

# ESP32 Technical Reference Manual



**Espressif Systems**

September 28, 2016

# About This Manual

**ESP32 Technical Reference Manual** targets application developers. The manual provides detailed and complete information on how to use the ESP32 memory and peripherals.

## Release Notes

Date	Version	Release notes
2016.08	V1.0	Initial release.
2016.09	V1.1	Added Chapter <a href="#">I2C Controller</a> .

## Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice. THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR 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 © 2016 Espressif Inc. All rights reserved.**

# Contents

<b>1</b>	<b>System and Memory</b>	<b>8</b>
1.1	Introduction	8
1.2	Features	8
1.3	Functional Description	10
1.3.1	Address Mapping	10
1.3.2	Embedded Memory	10
1.3.2.1	Internal ROM 0	11
1.3.2.2	Internal ROM 1	11
1.3.2.3	Internal SRAM 0	12
1.3.2.4	Internal SRAM 1	12
1.3.2.5	Internal SRAM 2	13
1.3.2.6	DMA	13
1.3.2.7	RTC FAST Memory	13
1.3.2.8	RTC SLOW Memory	13
1.3.3	External Memory	13
1.3.4	Peripherals	14
1.3.4.1	Asymmetric PID Controller Peripheral	15
1.3.4.2	Non-Contiguous Peripheral Memory Ranges	15
1.3.4.3	Memory Speed	16
<b>2</b>	<b>Interrupt Matrix</b>	<b>17</b>
2.1	Introduction	17
2.2	Features	17
2.3	Functional Description	17
2.3.1	Peripheral Interrupt Source	17
2.3.2	CPU Interrupt	20
2.3.3	Allocate Peripheral Interrupt Sources to Peripheral Interrupt on CPU	20
2.3.4	CPU NMI Interrupt Mask	21
2.3.5	Query Current Interrupt Status of Peripheral Interrupt Source	21
<b>3</b>	<b>Reset and Clock</b>	<b>22</b>
3.1	System Reset	22
3.1.1	Introduction	22
3.1.2	Reset Source	22
3.2	System Clock	23
3.2.1	Introduction	23
3.2.2	Clock Source	24
3.2.3	CPU Clock	24
3.2.4	Peripheral Clock	25
3.2.4.1	APB_CLK Source	25
3.2.4.2	REF_TICK Source	26
3.2.4.3	LEDC_SCLK Source	26
3.2.4.4	APLL_SCLK Source	26
3.2.4.5	PLL_D2_CLK Source	26

3.2.4.6	Clock Source Considerations	27
3.2.5	Wi-Fi BT Clock	27
3.2.6	RTC Clock	27
<b>4</b>	<b>IO_MUX and GPIO Matrix</b>	<b>28</b>
4.1	Introduction	28
4.2	Peripheral Input via GPIO Matrix	29
4.2.1	Summary	29
4.2.2	Functional Description	29
4.2.3	Simple GPIO Input	30
4.3	Peripheral Output via GPIO Matrix	30
4.3.1	Summary	30
4.3.2	Functional Description	30
4.3.3	Simple GPIO Output	31
4.4	Direct I/O via IO_MUX	31
4.4.1	Summary	31
4.4.2	Functional Description	31
4.5	RTC IO_MUX for Low Power and Analog I/O	32
4.5.1	Summary	32
4.5.2	Functional Description	32
4.6	Light-sleep Mode Pin Functions	32
4.7	Pad Hold Feature	32
4.8	I/O Pad Power Supply	33
4.8.1	VDD_SDIO Power Domain	33
4.9	Peripheral Signal List	33
4.10	IO_MUX Pad List	38
4.11	RTC_MUX Pin List	39
4.12	Register Summary	40
4.13	Registers	45
<b>5</b>	<b>I2C Controller</b>	<b>66</b>
5.1	Overview	66
5.2	Features	66
5.3	Functional Description	66
5.3.1	Introduction	66
5.3.2	Architecture	67
5.3.3	I2C Bus Timing	68
5.3.4	I2C cmd Structure	68
5.3.5	I2C Master Writes to Slave	69
5.3.6	I2C Master Reads from Slave	71
5.3.7	Interrupts	73
5.4	Register summary	74
5.5	Registers	76
<b>6</b>	<b>LED_PWM</b>	<b>87</b>
6.1	Introduction	87
6.2	Functional Description	87

6.2.1	Architecture	87
6.2.2	Timers	88
6.2.3	Channels	88
6.2.4	Interrupts	89
6.3	Register Summary	89
6.4	Registers	92

## 7 Remote Controller Peripheral 102

7.1	Introduction	102
7.2	Functional Description	102
7.2.1	RMT Architecture	102
7.2.2	RMT RAM	103
7.2.3	Clock	103
7.2.4	Transmitter	103
7.2.5	Receiver	104
7.2.6	Interrupts	104
7.3	Register Summary	105
7.4	Registers	106

## 8 PULSE\_CNT 111

8.1	Introduction	111
8.2	Functional Description	111
8.2.1	Architecture	111
8.2.2	Counter Channel Inputs	111
8.2.3	Watchpoints	112
8.2.4	Examples	113
8.2.5	Interrupts	113
8.3	Register Summary	113
8.4	Registers	115

## 9 64-bit Timers 119

9.1	Introduction	119
9.2	Functional Description	119
9.2.1	16-bit Prescaler	119
9.2.2	64-bit Time-base Counter	119
9.2.3	Alarm Generation	120
9.2.4	MWDT	120
9.2.5	Interrupts	120
9.3	Register summary	120
9.4	Registers	122

## 10 Watchdog Timers 129

10.1	Introduction	129
10.2	Features	129
10.3	Functional Description	129
10.3.1	Clock	129
10.3.1.1	Operating Procedure	130

10.3.1.2 Write Protection	130
10.3.1.3 Flash Boot Protection	130
10.3.1.4 Registers	131

## **11 AES Accelerator** 132

11.1 Introduction	132
11.2 Features	132
11.3 Functional Description	132
11.3.1 AES Algorithm Operations	132
11.3.2 Key, Plaintext and Ciphertext	132
11.3.3 Endianness	133
11.3.4 Encryption and Decryption Operations	135
11.3.5 Speed	135
11.4 Register summary	135
11.5 Registers	137

## **12 SHA Accelerator** 139

12.1 Introduction	139
12.2 Features	139
12.3 Functional Description	139
12.3.1 Padding and Parsing the Message	139
12.3.2 Message Digest	139
12.3.3 Hash Operation	140
12.3.4 Speed	140
12.4 Register Summary	140
12.5 Registers	142

## List of Tables

1	Address Mapping	10
2	Embedded Memory Address Mapping	11
3	Module with DMA	13
4	External Memory Address Mapping	14
5	Peripheral Address Mapping	14
6	PRO_CPU, APP_CPU Interrupt Configuration	18
7	CPU Interrupts	20
8	PRO_CPU and APP_CPU reset reason values	22
9	CPU_CLK Source	24
10	CPU_CLK Derivation	25
11	Peripheral Clock Usage	25
12	APB_CLK Derivation	26
13	REF_TICK Derivation	26
14	LEDC_SCLK Derivation	26
15	IO_MUX Light-sleep Pin Function Registers	32
16	GPIO Matrix Peripheral Signals	34
17	IO_MUX Pad Summary	38
18	RTC_MUX Pin Summary	39
27	Operation Mode	132
28	AES Text Endianness	133
29	AES-128 Key Endianness	134
30	AES-192 Key Endianness	134
31	AES-256 Key Endianness	134

## List of Figures

1	System Structure	9
2	System Address Mapping	9
3	Interrupt Matrix Structure	17
4	System Reset	22
5	System Clock	23
6	IO_MUX, RTC IO_MUX and GPIO Matrix Overview	28
7	Peripheral Input via IO_MUX, GPIO Matrix	29
8	Output via GPIO Matrix	31
9	ESP32 I/O Pad Power Sources	33
10	I2C Master Architecture	67
11	I2C Slave Architecture	67
12	I2C Sequence Chart	68
13	Structure of The I2C Command Register	68
14	I2C Master Writes to Slave with 7-bit Address	69
15	I2C Master Writes to Slave with 10-bit Address	70
16	I2C Master Writes to addrM in RAM of Slave with 7-bit Address	70
17	I2C Master Writes to Slave with 7-bit Address in Two Segments	71
18	I2C Master Reads from Slave with 7-bit Address	71
19	I2C Master Reads from Slave with 10-bit Address	72
20	I2C Master Reads N Bytes of Data from addrM in Slave with 7-bit Address	72
21	I2C Master Reads from Slave with 7-bit Address in Two Segments	73
22	LED_PWM Architecture	87
23	LED_PWM High-speed Channel Diagram	87
24	LED PWM Output Signal Diagram	88
25	Output Signal Diagram of Gradient Duty Cycle	89
26	RMT Architecture	102
27	Data Structure	103
28	PULSE_CNT Architecture	111
29	PULSE_CNT Upcounting Diagram	113
30	PULSE_CNT Downcounting Diagram	113



# 1. System and Memory

## 1.1 Introduction

The ESP32 is a dual-core system with two Harvard Architecture Xtensa LX6 CPUs. All embedded memory, external memory and peripherals are located on the data bus and/or the instruction bus of these CPUs.

With some minor exceptions (see below), the address mapping of two CPUs is symmetric, meaning they use the same addresses to access the same memory. Multiple peripherals in the system can access embedded memory via DMA.

The two CPUs are named “PRO\_CPU” and “APP\_CPU” (for “protocol” and “application”), however for most purposes the two CPUs are interchangeable.

## 1.2 Features

- Address Space
  - Symmetric address mapping
  - 4 GB (32-bit) address space for both data bus and instruction bus
  - 1296 KB embedded memory address space
  - 19704 KB external memory address space
  - 512 KB peripheral address space
  - Some embedded and external memory regions can be accessed by either data bus or instruction bus
  - 328 KB DMA address space
- Embedded Memory
  - 448 KB Internal ROM
  - 520 KB Internal SRAM
  - 8 KB RTC FAST Memory
  - 8 KB RTC SLOW Memory
- External Memory

Off-chip SPI memory can be mapped into the available address space as external memory. Parts of the embedded memory can be used as transparent cache for this external memory.

  - Supports up to 16 MB off-Chip SPI Flash.
  - Supports up to 8 MB off-Chip SPI SRAM.
- Peripherals
  - 41 peripherals
- DMA
  - 13 modules are capable of DMA operation

Figure 1 block diagram illustrates the system structure, the block diagram in Figure 2 illustrates the address map structure.

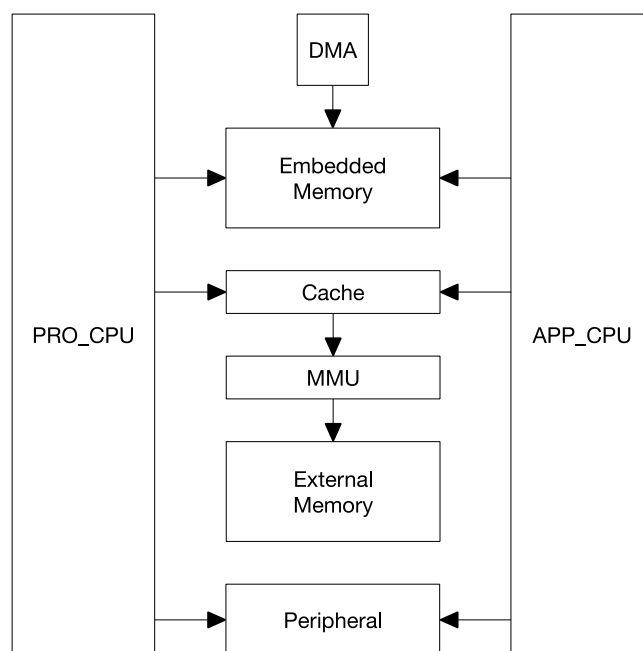


Figure 1: System Structure

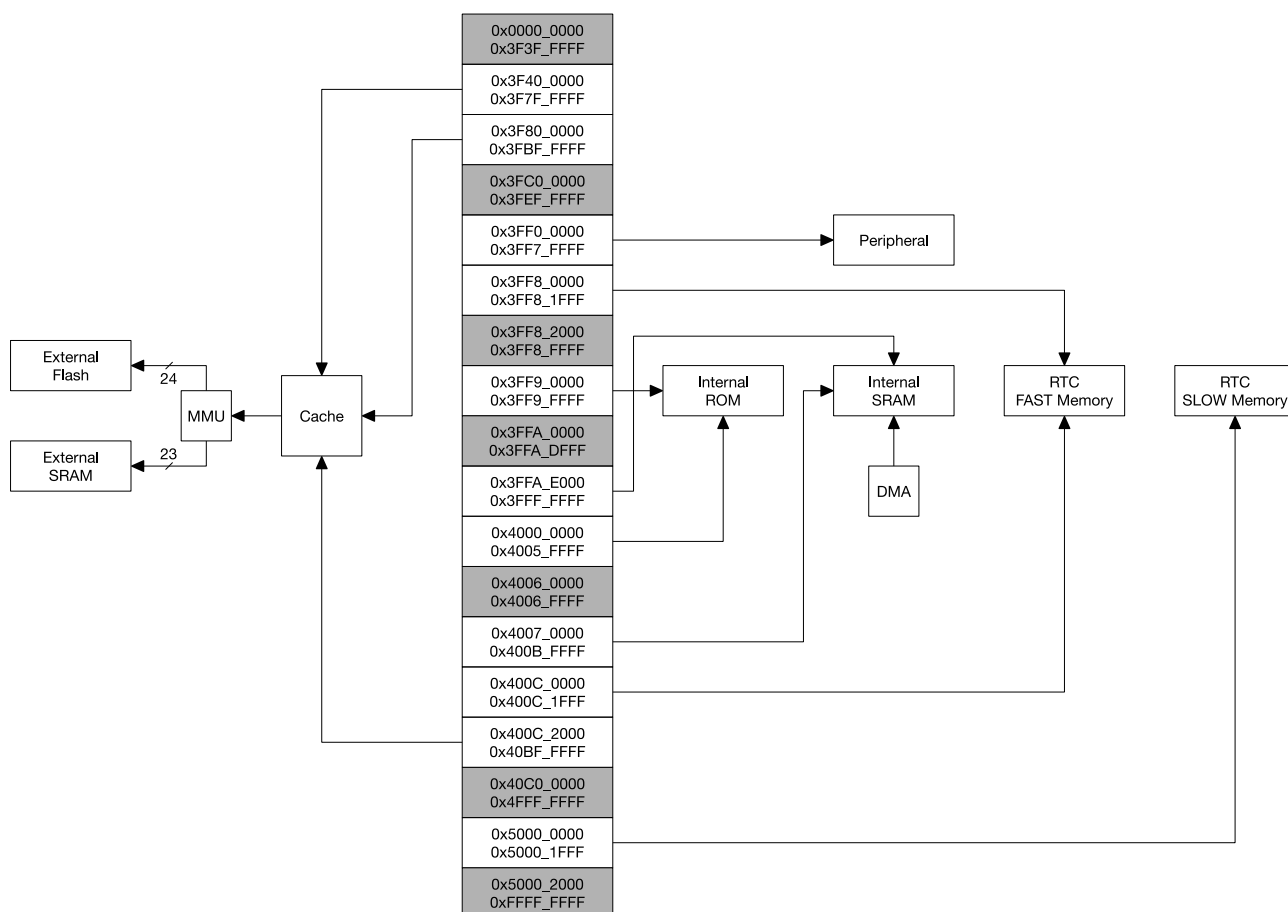


Figure 2: System Address Mapping

## 1.3 Functional Description

### 1.3.1 Address Mapping

Each of the two Harvard Architecture Xtensa LX6 CPUs has 4 GB (32-bit) address space. Address spaces are symmetric between the two CPUs.

Addresses below 0x4000\_0000 are serviced using the data bus. Addresses in the range 0x4000\_0000 ~ 0x4FFF\_FFFF are serviced using the instruction bus. Finally, addresses over and including 0x5000\_0000 are shared by the data and instruction bus.

The data bus and instruction bus are both little-endian: for example, byte addresses 0x0, 0x1, 0x2, 0x3 access the least significant, second least significant, second most significant, and most significant bytes of the 32-bit word stored at address 0x0, respectively. The CPU can access data bus addresses via aligned or non-aligned byte, half-word and word read and write operations. The CPU can read and write data through the instruction bus, but only in a **word aligned manner**; non-word-aligned access will cause a CPU exception.

Each CPU can directly access embedded memory through both the data bus and the instruction bus, external memory which is mapped into the address space (via transparent caching & MMU) and peripherals. Table 1 illustrates address ranges that can be accessed by each CPU's data bus and instruction bus.

Some embedded memories and some external memories can be accessed via the data bus or the instruction bus. In these cases, the same memory is available to either of the CPUs at two address ranges.

**Table 1: Address Mapping**

Bus Type	Boundary Address		Size	Target
	Low Address	High Address		
	0x0000_0000	0x3F3F_FFFF		Reserved
Data	0x3F40_0000	0x3F7F_FFFF	4 MB	External Memory
Data	0x3F80_0000	0x3FBF_FFFF	4 MB	External Memory
	0x3FC0_0000	0x3FEF_FFFF	3 MB	Reserved
Data	0x3FF0_0000	0x3FF7_FFFF	512 KB	Peripheral
Data	0x3FF8_0000	0x3FFF_FFFF	512 KB	Embedded Memory
Instruction	0x4000_0000	0x400C_1FFF	776 KB	Embedded Memory
Instruction	0x400C_2000	0x40BF_FFFF	11512 KB	External Memory
	0x40C0_0000	0x4FFF_FFFF	244 MB	Reserved
Data Instruction	0x5000_0000	0x5000_1FFF	8 KB	Embedded Memory
	0x5000_2000	0xFFFF_FFFF		Reserved

### 1.3.2 Embedded Memory

The Embedded Memory consists of four segments: internal ROM (448 KB), internal SRAM (520 KB), RTC FAST memory (8 KB) and RTC SLOW memory (8 KB).

The 448 KB internal ROM is divided into two parts: Internal ROM 0 (384 KB) and Internal ROM 1 (64 KB).

The 520 KB internal SRAM is divided into three parts: Internal SRAM 0 (192 KB), Internal SRAM 1 (128 KB), and Internal SRAM 2 (200 KB).

RTC FAST Memory and RTC SLOW Memory are both implemented as SRAM.

Table 2 lists all embedded memories and their address ranges on the data and instruction buses.

