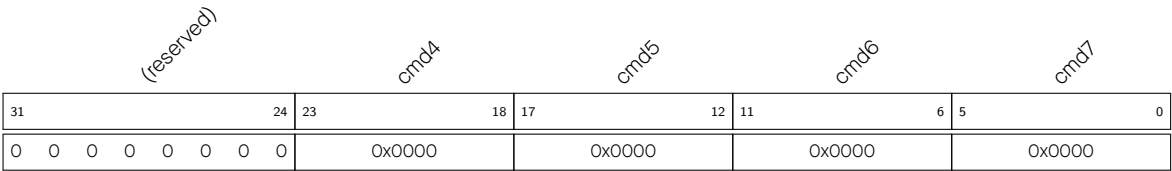
36.5.8 Pattern Table

FSM contains a pattern table consisting of the `APB_SARADC_SAR_PATT_TAB1_REG` and `APB_SARADC_SAR_PATT_TAB2_REG` registers. Each register contains four patterns and each pattern is 6 bits wide, as Figure 36.5-3 and Figure 36.5-4 show.



cmd x represents patterns 0 ~ 3.

Figure 36.5-3. APB_SARADC_SAR_PATT_TAB1_REG Contains Patterns 0 - 3



cmd x represents patterns 4 ~ 7.

Figure 36.5-4. APB_SARADC_SAR_PATT_TAB2_REG Contains Patterns 4 - 7

Each pattern is 6 bits wide, consisting of three fields where conversion rules are stored. Figure 36.5-5 shows the pattern format and is followed by the descriptions of each field.



Figure 36.5-5. Pattern Structure

atten Configures attenuation:

- 0: 0 dB
- 1: 2.5 dB
- 2: 6 dB
- 3: 12 dB

ch_sel Configures channel:

- 0: Channel 0
- 1: Channel 1
- 2: Channel 2
- 3: Channel 3
- 4: Channel 4

5: Channel 5

(reserved) Reserved

Pattern Configuration Example

Note: When using multi-channel continuous sampling, the attenuation for each channel must be set to be consistent. In this example, two channels are selected for multi-channel sampling:

- Channel 0, with an attenuation of 2.5 dB
- Channel 2, with an attenuation of 2.5 dB

The detailed configuration is as follows:

- Configure the first pattern (cmd0):

(reserved)		ch_sel	atten	
5	4	2	1	0
0	0		1	

Figure 36.5-6. cmd0 configuration

atten write 1 to this field, to set the attenuation to 2.5 dB.

ch_sel write 0 to this field, to select channel 0.

- Configure the second pattern (cmd1):

(reserved)		ch_sel	atten	
5	4	2	1	0
0	2		1	

Figure 36.5-7. cmd1 Configuration

atten write 1 to this field, to set the attenuation to 2.5 dB.

ch_sel write 2 to this field, to select channel 2.

- Configure [APB_SARADC_SAR_PATT_LEN](#) to 1. Then patterns cmd0 and cmd1 will be used.
- Enable the timer so that the DIG ADC controller starts sampling the two channels periodically.

36.5.9 ADC Filters

The DIG ADC controller provides two filters for filtering ADC converted data in multi-channel sampling mode. Both filters can be configured to any ADC channel, but cannot be configured to the same channel. If the two filters are configured to the same channel, the first one takes effect.

The filtered data is determined by the following equation:

$$data_{cur} = \frac{(k-1)data_{prev}}{k} + \frac{data_{in}}{k} + 0.5$$

- $data_{cur}$: the filtered data
- $data_{in}$: the ADC converted data
- $data_{prev}$: the last filtered data
- k : the filter coefficient

The filters are configured as follows:

- Configure `APB_SARADC_FILTER_CHANNEL $x$` to select the ADC channel for filter x ($x=0, 1$).
- Configure `APB_SARADC_FILTER_FACTOR $x$` to set the coefficient k for filter x .

36.5.10 Threshold Monitors

Two threshold monitors are available in the DIG ADC controller to monitor the filtered data in multi-channel sampling mode. When the data is above the high threshold, a high threshold interrupt is triggered; when the data is below the low threshold, a low threshold interrupt is triggered. Both monitors can be configured to any ADC channel, but cannot be configured to the same channel.

Threshold monitors are configured as follows:

- Set `APB_SARADC_THRES $x$ _EN` to enable threshold monitor x ($x=0, 1$).
- Configure `APB_SARADC_THRES $x$ _LOW` to set a low threshold.
- Configure `APB_SARADC_THRES $x$ _HIGH` to set a high threshold.
- Configure `APB_SARADC_THRES $x$ _CHANNEL` to select the channel to monitor.
- Set `APB_SARADC_THRES_ALL_EN` to enable threshold monitoring functionality.

36.5.11 GDMA Support

As SAR ADC has only one data register `APB_SARADC_DATA` for storing converted result from one-shot sampling, it is necessary to enable GDMA when converting data on multiple channels so that the data can be transferred to memory continuously.

GDMA support is triggered by the timer in the DIG ADC controller. Users can switch the DMA data path to the DIG ADC controller by configuring `APB_SARADC_APB_ADC_TRANS`. For specific DMA configuration, please refer to Chapter 3 *GDMA Controller (GDMA)*.

GDMA Data Format

The ADC eventually passes 32-bit data to GDMA. The data format is shown in Figure 36.5-8.

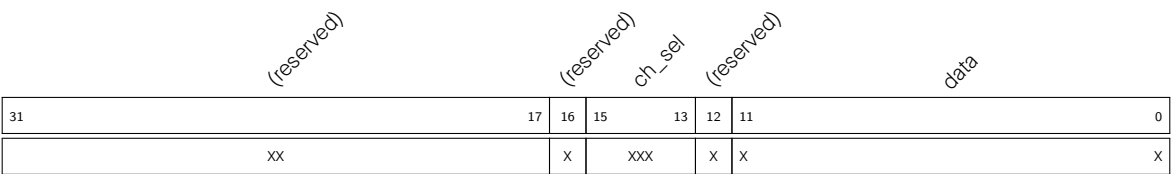


Figure 36.5-8. DMA Data Format

data 12-bit ADC conversion result

ch_sel 3-bit channel information

36.6 Event Task Matrix Feature

The SAR ADC on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows SAR ADC's ETM tasks to be triggered by any peripherals' ETM events, or SAR ADC's ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM tasks and events related to SAR ADC and temperature sensor. For more information, please refer to Chapter [8 Event Task Matrix \(ETM\)](#).

The SAR ADC can receive the following ETM tasks:

- ADC_TASK_SAMPLE0: ADC starts one-shot sampling when this task is triggered.
- ADC_TASK_START0: ADC starts multi-channel sampling when this task is triggered.
- ADC_TASK_STOP0: ADC stops sampling when this task is triggered.

The SAR ADC can generate the following ETM events:

- ADC_EVT_CONV_CMPLT0: Generated each time ADC completes a sampling in either one-shot sampling mode or multi-channel sampling mode.
- ADC_EVT_EQ_ABOVE_THRESH x : Generated when the ADC filtered data is above the threshold. $x = 0, 1$, representing threshold monitor 0, 1.
- ADC_EVT_EQ_BELOW_THRESH x : Generated when the ADC filtered data is below the threshold. $x = 0, 1$, representing threshold monitor 0, 1.
- ADC_EVT_STARTED0: Generated when ADC begins sampling; one-shot sampling will not trigger this event.
- ADC_EVT_STOPPED0: Generated when ADC stops sampling, one-shot sampling will not trigger this event.

In practical applications, SAR ADC's ETM events can trigger its own ETM tasks. For example, the ADC_EVT_EQ_ABOVE_THRESH x event can trigger the ADC_TASK_STOP0 task.

36.7 Interrupts

ESP32-C5's SAR ADC can generate the APB_ADC_INTR interrupt signal that will be sent to the [Interrupt Matrix](#). The following internal interrupt sources from SAR ADC can generate the interrupt signal APB_ADC_INTR:

- APB_SARADC_ADC_DONE_INT: Triggered when SAR ADC completes one conversion.
- APB_SARADC_THRES x _HIGH_INT: Triggered when the filtered data is above the high threshold of monitor x .
- APB_SARADC_THRES x _LOW_INT: Triggered when the filtered data is below the low threshold of monitor x .

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to Chapter [3 Interrupt Matrix \[to be added later\]](#) > Section [3.1 Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [36.9 Register Summary](#).

36.8 Programming Procedure

36.8.1 Configuring One-shot Sampling Mode

The one-shot sampling mode can be configured with the following procedure:

1. Set `PMU_XPD_PERIF_I2C` and `PMU_PERIF_I2C_RST` to power up SAR ADC.
2. Configure `PCR_SARADC_CLKM_SEL` to select ADC clock source.
3. Configure `PCR_SARADC_CLKM_DIV_NUM` and `PCR_SAR1_CLK_DIV_NUM` to set clock division.
4. Set `PCR_SARADC_CLKM_EN` to enable ADC clock.
5. Set `APB_SARADC_ONETIME_SAMPLE` to enable the one-shot sampling mode.
6. Configure `APB_SARADC_ONETIME_CHANNEL` to select sampled channel.
7. Configure `APB_SARADC_ONETIME_ATTEN` to set attenuation as needed.
8. Set `APB_SARADC_ONETIME_START` to start one-shot sampling.

Once sampling is complete, an `APB_SARADC_ADC_DONE_INT_RAW` interrupt is generated. Software can read conversion result from `APB_SARADC_ADC_DATA`. To switch the sampled channel, program from step 6.

36.8.2 Configuring Multi-Channel Sampling Mode

The multi-channel sampling mode can be configured with the following procedure:

1. Set `PMU_XPD_PERIF_I2C` and `PMU_PERIF_I2C_RST` to power up SAR ADC.
2. Configure `PCR_SARADC_CLKM_SEL` to select ADC clock source.
3. Configure `PCR_SARADC_CLKM_DIV_NUM` and `PCR_SAR1_CLK_DIV_NUM` to set clock division.
4. Set `PCR_SARADC_CLKM_EN` to enable ADC clock.
5. Configure the pattern table as described in Section [36.5.8 Pattern Table](#).
6. Configure channels and filtering coefficients as described in Section [36.5.9 ADC Filters](#).
7. Configure threshold monitoring as described in Section [36.5.10 Threshold Monitors](#), as needed.
8. Set `APB_SARADC_APB_ADC_TRANS` to use GDMA.
9. Configure `APB_SARADC_TIMER_TARGET` to set trigger target for DIG ADC timer.
10. Set `APB_SARADC_TIMER_EN` to enable the timer.

The timer timeout will trigger DIG ADC FSM to start sampling according to the pattern table. The conversion result will be automatically stored in memory. When the sampling reaches the number of limit set in `APB_SARADC_APB_ADC_EOF_NUM`, it will terminate.

Once sampling is complete, an `APB_SARADC_ADC_DONE_INT_RAW` interrupt will be generated.

36.9 Register Summary

The addresses in this section are relative to SAR ADC Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

The abbreviations given in Column **Access** are explained in Section *Access Types for Registers*.