**Table 2: Embedded Memory Address Mapping**

Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Data	0x3FF8_0000	0x3FF8_1FFF	8 KB	RTC FAST Memory	PRO_CPU Only
	0x3FF8_2000	0x3FF8_FFFF	56 KB	Reserved	-
Data	0x3FF9_0000	0x3FF9_FFFF	64 KB	Internal ROM 1	-
	0x3FFA_0000	0x3FFA_DFFF	56 KB	Reserved	-
Data	0x3FFA_E000	0x3FFD_FFFF	200 KB	Internal SRAM 2	DMA
Data	0x3FFE_0000	0x3FFF_FFFF	128 KB	Internal SRAM 1	DMA
Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Instruction	0x4000_0000	0x4000_7FFF	32 KB	Internal ROM 0	Remap
Instruction	0x4000_8000	0x4005_FFFF	352 KB	Internal ROM 0	-
	0x4006_0000	0x4006_FFFF	64 KB	Reserved	-
Instruction	0x4007_0000	0x4007_FFFF	64 KB	Internal SRAM 0	Cache
Instruction	0x4008_0000	0x4009_FFFF	128 KB	Internal SRAM 0	-
Instruction	0x400A_0000	0x400A_FFFF	64 KB	Internal SRAM 1	-
Instruction	0x400B_0000	0x400B_7FFF	32 KB	Internal SRAM 1	Remap
Instruction	0x400B_8000	0x400B_FFFF	32 KB	Internal SRAM 1	-
Instruction	0x400C_0000	0x400C_1FFF	8 KB	RTC FAST Memory	PRO_CPU Only
Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Data Instruction	0x5000_0000	0x5000_1FFF	8 KB	RTC SLOW Memory	-

### 1.3.2.1 Internal ROM 0

The capacity of Internal ROM 0 is 384 KB, It is accessible by both CPUs through the address range 0x4000\_0000 ~ 0x4005\_FFFF, which is on the instruction bus.

The address range of the first 32 KB of the ROM 0 (0x4000\_0000 ~ 0x4000\_7FFF) can be re-mapped to access a part of Internal SRAM 1 that normally resides in the memory range 0x400B\_0000 ~ 0x400B\_7FFF instead. While remapping, such 32 KB SRAM can not be accessed by address range 0x400B\_0000 ~ 0x400B\_7FFF any more, but it can still be accessible through the data bus (0x3FFE\_8000 ~ 0x3FFE\_FFFF). This can be done on a per-CPU basis: setting bit 0 of register DPORT\_PRO\_BOOT\_REMAP\_CTRL\_REG or DPORT\_APP\_BOOT\_REMAP\_CTRL\_REG will remap SRAM for the PRO\_CPU and APP\_CPU, respectively.

### 1.3.2.2 Internal ROM 1

The capacity of Internal ROM 1 is 64 KB. It can be read by either CPU at address range 0x3FF9\_0000 ~ 0x3FF9\_FFFF of the data bus.

### 1.3.2.3 Internal SRAM 0

The capacity of Internal SRAM 0 is 192 KB. Hardware can be configured to use the first 64KB to cache external memory access. When not used as cache, the first 64KB can be read and written by either CPU at addresses 0x4007\_0000 ~ 0x4007\_7FFF of the instruction bus. The remaining 128 KB can always be read and written by either CPU at addresses 0x4007\_8000 ~ 0x4007\_FFFF of instruction bus.

### 1.3.2.4 Internal SRAM 1

The capacity of Internal SRAM 1 is 128 KB. Either CPU can read and write this memory at addresses 0x3FFE\_0000 ~ 0x3FFF\_FFFF of the data bus, and also at addresses 0x400A\_0000 ~ 0x400B\_FFFF of the instruction bus.

The address range accessed via the instruction bus is in reverse order (word-wise) compared to access via the data bus. That is to say, address

0x3FFE\_0000 and 0x400B\_FFFC access the same word

0x3FFE\_0004 and 0x400B\_FFF8 access the same word

0x3FFE\_0008 and 0x400B\_FFF4 access the same word

.....

0x3FFF\_FFF4 and 0x400A\_0008 access the same word

0x3FFF\_FFF8 and 0x400A\_0004 access the same word

0x3FFF\_FFFC and 0x400A\_0000 access the same word

The data bus and instruction bus of the CPU are still both little endian, so the byte order of individual words is not reversed between address spaces. For example, address

0x3FFE\_0000 accesses the least significant byte in the word accessed by 0x400B\_FFFC.

0x3FFE\_0001 accesses the second least significant byte in the word accessed by 0x400B\_FFFC.

0x3FFE\_0002 accesses the second most significant byte in the word accessed by 0x400B\_FFFC.

0x3FFE\_0003 accesses the most significant byte in the word accessed by 0x400B\_FFFC.

0x3FFE\_0004 accesses the least significant byte in the word accessed by 0x400B\_FFF8.

0x3FFE\_0005 accesses the second least significant byte in the word accessed by 0x400B\_FFF8.

0x3FFE\_0006 accesses the second most significant byte in the word accessed by 0x400B\_FFF8.

0x3FFE\_0007 accesses the most significant byte in the word accessed by 0x400B\_FFF8.

.....

0x3FFF\_FFF8 accesses the least significant byte in the word accessed by 0x400A\_0004.

0x3FFF\_FFF9 accesses the second least significant byte in the word accessed by 0x400A\_0004.

0x3FFF\_FFFA accesses the second most significant byte in the word accessed by 0x400A\_0004.

0x3FFF\_FFFB accesses the most significant byte in the word accessed by 0x400A\_0004.

0x3FFF\_FFFC accesses the least significant byte in the word accessed by 0x400A\_0000.

0x3FFF\_FFFD accesses the second most significant byte in the word accessed by 0x400A\_0000.

0x3FFF\_FFFE accesses the second most significant byte in the word accessed by 0x400A\_0000.

0x3FFF\_FFFF accesses the most significant byte in the word accessed by 0x400A\_0000.

Part of this memory can be remapped to the ROM 0 address space. See [Internal Rom 0](#) for more information.

### 1.3.2.5 Internal SRAM 2

The capacity of Internal SRAM 2 is 200 KB. It can be read and written by either CPU at addresses 0x3FFA\_E000 ~ 0x3FFD\_FFFF on the data bus.

### 1.3.2.6 DMA

DMA uses the same addressing as the CPU data bus to read and write Internal SRAM 1 and Internal SRAM 2. This means DMA uses address range 0x3FFE\_0000 ~ 0x3FFF\_FFFF to read and write Internal SRAM 1 and address range 0x3FFA\_E000 ~ 0x3FFD\_FFFF to read and write Internal SRAM 2.

In the ESP32, 13 peripherals are equipped with DMA. Table 3 lists these peripherals.

**Table 3: Module with DMA**

UART0	UART1	UART2
SPI1	SPI2	SPI3
I2S0	I2S1	
SDIO Slave	SDMMC	
EMAC		
BT	WIFI	

### 1.3.2.7 RTC FAST Memory

RTC FAST Memory is 8 KB of SRAM. It can be read and written by PRO\_CPU only at address range 0x3FF8\_0000 ~ 0x3FF8\_1FFF on the data bus or address range 0x400C\_0000 ~ 0x400C\_1FFF on the instruction bus. Unlike most other memory regions, RTC FAST memory cannot be accessed by the APP\_CPU.

The two address ranges of PRO\_CPU access RTC FAST Memory in the same order, so for example, address 0x3FF8\_0000 and 0x400C\_0000 access the same word. **On the APP\_CPU, these address ranges do not provide access to RTC FAST Memory or any other memory location.**

### 1.3.2.8 RTC SLOW Memory

RTC SLOW Memory is 8 KB of SRAM which can be read from and written by either CPU at address range 0x5000\_0000 ~ 0x5000\_1FFF. This address range is shared by both the data bus and the instruction bus.

## 1.3.3 External Memory

The ESP32 can access external SPI flash and SPI SRAM as external memory. Table 4 provides a list of external memories that can be accessed by either CPU at a range of addresses on the data and instruction buses. When a CPU accesses external memory through the Cache and MMU, the cache will map the CPU's address to an external physical memory address (in the external memory's address space), according to the MMU settings. Due to this address mapping, the ESP32 can address up to 16 MB External Flash and 8 MB External SRAM.

**Table 4: External Memory Address Mapping**

Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Data	0x3F40_0000	0x3F7F_FFFF	4 MB	External Flash	Read
Data	0x3F80_0000	0x3FBF_FFFF	4 MB	External SRAM	Read and Write
Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Instruction	0x400C_2000	0x40BF_FFFF	11512 KB	External Flash	Read

### 1.3.4 Peripherals

The ESP32 has 41 peripherals. Table 5 specifically describes the peripherals their respective address ranges.

Almost all peripheral modules can be accessed by either CPU at the same address, the only exception being the PID Controller.

**Table 5: Peripheral Address Mapping**

Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Data	0x3FF0_0000	0x3FF0_0FFF	4 KB	DPort Register	
Data	0x3FF0_1000	0x3FF0_1FFF	4 KB	AES Accelerator	
Data	0x3FF0_2000	0x3FF0_2FFF	4 KB	RSA Accelerator	
Data	0x3FF0_3000	0x3FF0_3FFF	4 KB	SHA Accelerator	
Data	0x3FF0_4000	0x3FF0_4FFF	4 KB	Secure Boot	
	0x3FF0_5000	0x3FF0_FFFF	44 KB	Reserved	
Data	0x3FF1_0000	0x3FF1_3FFF	16 KB	Cache MMU Table	
	0x3FF1_4000	0x3FF1_EFFF	44 KB	Reserved	
Data	0x3FF1_F000	0x3FF1_FFFF	4 KB	PID Controller	Per-CPU peripheral
	0x3FF2_0000	0x3FF3_FFFF	128 KB	Reserved	
Data	0x3FF4_0000	0x3FF4_0FFF	4 KB	UART0	
	0x3FF4_1000	0x3FF4_1FFF	4 KB	Reserved	
Data	0x3FF4_2000	0x3FF4_2FFF	4 KB	SPI1	
Data	0x3FF4_3000	0x3FF4_3FFF	4 KB	SPI0	
Data	0x3FF4_4000	0x3FF4_4FFF	4 KB	GPIO	
	0x3FF4_5000	0x3FF4_7FFF	12 KB	Reserved	
Data	0x3FF4_8000	0x3FF4_8FFF	4 KB	RTC	
Data	0x3FF4_9000	0x3FF4_9FFF	4 KB	IO MUX	
	0x3FF4_A000	0x3FF4_AFFF	4 KB	Reserved	
Data	0x3FF4_B000	0x3FF4_BFFF	4 KB	SDIO Slave	One of three parts
Data	0x3FF4_C000	0x3FF4_CFFF	4 KB	UDMA1	
	0x3FF4_D000	0x3FF4_EFFF	8 KB	Reserved	
Data	0x3FF4_F000	0x3FF4_FFFF	4 KB	I2S0	
Data	0x3FF5_0000	0x3FF5_0FFF	4 KB	UART1	
	0x3FF5_1000	0x3FF5_2FFF	8 KB	Reserved	
Data	0x3FF5_3000	0x3FF5_3FFF	4 KB	I2C0	
Data	0x3FF5_4000	0x3FF5_4FFF	4 KB	UDMA0	

Bus Type	Boundary Address		Size	Target	Comment
	Low Address	High Address			
Data	0x3FF5_5000	0x3FF5_5FFF	4 KB	SDIO Slave	One of three parts
Data	0x3FF5_6000	0x3FF5_6FFF	4 KB	RMT	
Data	0x3FF5_7000	0x3FF5_7FFF	4 KB	PCNT	
Data	0x3FF5_8000	0x3FF5_8FFF	4 KB	SDIO Slave	One of three parts
Data	0x3FF5_9000	0x3FF5_9FFF	4 KB	LED PWM	
Data	0x3FF5_A000	0x3FF5_AFFF	4 KB	Efuse Controller	
Data	0x3FF5_B000	0x3FF5_BFFF	4 KB	Flash Encryption	
	0x3FF5_C000	0x3FF5_DFFF	8 KB	Reserved	
Data	0x3FF5_E000	0x3FF5_EFFF	4 KB	PWM0	
Data	0x3FF5_F000	0x3FF5_FFFF	4 KB	TIMG0	
Data	0x3FF6_0000	0x3FF6_0FFF	4 KB	TIMG1	
	0x3FF6_1000	0x3FF6_3FFF	12 KB	Reserved	
Data	0x3FF6_4000	0x3FF6_4FFF	4 KB	SPI2	
Data	0x3FF6_5000	0x3FF6_5FFF	4 KB	SPI3	
Data	0x3FF6_6000	0x3FF6_6FFF	4 KB	SYSCON	
Data	0x3FF6_7000	0x3FF6_7FFF	4 KB	I2C1	
Data	0x3FF6_8000	0x3FF6_8FFF	4 KB	SDMMC	
Data	0x3FF6_9000	0x3FF6_AFFF	8 KB	EMAC	
	0x3FF6_B000	0x3FF6_BFFF	4 KB	Reserved	
Data	0x3FF6_C000	0x3FF6_CFFF	4 KB	PWM1	
Data	0x3FF6_D000	0x3FF6_DFFF	4 KB	I2S1	
Data	0x3FF6_E000	0x3FF6_EFFF	4 KB	UART2	
Data	0x3FF6_F000	0x3FF6_FFFF	4 KB	PWM2	
Data	0x3FF7_0000	0x3FF7_0FFF	4 KB	PWM3	
	0x3FF7_1000	0x3FF7_4FFF	16 KB	Reserved	
Data	0x3FF7_5000	0x3FF7_5FFF	4 KB	RNG	
	0x3FF7_6000	0x3FF7_FFFF	40 KB	Reserved	

#### 1.3.4.1 Asymmetric PID Controller Peripheral

There are two PID Controllers in the system. They serve the PRO\_CPU and the APP\_CPU, respectively. **The PRO\_CPU and the APP\_CPU can only access their own PID Controller and not their counterpart's PID Controller.** Each CPU uses the same memory range 0x3FF1\_F000 ~ 3FF1\_FFFF to access its own PID Controller.

#### 1.3.4.2 Non-Contiguous Peripheral Memory Ranges

The SDIO Slave peripheral consists of three parts and the two CPUs use non-contiguous addresses to access these. The three parts are accessed at the address ranges 0x3FF4\_B000 ~ 3FF4\_BFFF, 0x3FF5\_5000 ~ 3FF5\_5FFF and 0x3FF5\_8000 ~ 3FF5\_8FFF of each CPU's data bus. Similar to other peripherals, access to this peripheral is identical for both CPUs.



### 1.3.4.3 Memory Speed

The ROM as well as the SRAM are both clocked from CPU\_CLK and can be accessed by the CPU in a single cycle. The RTC FAST memory is clocked from the APB\_CLOCK and the RTC SLOW memory from the FAST\_CLOCK, so access to these memories may be slower. DMA uses the APB\_CLK to access memory.

Internally, the SRAM is organized in 32K-sized banks. Each CPU and DMA channel can access the SRAM at full speed and simultaneously, provided they access address different memory banks.

## 2. Interrupt Matrix

### 2.1 Introduction

The Interrupt Matrix embedded in the ESP32 independently allocates peripheral interrupt sources to the two CPUs' peripheral interrupts. This configuration is highly flexible in order to meet many different needs.

### 2.2 Features

- Accepts 71 peripheral interrupt sources as input.
- Generates 26 peripheral interrupt sources per CPU as output (52 total).
- CPU NMI Interrupt Mask.
- Queries current interrupt status of peripheral interrupt sources.

The structure of the Interrupt Matrix is shown in Figure 3.

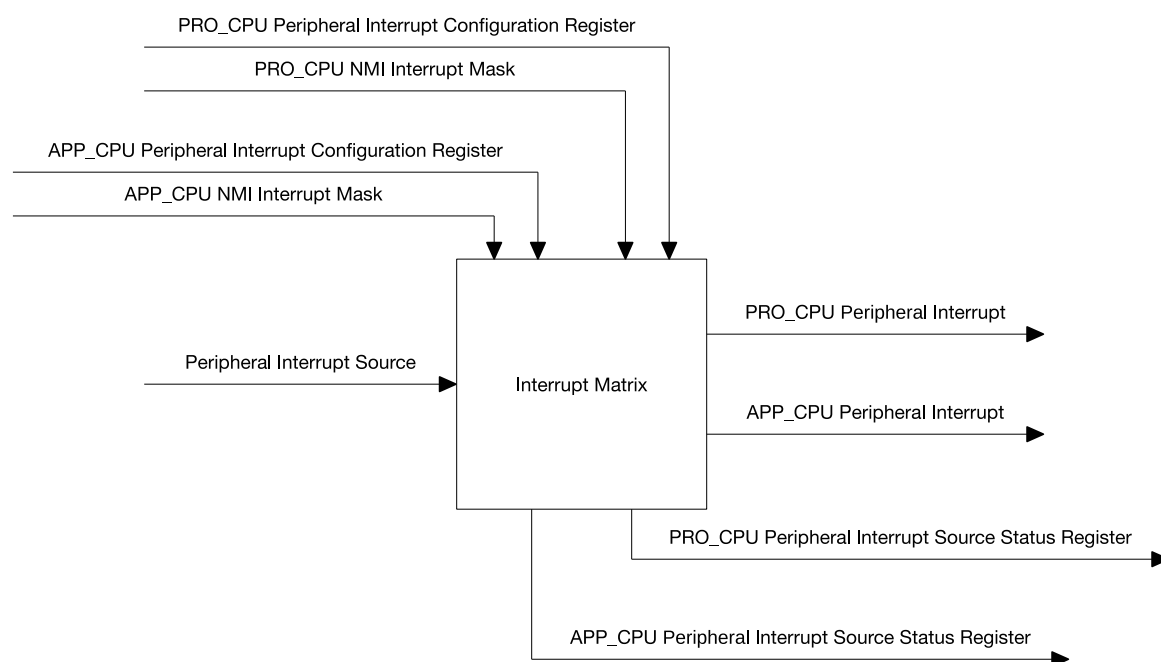


Figure 3: Interrupt Matrix Structure

### 2.3 Functional Description

#### 2.3.1 Peripheral Interrupt Source

ESP32 has 71 peripheral interrupt sources in total. All peripheral interrupt sources are listed in table 6. 67 of 71 ESP32 peripheral interrupt sources can be allocated to either CPU.

The four remaining peripheral interrupt sources are CPU-specific, two per CPU. GPIO\_INTERRUPT\_PRO and GPIO\_INTERRUPT\_PRO\_NMI can only be allocated to PRO\_CPU. GPIO\_INTERRUPT\_APP and GPIO\_INTERRUPT\_APP\_NMI can only be allocated to APP\_CPU. As a result, PRO\_CPU and APP\_CPU each have 69 peripheral interrupt sources.