Name	Description	Address	Access
Configuration Registers			
APB_SARADC_CTRL_REG	SAR ADC control register 1	0x0000	R/W
APB_SARADC_CTRL2_REG	SAR ADC control register 2	0x0004	R/W
APB_SARADC_FILTER_CTRL1_REG	Filtering control register 1	0x0008	R/W
APB_SARADC_SAR_PATT_TAB1_REG	Pattern table register 1	0x0018	R/W
APB_SARADC_SAR_PATT_TAB2_REG	Pattern table register 2	0x001C	R/W
APB_SARADC_ONETIME_SAMPLE_REG	Configuration register for one-shot sampling	0x0020	R/W
APB_SARADC_FILTER_CTRL0_REG	Filtering control register 0	0x0028	R/W
APB_SARADC_SAR1DATA_STATUS_REG	SAR ADC conversion data storage register	0x002C	RO
APB_SARADC_THRES0_CTRL_REG	Filtered data threshold control register 0	0x0034	R/W
APB_SARADC_THRES1_CTRL_REG	Filtered data threshold control register 1	0x0038	R/W
APB_SARADC_THRES_CTRL_REG	Filtered data threshold control register	0x003C	R/W
APB_SARADC_INT_ENA_REG	Enable register of SAR ADC interrupts	0x0040	R/W
APB_SARADC_INT_RAW_REG	Raw register of SAR ADC interrupts	0x0044	R/WTC/SS
APB_SARADC_INT_ST_REG	State register of SAR ADC interrupts	0x0048	RO
APB_SARADC_INT_CLR_REG	Clear register of SAR ADC interrupts	0x004C	WT
APB_SARADC_DMA_CONF_REG	DMA configuration register for SAR ADC	0x0050	R/W
APB_SARADC_CTRL_DATE_REG	Version control register	0x03FC	R/W

36.10 Registers

The addresses in this section are relative to SAR ADC Controller base address provided in Table 4.3-2 in Chapter 4 *System and Memory*.

Register 36.1. APB_SARADC_CTRL_REG (0x0000)

(reserved)		APB_SARADC_XPD_SAR_FORCE		(reserved)		APB_SARADC_SAR_PATT_P_CLEAR		(reserved)		APB_SARADC_SAR_PATT_LEN		(reserved)		APB_SARADC_SAR_CLK_GATED		(reserved)		APB_SARADC_START		APB_SARADC_START_FORCE	
31	29	28	27	26	24	23	22	18	17	15	14	7	6	5	2	1	0	Reset			
0		0		0 0 0		0	0 0 0 0 0		7		4		1	0 0 0 0		0	0				

APB_SARADC_START_FORCE Configures whether to use software to enable SAR ADC.

- 0: Select FSM to start SAR ADC
 - 1: Select software to start SAR ADC
- (R/W)

APB_SARADC_START Configures whether to start SAR ADC by software.

- 0: No effect
 - 1: Start SAR ADC by software
- Valid only when [APB_SARADC_START_FORCE](#) = 1.
- (R/W)

APB_SARADC_SAR_CLK_GATED Configures whether to enable SAR ADC clock gate.

- 0: Disable
 - 1: Enable
- (R/W)

APB_SARADC_SAR_PATT_LEN Configures how many patterns will be used.

- 0: Use only cmd0
 - 1: Use cmd0 and cmd1
 - n: Use cmd0 to cmdn, the maximum n is 7
- (R/W)

APB_SARADC_SAR_PATT_P_CLEAR Configures whether to clear the pointer of pattern table for DIG ADC controller.

- 0: No effect
 - 1: Clear
- (R/W)

APB_SARADC_XPD_SAR_FORCE Configures whether to forcibly power up SAR ADC.

- 0: No effect
 - 1: Forcibly power up SAR ADC
- (R/W)

Register 36.2. APB_SARADC_CTRL2_REG (0x0004)

(reserved)								APB_SARADC_TIMER_EN																APB_SARADC_TIMER_TARGET																(reserved)				(reserved)				APB_SARADC_SAR1_INV																APB_SARADC_MAX_MEAS_NUM																APB_SARADC_MEAS_NUM_LIMIT															
31								25								24								23								12								11				10				9				8				1								0																															
0 0 0 0 0 0 0 0								0								10								0				0				0				255								0				Reset																																															

APB_SARADC_MEAS_NUM_LIMIT Configures whether to enable the limitation of SAR ADC's maximum conversion times.

0: Disable

1: Enable

(R/W)

APB_SARADC_MAX_MEAS_NUM Configures the SAR ADC's maximum conversion times. (R/W)

APB_SARADC_SAR1_INV Configures whether to invert the data of SAR ADC.

0: No effect

1: Invert the data of SAR ADC

(R/W)

APB_SARADC_TIMER_TARGET Configures SAR ADC timer target. (R/W)

APB_SARADC_TIMER_EN Configures whether to enable SAR ADC timer trigger.

0: Disable

1: Enable

(R/W)

Register 36.3. APB_SARADC_FILTER_CTRL1_REG (0x0008)

APB_SARADC_FILTER_FACTOR0																												APB_SARADC_FILTER_FACTOR1																												(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
31	29	28	26	25																								0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Reset

APB_SARADC_FILTER_FACTOR1 Configures the filter coefficient k for SAR ADC filter 1.

0: $k=0$ (i.e., the filter is disabled)

1: $k=2$

2: $k=4$

3: $k=8$

4: $k=16$

5: $k=32$

6: $k=64$

(R/W)

APB_SARADC_FILTER_FACTOR0 Configures the filter coefficient k for SAR ADC filter 0 (same as above). (R/W)

Register 36.4. APB_SARADC_SAR_PATT_TAB1_REG (0x0018)

(reserved)								APB_SARADC_SAR_PATT_TAB1															
31	24	23																					0
0	0	0	0	0	0	0	0	0	0xffffffff														0

Reset

APB_SARADC_SAR_PATT_TAB1 Configures pattern 0 ~ 3 (each pattern takes six bits). For details see Section 36.5.8 [Pattern Table](#). (R/W)

Register 36.5. APB_SARADC_SAR_PATT_TAB2_REG (0x001C)



APB_SARADC_SAR_PATT_TAB2 Configures pattern 4 ~ 7 (each pattern takes six bits). For details see Section 36.5.8 *Pattern Table*. (R/W)

Register 36.6. APB_SARADC_ONETIME_SAMPLE_REG (0x0020)

Diagram illustrating the structure of the `APB_SARADC_ONETIME` register. The register is 32 bits wide, divided into four 8-bit fields:

- `APB_SARADC_ONETIME_SAMPLE` (bits 31-24)
- `APB_SARADC_ONETIME_START` (bits 23-16)
- `APB_SARADC_ONETIME_CHANNEL` (bits 15-8)
- `APB_SARADC_ONETIME_ATTEN` (bits 7-0)

The diagram shows the bit positions for each field, with bit 31 being the most significant and bit 0 being the least significant. The fields are labeled as (reserved) in the original image, indicating they are not used for configuration.