Table 6: PRO\_CPU, APP\_CPU Interrupt Configuration

PRO_CPU				APP_CPU			
Peripheral Interrupt Configuration Register	Status Register		Peripheral Interrupt Source		Status Register		Peripheral Interrupt Configuration Register
	Bit	No.	Name	No.	Bit	No.	
PRO_MAC_INTR_MAP_REG	0	0	MAC_INTR	0	0	0	APP_MAC_INTR_MAP_REG
PRO_MAC_NMI_MAP_REG	1	1	MAC_NMI	1	1	1	APP_MAC_NMI_MAP_REG
PRO_BB_INT_MAP_REG	2	2	BB_INT	2	2	2	APP_BB_INT_MAP_REG
PRO_BT_MAC_INT_MAP_REG	3	3	BT_MAC_INT	3	3	3	APP_BT_MAC_INT_MAP_REG
PRO_BT_BB_INT_MAP_REG	4	4	BT_BB_INT	4	4	4	APP_BT_BB_INT_MAP_REG
PRO_BT_BB_NMI_MAP_REG	5	5	BT_BB_NMI	5	5	5	APP_BT_BB_NMI_MAP_REG
PRO_RWBTT_IRQ_MAP_REG	6	6	RWBTT_IRQ	6	6	6	APP_RWBTT_IRQ_MAP_REG
PRO_BT_BB_NMI_MAP_REG	5	5	BT_BB_NMI	5	5	5	APP_BT_BB_NMI_MAP_REG
PRO_RWBTT_IRQ_MAP_REG	6	6	RWBTT_IRQ	6	6	6	APP_RWBTT_IRQ_MAP_REG
PRO_RWBLE_IRQ_MAP_REG	7	7	RWBLE_IRQ	7	7	7	APP_RWBLE_IRQ_MAP_REG
PRO_RWBTT_NMI_MAP_REG	8	8	RWBTT_NMI	8	8	8	APP_RWBTT_NMI_MAP_REG
PRO_RWBLE_NMI_MAP_REG	9	9	RWBLE_NMI	9	9	9	APP_RWBLE_NMI_MAP_REG
PRO_SLCO_INTR_MAP_REG	10	10	SLCO_INTR	10	10	10	APP_SLCO_INTR_MAP_REG
PRO_SLC1_INTR_MAP_REG	11	11	SLC1_INTR	11	11	11	APP_SLC1_INTR_MAP_REG
PRO_UHCIO_INTR_MAP_REG	12	12	UHCIO_INTR	12	12	12	APP_UHCIO_INTR_MAP_REG
PRO_UHC1_INTR_MAP_REG	13	13	UHC1_INTR	13	13	13	APP_UHC1_INTR_MAP_REG
PRO_TG_T0_LEVEL_INT_MAP_REG	14	14	TG_T0_LEVEL_INT	14	14	14	APP_TG_T0_LEVEL_INT_MAP_REG
PRO_TG_T1_LEVEL_INT_MAP_REG	15	15	TG_T1_LEVEL_INT	15	15	15	APP_TG_T1_LEVEL_INT_MAP_REG
PRO_TG_WDT_LEVEL_INT_MAP_REG	16	16	TG_WDT_LEVEL_INT	16	16	16	APP_TG_WDT_LEVEL_INT_MAP_REG
PRO_TG_LACT_LEVEL_INT_MAP_REG	17	17	TG_LACT_LEVEL_INT	17	17	17	APP_TG_LACT_LEVEL_INT_MAP_REG
PRO_TG1_T0_LEVEL_INT_MAP_REG	18	18	TG1_T0_LEVEL_INT	18	18	18	APP_TG1_T0_LEVEL_INT_MAP_REG
PRO_TG1_T1_LEVEL_INT_MAP_REG	19	19	TG1_T1_LEVEL_INT	19	19	19	APP_TG1_T1_LEVEL_INT_MAP_REG
PRO_TG1_WDT_LEVEL_INT_MAP_REG	20	20	TG1_WDT_LEVEL_INT	20	20	20	APP_TG1_WDT_LEVEL_INT_MAP_REG
PRO_TG1_LACT_LEVEL_INT_MAP_REG	21	21	TG1_LACT_LEVEL_INT	21	21	21	APP_TG1_LACT_LEVEL_INT_MAP_REG
PRO_GPIO_INTERRUPT_PRO_MAP_REG	22	22	GPIO_INTERRUPT_PRO	22	22	22	APP_GPIO_INTERRUPT_APP_MAP_REG
PRO_GPIO_INTERRUPT_PRO_NMI_MAP_REG	23	23	GPIO_INTERRUPT_PRO_NMI	23	23	23	APP_GPIO_INTERRUPT_APP_NMI_MAP_REG
PRO_CPU_INTR_FROM_CPU_0_MAP_REG	24	24	CPU_INTR_FROM_CPU_0	24	24	24	APP_CPU_INTR_FROM_CPU_0_MAP_REG
PRO_CPU_INTR_FROM_CPU_1_MAP_REG	25	25	CPU_INTR_FROM_CPU_1	25	25	25	APP_CPU_INTR_FROM_CPU_1_MAP_REG
PRO_CPU_INTR_FROM_CPU_2_MAP_REG	26	26	CPU_INTR_FROM_CPU_2	26	26	26	APP_CPU_INTR_FROM_CPU_2_MAP_REG
PRO_CPU_INTR_FROM_CPU_3_MAP_REG	27	27	CPU_INTR_FROM_CPU_3	27	27	27	APP_CPU_INTR_FROM_CPU_3_MAP_REG
PRO_SPI_INTR_0_MAP_REG	28	28	SPI_INTR_0	28	28	28	APP_SPI_INTR_0_MAP_REG
PRO_SPI_INTR_1_MAP_REG	29	29	SPI_INTR_1	29	29	29	APP_SPI_INTR_1_MAP_REG
PRO_SPI_INTR_2_MAP_REG	30	30	SPI_INTR_2	30	30	30	APP_SPI_INTR_2_MAP_REG
PRO_SPI_INTR_3_MAP_REG	31	31	SPI_INTR_3	31	31	31	APP_SPI_INTR_3_MAP_REG
PRO_I2S0_INT_MAP_REG	0	32	I2S0_INT	32	0	0	APP_I2S0_INT_MAP_REG
PRO_I2S1_INT_MAP_REG	1	33	I2S1_INT	33	1	1	APP_I2S1_INT_MAP_REG
PRO_UART_INTR_MAP_REG	2	34	UART_INTR	34	2	2	APP_UART_INTR_MAP_REG
PRO_UART1_INTR_MAP_REG	3	35	UART1_INTR	35	3	3	APP_UART1_INTR_MAP_REG
PRO_UART2_INTR_MAP_REG	4	36	UART2_INTR	36	4	4	APP_UART2_INTR_MAP_REG
PRO_SDIO_HOST_INTERRUPT_MAP_REG	5	37	SDIO_HOST_INTERRUPT	37	5	5	APP_SDIO_HOST_INTERRUPT_MAP_REG
PRO_EMAC_INT_MAP_REG	6	38	EMAC_INT	38	6	6	APP_EMAC_INT_MAP_REG
PRO_PWM0_INTR_MAP_REG	7	39	PWM0_INTR	39	7	7	APP_PWM0_INTR_MAP_REG
PRO_PWM1_INTR_MAP_REG	8	40	PWM1_INTR	40	8	8	APP_PWM1_INTR_MAP_REG
PRO_PWM2_INTR_MAP_REG	9	41	PWM2_INTR	41	9	9	APP_PWM2_INTR_MAP_REG
PRO_PWM3_INTR_MAP_REG	10	42	PWM3_INTR	42	10	10	APP_PWM3_INTR_MAP_REG
PRO_LEDC_INT_MAP_REG	11	43	LEDC_INT	43	11	11	APP_LEDC_INT_MAP_REG
PRO_EFUSE_INT_MAP_REG	12	44	EFUSE_INT	44	12	12	APP_EFUSE_INT_MAP_REG
PRO_CAN_INT_MAP_REG	13	45	CAN_INT	45	13	13	APP_CAN_INT_MAP_REG
PRO_RTC_CORE_INTR_MAP_REG	14	46	RTC_CORE_INTR	46	14	14	APP_RTC_CORE_INTR_MAP_REG
PRO_RMT_INTR_MAP_REG	15	47	RMT_INTR	47	15	15	APP_RMT_INTR_MAP_REG
PRO_PCNT_INTR_MAP_REG	16	48	PCNT_INTR	48	16	16	APP_PCNT_INTR_MAP_REG
PRO_I2C_EXT0_INTR_MAP_REG	17	49	I2C_EXT0_INTR	49	17	17	APP_I2C_EXT0_INTR_MAP_REG
PRO_I2C_EXT1_INTR_MAP_REG	18	50	I2C_EXT1_INTR	50	18	18	APP_I2C_EXT1_INTR_MAP_REG
PRO_RSA_INTR_MAP_REG	19	51	RSA_INTR	51	19	19	APP_RSA_INTR_MAP_REG
PRO_SPI1_DMA_INT_MAP_REG	20	52	SPI1_DMA_INT	52	20	20	APP_SPI1_DMA_INT_MAP_REG

PRO_CPU				APP_CPU			
Peripheral Interrupt Configuration Register	Peripheral Interrupt Source			Peripheral Interrupt Source			Peripheral Interrupt Configuration Register
	Status Register Name	No.	Name	No.	Status Register Name	Bit	
PRO_SPI2_DMA_INT_MAP_REG	PRO_INTR_STATUS_REG_1	53	SPI2_DMA_INT	53	APP_INTR_STATUS_REG_1	21	APP_SPI2_DMA_INT_MAP_REG
PRO_SPI3_DMA_INT_MAP_REG		54	SPI3_DMA_INT	54		22	APP_SPI3_DMA_INT_MAP_REG
PRO_WDG_INT_MAP_REG		55	WDG_INT	55		23	APP_WDG_INT_MAP_REG
PRO_TIMER_INT1_MAP_REG		56	TIMER_INT1	56		24	APP_TIMER_INT1_MAP_REG
PRO_TIMER_INT2_MAP_REG		57	TIMER_INT2	57		25	APP_TIMER_INT2_MAP_REG
PRO_TG_T0_EDGE_INT_MAP_REG		58	TG_T0_EDGE_INT	58		26	APP_TG_T0_EDGE_INT_MAP_REG
PRO_TG_T1_EDGE_INT_MAP_REG		59	TG_T1_EDGE_INT	59		27	APP_TG_T1_EDGE_INT_MAP_REG
PRO_TG_WDT_EDGE_INT_MAP_REG		60	TG_WDT_EDGE_INT	60		28	APP_TG_WDT_EDGE_INT_MAP_REG
PRO_TG_LACT_EDGE_INT_MAP_REG		61	TG_LACT_EDGE_INT	61		29	APP_TG_LACT_EDGE_INT_MAP_REG
PRO_TG1_T0_EDGE_INT_MAP_REG		62	TG1_T0_EDGE_INT	62		30	APP_TG1_T0_EDGE_INT_MAP_REG
PRO_TG1_T1_EDGE_INT_MAP_REG		63	TG1_T1_EDGE_INT	63		31	APP_TG1_T1_EDGE_INT_MAP_REG
PRO_TG1_WDT_EDGE_INT_MAP_REG	PRO_INTR_STATUS_REG_2	64	TG1_WDT_EDGE_INT	64	APP_INTR_STATUS_REG_2	0	APP_TG1_WDT_EDGE_INT_MAP_REG
PRO_TG1_LACT_EDGE_INT_MAP_REG		65	TG1_LACT_EDGE_INT	65		1	APP_TG1_LACT_EDGE_INT_MAP_REG
PRO_MMU_IA_INT_MAP_REG		66	MMU_IA_INT	66		2	APP_MMU_IA_INT_MAP_REG
PRO_MPU_IA_INT_MAP_REG		67	MPU_IA_INT	67		3	APP_MPU_IA_INT_MAP_REG
PRO_CACHE_IA_INT_MAP_REG		68	CACHE_IA_INT	68		4	APP_CACHE_IA_INT_MAP_REG

### 2.3.2 CPU Interrupt

Both of the two CPUs (PRO and APP) have 32 interrupts each, of which 26 are peripheral interrupts. All interrupts in a CPU are listed in Table 7.

**Table 7: CPU Interrupts**

No.	Category	Type	Priority Level
0	Peripheral	Level-Triggered	1
1	Peripheral	Level-Triggered	1
2	Peripheral	Level-Triggered	1
3	Peripheral	Level-Triggered	1
4	Peripheral	Level-Triggered	1
5	Peripheral	Level-Triggered	1
6	Internal	Timer.0	1
7	Internal	Software	1
8	Peripheral	Level-Triggered	1
9	Peripheral	Level-Triggered	1
10	Peripheral	Edge-Triggered	1
11	Internal	Profiling	3
12	Peripheral	Level-Triggered	1
13	Peripheral	Level-Triggered	1
14	Peripheral	NMI	NMI
15	Internal	Timer.1	3
16	Internal	Timer.2	5
17	Peripheral	Level-Triggered	1
18	Peripheral	Level-Triggered	1
19	Peripheral	Level-Triggered	2
20	Peripheral	Level-Triggered	2
21	Peripheral	Level-Triggered	2
22	Peripheral	Edge-Triggered	3
23	Peripheral	Level-Triggered	3
24	Peripheral	Level-Triggered	4
25	Peripheral	Level-Triggered	4
26	Peripheral	Level-Triggered	5
27	Peripheral	Level-Triggered	3
28	Peripheral	Edge-Triggered	4
29	Internal	Software	3
30	Peripheral	Edge-Triggered	4
31	Peripheral	Level-Triggered	5

### 2.3.3 Allocate Peripheral Interrupt Sources to Peripheral Interrupt on CPU

In this section:

- Source\_X stands for any particular peripheral interrupt source.
- PRO\_X\_MAP\_REG (or APP\_X\_MAP\_REG) stands for any particular peripheral interrupt configuration

register of the PRO\_CPU (or APP\_CPU). The peripheral interrupt configuration register corresponds to the peripheral interrupt source Source\_X. Referring to Table 6, the registers listed under “PRO\_CPU (APP\_CPU) - Peripheral Interrupt Configuration Register” correspond to the peripheral interrupt sources listed in “Peripheral Interrupt Source - Name”.

- Interrupt\_P stands for CPU peripheral interrupt, numbered as Num\_P. Num\_P can take the ranges 0 ~ 5, 8 ~ 10, 12 ~ 14, 17 ~ 28, 30 ~ 31.
- Interrupt\_I stands for CPU internal interrupt numbered as Num\_I. Num\_I can take values 6, 7, 11, 15, 16, 29.

Using this terminology, the possible operations of the Interrupt Matrix controller can be described as follows:

- **Allocate peripheral interrupt source Source\_X to CPU (PRO\_CPU or APP\_CPU)**  
Set PRO\_X\_MAP\_REG or APP\_X\_MAP\_REG to Num\_P. Num\_P can be any CPU peripheral interrupt number. CPU interrupts can be shared between multiple peripherals (see below).
- **Disable peripheral interrupt source Source\_X for CPU (PRO\_CPU or APP\_CPU)**  
Set PRO\_X\_MAP\_REG or APP\_X\_MAP\_REG for peripheral interrupt source to any Num\_I. The specific choice of internal interrupt number does not change behaviour, as none of the interrupt numbered as Num\_I are connected to either CPU.
- **Allocate multiple peripheral sources Source\_X<sub>n</sub> ORed to PRO\_CPU (APP\_CPU) peripheral interrupt**  
Set multiple PRO\_X<sub>n</sub>\_MAP\_REG (APP\_X<sub>n</sub>\_MAP\_REG) to the same Num\_P. Any of these peripheral interrupts will trigger CPU Interrupt\_P.

### 2.3.4 CPU NMI Interrupt Mask

The Interrupt Matrix temporarily masks all peripheral interrupt sources allocated to PRO\_CPU's ( or APP\_CPU's ) NMI interrupt if it receives the signal PRO\_CPU NMI Interrupt Mask ( or APP\_CPU NMI Interrupt Mask ) from the peripheral PID Controller respectively.

### 2.3.5 Query Current Interrupt Status of Peripheral Interrupt Source

The current interrupt status of a peripheral interrupt source can be read via the bit value in PRO\_INTR\_STATUS\_REG\_<sub>n</sub> (APP\_INTR\_STATUS\_REG\_<sub>n</sub>) as shown in the mapping in Table 6.

## 3. Reset and Clock

### 3.1 System Reset

#### 3.1.1 Introduction

The ESP32 has three reset levels: CPU reset, Core reset, and System reset. None of these reset levels clear the RAM. Figure 4 shows the subsystems included in each reset level.

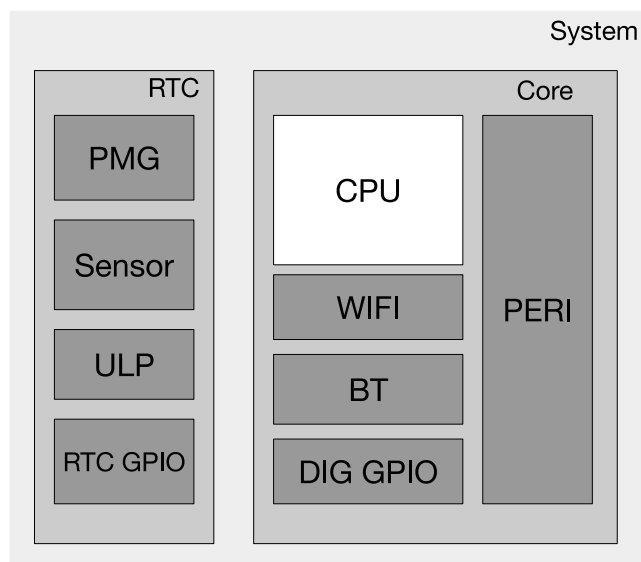


Figure 4: System Reset

- CPU reset: Only resets the registers of one or both the CPU cores.
- Core reset: Resets all the digital registers, including CPU cores, external GPIO and digital GPIO. The RTC is not reset.
- System reset: Resets all the registers on the chip, including those of the RTC.

#### 3.1.2 Reset Source

While most of the time the APP\_CPU and PRO\_CPU will be reset simultaneously, some reset sources are able to reset only one of the two cores. The reset reason for each core can be looked up individually: the PRO\_CPU reset reason will be stored in RTC\_CNTL\_RESET\_CAUSE\_PROCPU, the reset reason for the APP\_CPU in APP\_CNTL\_RESET\_CAUSE\_PROCPU. Table 8 shows the possible reset reason values that can be read from these registers.

Table 8: PRO\_CPU and APP\_CPU reset reason values

PRO	APP	Source	Reset Type	Note
0x01	0x01	Chip Power On Reset	System Reset	-
0x10	0x10	RWDT System Reset	System Reset	Refer to <a href="#">WDT Chapter</a> .
0x0F	0x0F	Brown Out Reset	System Reset	Refer to Power Management Chapter.
0x03	0x03	Software System Reset	Core Reset	Configure RTC_CNTL_SW_SYS_RST register.
0x05	0x05	Deep Sleep Rest	Core Reset	Refer to Power Management Chapter.
0x07	0x07	MWDT0 Global Reset	Core Reset	Refer to <a href="#">WDT Chapter</a> .