APB_SARADC_ONETIME_ATTEN Configures the attenuation for a one-shot sampling.

0: 0 dB

1: 2.5 dB

2: 6 dB

3: 12 dB

(R/W)

APB_SARADC_ONETIME_CHANNEL Configures the channel for a one-shot sampling.

0: Channel 0

1: Channel 1

2: Channel 2

3: Channel 3

4: Channel 4

5: Channel 5

(R/W)

APB_SARADC_ONETIME_START Configures whether to start SAR ADC one-shot sampling.

0: No effect

1: Start

(R/W)

APB_SARADC_ONETIME_SAMPLE Configures whether to enable SAR ADC one-shot sampling.

0: Disable

1: Enable

(R/W)

Register 36.7. APB_SARADC_FILTER_CTRL0_REG (0x0028)

APB_SARADC_FILTER_RESET						(reserved)						APB_SARADC_FILTER_CHANNEL0						APB_SARADC_FILTER_CHANNEL1						(reserved)							
31	30					26	25	22			21	18			17																0
0	0	0	0	0	0	13			13			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0															Reset				

Reset

APB_SARADC_FILTER_CHANNEL1 Configures the filter channel for SAR ADC filter 1. (R/W)**APB_SARADC_FILTER_CHANNEL0** Configures the filter channel for SAR ADC filter 0. (R/W)**APB_SARADC_FILTER_RESET** Configures whether to reset SAR ADC filter.

0: No effect

1: Reset

(R/W)

Register 36.8. APB_SARADC_SAR1DATA_STATUS_REG (0x002C)

(reserved)																APB_SARADC_DATA																
31																0																
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0																Reset

Reset

APB_SARADC_DATA Stores SAR ADC conversion data in one-shot sampling mode. (RO)

Register 36.9. APB_SARADC_THRESO_CTRL_REG (0x0034)

(reserved)																APB_SARADC_THRESO_LOW																APB_SARADC_THRESO_HIGH																(reserved)																APB_SARADC_THRESO_CHANNEL															
31	30															18	17																5	4	3	0																																											
0	0															0x1fff															0	13															Reset																																

Reset

APB_SARADC_THRESO_CHANNEL Configures the channel for SAR ADC monitor 0. (R/W)**APB_SARADC_THRESO_HIGH** Configures the high threshold for SAR ADC monitor 0. (R/W)**APB_SARADC_THRESO_LOW** Configures the low threshold for SAR ADC monitor 0. (R/W)

Register 36.10. APB_SARADC_THRES1_CTRL_REG (0x0038)

(reserved)		APB_SARADC_THRES1_LOW																APB_SARADC_THRES1_HIGH																(reserved)		APB_SARADC_THRES1_CHANNEL														
31	30																	17																	5	4	3													0
0	0																0x1fff																0	13												Reset				

APB_SARADC_THRES1_CHANNEL Configures the channel for SAR ADC monitor 1. (R/W)

APB_SARADC_THRES1_HIGH Configures the high threshold for SAR ADC monitor 1. (R/W)

APB_SARADC_THRES1_LOW Configures the low threshold for SAR ADC monitor 1. (R/W)

Register 36.11. APB_SARADC_THRES_CTRL_REG (0x003C)

APB_SARADC_THRES0_EN																															APB_SARADC_THRES1_EN																															(reserved)																															APB_SARADC_THRES_ALL_EN																															(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
31	30	29	28	27	26																										0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

APB_SARADC_THRES_ALL_EN Configures whether to enable the threshold monitoring for all configured channels.

0: Disable

1: Enable

(R/W)

APB_SARADC_THRES1_EN Configures whether to enable threshold monitor 1.

0: Disable

1: Enable

(R/W)

APB_SARADC_THRES0_EN Configures whether to enable threshold monitor 0.

0: Disable

1: Enable

(R/W)

Register 36.12. APB_SARADC_INT_ENA_REG (0x0040)

APB_SARADC_ADC_DONE_INT_ENA																															(reserved)																															APB_SARADC_THRES0_HIGH_INT_ENA																															APB_SARADC_THRES1_HIGH_INT_ENA																															APB_SARADC_THRES0_LOW_INT_ENA																															APB_SARADC_THRES1_LOW_INT_ENA																															APB_SARADC_TSENS_INT_ENA																															(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
31	30	29	28	27	26	25	24																								0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

APB_SARADC_TSENS_INT_ENA Write 1 to enable the [APB_SARADC_TSENS_INT](#) [TBA] interrupt. (R/W)

APB_SARADC_THRES1_LOW_INT_ENA Write 1 to enable the [APB_SARADC_THRES1_LOW_INT](#) interrupt. (R/W)

APB_SARADC_THRES0_LOW_INT_ENA Write 1 to enable the [APB_SARADC_THRES0_LOW_INT](#) interrupt. (R/W)

APB_SARADC_THRES1_HIGH_INT_ENA Write 1 to enable the [APB_SARADC_THRES1_HIGH_INT](#) interrupt. (R/W)

APB_SARADC_THRES0_HIGH_INT_ENA Write 1 to enable the [APB_SARADC_THRES0_HIGH_INT](#) interrupt. (R/W)

APB_SARADC_ADC_DONE_INT_ENA Write 1 to enable the [APB_SARADC_ADC_DONE_INT](#) interrupt. (R/W)