PRO	APP	APP Source	Reset Type	Note
0x08	0x08	MWDT1 Global Reset	Core Reset	Refer to <a href="#">WDT Chapter</a> .
0x09	0x09	RWDT Core Reset	Core Reset	Refer to <a href="#">WDT Chapter</a> .
0x0B	-	MWDT0 CPU Reset	CPU Reset	Refer to <a href="#">WDT Chapter</a> .
0x0C	-	Software CPU Reset	CPU Reset	Configure RTC_CNTL_SW_APPCPU_RST register.
-	0x0B	MWDT1 CPU Reset	CPU Reset	Refer to <a href="#">WDT Chapter</a> .
-	0x0C	Software CPU Reset	CPU Reset	Configure RTC_CNTL_SW_APPCPU_RST register.
0x0D	0x0D	RWDT CPU Reset	CPU Reset	Refer to <a href="#">WDT Chapter</a> .
-	0xE	PRO CPU Reset	CPU Reset	Indicates that the PRO CPU has independently reset the APP CPU by configuring the DPORT_APPCPU_RESETTING register.

## 3.2 System Clock

### 3.2.1 Introduction

The ESP32 integrates multiple clock sources for the CPU cores, the peripherals and the RTC. These clocks can be configured to meet different requirements. Figure 5 shows the system clock structure.

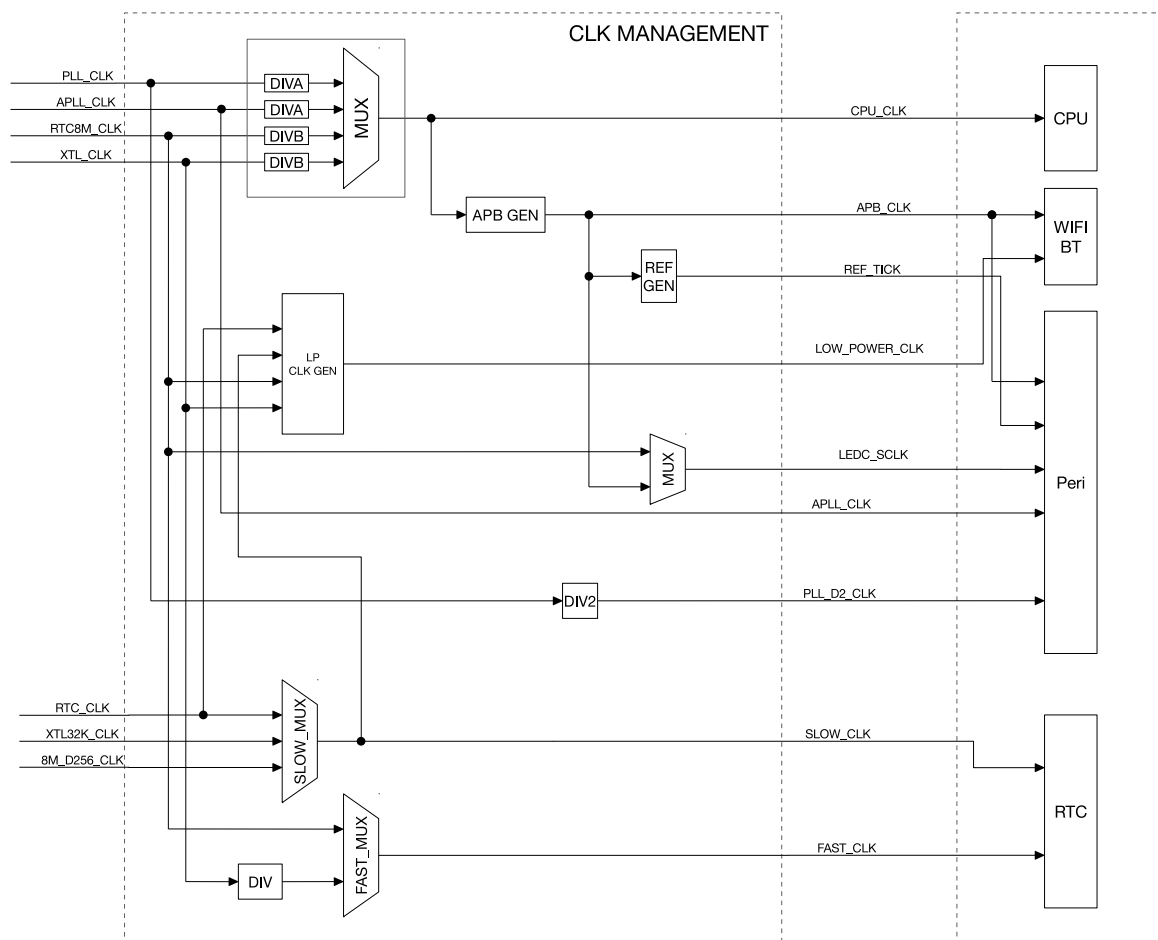


Figure 5: System Clock



### 3.2.2 Clock Source

The ESP32 can use an external crystal oscillator, an internal PLL or an oscillating circuit as a clock source. Specifically, the clock sources available are:

- High Speed Clocks
  - PLL\_CLK is an internal PLL clock with a frequency of 320 MHz.
  - XTL\_CLK is a clock signal generated using an external crystal with a frequency range of 2 ~ 40 MHz.
- Low Power Clocks
  - XTL32K\_CLK is a clock generated using an external crystal with a frequency of 32 KHz.
  - RTC8M\_CLK is an internal clock with a default frequency of 8 MHz. This frequency is adjustable.
  - RTC8M\_D256\_CLK is divided from RTC8M\_CLK 256. Its frequency is (RTC8M\_CLK / 256). With the default RTC8M\_CLK frequency of 8 MHz, this clock runs at 31.250 KHz.
  - RTC\_CLK is an internal low power clock with a default frequency of 150 KHz. This frequency is adjustable.
- Audio Clock
  - APLL\_CLK is an internal Audio PLL clock with a frequency range of 16 ~ 128 MHz.

### 3.2.3 CPU Clock

As Figure 5 shows, CPU\_CLK is the master clock for both CPU cores. CPU\_CLK clock can be as high as 160 MHz when the CPU is in high performance mode. Alternatively, the CPU can run at lower frequencies to reduce power consumption.

The CPU\_CLK clock source is determined by the RTC\_CNTL\_SOC\_CLK\_SEL register. PLL\_CLK, APLL\_CLK, RTC8M\_CLK and XTL\_CLK can be set as the CPU\_CLK source; see Table 9 and 10.

**Table 9: CPU\_CLK Source**

RTC_CNTL_SOC_CLK_SEL Value	Clock Source
0	XTL_CLK
1	PLL_CLK
2	RTC8M_CLK
3	APLL_CLK

**Table 10: CPU\_CLK Derivation**

Clock Source	SEL *	CPU Clock
0 / XTL_CLK	-	CPU_CLK = XTL_CLK / (APB_CTRL_PRE_DIV_CNT+1) APB_CTRL_PRE_DIV_CNT range is 0 ~ 1023. Default is 0.
1 / PLL_CLK	0	CPU_CLK = PLL_CLK / 4 CPU_CLK frequency is 80 MHz
1 / PLL_CLK	1	CPU_CLK = PLL_CLK / 2 CPU_CLK frequency is 160 MHz
2 / RTC8M_CLK	-	CPU_CLK = RTC8M_CLK / (APB_CTRL_PRE_DIV_CNT+1) APB_CTRL_PRE_DIV_CNT range is 0 ~ 1023. Default is 0.
3 / APLL_CLK	0	CPU_CLK = APLL_CLK / 4
3 / APLL_CLK	1	CPU_CLK = APLL_CLK / 2

\*SEL: DPORT\_CPUPERIOD\_SEL value

### 3.2.4 Peripheral Clock

Peripheral clocks include APB\_CLK, REF\_TICK, LEDC\_SCLK, APLL\_CLK and PLL\_D2\_CLK.

Table 11 shows which clocks can be used by which peripherals.

**Table 11: Peripheral Clock Usage**

Peripherals	APB_CLK	REF_TICK	LEDC_SCLK	APLL_CLK	PLL_D2_CLK
EMAC	Y	N	N	Y	N
TIMG	Y	N	N	N	N
I2S	Y	N	N	Y	Y
UART	Y	Y	N	N	N
RMT	Y	Y	N	N	N
LED PWM	Y	Y	Y	N	N
PWM	Y	N	N	N	N
I2C	Y	N	N	N	N
SPI	Y	N	N	N	N
PCNT	Y	N	N	N	N
Efuse Controller	Y	N	N	N	N
SDIO Slave	Y	N	N	N	N
SDMMC	Y	N	N	N	N

#### 3.2.4.1 APB\_CLK Source

The APB\_CLK is derived from CPU\_CLK as detailed in Table 12. The division factor depends on the CPU\_CLK source.

**Table 12: APB\_CLK Derivation**

CPU_CLK Source	APB_CLK
PLL_CLK	PLL_CLK / 4
APLL_CLK	CPU_CLK / 2
XTAL_CLK	CPU_CLK
RTC8M_CLK	CPU_CLK

### 3.2.4.2 REF\_TICK Source

REF\_TICK is derived from APB\_CLK via a divider. The divider value used depends on the APB\_CLK source, which in turn depends on the CPU\_CLK source.

By configuring correct divider values for each APB\_CLK source, the user can ensure that the REF\_TICK frequency does not change when CPU\_CLK changes source causing the APB\_CLK frequency to change.

Clock divider registers are shown in Table 13.

**Table 13: REF\_TICK Derivation**

CPU_CLK & APB_CLK Source	Clock Divider Register
PLL_CLK	APB_CTRL_PLL_TICK_NUM
XTAL_CLK	APB_CTRL_XTAL_TICK_NUM
APLL_CLK	APB_CTRL_APLL_TICK_NUM
RTC8M_CLK	APB_CTRL_CK8M_TICK_NUM

### 3.2.4.3 LEDC\_SCLK Source

The LEDC\_SCLK clock source is selected by the LEDC\_APB\_CLK\_SEL register, as shown in Table 14.

**Table 14: LEDC\_SCLK Derivation**

LEDC_APB_CLK_SEL Value	LEDC_SCLK Source
1	RTC8M_CLK
0	APB_CLK

### 3.2.4.4 APLL\_SCLK Source

The APLL\_CLK is sourced from PLL\_CLK, with its output frequency configured using the APLL configuration registers.

### 3.2.4.5 PLL\_D2\_CLK Source

PLL\_D2\_CLK is half the PLL\_CLK frequency.

### 3.2.4.6 Clock Source Considerations

Most peripherals will operate using the APB\_CLK frequency as a reference. When this frequency changes, the peripherals will need to update their clock configuration to operate at the same frequency after the change. Peripherals accessing REF\_TICK can continue operating normally when switching clock sources, without changing clock source. Please refer to Table 11 for details.

The LED PWM module can use RTC8M\_CLK as a clock source, meaning that it can be used when APB\_CLK is disabled. In other words, when the system is in low power consumption mode (refer to power manager module), normal peripherals will be halted (APB\_CLK is turned off), but the LED PWM can work normally via RTC8M\_CLK.

### 3.2.5 Wi-Fi BT Clock

Wi-Fi and BT can only operate if APB\_CLK has PLL\_CLK as its clock source. Suspending PLL\_CLK requires Wi-Fi and BT to both have entered low power consumption mode first.

For LOW\_POWER\_CLK, one of RTC\_CLK, SLOW\_CLK, RTC8M\_CLK or XTL\_CLK can be selected as the low power consumption mode clock source for Wi-Fi and BT.

### 3.2.6 RTC Clock

The clock sources of SLOW\_CLK and FAST\_CLK are low frequency clocks. The RTC module can operate when most other clocks are stopped.

SLOW\_CLK is used to clock the Power Management module. It can be sourced from RTC\_CLK, XTL32K\_CLK or RTC8M\_D256\_CLK

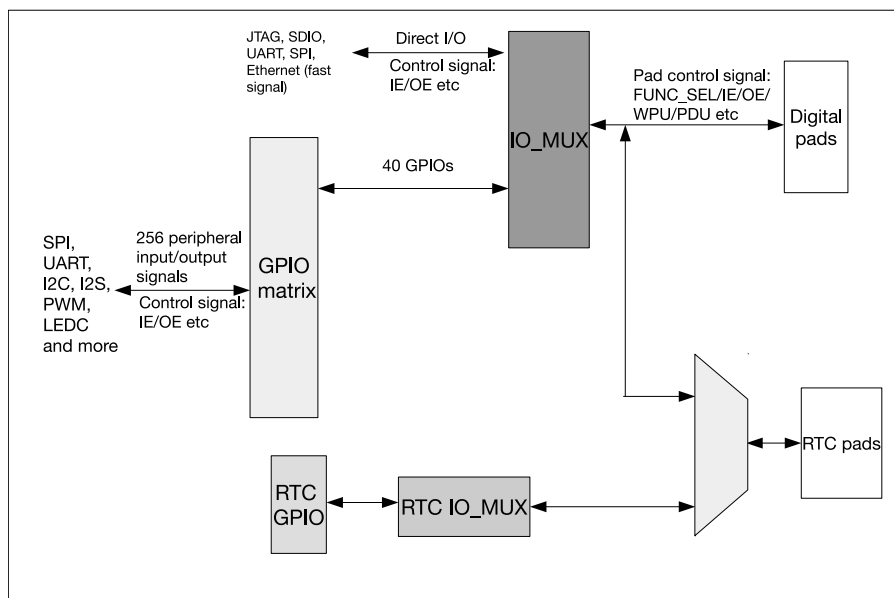
FAST\_CLK is used to clock the On-chip Sensor module. It can be sourced from a divided XTL\_CLK or from RTC8M\_CLK.

## 4. IO\_MUX and GPIO Matrix

### 4.1 Introduction

The ESP32 chip features 40 physical GPIO pads. Some GPIO pads cannot be used or do not have the corresponding pin on the chip package. Each pad can be used as a general purpose I/O or can be connected to an internal peripheral signal. The IO\_MUX, RTC IO\_MUX and the GPIO matrix are responsible for routing signals from the peripherals to GPIO pads. Together these systems provide highly configurable I/O.

This chapter describes the signal selection and connection between the digital pads (FUNC\_SEL, IE, OE, WPU, WDU, etc), 256 peripheral input/output signals (control signals: SIG\_IN\_SEL, SIG\_OUT\_SEL, IE, OE, etc), fast peripheral input/output signals (control signals: IE, OE, etc), and RTC IO\_MUX.



**Figure 6: IO\_MUX, RTC IO\_MUX and GPIO Matrix Overview**

1. The IO\_MUX contains one register per GPIO pad. Each pad can be configured for "GPIO" function (connected to the GPIO Matrix) or configured for a direct function (bypassing the GPIO Matrix. Some high speed digital functions (Ethernet, SDIO, SPI, JTAG, UART) can bypass the GPIO Matrix for better high frequency digital performance. In this case, the IO\_MUX is used to connect these pads directly to the peripheral.)

Refer to Section 4.10 for a list of IO\_MUX functions for each I/O pad.

2. The GPIO Matrix is a full switching matrix between the peripheral input/output signals and the pads.
  - For input to the chip: Each of the 256 internal peripheral inputs can select any GPIO pad as the input source.
  - For output from the chip: The output signal of each of the 40 GPIO pads can be from one of the 256 peripheral output signals.

Refer to Section 4.9 for a list of GPIO Matrix peripheral signals.

3. RTC IO\_MUX is used to connect GPIO pads to their low power and analog functions. Only a subset of GPIO pads have these optional "RTC" functions.

Refer to Section 4.11 for a list of RTC IO\_MUX functions.

## 4.2 Peripheral Input via GPIO Matrix

### 4.2.1 Summary

To receive a peripheral input signal via the GPIO Matrix, the GPIO Matrix is configured to source the peripheral signal's input index (0-255) from one of the 40 GPIOs (0-39).

The input signal is read from the GPIO pad through the IO\_MUX. The IO\_MUX must be configured to set the chosen pad to "GPIO" function. This causes the GPIO pad input signal to be routed into the GPIO Matrix which in turn routes it to the selected peripheral input.

### 4.2.2 Functional Description

Figure 7 shows the logic for input selection via GPIO Matrix.

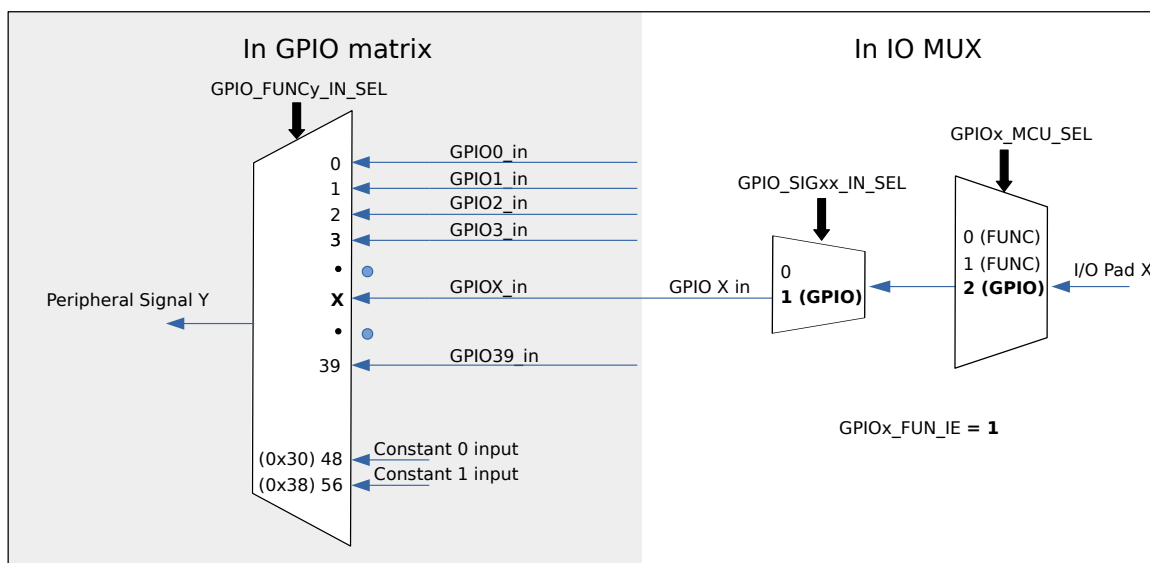


Figure 7: Peripheral Input via IO\_MUX, GPIO Matrix

To read GPIO pad  $X$  into peripheral signal  $Y$ , follow these steps:

1. Configure the GPIO\_FUNC $y$ \_IN\_SEL\_CFG register for peripheral signal  $Y$  in the GPIO Matrix:
  - Set the GPIO\_FUNC $x$ \_IN\_SEL field to the number of the GPIO pad  $X$  to read from.
2. Configure the GPIO\_FUNC $x$ \_OUT\_SEL\_CFG and GPIO\_ENABLE\_DATA[ $x$ ] for GPIO pad  $X$  in the GPIO Matrix:
  - For input only signals, the pad output can be disabled by setting the GPIO\_FUNC $x$ \_OEN\_SEL bits to one and GPIO\_ENABLE\_DATA[ $x$ ] to zero. Otherwise, there is no need to disable output.
3. Configure the IO\_MUX register for GPIO pad  $X$ :
  - Set the function field to GPIO.
  - Enable the input by setting the  $xx\_FUN\_IE$  bit.
  - Set  $xx\_FUN\_WPU$  and  $xx\_FUN\_WPD$  fields as desired to enable internal pull-up/pull-down resistors.

Notes:

- One input pad can be connected to multiple input\_signals.

- The input signal can be inverted with GPIO\_FUNC $x$ \_IN\_INV\_SEL.
- It is possible to have a peripheral read a constant low or constant high input value without connecting this input to a pad. This can be done by selecting a special GPIO\_FUNC $y$ \_IN\_SEL input instead of a GPIO number:
  - When GPIO\_FUNC $x$ \_IN\_SEL is 0x30, input\_signal $x$  is always 0
  - When GPIO\_FUNC $x$ \_IN\_SEL is 0x38, input\_signal $x$  is always 1.

### 4.2.3 Simple GPIO Input

The GPIO\_IN\_DATA register holds the input values of each GPIO pad.

The input value of any GPIO pin can be read at any time without configuring the GPIO Matrix for a particular peripheral signal. However, it is necessary to configure the  $xx\_FUN\_IE$  register for pad  $x$ , as shown in Section 4.2.2.

## 4.3 Peripheral Output via GPIO Matrix

### 4.3.1 Summary

To output a signal from a peripheral via the GPIO Matrix, the GPIO Matrix is configured to route the peripheral output signal (0-255) to one of the first 34 GPIOs (0-33). (Note that GPIO pads 34-39 cannot be used as outputs.)

The output signal is routed from the peripheral into the GPIO Matrix. It is then routed into the IO\_MUX, which is configured to set the chosen pad to "GPIO" function. This causes the output GPIO signal to be connected to the pad.

### 4.3.2 Functional Description

One of 256 input signals can be selected to go through the GPIO matrix into the IO\_MUX and then to a pad. Figure 8 illustrates the configuration.

To output peripheral signal  $y$  to particular GPIO pad  $x$ , follow these steps:

1. Configure the GPIO\_FUNC $x$ \_OUT\_SEL\_CFG register and GPIO\_ENABLE\_DATA $[x]$  of GPIO  $x$  in the GPIO Matrix:
  - Set GPIO\_FUNC $x$ \_OUT\_SEL to the index of desired peripheral output signal  $y$ .
  - Set the GPIO\_FUNC $x$ \_OEN\_SEL bits and GPIO\_ENABLE\_DATA $[x]$  to enable output mode by force, OR, clear GPIO\_FUNC $x$ \_OEN\_SEL to zero so that the output enable signal will be decided by the internal logic function.
2. Optionally, to enable open drain mode set the GPIO\_PIN $x$ \_PAD\_DRIVER bit in the GPIO\_PIN $x$  register.
3. Configure the I/O mux register for GPIO pad  $x$ :
  - Set the function field to GPIO.
  - Set the  $xx\_FUN\_DRV$  field to the desired value for output strength. The higher the value is, the higher the output strength is. Pull up/down the pad by configuring  $xx\_FUNC\_WPU$  and  $xx\_FUNC\_WPD$  registers in open drain mode.

Notes:

- The output signal from a single peripheral can be sent to multiple pads simultaneously.
- Only the first 34 GPIOs (0-33) can be used as outputs.
- The output signal can be inverted by setting the GPIO\_FUNC<sub>x</sub>\_OUT\_INV\_SEL bit.

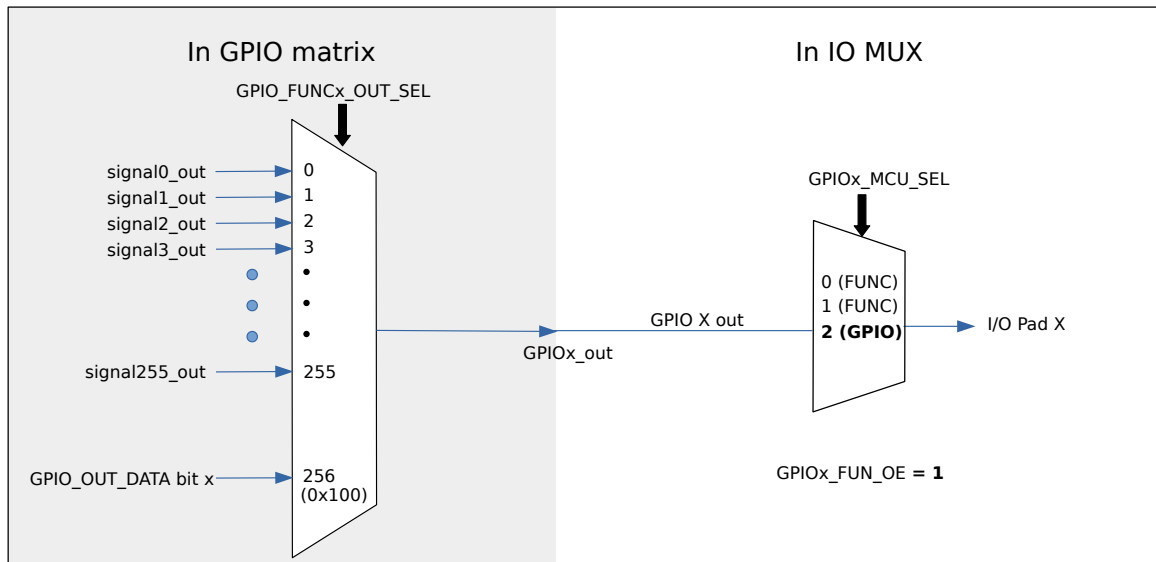


Figure 8: Output via GPIO Matrix

### 4.3.3 Simple GPIO Output

The GPIO Matrix can also be used for simple GPIO output - setting a bit in the GPIO\_OUT\_DATA register will write to the corresponding GPIO pad.

To configure a pad as simple GPIO output, the GPIO Matrix GPIO\_FUNC<sub>x</sub>\_OUT\_SEL register is configured with a special peripheral index value (0x100).

## 4.4 Direct I/O via IO\_MUX

### 4.4.1 Summary

Some high speed digital functions (Ethernet, SDIO, SPI, JTAG, UART) can bypass the GPIO Matrix for better high frequency digital performance. In this case, the IO\_MUX is used to connect these pads directly to the peripheral.

Selecting this option is less flexible than using the GPIO Matrix, as the IO\_MUX register for each GPIO pad can only select from a limited number of functions. However, better high frequency digital performance will be maintained.

### 4.4.2 Functional Description

Two registers must be configured in order to bypass the GPIO Matrix for peripheral I/O:

1. IO\_MUX for the GPIO pad must be set to the desired pad function (Section 4.10 has a list of pad functions).
2. For inputs, the SIG\_IN\_SEL register must be set to route the input directly to the peripheral.



## 4.5 RTC IO\_MUX for Low Power and Analog I/O

### 4.5.1 Summary

18 pins have low power (RTC domain) capabilities and analog functions which are handled by the RTC subsystem of ESP32. The IO\_MUX and GPIO Matrix are not used for these functions, instead the RTC\_MUX is used to redirect the I/O to the RTC subsystem.

When configured as RTC GPIOs, the output pads can still retain the output level value when the chip is in Deep-sleep mode, and the input pads can wake up the chip from Deep-sleep.

Section 4.11 has a list of RTC\_MUX pins and their functions.

### 4.5.2 Functional Description

Each pad with analog and RTC functions is controlled by the RTC\_IO\_TOUCH\_PAD $x$ \_TO\_GPIO bit in the RTC\_GPIO\_PIN $x$  register. By default this bit is set to 1, routing all I/O via the IO\_MUX subsystem as described in earlier subsections.

If the RTC\_IO\_TOUCH\_PAD $x$ \_TO\_GPIO bit is cleared, then I/O to and from that pad is routed to the RTC subsystem instead. In this mode, the RTC\_GPIO\_PIN $x$  register is used for digital I/O and the analog features of the pad are also available. See Section 4.11 for a list of RTC pin functions.

See 4.11 for a table mapping GPIO pads to their RTC equivalent pins and analog functions. Note that the RTC\_IO\_PIN $x$  registers use the RTC GPIO pin numbering, not the GPIO pad numbering.

## 4.6 Light-sleep Mode Pin Functions

Pins can have different functions when the ESP32 is in Light-sleep mode. If the GPIO $xx$ \_SLP\_SEL bit in the IO\_MUX register for a GPIO pad is set to 1, a different set of registers is used to control the pad when the ESP32 is in Light-sleep mode:

**Table 15: IO\_MUX Light-sleep Pin Function Registers**

IO_MUX Function	Normal Execution OR GPIO $xx$ _SLP_SEL = 0	Light-sleep Mode AND GPIO $xx$ _SLP_SEL = 1
Output Drive Strength	GPIO $xx$ _FUNC_DRV	GPIO $xx$ _MCU_DRV
Pullup Resistor	GPIO $xx$ _FUNC_WPU	GPIO $xx$ _MCU_WPU
Pulldown Resistor	GPIO $xx$ _FUNC_WPD	GPIO $xx$ _MCU_WPD
Output Enable	(From GPIO Matrix _OEN field)	GPIO $xx$ _MCU_OE

If GPIO $xx$ \_SLP\_SEL is set to 0, the pin functions remain the same in both normal execution and Light-sleep modes.

## 4.7 Pad Hold Feature

Each IO pad (including the RTC pads) has an individual hold function controlled by a RTC register. When the pad 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 configuration or GPIO configuration is modified. Users can use the hold function for the pads

to retain the pad state through a core reset and system reset triggered by watchdog time-out or Deep-sleep events.

## 4.8 I/O Pad Power Supply

IO pad power supply is shown in Figure 9.

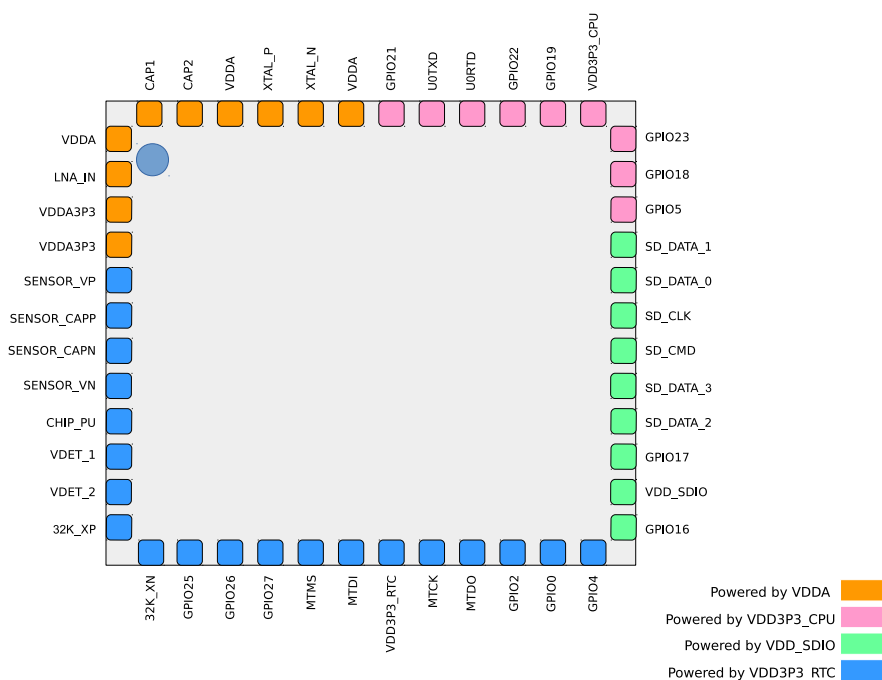


Figure 9: ESP32 I/O Pad Power Sources

- Pads marked blue are RTC pads that have their individual analog function and can also act as normal digital IO pads. For details, please refer to Section 4.11.
- Pads marked pink and green only have digital functions.
- Pads marked green can be powered externally or internally via VDD\_SDIO (see below).

### 4.8.1 VDD\_SDIO Power Domain

VDD\_SDIO can source or sink current, allowing this power domain to be powered externally or internally. To power VDD\_SDIO externally, apply the same power supply of VDD3P3\_RTC to the VDD\_SDIO pad.

Without an external power supply, the internal regulator will supply VDD\_SDIO. The VDD\_SDIO voltage can be configured to be either 1.8V or 3.3V (the same as that at VRTC), depending on the state of the MTDI pad at reset - a high level configures 1.8V and a low level configures 3.3V. Setting the efuse bit decides the default voltage of the VDD\_SDIO. In addition, software can change the voltage of the VDD\_SDIO by configuring register bits.

## 4.9 Peripheral Signal List

Table 16 contains a list of Peripheral Input/Output signals used by the GPIO Matrix:

Table 16: GPIO Matrix Peripheral Signals

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
0	SPICLK_in	SPICLK_out	YES
1	SPIQ_in	SPIQ_out	YES
2	SPID_in	SPID_out	YES
3	SPIHD_in	SPIHD_out	YES
4	SPIWP_in	SPIWP_out	YES
5	SPICS0_in	SPICS0_out	YES
6	SPICS1_in	SPICS1_out	
7	SPICS2_in	SPICS2_out	
8	HSPICLK_in	HSPICLK_out	YES
9	HSPIQ_in	HSPIQ_out	YES
10	HSPID_in	HSPID_out	YES
11	HSPICS0_in	HSPICS0_out	YES
12	HSPIHD_in	HSPIHD_out	YES
13	HSPIWP_in	HSPIWP_out	YES
14	U0RXD_in	U0TXD_out	YES
15	U0CTS_in	U0RTS_out	YES
16	U0DSR_in	U0DTR_out	
17	U1RXD_in	U1TXD_out	YES
18	U1CTS_in	U1RTS_out	YES
23	I2S0O_BCK_in	I2S0O_BCK_out	
24	I2S1O_BCK_in	I2S1O_BCK_out	
25	I2S0O_WS_in	I2S0O_WS_out	
26	I2S1O_WS_in	I2S1O_WS_out	
27	I2S0I_BCK_in	I2S0I_BCK_out	
28	I2S0I_WS_in	I2S0I_WS_out	
29	I2CEXT0_SCL_in	I2CEXT0_SCL_out	
30	I2CEXT0_SDA_in	I2CEXT0_SDA_out	
31	pwm0_sync0_in	sdio_tohost_int_out	
32	pwm0_sync1_in	pwm0_out0a	
33	pwm0_sync2_in	pwm0_out0b	
34	pwm0_f0_in	pwm0_out1a	
35	pwm0_f1_in	pwm0_out1b	
36	pwm0_f2_in	pwm0_out2a	
37		pwm0_out2b	
39	pcnt_sig_ch0_in0		
40	pcnt_sig_ch1_in0		
41	pcnt_ctrl_ch0_in0		
42	pcnt_ctrl_ch1_in0		
43	pcnt_sig_ch0_in1		
44	pcnt_sig_ch1_in1		
45	pcnt_ctrl_ch0_in1		
46	pcnt_ctrl_ch1_in1		
47	pcnt_sig_ch0_in2		

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
48	pcnt_sig_ch1_in2		
49	pcnt_ctrl_ch0_in2		
50	pcnt_ctrl_ch1_in2		
51	pcnt_sig_ch0_in3		
52	pcnt_sig_ch1_in3		
53	pcnt_ctrl_ch0_in3		
54	pcnt_ctrl_ch1_in3		
55	pcnt_sig_ch0_in4		
56	pcnt_sig_ch1_in4		
57	pcnt_ctrl_ch0_in4		
58	pcnt_ctrl_ch1_in4		
61	HSPICS1_in	HSPICS1_out	
62	HSPICS2_in	HSPICS2_out	
63	VSPICLK_in	VSPICLK_out_mux	YES
64	VSPIQ_in	VSPIQ_out	YES
65	VSPID_in	VSPID_out	YES
66	VSPICLK_in	VSPICLK_out	YES
67	VSPICLK_in	VSPICLK_out	YES
68	VSPICS0_in	VSPICS0_out	YES
69	VSPICS1_in	VSPICS1_out	
70	VSPICS2_in	VSPICS2_out	
71	pcnt_sig_ch0_in5	ledc_hs_sig_out0	
72	pcnt_sig_ch1_in5	ledc_hs_sig_out1	
73	pcnt_ctrl_ch0_in5	ledc_hs_sig_out2	
74	pcnt_ctrl_ch1_in5	ledc_hs_sig_out3	
75	pcnt_sig_ch0_in6	ledc_hs_sig_out4	
76	pcnt_sig_ch1_in6	ledc_hs_sig_out5	
77	pcnt_ctrl_ch0_in6	ledc_hs_sig_out6	
78	pcnt_ctrl_ch1_in6	ledc_hs_sig_out7	
79	pcnt_sig_ch0_in7	ledc_ls_sig_out0	
80	pcnt_sig_ch1_in7	ledc_ls_sig_out1	
81	pcnt_ctrl_ch0_in7	ledc_ls_sig_out2	
82	pcnt_ctrl_ch1_in7	ledc_ls_sig_out3	
83	rmt_sig_in0	ledc_ls_sig_out4	
84	rmt_sig_in1	ledc_ls_sig_out5	
85	rmt_sig_in2	ledc_ls_sig_out6	
86	rmt_sig_in3	ledc_ls_sig_out7	
87	rmt_sig_in4	rmt_sig_out0	
88	rmt_sig_in5	rmt_sig_out1	
89	rmt_sig_in6	rmt_sig_out2	
90	rmt_sig_in7	rmt_sig_out3	
91		rmt_sig_out4	
92		rmt_sig_out5	
93		rmt_sig_out6	