Register 36.13. APB_SARADC_INT_RAW_REG (0x0044)

[illegible]

APB_SARADC_TSENS_INT_RAW The raw interrupt status of the [APB_SARADC_TSENS_INT](#) interrupt.
(R/WTC/SS)

APB_SARADC_THRES1_LOW_INT_RAW The raw interrupt status of the [APB_SARADC_THRES1_LOW_INT](#) interrupt. (R/WTC/SS)

APB_SARADC_THRESO_LOW_INT_RAW The raw interrupt status of the [APB_SARADC_THRESO_LOW_INT](#) interrupt. (R/WTC/SS)

APB_SARADC_THRES1_HIGH_INT_RAW The raw interrupt status of the [APB_SARADC_THRES1_HIGH_INT](#) interrupt. (R/WTC/SS)

APB_SARADC_THRESO_HIGH_INT_RAW The raw interrupt status of the [APB_SARADC_THRESO_HIGH_INT](#) interrupt. (R/WTC/SS)

APB_SARADC_ADC_DONE_INT_RAW The raw interrupt status of the [APB_SARADC_ADC_DONE_INT](#) interrupt. (R/WTC/SS)

Register 36.14. APB_SARADC_INT_ST_REG (0x0048)

Diagram illustrating the APB_SARADC register structure. The register is 32 bits wide, with bits 31-24 reserved. The structure is as follows:

- Bits 31-24: (reserved)
- Bits 23-16: APB_SARADC_THRES0_HIGH_INT_ST
- Bits 15-8: APB_SARADC_THRES0_LOW_INT_ST
- Bits 7-0: APB_SARADC_THRES1_LOW_INT_ST

APB_SARADC_TSENS_INT_ST The masked interrupt status of the [APB_SARADC_TSENS_INT](#) interrupt. (RO)

APB_SARADC_THRES1_LOW_INT_ST The masked interrupt status of the [APB_SARADC_THRES1_LOW_INT](#) interrupt. (RO)

APB_SARADC_THRESO_LOW_INT_ST The masked interrupt status of the [APB_SARADC_THRESO_LOW_INT](#) interrupt. (RO)

APB_SARADC_THRES1_HIGH_INT_ST The masked interrupt status of the [APB_SARADC_THRES1_HIGH_INT](#) interrupt. (RO)

APB_SARADC_THRESO_HIGH_INT_ST The masked interrupt status of the [APB_SARADC_THRESO_HIGH_INT](#) interrupt. (RO)

APB_SARADC_ADC_DONE_INT_ST The masked interrupt status of the [APB_SARADC_ADC_DONE_INT](#) interrupt. (RO)

Register 36.15. APB_SARADC_INT_CLR_REG (0x004C)

Diagram illustrating the structure of the APB_SARADC_TSENSE1 register (32 bits):

- Bits 31 to 24: (reserved)
- Bits 23 to 16: APB_SARADC_TSENSE1_INT_CLR
- Bits 15 to 8: APB_SARADC_THRES1_LOW_INT_CLR
- Bits 7 to 0: APB_SARADC_THRES1_HIGH_INT_CLR

APB_SARADC_TSENS_INT_CLR Write 1 to clear the [APB_SARADC_TSENS_INT](#) interrupt. (WT)

APB_SARADC_THRES1_LOW_INT_CLR Write 1 to clear the **APB_SARADC_THRES1_LOW_INT** interrupt. (WT)

APB_SARADC_THRESO_LOW_INT_CLR Write 1 to clear the **APB_SARADC_THRESO_LOW_INT** interrupt. (WT)

APB_SARADC_THRES1_HIGH_INT_CLR Write 1 to clear the **APB_SARADC_THRES1_HIGH_INT** interrupt. (WT)

APB_SARADC_THRESO_HIGH_INT_CLR Write 1 to clear the **APB_SARADC_THRESO_HIGH_INT** interrupt. (WT)

APB_SARADC_ADC_DONE_INT_CLR Write 1 to clear the **APB_SARADC_ADC_DONE_INT** interrupt.
(WT)

APB_SARADC_APB_ADC_TRANS
APB_SARADC_APB_ADC_RESET_FSM

(reserved)

APB_SARADC_APB_ADC_EOF_NUM

APB_SARADC_APB_ADC_TRANS Configures whether to let DIG ADC controller use GDMA.
 0: No effect
 1: DIG ADC controller uses DMA
 (R/W)

Chapter 37

Analog Voltage Comparator

37.1 Introduction

ESP32-C5 integrates an analog voltage comparator. The comparators rely on special pads that support voltage comparison functionality to monitor voltage changes on these pads. The analog voltage comparator has two pads associated with it, for the main voltage and the reference voltage respectively. The voltage comparison result generated by the analog voltage comparator can be used as Event Task Matrix (ETM) events to drive ETM tasks of other peripherals or trigger interrupts.

37.2 Feature List

The analog voltage comparator has the following features:

- Voltage comparison
 - Configurable voltage comparison mode
 - Configurable reference voltage
- Interrupt upon changes of voltage comparison result
- ETM event generation

37.3 Architectural Overview

Figure 37.3-1 shows the structure of the analog voltage comparator.