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
94		rmt_sig_out7	
95	I2CEXT1_SCL_in	I2CEXT1_SCL_out	
96	I2CEXT1_SDA_in	I2CEXT1_SDA_out	
97	host_card_detect_n_1	host_ccmd_od_pullup_en_n	
98	host_card_detect_n_2	host_rst_n_1	
99	host_card_write_prt_1	host_rst_n_2	
100	host_card_write_prt_2	gpio_sd0_out	
101	host_card_int_n_1	gpio_sd1_out	
102	host_card_int_n_2	gpio_sd2_out	
103	pwm1_sync0_in	gpio_sd3_out	
104	pwm1_sync1_in	gpio_sd4_out	
105	pwm1_sync2_in	gpio_sd5_out	
106	pwm1_f0_in	gpio_sd6_out	
107	pwm1_f1_in	gpio_sd7_out	
108	pwm1_f2_in	pwm1_out0a	
109	pwm0_cap0_in	pwm1_out0b	
110	pwm0_cap1_in	pwm1_out1a	
111	pwm0_cap2_in	pwm1_out1b	
112	pwm1_cap0_in	pwm1_out2a	
113	pwm1_cap1_in	pwm1_out2b	
114	pwm1_cap2_in	pwm2_out1h	
115	pwm2_fta	pwm2_out1l	
116	pwm2_ftb	pwm2_out2h	
117	pwm2_cap1_in	pwm2_out2l	
118	pwm2_cap2_in	pwm2_out3h	
119	pwm2_cap3_in	pwm2_out3l	
120	pwm3_fta	pwm2_out4h	
121	pwm3_ftb	pwm2_out4l	
122	pwm3_cap1_in		
123	pwm3_cap2_in		
124	pwm3_cap3_in		
140	I2S0I_DATA_in0	I2S0O_DATA_out0	
141	I2S0I_DATA_in1	I2S0O_DATA_out1	
142	I2S0I_DATA_in2	I2S0O_DATA_out2	
143	I2S0I_DATA_in3	I2S0O_DATA_out3	
144	I2S0I_DATA_in4	I2S0O_DATA_out4	
145	I2S0I_DATA_in5	I2S0O_DATA_out5	
146	I2S0I_DATA_in6	I2S0O_DATA_out6	
147	I2S0I_DATA_in7	I2S0O_DATA_out7	
148	I2S0I_DATA_in8	I2S0O_DATA_out8	
149	I2S0I_DATA_in9	I2S0O_DATA_out9	
150	I2S0I_DATA_in10	I2S0O_DATA_out10	
151	I2S0I_DATA_in11	I2S0O_DATA_out11	
152	I2S0I_DATA_in12	I2S0O_DATA_out12	

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
153	I2S0I_DATA_in13	I2S0O_DATA_out13	
154	I2S0I_DATA_in14	I2S0O_DATA_out14	
155	I2S0I_DATA_in15	I2S0O_DATA_out15	
156		I2S0O_DATA_out16	
157		I2S0O_DATA_out17	
158		I2S0O_DATA_out18	
159		I2S0O_DATA_out19	
160		I2S0O_DATA_out20	
161		I2S0O_DATA_out21	
162		I2S0O_DATA_out22	
163		I2S0O_DATA_out23	
164	I2S1I_BCK_in	I2S1I_BCK_out	
165	I2S1I_WS_in	I2S1I_WS_out	
166	I2S1I_DATA_in0	I2S1O_DATA_out0	
167	I2S1I_DATA_in1	I2S1O_DATA_out1	
168	I2S1I_DATA_in2	I2S1O_DATA_out2	
169	I2S1I_DATA_in3	I2S1O_DATA_out3	
170	I2S1I_DATA_in4	I2S1O_DATA_out4	
171	I2S1I_DATA_in5	I2S1O_DATA_out5	
172	I2S1I_DATA_in6	I2S1O_DATA_out6	
173	I2S1I_DATA_in7	I2S1O_DATA_out7	
174	I2S1I_DATA_in8	I2S1O_DATA_out8	
175	I2S1I_DATA_in9	I2S1O_DATA_out9	
176	I2S1I_DATA_in10	I2S1O_DATA_out10	
177	I2S1I_DATA_in11	I2S1O_DATA_out11	
178	I2S1I_DATA_in12	I2S1O_DATA_out12	
179	I2S1I_DATA_in13	I2S1O_DATA_out13	
180	I2S1I_DATA_in14	I2S1O_DATA_out14	
181	I2S1I_DATA_in15	I2S1O_DATA_out15	
182		I2S1O_DATA_out16	
183		I2S1O_DATA_out17	
184		I2S1O_DATA_out18	
185		I2S1O_DATA_out19	
186		I2S1O_DATA_out20	
187		I2S1O_DATA_out21	
188		I2S1O_DATA_out22	
189		I2S1O_DATA_out23	
190	I2S0I_H_SYNC	pwm3_out1h	
191	I2S0I_V_SYNC	pwm3_out1l	
192	I2S0I_H_ENABLE	pwm3_out2h	
193	I2S1I_H_SYNC	pwm3_out2l	
194	I2S1I_V_SYNC	pwm3_out3h	
195	I2S1I_H_ENABLE	pwm3_out3l	
196		pwm3_out4h	

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
197		pwm3_out4l	
198	U2RXD_in	U2TXD_out	YES
199	U2CTS_in	U2RTS_out	YES
200	emac_mdc_i	emac_mdc_o	
201	emac_mdio_i	emac_mdio_o	
202	emac_crs_i	emac_crs_o	
203	emac_col_i	emac_col_o	
204	pcmfsync_in	bt_audio0_irq	
205	pcmclk_in	bt_audio1_irq	
206	pcmdin	bt_audio2_irq	
207		ble_audio0_irq	
208		ble_audio1_irq	
209		ble_audio2_irq	
210		pcmfsync_out	
211		pcmclk_out	
212		pcmdout	
213		ble_audio_sync0_p	
214		ble_audio_sync1_p	
215		ble_audio_sync2_p	
224		sig_in_func224	
225		sig_in_func225	
226		sig_in_func226	
227		sig_in_func227	
228		sig_in_func228	

**Direct I/O in IO\_MUX "YES"** means this signal is also available directly via IO\_MUX. To apply GPIO Matrix for these signals, their corresponding SIG\_IN\_SEL register must be cleared.

## 4.10 IO\_MUX Pad List

Table 17 shows the IO\_MUX functions for each I/O pad:

**Table 17: IO\_MUX Pad Summary**

GPIO	Pad Name	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Reset	Notes
0	GPIO0	GPIO0	CLK_OUT1	GPIO0	-	-	EMAC_TX_CLK	3	R
1	U0TXD	U0TXD	CLK_OUT3	GPIO1	-	-	EMAC_RXD2	3	-
2	GPIO2	GPIO2	HSPIWP	GPIO2	HS2_DATA0	SD_DATA0	-	2	R
3	U0RXD	U0RXD	CLK_OUT2	GPIO3	-	-	-	3	-
4	GPIO4	GPIO4	HSPIHD	GPIO4	HS2_DATA1	SD_DATA1	EMAC_TX_ER	2	R
5	GPIO5	GPIO5	VSPICS0	GPIO5	HS1_DATA6	-	EMAC_RX_CLK	3	-
6	SD_CLK	SD_CLK	SPICLK	GPIO6	HS1_CLK	U1CTS	-	3	-
7	SD_DATA_0	SD_DATA0	SPIQ	GPIO7	HS1_DATA0	U2RTS	-	3	-
8	SD_DATA_1	SD_DATA1	SPID	GPIO8	HS1_DATA1	U2CTS	-	3	-
9	SD_DATA_2	SD_DATA2	SPIHD	GPIO9	HS1_DATA2	U1RXD	-	3	-
10	SD_DATA_3	SD_DATA3	SPIWP	GPIO10	HS1_DATA3	U1TXD	-	3	-
11	SD_CMD	SD_CMD	SPICS0	GPIO11	HS1_CMD	U1RTS	-	3	-
12	MTDI	MTDI	HSPIQ	GPIO12	HS2_DATA2	SD_DATA2	EMAC_TXD3	2	R

GPIO	Pad Name	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Reset	Notes
13	MTCK	MTCK	HSPID	GPIO13	HS2_DATA3	SD_DATA3	EMAC_RX_ER	1	R
14	MTMS	MTMS	HSPICLK	GPIO14	HS2_CLK	SD_CLK	EMAC_TXD2	1	R
15	MTDO	MTDO	HSPICS0	GPIO15	HS2_CMD	SD_CMD	EMAC_RXD3	3	R
16	GPIO16	GPIO16	-	GPIO16	HS1_DATA4	U2RXD	EMAC_CLK_OUT	1	-
17	GPIO17	GPIO17	-	GPIO17	HS1_DATA5	U2TXD	EMAC_CLK_180	1	-
18	GPIO18	GPIO18	VSPICLK	GPIO18	HS1_DATA7	-	-	1	-
19	GPIO19	GPIO19	VSPIQ	GPIO19	U0CTS	-	EMAC_TXD0	1	-
20	GPIO20	GPIO20	-	GPIO20	-	-	-	1	-
21	GPIO21	GPIO21	VSPICLK	GPIO21	-	-	EMAC_TX_EN	1	-
22	GPIO22	GPIO22	VSPICLK	GPIO22	U0RTS	-	EMAC_TXD1	1	-
23	GPIO23	GPIO23	VSPID	GPIO23	HS1_STROBE	-	-	1	-
25	GPIO25	GPIO25	-	GPIO25	-	-	EMAC_RXD0	0	R
26	GPIO26	GPIO26	-	GPIO26	-	-	EMAC_RXD1	0	R
27	GPIO27	GPIO27	-	GPIO27	-	-	EMAC_RX_DV	1	R
32	32K_XP	GPIO32	-	GPIO32	-	-	-	0	R
33	32K_XN	GPIO33	-	GPIO33	-	-	-	0	R
34	VDET_1	GPIO34	-	GPIO34	-	-	-	0	R, I
35	VDET_2	GPIO35	-	GPIO35	-	-	-	0	R, I
36	SENSOR_VP	GPIO36	-	GPIO36	-	-	-	0	R, I
37	SENSOR_CAPP	GPIO37	-	GPIO37	-	-	-	0	R, I
38	SENSOR_CAPN	GPIO38	-	GPIO38	-	-	-	0	R, I
39	SENSOR_VN	GPIO39	-	GPIO39	-	-	-	0	R, I

## Reset Configurations

"Reset" column shows each pad's default configurations after reset:

- **0** - IE=0 (input disabled).
- **1** - IE=1 (input enabled).
- **2** - IE=1, WPD=1 (input enabled, pulldown resistor).
- **3** - IE=1, WPU=1 (input enabled, pullup resistor).

## Notes

- **R** - Pad has RTC/analog functions via RTC\_MUX.
- **I** - Pad can only be configured as input GPIO.

Refer to [ESP32 Pin List](#) datasheet for more details and a complete table of pin functions.

## 4.11 RTC\_MUX Pin List

Table 18 shows the RTC pins and how they correspond to GPIO pads:

**Table 18: RTC\_MUX Pin Summary**

RTC GPIO Num	GPIO Num	Pad Name	Analog Function		
			1	2	3
0	36	SENSOR_VP	ADC_H	ADC1_CH0	-
1	37	SENSOR_CAPP	ADC_H	ADC1_CH1	-
2	38	SENSOR_CAPN	ADC_H	ADC1_CH2	-
3	39	SENSOR_VN	ADC_H	ADC1_CH3	-
4	34	VDET_1	-	ADC1_CH6	-



RTC GPIO Num	GPIO Num	Pad Name	Analog Function		
			1	2	3
5	35	VDET_2	-	ADC1_CH7	-
6	25	GPIO25	DAC_1	ADC2_CH8	-
7	26	GPIO26	DAC_2	ADC2_CH9	-
8	33	32K_XN	XTAL_32K_N	ADC1_CH5	TOUCH8
9	32	32K_XP	XTAL_32K_P	ADC1_CH4	TOUCH9
10	4	GPIO4	-	ADC2_CH0	TOUCH0
11	0	GPIO0	-	ADC2_CH1	TOUCH1
12	2	GPIO2	-	ADC2_CH2	TOUCH2
13	15	MTDO	-	ADC2_CH3	TOUCH3
14	13	MTCK	-	ADC2_CH4	TOUCH4
15	12	MTDI	-	ADC2_CH5	TOUCH5
16	14	MTMS	-	ADC2_CH6	TOUCH6
17	27	GPIO27	-	ADC2_CH7	TOUCH7

## 4.12 Register Summary

Name	Description	Address	Access
<a href="#">GPIO_OUT_REG</a>	GPIO 0-31 output register_REG	0x3FF44004	R/W
<a href="#">GPIO_OUT_W1TS_REG</a>	GPIO 0-31 output register_W1TS_REG	0x3FF44008	RO
<a href="#">GPIO_OUT_W1TC_REG</a>	GPIO 0-31 output register_W1TC_REG	0x3FF4400C	RO
<a href="#">GPIO_OUT1_REG</a>	GPIO 0-31 output register1_REG	0x3FF44010	R/W
<a href="#">GPIO_OUT1_W1TS_REG</a>	GPIO 0-31 output register1_W1TS_REG	0x3FF44014	RO
<a href="#">GPIO_OUT1_W1TC_REG</a>	GPIO 0-31 output register1_W1TC_REG	0x3FF44018	RO
<a href="#">GPIO_OUT_W1TS_REG</a>	GPIO 0-31 output bit set register_REG	0x3FF44008	RO
<a href="#">GPIO_OUT_W1TC_REG</a>	GPIO 0-31 output bit clear register_REG	0x3FF4400C	RO
<a href="#">GPIO_OUT1_REG</a>	GPIO 32-39 output register_REG	0x3FF44010	R/W
<a href="#">GPIO_OUT1_W1TS_REG</a>	GPIO 32-39 output register_W1TS_REG	0x3FF44014	RO
<a href="#">GPIO_OUT1_W1TC_REG</a>	GPIO 32-39 output register_W1TC_REG	0x3FF44018	RO
<a href="#">GPIO_OUT1_W1TS_REG</a>	GPIO 32-39 output bit set register_REG	0x3FF44014	RO
<a href="#">GPIO_OUT1_W1TC_REG</a>	GPIO 32-39 output bit clear register_REG	0x3FF44018	RO
<a href="#">GPIO_ENABLE_REG</a>	GPIO 0-31 output enable register_REG	0x3FF44020	R/W
<a href="#">GPIO_ENABLE_W1TS_REG</a>	GPIO 0-31 output enable register_W1TS_REG	0x3FF44024	RO
<a href="#">GPIO_ENABLE_W1TC_REG</a>	GPIO 0-31 output enable register_W1TC_REG	0x3FF44028	RO
<a href="#">GPIO_ENABLE1_REG</a>	GPIO 0-31 output enable register1_REG	0x3FF4402C	R/W
<a href="#">GPIO_ENABLE1_W1TS_REG</a>	GPIO 0-31 output enable register1_W1TS_REG	0x3FF44030	RO
<a href="#">GPIO_ENABLE1_W1TC_REG</a>	GPIO 0-31 output enable register1_W1TC_REG	0x3FF44034	RO
<a href="#">GPIO_ENABLE_W1TS_REG</a>	GPIO 0-31 output enable bit set register_REG	0x3FF44024	RO
<a href="#">GPIO_ENABLE_W1TC_REG</a>	GPIO 0-31 output enable bit clear register_REG	0x3FF44028	RO
<a href="#">GPIO_ENABLE1_REG</a>	GPIO 32-39 output enable register_REG	0x3FF4402C	R/W
<a href="#">GPIO_ENABLE1_W1TS_REG</a>	GPIO 32-39 output enable register_W1TS_REG	0x3FF44030	RO
<a href="#">GPIO_ENABLE1_W1TC_REG</a>	GPIO 32-39 output enable register_W1TC_REG	0x3FF44034	RO
<a href="#">GPIO_ENABLE1_W1TS_REG</a>	GPIO 32-39 output enable bit set register_REG	0x3FF44030	RO

Name	Description	Address	Access
<a href="#">GPIO_ENABLE1_W1TC_REG</a>	GPIO 32-39 output enable bit clear register_REG	0x3FF44034	RO
<a href="#">GPIO_STRAP_REG</a>	Bootstrap pin value register_REG	0x3FF44038	RO
<a href="#">GPIO_IN_REG</a>	GPIO 0-31 input register_REG	0x3FF4403C	RO
<a href="#">GPIO_IN1_REG</a>	GPIO 0-31 input register1_REG	0x3FF44040	RO
<a href="#">GPIO_IN1_REG</a>	GPIO 32-39 input register_REG	0x3FF44040	RO
<a href="#">GPIO_STATUS_REG</a>	GPIO 0-31 interrupt status register_REG	0x3FF44044	R/W
<a href="#">GPIO_STATUS_W1TS_REG</a>	GPIO 0-31 interrupt status register_W1TS_REG	0x3FF44048	RO
<a href="#">GPIO_STATUS_W1TC_REG</a>	GPIO 0-31 interrupt status register_W1TC_REG	0x3FF4404C	RO
<a href="#">GPIO_STATUS1_REG</a>	GPIO 0-31 interrupt status register1_REG	0x3FF44050	R/W
<a href="#">GPIO_STATUS1_W1TS_REG</a>	GPIO 0-31 interrupt status register1_W1TS_REG	0x3FF44054	RO
<a href="#">GPIO_STATUS1_W1TC_REG</a>	GPIO 0-31 interrupt status register1_W1TC_REG	0x3FF44058	RO
<a href="#">GPIO_STATUS_W1TS_REG</a>	GPIO 0-31 interrupt status bit set register_REG	0x3FF44048	RO
<a href="#">GPIO_STATUS_W1TC_REG</a>	GPIO 0-31 interrupt status bit clear register_REG	0x3FF4404C	RO
<a href="#">GPIO_STATUS1_REG</a>	GPIO 32-39 interrupt status register_REG	0x3FF44050	R/W
<a href="#">GPIO_STATUS1_W1TS_REG</a>	GPIO 32-39 interrupt status register_W1TS_REG	0x3FF44054	RO
<a href="#">GPIO_STATUS1_W1TC_REG</a>	GPIO 32-39 interrupt status register_W1TC_REG	0x3FF44058	RO
<a href="#">GPIO_STATUS1_W1TS_REG</a>	GPIO 32-39 interrupt status bit set register_REG	0x3FF44054	RO
<a href="#">GPIO_STATUS1_W1TC_REG</a>	GPIO 32-39 interrupt status bit clear register_REG	0x3FF44058	RO
<a href="#">GPIO_ACPU_INT_REG</a>	GPIO 0-31 APP_CPU interrupt status_REG	0x3FF44060	RO
<a href="#">GPIO_ACPU_INT1_REG</a>	GPIO 0-31 APP_CPU interrupt status1_REG	0x3FF44074	RO
<a href="#">GPIO_ACPU_NMI_INT_REG</a>	GPIO 0-31 APP_CPU non-maskable interrupt status_REG	0x3FF44064	RO
<a href="#">GPIO_ACPU_NMI_INT1_REG</a>	GPIO 0-31 APP_CPU non-maskable interrupt status1_REG	0x3FF44078	RO
<a href="#">GPIO_PCPU_INT_REG</a>	GPIO 0-31 PRO_CPU interrupt status_REG	0x3FF44068	RO
<a href="#">GPIO_PCPU_INT1_REG</a>	GPIO 0-31 PRO_CPU interrupt status1_REG	0x3FF4407C	RO
<a href="#">GPIO_PCPU_NMI_INT_REG</a>	GPIO 0-31 PRO_CPU non-maskable interrupt status_REG	0x3FF4406C	RO
<a href="#">GPIO_PCPU_NMI_INT1_REG</a>	GPIO 0-31 PRO_CPU non-maskable interrupt status1_REG	0x3FF44080	RO
<a href="#">GPIO_ACPU_INT1_REG</a>	GPIO 32-39 APP_CPU interrupt status_REG	0x3FF44074	RO
<a href="#">GPIO_ACPU_NMI_INT1_REG</a>	GPIO 32-39 APP_CPU non-maskable interrupt status_REG	0x3FF44078	RO
<a href="#">GPIO_PCPU_INT1_REG</a>	GPIO 32-39 PRO_CPU interrupt status_REG	0x3FF4407C	RO
<a href="#">GPIO_PCPU_NMI_INT1_REG</a>	GPIO 32-39 PRO_CPU non-maskable interrupt status_REG	0x3FF44080	RO
<a href="#">GPIO_PIN0_REG</a>	Configuration for GPIO pin 0_REG	0x3FF44088	R/W
<a href="#">GPIO_PIN1_REG</a>	Configuration for GPIO pin 1_REG	0x3FF4408C	R/W
<a href="#">GPIO_PIN2_REG</a>	Configuration for GPIO pin 2_REG	0x3FF44090	R/W
...	...		
<a href="#">GPIO_PIN38_REG</a>	Configuration for GPIO pin 38_REG	0x3FF44120	R/W
<a href="#">GPIO_PIN39_REG</a>	Configuration for GPIO pin 39_REG	0x3FF44124	R/W
<a href="#">GPIO_FUNC0_IN_SEL_CFG_REG</a>	Peripheral function 0 input selection register_REG	0x3FF44130	R/W
<a href="#">GPIO_FUNC1_IN_SEL_CFG_REG</a>	Peripheral function 1 input selection register_REG	0x3FF44134	R/W

Name	Description	Address	Access
...	...		
<a href="#">GPIO_FUNC254_IN_SEL_CFG_REG</a>	Peripheral function 254 input selection register_REG	0x3FF44528	R/W
<a href="#">GPIO_FUNC255_IN_SEL_CFG_REG</a>	Peripheral function 255 input selection register_REG	0x3FF4452C	R/W
<a href="#">GPIO_FUNC0_OUT_SEL_CFG_REG</a>	Peripheral output selection for GPIO 0_REG	0x3FF44530	R/W
<a href="#">GPIO_FUNC1_OUT_SEL_CFG_REG</a>	Peripheral output selection for GPIO 1_REG	0x3FF44534	R/W
...	...		
<a href="#">GPIO_FUNC38_OUT_SEL_CFG_REG</a>	Peripheral output selection for GPIO 38_REG	0x3FF445C8	R/W
<a href="#">GPIO_FUNC39_OUT_SEL_CFG_REG</a>	Peripheral output selection for GPIO 39_REG	0x3FF445CC	R/W

Name	Description	Address	Access
<a href="#">IO_MUX_GPIO36_REG</a>	Configuration register for pad GPIO36	0x3FF53004	R/W
<a href="#">IO_MUX_GPIO37_REG</a>	Configuration register for pad GPIO37	0x3FF53008	R/W
<a href="#">IO_MUX_GPIO38_REG</a>	Configuration register for pad GPIO38	0x3FF5300C	R/W
<a href="#">IO_MUX_GPIO39_REG</a>	Configuration register for pad GPIO39	0x3FF53010	R/W
<a href="#">IO_MUX_GPIO34_REG</a>	Configuration register for pad GPIO34	0x3FF53014	R/W
<a href="#">IO_MUX_GPIO35_REG</a>	Configuration register for pad GPIO35	0x3FF53018	R/W
<a href="#">IO_MUX_GPIO32_REG</a>	Configuration register for pad GPIO32	0x3FF5301C	R/W
<a href="#">IO_MUX_GPIO33_REG</a>	Configuration register for pad GPIO33	0x3FF53020	R/W
<a href="#">IO_MUX_GPIO25_REG</a>	Configuration register for pad GPIO25	0x3FF53024	R/W
<a href="#">IO_MUX_GPIO26_REG</a>	Configuration register for pad GPIO26	0x3FF53028	R/W
<a href="#">IO_MUX_GPIO27_REG</a>	Configuration register for pad GPIO27	0x3FF5302C	R/W
<a href="#">IO_MUX_MTMS_REG</a>	Configuration register for pad MTMS	0x3FF53030	R/W
<a href="#">IO_MUX_MTDI_REG</a>	Configuration register for pad MTDI	0x3FF53034	R/W
<a href="#">IO_MUX_MTCK_REG</a>	Configuration register for pad MTCK	0x3FF53038	R/W
<a href="#">IO_MUX_MTDO_REG</a>	Configuration register for pad MTDO	0x3FF5303C	R/W
<a href="#">IO_MUX_GPIO2_REG</a>	Configuration register for pad GPIO2	0x3FF53040	R/W
<a href="#">IO_MUX_GPIO0_REG</a>	Configuration register for pad GPIO0	0x3FF53044	R/W
<a href="#">IO_MUX_GPIO4_REG</a>	Configuration register for pad GPIO4	0x3FF53048	R/W
<a href="#">IO_MUX_GPIO16_REG</a>	Configuration register for pad GPIO16	0x3FF5304C	R/W
<a href="#">IO_MUX_GPIO17_REG</a>	Configuration register for pad GPIO17	0x3FF53050	R/W
<a href="#">IO_MUX_SD_DATA2_REG</a>	Configuration register for pad SD_DATA2	0x3FF53054	R/W
<a href="#">IO_MUX_SD_DATA3_REG</a>	Configuration register for pad SD_DATA3	0x3FF53058	R/W
<a href="#">IO_MUX_SD_CMD_REG</a>	Configuration register for pad SD_CMD	0x3FF5305C	R/W
<a href="#">IO_MUX_SD_CLK_REG</a>	Configuration register for pad SD_CLK	0x3FF53060	R/W
<a href="#">IO_MUX_SD_DATA0_REG</a>	Configuration register for pad SD_DATA0	0x3FF53064	R/W
<a href="#">IO_MUX_SD_DATA1_REG</a>	Configuration register for pad SD_DATA1	0x3FF53068	R/W
<a href="#">IO_MUX_GPIO5_REG</a>	Configuration register for pad GPIO5	0x3FF5306C	R/W
<a href="#">IO_MUX_GPIO18_REG</a>	Configuration register for pad GPIO18	0x3FF53070	R/W
<a href="#">IO_MUX_GPIO19_REG</a>	Configuration register for pad GPIO19	0x3FF53074	R/W
<a href="#">IO_MUX_GPIO20_REG</a>	Configuration register for pad GPIO20	0x3FF53078	R/W
<a href="#">IO_MUX_GPIO21_REG</a>	Configuration register for pad GPIO21	0x3FF5307C	R/W
<a href="#">IO_MUX_GPIO22_REG</a>	Configuration register for pad GPIO22	0x3FF53080	R/W

Name	Description	Address	Access
<a href="#">IO_MUX_U0RXD_REG</a>	Configuration register for pad U0RXD	0x3FF53084	R/W
<a href="#">IO_MUX_U0TXD_REG</a>	Configuration register for pad U0TXD	0x3FF53088	R/W
<a href="#">IO_MUX_GPIO23_REG</a>	Configuration register for pad GPIO23	0x3FF5308C	R/W
<a href="#">IO_MUX_GPIO24_REG</a>	Configuration register for pad GPIO24	0x3FF53090	R/W

Name	Description	Address	Access
<b>GPIO configuration / data registers</b>			
<a href="#">RTCIO_RTC_GPIO_OUT_REG</a>	RTC GPIO output register_REG	0x3FF48000	R/W
<a href="#">RTCIO_RTC_GPIO_OUT_W1TS_REG</a>	RTC GPIO output register_W1TS_REG	0x3FF48001	WO
<a href="#">RTCIO_RTC_GPIO_OUT_W1TC_REG</a>	RTC GPIO output register_W1TC_REG	0x3FF48002	WO
<a href="#">RTCIO_RTC_GPIO_OUT_W1TS_REG</a>	RTC GPIO output bit set register_REG	0x3FF48001	WO
<a href="#">RTCIO_RTC_GPIO_OUT_W1TC_REG</a>	RTC GPIO output bit clear register_REG	0x3FF48002	WO
<a href="#">RTCIO_RTC_GPIO_ENABLE_REG</a>	RTC GPIO output enable register_REG	0x3FF48003	R/W
<a href="#">RTCIO_RTC_GPIO_ENABLE_W1TS_REG</a>	RTC GPIO output enable register_W1TS_REG	0x3FF48004	WO
<a href="#">RTCIO_RTC_GPIO_ENABLE_W1TC_REG</a>	RTC GPIO output enable register_W1TC_REG	0x3FF48005	WO
<a href="#">RTCIO_RTC_GPIO_ENABLE_W1TS_REG</a>	RTC GPIO output enable bit set register_REG	0x3FF48004	WO
<a href="#">RTCIO_RTC_GPIO_ENABLE_W1TC_REG</a>	RTC GPIO output enable bit clear register_REG	0x3FF48005	WO
<a href="#">RTCIO_RTC_GPIO_STATUS_REG</a>	RTC GPIO interrupt status register_REG	0x3FF48006	R/W
<a href="#">RTCIO_RTC_GPIO_STATUS_W1TS_REG</a>	RTC GPIO interrupt status register_W1TS_REG	0x3FF48007	WO
<a href="#">RTCIO_RTC_GPIO_STATUS_W1TC_REG</a>	RTC GPIO interrupt status register_W1TC_REG	0x3FF48008	WO
<a href="#">RTCIO_RTC_GPIO_STATUS_W1TS_REG</a>	RTC GPIO interrupt status bit set register_REG	0x3FF48007	WO
<a href="#">RTCIO_RTC_GPIO_STATUS_W1TC_REG</a>	RTC GPIO interrupt status bit clear register_REG	0x3FF48008	WO
<a href="#">RTCIO_RTC_GPIO_IN_REG</a>	RTC GPIO input register_REG	0x3FF48009	RO
<a href="#">RTCIO_RTC_GPIO_PIN0_REG</a>	RTC configuration for pin 0_REG	0x3FF4800A	R/W
<a href="#">RTCIO_RTC_GPIO_PIN1_REG</a>	RTC configuration for pin 1_REG	0x3FF4800B	R/W
<a href="#">RTCIO_RTC_GPIO_PIN2_REG</a>	RTC configuration for pin 2_REG	0x3FF4800C	R/W
<a href="#">RTCIO_RTC_GPIO_PIN3_REG</a>	RTC configuration for pin 3_REG	0x3FF4800D	R/W
<a href="#">RTCIO_RTC_GPIO_PIN4_REG</a>	RTC configuration for pin 4_REG	0x3FF4800E	R/W
<a href="#">RTCIO_RTC_GPIO_PIN5_REG</a>	RTC configuration for pin 5_REG	0x3FF4800F	R/W
<a href="#">RTCIO_RTC_GPIO_PIN6_REG</a>	RTC configuration for pin 6_REG	0x3FF48010	R/W
<a href="#">RTCIO_RTC_GPIO_PIN7_REG</a>	RTC configuration for pin 7_REG	0x3FF48011	R/W
<a href="#">RTCIO_RTC_GPIO_PIN8_REG</a>	RTC configuration for pin 8_REG	0x3FF48012	R/W
<a href="#">RTCIO_RTC_GPIO_PIN9_REG</a>	RTC configuration for pin 9_REG	0x3FF48013	R/W
<a href="#">RTCIO_RTC_GPIO_PIN10_REG</a>	RTC configuration for pin 10_REG	0x3FF48014	R/W
<a href="#">RTCIO_RTC_GPIO_PIN11_REG</a>	RTC configuration for pin 11_REG	0x3FF48015	R/W
<a href="#">RTCIO_RTC_GPIO_PIN12_REG</a>	RTC configuration for pin 12_REG	0x3FF48016	R/W
<a href="#">RTCIO_RTC_GPIO_PIN13_REG</a>	RTC configuration for pin 13_REG	0x3FF48017	R/W
<a href="#">RTCIO_RTC_GPIO_PIN14_REG</a>	RTC configuration for pin 14_REG	0x3FF48018	R/W
<a href="#">RTCIO_RTC_GPIO_PIN15_REG</a>	RTC configuration for pin 15_REG	0x3FF48019	R/W
<a href="#">RTCIO_RTC_GPIO_PIN16_REG</a>	RTC configuration for pin 16_REG	0x3FF4801A	R/W
<a href="#">RTCIO_RTC_GPIO_PIN17_REG</a>	RTC configuration for pin 17_REG	0x3FF4801B	R/W

Name	Description	Address	Access
<a href="#">RTCIO_DIG_PAD_HOLD_REG</a>	RTC GPIO hold register_REG	0x3FF4801D	R/W
<b>GPIO RTC functions configuration registers</b>			
<a href="#">RTCIO_HALL_SENS_REG</a>	Hall sensor configuration_REG	0x3FF4801E	R/W
<a href="#">RTCIO_SENSOR_PADS_REG</a>	Sensor pads configuration register_REG	0x3FF4801F	R/W
<a href="#">RTCIO_ADC_PAD_REG</a>	ADC configuration register_REG	0x3FF48020	R/W
<a href="#">RTCIO_PAD_DAC1_REG</a>	DAC1 configuration register_REG	0x3FF48021	R/W
<a href="#">RTCIO_PAD_DAC2_REG</a>	DAC2 configuration register_REG	0x3FF48022	R/W
<a href="#">RTCIO_XTAL_32K_PAD_REG</a>	32KHz crystal pads configuration register_REG	0x3FF48023	R/W
<a href="#">RTCIO_TOUCH_CFG_REG</a>	Touch sensor configuration register_REG	0x3FF48024	R/W
<a href="#">RTCIO_EXT_WAKEUP0_REG</a>	External wake up configuration register_REG	0x3FF4802F	R/W
<a href="#">RTCIO_XTL_EXT_CTR_REG</a>	Crystal power down enable gpio source_REG	0x3FF48030	R/W
<a href="#">RTCIO_SAR_I2C_IO_REG</a>	RTC I2C pad selection_REG	0x3FF48031	R/W

4.13 Registers

Register 4.1: GPIO\_OUT\_REG (0x0004)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO\_OUT\_REG GPIO0-31 output value. (R/W)

Register 4.2: GPIO\_OUT\_W1TS\_REG (0x0008)

31																															0
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO\_OUT\_W1TS\_REG GPIO0-31 output set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_OUT\_DATA will be set. (RO)

Register 4.3: GPIO\_OUT\_W1TC\_REG (0x000c)

31																															0																														
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset																									

GPIO\_OUT\_W1TC\_REG GPIO0-31 output clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_OUT\_DATA will be cleared. (RO)

Register 4.4: GPIO\_OUT1\_REG (0x0010)

(reserved)																								GPIO_OUT_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	0	0	x	x	x	x	x	x	x	x	Reset

GPIO\_OUT\_DATA GPIO32-39 output value. (R/W)

**Register 4.5: GPIO\_OUT1\_W1TS\_REG (0x0014)**

(reserved)																								GPIO_OUT_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	x	x	x	x	x	x	x	x	Reset	

Reset

**GPIO\_OUT\_DATA** GPIO32-39 output value set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_OUT1\_DATA will be set. (RO)

**Register 4.6: GPIO\_OUT1\_W1TC\_REG (0x0018)**

(reserved)																								GPIO_OUT_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	0	x	x	x	x	x	x	x	x	Reset

Reset

**GPIO\_OUT\_DATA** GPIO32-39 output value clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_OUT1\_DATA will be cleared. (RO)

**Register 4.7: GPIO\_ENABLE\_REG (0x0020)**

31																															0			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

Reset

**GPIO\_ENABLE\_REG** GPIO0-31 output enable. (R/W)

**Register 4.8: GPIO\_ENABLE\_W1TS\_REG (0x0024)**

31																													0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

Reset

**GPIO\_ENABLE\_W1TS\_REG** GPIO0-31 output enable set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_ENABLE will be set. (RO)

**Register 4.9: GPIO\_ENABLE\_W1TC\_REG (0x0028)**

31																															0			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

Reset

**GPIO\_ENABLE\_W1TC\_REG** GPIO0-31 output enable clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_ENABLE will be cleared. (RO)

**Register 4.10: GPIO\_ENABLE1\_REG (0x002c)**

<div>(reserved)</div>																												<div>GPIO_ENABLE_DATA</div>															
31																												7								0							
0 0																												x x x x x x x x x								Reset							

**GPIO\_ENABLE\_DATA** GPIO32-39 output enable. (R/W)

**Register 4.11: GPIO\_ENABLE1\_W1TS\_REG (0x0030)**

(reserved)																								GPIO_ENABLE_DATA															
31																								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	x	x	x	x	x	x	x	x	Reset							

**GPIO\_ENABLE\_DATA** GPIO32-39 output enable set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_ENABLE1 will be set. (RO)

**Register 4.12: GPIO\_ENABLE1\_W1TC\_REG (0x0034)**

(reserved)																								GPIO_ENABLE_DATA															
31																								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	x	x	x	x	x	x	x	x	Reset					

**GPIO\_ENABLE\_DATA** GPIO32-39 output enable clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_ENABLE1 will be cleared. (RO)

**Register 4.13: GPIO\_STRAP\_REG (0x0038)**

(reserved)																GPIO_STRAPPING													
31																	16	15											0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	Reset

**GPIO\_STRAPPING** GPIO strapping results: boot\_sel\_chip[5:0]: MTDI, GPIO0, GPIO2, GPIO4, MTDO, GPIO5.



**Register 4.14: GPIO\_IN\_REG (0x003c)**

31	0
x x	Reset

**GPIO\_IN\_REG** GPIO0-31 input value. Each bit represents pad input value, 1 for high level and 0 for low level. (RO)

**Register 4.15: GPIO\_IN1\_REG (0x0040)**

31	8	7	0
(reserved)			
GPIO_IN_DATA_NEXT			
0 0	x x x x x x x x	Reset	

**GPIO\_IN\_DATA\_NEXT** GPIO32-39 input value. Each bit represents pad input value. (RO)

**Register 4.16: GPIO\_STATUS\_REG (0x0044)**

31	0
x x	Reset

**GPIO\_STATUS\_REG** GPIO0-31 interrupt status register. Each bit can be the two interrupt sources for the two CPUs. The enable bits in GPIO\_STATUS\_INTERRUPT corresponding to the 0-4 bits in GPIO\_PIN<sub>n</sub>\_REG should be set to 1. (R/W)

**Register 4.17: GPIO\_STATUS\_W1TS\_REG (0x0048)**

31	0
x x	Reset

**GPIO\_STATUS\_W1TS\_REG** GPIO0-31 interrupt status set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_STATUS\_INTERRUPT will be set. (RO)

**Register 4.18: GPIO\_STATUS\_W1TC\_REG (0x004c)**

31	0
x x	Reset

**GPIO\_STATUS\_W1TC\_REG** GPIO0-31 interrupt status clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_STATUS\_INTERRUPT will be cleared. (RO)

### Register 4.19: GPIO\_STATUS1\_REG (0x0050)

31																									8	7									0
0 0																								x x x x x x x x								Reset			

**GPIO\_STATUS\_INTERRUPT** GPIO32-39 interrupt status. (R/W)

### Register 4.20: GPIO\_STATUS1\_W1TS\_REG (0x0054)

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	x	x	x	x	x	x	x	x

Reset

**GPIO\_STATUS\_INTERRUPT** GPIO32-39 interrupt status set register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_STATUS\_INTERRUPT1 will be set. (RO)

### Register 4.21: GPIO\_STATUS1\_W1TC\_REG (0x0058)

Diagram illustrating the structure of the GPIO Status Interrupt register. The register is 32 bits wide, divided into two main sections:

- Reserved (bits 31-8):** Labeled "(reserved)".
- GPIO\_STATUS\_INTERRUPT (bits 7-0):** Labeled "GPIO\_STATUS\_INTERRUPT". This section contains 8 bits, with bit 0 labeled "Reset".