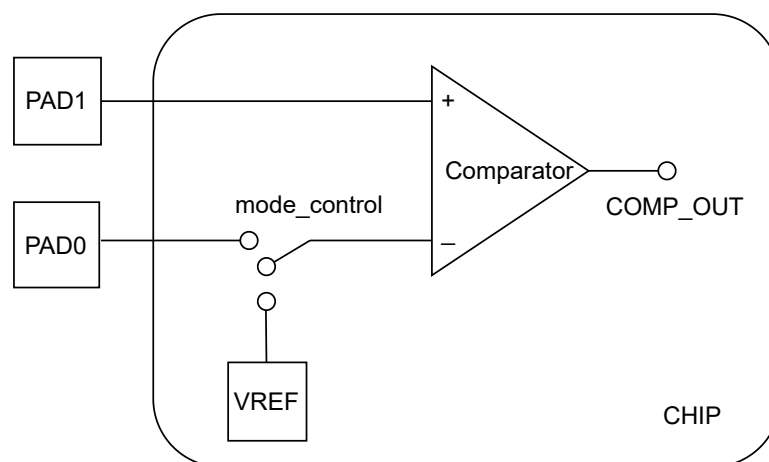


Figure 37.3-1. Analog Voltage Comparator Architecture

As shown in the figure above, the analog voltage comparator consists of:

- Two pad interfaces:
 - PAD1: Main voltage interface
 - PADO: External reference voltage interface
- VREF: Internal reference voltage interface
- mode_control: Register-controlled switch for selecting the reference voltage source
- COMP_OUT: Comparator voltage comparison result, which is an on-chip internal signal that cannot be directly accessed externally
 - If the main voltage is higher than the reference voltage, the COMP_OUT signal is high.
 - If the main voltage is lower than the reference voltage, the COMP_OUT signal is low.

37.4 Functional Description

The analog voltage comparator has the following functions:

- **Voltage comparison**

The analog voltage comparator relies on specialized pads that support voltage comparison. Table below shows the mapping between the PAD of analog voltage comparator and GPIO pins.

Table 37.4-1. Mapping Between PAD and GPIO

Analog Voltage Comparator	PADO	PAD1
Analog voltage comparator	GPIO9	GPIO8

The comparison mode and reference voltage are configurable. For detailed configuration procedures, see [Section 37.7 Programming Procedures](#).

The voltage comparison results are output as the COMP_OUT signal.

- **Voltage comparison mode control**

The analog voltage comparator compares the main voltage with the reference voltage, which can either be internal or external. There are two voltage comparison modes depending on the reference voltage selection:

- Comparing the main voltage with the external reference voltage.
- Comparing the main voltage with the internal reference voltage.

The voltage comparison mode can be configured through [GPIO_EXT_PAD_COMP_CONFIG_0_REG](#).

- **Reference voltage configuration**

The internal reference voltage range is $0 \sim (0.7 \times VDDPST1)$ V with a step of $0.1 \times VDDPST1$ V, where VDDPST1 is the supply voltage of the analog voltage comparator which is equal to the power supply voltage of the chip (usually 3.3 V). The internal reference voltage value can be configured through LPSYSREG_DREF_COMP n ($n = 0 \sim 1$).

The external reference voltage is accessed directly from the pad with an input range of $0 \sim (0.7 \times V_{DDPST1})$ V, same as the internal reference voltage, requiring no additional configuration.

- **Voltage comparison interrupt processing**

The voltage comparison result, i.e., the COMP_OUT signal, can be either high or low. Whenever the output value changes, a corresponding interrupt signal is generated.

37.5 Event Task Matrix Feature

The analog voltage comparator on ESP32-C5 supports the Event Task Matrix (ETM) function, which allows analog voltage comparator ETM events to trigger any peripherals' ETM tasks. This section introduces the ETM events related to analog voltage comparator. For more information, please refer to [Chapter 8 Event Task Matrix \(ETM\)](#).

The analog voltage comparator can generate the following ETM events:

- GPIO_EVT_ZERO_DET_POS: Indicates that the COMP_OUT signal changes from low level to high level, i.e., the main voltage changes from below to above the reference voltage.
- GPIO_EVT_ZERO_DET_NEG: Indicates that the COMP_OUT signal changes from high level to low level, i.e., the main voltage changes from higher to lower than the reference voltage.

The analog voltage comparator does not support any ETM tasks.

37.6 Interrupts

ESP32-C5's analog voltage comparator can generate the GPIO_PAD_COMP_INT interrupt signal that will be sent to the [Interrupt Matrix](#).

There are several internal interrupt sources from analog voltage comparator that can generate the above interrupt signal. The interrupt sources from analog voltage comparator are listed with their trigger conditions and the resulted interrupt signal(s) in [Table 37.6-1](#).

Table 37.6-1. Analog Voltage Comparator's Internal Interrupt Sources

Internal Interrupt Source	Trigger Condition	Interrupt Signal
GPIO_COMP_ALL_INT	Any COMP_OUT signal transition occurs	GPIO_PAD_COMP_INT
GPIO_COMP_NEG_INT	The COMP_OUT signal changes from high level to low level	GPIO_PAD_COMP_INT
GPIO_COMP_POS_INT	The COMP_OUT signal changes from low level to high level	GPIO_PAD_COMP_INT

Note:

For definitions of *interrupt*, *interrupt signal*, *interrupt source*, and their correlations, please refer to [Chapter 3 Interrupt Matrix \[to be added later\]](#) > [Section 3.1 Interrupt Terminology](#).

Each interrupt source can be configured by a common set of registers that are described in Section [Interrupt Configuration Registers](#). The specific registers can be found in Section [5.18.1 HP GPIO Matrix Register Summary](#).

37.7 Programming Procedures

The programming procedure for the analog voltage comparator is as follows:

Note:

For more information about the register fields mentioned in this section, refer to Chapter [5 GPIO Matrix and IO MUX](#).

1. Set [PCR_IOMUX_FUNC_CLK_EN](#) to 1.
2. Enable the comparator by setting [GPIO_EXT_XPD_COMP_0](#) to 1.
3. Disable normal digital pad functions (including IE, OE, WPU, and WPD) for PAD1, and if using the external reference voltage, also for PADO. For detailed configuration, see Chapter [5 GPIO Matrix and IO MUX](#).
4. Configure the comparison mode by setting [GPIO_EXT_PAD_COMP_CONFIG_0_REG](#):
 - 0: Configures to compare the main voltage with the internal reference voltage;
 - 1: Configures to compare the main voltage with the external reference voltage.
5. Configure the internal reference voltage through [GPIO_EXT_DREF_COMP](#), which offers a voltage range of $0 \sim (0.7 \times VDDPST1)$ V with a step of $0.1 \times VDDPST1$ V.

Part VII

Appendix

This part contains the following information starting from the next page:

- [*Related Documentation and Resources*](#)
- [*Glossary*](#)
- [*Programming Reserved Register Field*](#)
- [*Interrupt Configuration Registers*](#)
- [*Revision History*](#)

Related Documentation and Resources

Related Documentation

- [ESP32-C5 Series Datasheet](#) – Specifications of the ESP32-C5 hardware.
- [ESP32-C5 Hardware Design Guidelines](#) – Guidelines on how to integrate the ESP32-C5 into your hardware product.
- [Certificates](#)
<https://espressif.com/en/support/documents/certificates>
- [ESP32-C5 Product/Process Change Notifications \(PCN\)](#)
<https://espressif.com/en/support/documents/pcns?keys=ESP32-C5>
- [ESP32-C5 Advisories](#) – Information on security, bugs, compatibility, component reliability.
<https://espressif.com/en/support/documents/advisories?keys=ESP32-C5>
- [Documentation Updates and Update Notification Subscription](#)
<https://espressif.com/en/support/download/documents>

Developer Zone

- [ESP-IDF Programming Guide for ESP32-C5](#) – Extensive documentation for the ESP-IDF development framework.
- [ESP-IDF](#) and other development frameworks on GitHub.
<https://github.com/espressif>
- [ESP32 BBS Forum](#) – Engineer-to-Engineer (E2E) Community for Espressif products where you can post questions, share knowledge, explore ideas, and help solve problems with fellow engineers.
<https://esp32.com/>
- [The ESP Journal](#) – Best Practices, Articles, and Notes from Espressif folks.
<https://blog.espressif.com/>
- See the tabs *SDKs and Demos*, *Apps*, *Tools*, *AT Firmware*.
<https://espressif.com/en/support/download/sdks-demos>

Products

- [ESP32-C5 Series SoCs](#) – Browse through all ESP32-C5 SoCs.
<https://espressif.com/en/products/socs?id=ESP32-C5>
- [ESP32-C5 Series Modules](#) – Browse through all ESP32-C5-based modules.
<https://espressif.com/en/products/modules?id=ESP32-C5>
- [ESP32-C5 Series DevKits](#) – Browse through all ESP32-C5-based devkits.
<https://espressif.com/en/products/devkits?id=ESP32-C5>
- [ESP Product Selector](#) – Find an Espressif hardware product suitable for your needs by comparing or applying filters.
<https://products.espressif.com/#/product-selector?language=en>

Contact Us

- See the tabs *Sales Questions*, *Technical Enquiries*, *Circuit Schematic & PCB Design Review*, *Get Samples* (Online stores), *Become Our Supplier*, *Comments & Suggestions*.
<https://espressif.com/en/contact-us/sales-questions>

Glossary

Abbreviations for Peripherals

ADC	ADC Controller
AES	AES (Advanced Encryption Standard) Accelerator
CAN FD	Controller Area Network Flexible Data-Rate
DSA	Digital Signature Algorithm
ECC	ECC (Elliptic Curve Cryptography) Accelerator
ECDSA	Elliptic Curve Digital Signature Algorithm
eFuse	eFuse Controller
ETM	Event Task Matrix
GDMA	GDMA (General Direct Memory Access) Controller
HMAC	HMAC (Hash-based Message Authentication Code) Accelerator
I2C	I2C (Inter-Integrated Circuit) Controller
I2S	I2S (Inter-IC Sound) Controller
LED	LED Control PWM (Pulse Width Modulation)
MCPWM	Motor Control PWM (Pulse Width Modulation)
PARLIO	Parallel IO Controller
PCNT	Pulse Count Controller
PMS	Permission Control
RMT	Remote Control Peripheral
RNG	Random Number Generator
RSA	RSA (Rivest Shamir Adleman) Accelerator
SDHOST	SD/MMC Host Controller
SHA	SHA (Secure Hash Algorithm) Accelerator
SPI	SPI (Serial Peripheral Interface) Controller
TIMG	Timer Group
TRACE	RISC-V Trace Encoder
TSENS	Temperature Sensor
UART	UART (Universal Asynchronous Receiver-Transmitter) Controller
WDT	Watchdog Timers
XTS_AES	External Memory Encryption and Decryption

Abbreviations Related to Registers

REG	Register.
SYSREG	System registers are a group of registers that control system reset, memory, clocks, software interrupts, power management, clock gating, etc.
ISO	Isolation. If a peripheral or other chip component is powered down, the pins, if any, to which its output signals are routed will go into a floating state. ISO registers isolate such pins and keep them at a certain determined value, so that the other non-powered-down peripherals/devices attached to these pins are not affected.

- NMI **Non-maskable interrupt** is a hardware interrupt that cannot be disabled or ignored by the CPU instructions. Such interrupts exist to signal the occurrence of a critical error.
- W1TS Abbreviation added to names of registers/fields to indicate that such register/field should be used to set a field in a corresponding register with a similar name. For example, the register `GPIO_ENABLE_W1TS_REG` should be used to set the corresponding fields in the register `GPIO_ENABLE_REG`.
- W1TC Same as *W1TS*, but used to clear a field in a corresponding register.

Access Types for Registers

Sections *Register Summary* and *Register Description* in TRM chapters specify access types for registers and their fields.

Most frequently used access types and their combinations are as follows:

• RO	• R/W/SS	• R/WS/SS/SC
• WO	• R/W/SS/SC	• R/SS/WTC
• WT	• R/WC/SS	• R/SC/WTC
• R/W	• R/WC/SC	• R/SS/SC/WTC
• R/W1	• R/WC/SS/SC	• RF/WF
• WL	• R/WS/SC	• R/SS/RC
• R/W/SC	• R/WS/SS	• varies

Descriptions of all access types are provided below.

- R **Read.** User application can read from this register/field; usually combined with other access types.
- RO **Read only.** User application can only read from this register/field.
- HRO **Hardware Read Only.** Only hardware can read from this register/field; used for storing default settings for variable parameters.
- W **Write.** User application can write to this register/field; usually combined with other access types.
- WO **Write only.** User application can only write to this register/field.
- W1 **Write Once.** User application can write to this register/field only once; only allowed to write 1; writing 0 is invalid.
- SS **Self set.** On a specified event, hardware automatically writes 1 to this register/field; used with 1-bit fields.
- SC **Self clear.** On a specified event, hardware automatically writes 0 to this register/field; used with 1-bit and multi-bit fields.
- SM **Self modify.** On a specified event, hardware automatically writes a specified value to this register/field; used with multi-bit fields.
- SU **Self update.** On a specified event, hardware automatically updates this register/field; used with multi-bit fields.
- RS **Read to set.** If user application reads from this register/field, hardware automatically writes 1 to it.
- RC **Read to clear.** If user application reads from this register/field, hardware automatically writes 0 to it.
- RF **Read from FIFO.** If user application writes new data to FIFO, the register/field automatically reads it.
- WF **Write to FIFO.** If user application writes new data to this register/field, it automatically passes the data to FIFO via APB bus.
- WS **Write any value to set.** If user application writes to this register/field, hardware automatically sets this register/field.

- W1S **Write 1 to set.** If user application writes 1 to this register/field, hardware automatically sets this register/field.
- W0S **Write 0 to set.** If user application writes 0 to this register/field, hardware automatically sets this register/field.
- WC **Write any value to clear.** If user application writes to this register/field, hardware automatically clears this register/field.
- W1C **Write 1 to clear.** If user application writes 1 to this register/field, hardware automatically clears this register/field.
- W0C **Write 0 to clear.** If user application writes 0 to this register/field, hardware automatically clears this register/field.
- WT **Write 1 to trigger an event.** If user application writes 1 to this field, this action triggers an event (pulse in the APB bus) or clears a corresponding WTC field (see WTC).
- WTC **Write to clear.** Hardware automatically clears this field if user application writes 1 to the corresponding WT field (see WT).
- W1T **Write 1 to toggle.** If user application writes 1 to this field, hardware automatically inverts the corresponding field; otherwise - no effect.
- W0T **Write 0 to toggle.** If user application writes 0 to this field, hardware automatically inverts the corresponding field; otherwise - no effect.
- WL **Write if a lock is deactivated.** If the lock is deactivated, user application can write to this register/field.
- varies **The access type varies.** Different fields of this register might have different access types.

Programming Reserved Register Field

Introduction

A field in a register is reserved if the field is not open to users, or produces unpredictable results if configured to values other than defaults.

Programming Reserved Register Field

The reserved fields should not be modified. It is not possible to write only part of a register since registers must always be written as a whole. As a result, to write an entire register that contains reserved fields, you can choose one of the following two options:

1. Read the value of the register, modify only the fields you want to configure and then write back the value so that reserved fields are untouched.

OR

2. Modify only the fields you want to configure and write back the default value of the reserved fields. The default value of a field is provided in the "Reset" line of a register diagram. For example, the default value of Field_A in [Register X](#) is 1.

Register 371. Register X (Address)

(reserved)												Field_C				(reserved)												Field_B		Field_A			
31												20		19		16		15		2												1	0
0 0 0 0 0 0 0 0 0 0 0 0												0000				0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0												0	1	Reset			

Suppose you want to set Field_A, Field_B, and Field_C of [Register X](#) to 0x0, 0x1, and 0x2, you can:

- Use option 1 and fill in the reserved fields with the value you have just read. Suppose the register reads as 0x0000_0003. Then, you can modify the fields you want to configure, thus writing 0x0002_0002 to the register.
- Use option 2 and fill in the reserved fields with their defaults, thus writing 0x0002_0002 to the register.

Interrupt Configuration Registers

Generally, the peripherals' internal interrupt sources can be configured by the following common set of registers:

- **RAW** (Raw Interrupt Status) register: This register indicates the raw interrupt status. Each bit in the register represents a specific internal interrupt source. When an interrupt source triggers, its RAW bit is set to 1.
- **ENA** (Enable) register: This register is used to enable or disable the internal interrupt sources. Each bit in the ENA register corresponds to an internal interrupt source.

By manipulating the ENA register, you can mask or unmask individual internal interrupt source as needed. When an internal interrupt source is masked (disabled), it will not generate an interrupt signal, but its value can still be read from the RAW register.

- **ST** (Status) register: This register reflects the status of enabled interrupt sources. Each bit in the ST register corresponds to a specific internal interrupt source. The ST bit being 1 means that both the corresponding RAW bit and ENA bit are 1, indicating that the interrupt source is triggered and not masked. The other combinations of the RAW bit and ENA bit will result in the ST bit being 0.

The configuration of ENA/RAW/ST registers is shown in Table 377-4.

- **CLR** (Clear) register: The CLR register is responsible for clearing the internal interrupt sources. Writing 1 to the corresponding bit in the CLR register clears the interrupt source.

Table 377-4. Configuration of ENA/RAW/ST Registers

ENA Bit Value	RAW Bit Value	ST Bit Value
0	Ignored	0
1	0	0
	1	1

Revision History

Date	Version	Release notes
2025-05-20	v0.1	Preliminary release



Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

ALL THIRD PARTY'S INFORMATION IN THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES TO ITS AUTHENTICITY AND ACCURACY.

NO WARRANTY IS PROVIDED TO THIS DOCUMENT FOR ITS MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, NOR DOES ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

Copyright © 2025 Espressif Systems (Shanghai) Co., Ltd. All rights reserved.

www.espressif.com