**GPIO\_STATUS\_INTERRUPT** GPIO32-39 interrupt status clear register. For every bit that is one in the value that is written here, the corresponding bit in GPIO\_STATUS\_INTERRUPT1 will be cleared. (RO)

### Register 4.22: GPIO\_ACPU\_INT\_REG (0x0060)

[illegible]

**GPIO\_ACPU\_INT\_REG** GPIO0-31 APP CPU interrupt status. (RO)

**Register 4.23: GPIO\_ACPU\_NMI\_INT\_REG (0x0064)**

31	0
x x	Reset

**GPIO\_ACPU\_NMI\_INT\_REG** GPIO0-31 APP CPU non-maskable interrupt status. (RO)

**Register 4.24: GPIO\_PCPU\_INT\_REG (0x0068)**

31	0
x x	Reset

**GPIO\_PCPU\_INT\_REG** GPIO0-31 PRO CPU interrupt status. (RO)

**Register 4.25: GPIO\_PCPU\_NMI\_INT\_REG (0x006c)**

31	0
x x	Reset

**GPIO\_PCPU\_NMI\_INT\_REG** GPIO0-31 PRO CPU non-maskable interrupt status. (RO)

**Register 4.26: GPIO\_ACPU\_INT1\_REG (0x0074)**

(reserved)																												GPIO_APPCPU_INT															
31																												7								0							
0 0																												x x x x x x x x x								Reset							

**GPIO\_APPCPU\_INT** GPIO32-39 APP CPU interrupt status. (RO)

**Register 4.27: GPIO\_ACPU\_NMI\_INT1\_REG (0x0078)**

<div>(reserved)</div>																												<div>GPIO_APPCPU_NMI_INT</div>															
31																												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	x	x	x	x	x	x	x	x	Reset							

**GPIO\_APPCPU\_NMI\_INT** GPIO32-39 APP CPU non-maskable interrupt status. (RO)

#### Register 4.28: GPIO\_PCPU\_INT1\_REG (0x007c)

[illegible]

**GPIO\_PROCPU\_INT** GPIO32-39 PRO CPU interrupt status. (RO)

#### Register 4.29: GPIO\_PCPU\_NMI\_INT1\_REG (0x0080)

(reserved)																								GPIO_PROCPU_NMI_INT															
31																								7								0							
0 0																								x x x x x x x x								Reset							

**GPIO\_PROCPU\_NMI\_INT** GPIO32-39 PRO CPU non-maskable interrupt status. (RO)

**Register 4.30: GPIO\_PIN $n$ \_REG ( $n$ : 0-39) (0x88+0x4\* $n$ )**

Register map for GPIO\_PIN2\_PAD\_DRIVER (32 bits):

- 31: (reserved)
- 18: (reserved)
- 17: GPIO\_PIN2\_INT\_ENA
- 16: (reserved)
- 15: (reserved)
- 14: (reserved)
- 13: (reserved)
- 12: (reserved)
- 11: (reserved)
- 10: (reserved)
- 9: GPIO\_PIN2\_WAKEUP\_ENABLE
- 8: (reserved)
- 7: (reserved)
- 6: GPIO\_PIN2\_INT\_TYPE
- 5: (reserved)
- 4: (reserved)
- 3: (reserved)
- 2: (reserved)
- 1: (reserved)
- 0: GPIO\_PIN2\_PAD\_DRIVER

**GPIO\_PIN $n$ \_INT\_ENA** Interrupt enable bits for pin  $n$ : (R/W)

- bit0: APP CPU interrupt enable;  
bit1: APP CPU non-maskable interrupt enable;  
bit3: PRO CPU interrupt enable;  
bit4: PRO CPU non-maskable interrupt enable.

**GPIO\_PIN<sub>n</sub>\_WAKEUP\_ENABLE** GPIO wake up enable. Will only wake the CPU from Light-sleep.  
(R/W)

**GPIO\_PIN<sub>n</sub>\_INT\_TYPE** Interrupt type selection: (R/W)

- 0: GPIO interrupt disable;
- 1: rising edge trigger;
- 2: falling edge trigger;
- 3: any edge trigger;
- 4: low level trigger;
- 5: high level trigger.

**GPIO\_PIN<sub>n</sub>\_PAD\_DRIVER** 0: normal output; 1: open drain output. (R/W)

**Register 4.31: GPIO\_FUNC<sub>m</sub> IN\_SEL\_CFG\_REG (*m*: 0-255) (0x130+0x4\**m*)**

Diagram of the GPIO\_FUNC\_IN\_SEL register. The register is 32 bits wide, with bits 31-8 reserved. Bits 7-6 are labeled GPIO\_FUNC\_IN\_SEL. Bits 5-0 are labeled GPIO\_FUNC\_IN\_SEL. The register is shown with a value of 0x00000000.

**GPIO\_SIG<sub>m</sub>\_IN\_SEL** Bypass the GPIO Matrix. 0: route through GPIO Matrix, 1: connect signal directly to peripheral configured in the IO MUX. (R/W)

**GPIO\_FUNC<sub>m</sub> IN INV SEL** Invert the input value. 1: invert; 0: do not invert. (R/W)

**GPIO\_FUNC*m*\_IN\_SEL** Selection control for peripheral input *m*. A value of 0-39 selects which of the 40 GPIO Matrix input pins this signal is connected to, or 0x38 for a constant high input or 0x30 for a constant low input. (R/W)

**Register 4.32: GPIO\_FUNC<sub>n</sub>\_OUT\_SEL\_CFG\_REG (*n*: 0-39) (0x530+0x4\**n*)**

Register 0x00000000: GPIO Function Selection Register

Bits 31-12: (reserved)

Bits 11-8: Control bits

Bit 11: x

Bit 10: x

Bit 9: x

Bit 8: x

Bits 7-0: Function selection bits (n)

Bit 7: GPIO\_FUNC<sub>n</sub>\_OEN\_SEL

Bit 6: GPIO\_FUNC<sub>n</sub>\_OEN\_SEL

Bit 5: GPIO\_FUNC<sub>n</sub>\_OUT\_INV\_SEL

Bit 4: GPIO\_FUNC<sub>n</sub>\_OUT\_INV\_SEL

Bit 3: GPIO\_FUNC<sub>n</sub>\_OUT\_SEL

Bit 2: GPIO\_FUNC<sub>n</sub>\_OUT\_SEL

Bit 1: GPIO\_FUNC<sub>n</sub>\_OUT\_SEL

Bit 0: GPIO\_FUNC<sub>n</sub>\_OUT\_SEL

Reset

**GPIO\_FUNCn\_OEN\_INV\_SEL** 1: Invert the output enable signal; 0: do not invert the output enable signal. (R/W)

**GPIO\_FUNCn\_OEN\_SEL** 1: Force the output enable signal to be sourced from bit *n* of GPIO\_ENABLE\_REG; 0: use output enable signal from peripheral. (R/W)

**GPIO\_FUNC<sub>n</sub>\_OUT\_INV\_SEL** 1: Invert the output value; 0: do not invert the output value. (R/W)

**GPIO\_FUNC***n***\_OUT\_SEL** Selection control for GPIO output *n*. A value of *s* ( $0 \leq s < 256$ ) connects peripheral output *s* to GPIO output *n*. A value of 256 selects bit *n* of GPIO\_DATA\_REG and GPIO\_ENABLE\_REG as the output value and output enable. (R/W)

**Register 4.33: IO\_MUX\_x\_REG (x: GPIO0-GPIO39) (0x10+4\*x)**

(reserved)																IO_X_MCU_SEL		IO_X_FUNC_DRV		IO_X_FUNC_IE		IO_X_FUNC_WPU		IO_X_FUNC_WPD		IO_X_MCU_DRV		IO_X_MCU_IE		IO_X_MCU_WPU		IO_X_MCU_WPD		IO_X_SLP_SEL		IO_X_MCU_OE		
31																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 0 0																0x0		0x2		0 0 0		0x0		0 0 0		0 0 0		0 0 0		0 0 0		Reset						

**IO\_x\_MCU\_SEL** Select the IO\_MUX function for this signal. 0 selects Function 1, 1 selects Function 2, etc. (R/W)

**IO\_x\_FUNC\_DRV** Select the drive strength of the pad. A higher value selects a higher strength. (R/W)

**IO\_x\_FUNC\_IE** Input enable of the pad. 1: input enabled; 0: input disabled. (R/W)

**IO\_x\_FUNC\_WPU** Pull-up enable of the pad. 1: internal pull-up enabled; 0: internal pull-up disabled. (R/W)

**IO\_x\_FUNC\_WPD** Pull-down enable of the pad. 1: internal pull-down enabled, 0: internal pull-down disabled. (R/W)

**IO\_x\_MCU\_DRV** Select the drive strength of the pad during sleep mode. A higher value selects a higher strength. (R/W)

**IO\_x\_MCU\_IE** Input enable of the pad during sleep mode. 1: input enabled; 0: input disabled. (R/W)

**IO\_x\_MCU\_WPU** Pull-up enable of the pad during sleep mode. 1: internal pull-up enabled; 0: internal pull-up disabled. (R/W)

**IO\_x\_MCU\_WPD** Pull-down enable of the pad during sleep mode. 1: internal pull-down enabled; 0: internal pull-down disabled. (R/W)

**IO\_x\_SLP\_SEL** Sleep mode selection of this pad. Set to 1 to put the pad in sleep mode. (R/W)

**IO\_x\_MCU\_OE** Output enable of the pad in sleep mode. 1: enable output; 0: disable output. (R/W)

**Register 4.34: RTCIO\_RTC\_GPIO\_OUT\_REG (0x0000)**

RTCIO_RTC_GPIO_OUT_DATA																	(reserved)														
3114																	2714														
x x x x x x x x x x x x x x x x x x x																	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0														Reset

**RTCIO\_RTC\_GPIO\_OUT\_DATA** GPIO0-17 output register. Bit14 is GPIO[0], bit15 is GPIO[1], etc. (R/W)

**Register 4.35: RTCIO\_RTC\_GPIO\_OUT\_W1TS\_REG (0x0001)**

RTCIO_RTC_GPIO_OUT_DATA_W1TS														(reserved)															
31														14	27													14	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	Reset

**RTCIO\_RTC\_GPIO\_OUT\_DATA\_W1TS** GPIO0-17 output set register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_OUT will be set. (WO)

**Register 4.36: RTCIO\_RTC\_GPIO\_OUT\_W1TC\_REG (0x0002)**

RTCIO_RTC_GPIO_OUT_DATA_W1TC														(reserved)																																									
31														14														27														14													
x x x x x x x x x x x x x x x x x x x x														0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0														Reset																											

**RTCIO\_RTC\_GPIO\_OUT\_DATA\_W1TC** GPIO0-17 output clear register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_OUT will be cleared. (WO)

**Register 4.37: RTCIO\_RTC\_GPIO\_ENABLE\_REG (0x0003)**

RTCIO_RTC_GPIO_ENABLE														(reserved)																
31														14	27														14	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	Reset

**RTCIO\_RTC\_GPIO\_ENABLE** GPIO0-17 output enable. Bit14 is GPIO[0], bit15 is GPIO[1], etc. 1 means this GPIO pad is output. (R/W)



**Register 4.38: RTCIO\_RTC\_GPIO\_ENABLE\_W1TS\_REG (0x0004)**

RTCIO_RTC_GPIO_ENABLE_W1TS																(reserved)														
31															14	27													14	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

**RTCIO\_RTC\_GPIO\_ENABLE\_W1TS** GPIO0-17 output enable set register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_ENABLE will be set. (WO)

**Register 4.39: RTCIO\_RTC\_GPIO\_ENABLE\_W1TC\_REG (0x0005)**

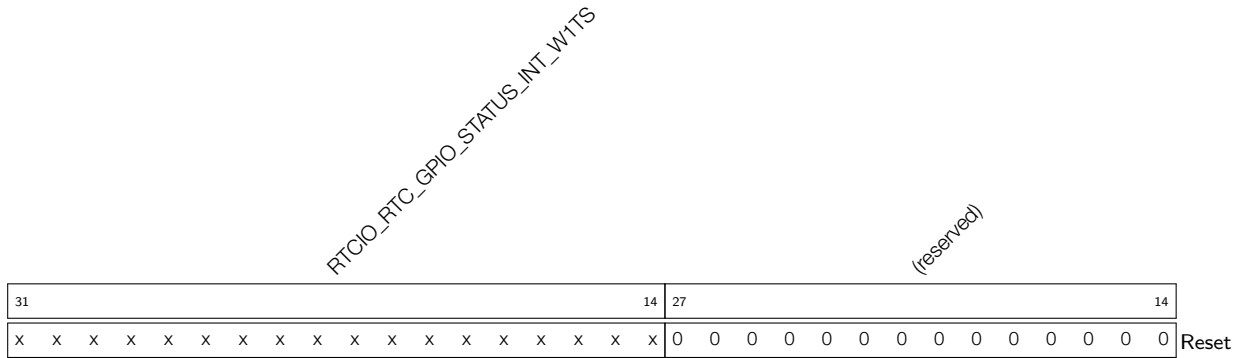
RTCIO_RTC_GPIO_ENABLE_W1TC																(reserved)														
31															14	27													14	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

**RTCIO\_RTC\_GPIO\_ENABLE\_W1TC** GPIO0-17 output enable clear register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_ENABLE will be cleared. (WO)

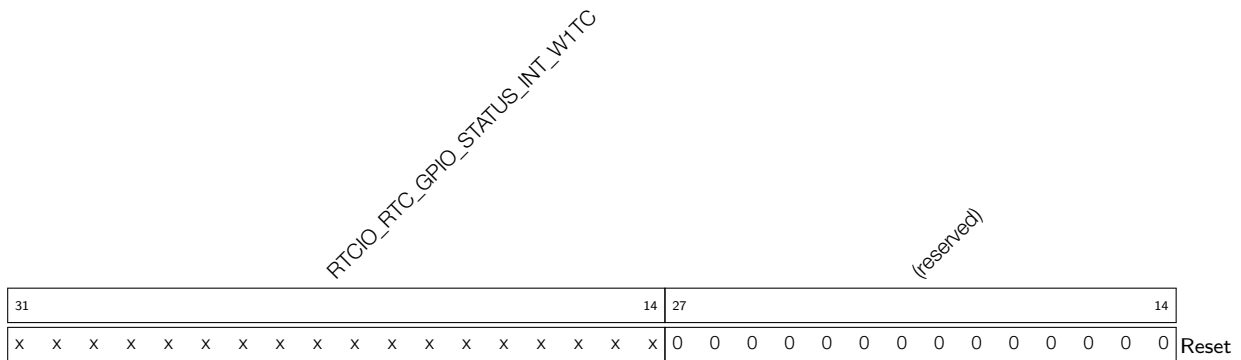
**Register 4.40: RTCIO\_RTC\_GPIO\_STATUS\_REG (0x0006)**

RTCIO_RTC_GPIO_STATUS_INT																(reserved)														
31															14	27													14	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

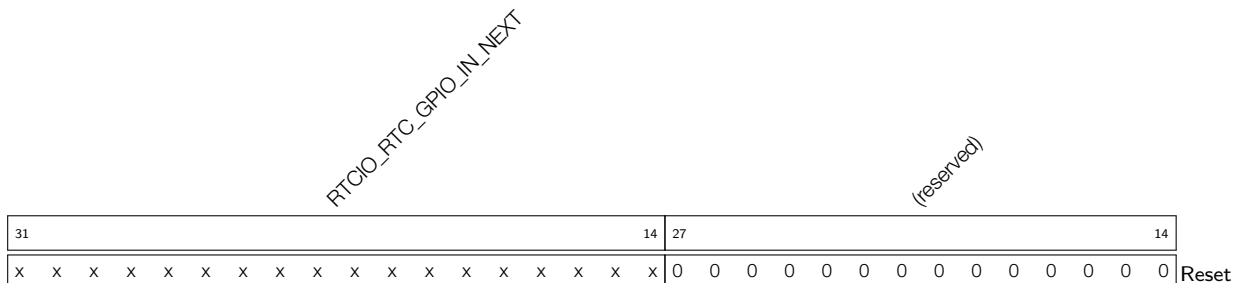
**RTCIO\_RTC\_GPIO\_STATUS\_INT** GPIO0-17 interrupt status. Bit14 is GPIO[0], bit15 is GPIO[1], etc. This register should be used together with RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_INT\_TYPE in RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_REG. 1 means there is corresponding interrupt, 0 means there is no interrupt. (R/W)

**Register 4.41: RTCIO\_RTC\_GPIO\_STATUS\_W1TS\_REG (0x0007)**

**RTCIO\_RTC\_GPIO\_STATUS\_INT\_W1TS** GPIO0-17 interrupt set register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_STATUS\_INT will be set. (WO)

**Register 4.42: RTCIO\_RTC\_GPIO\_STATUS\_W1TC\_REG (0x0008)**

**RTCIO\_RTC\_GPIO\_STATUS\_INT\_W1TC** GPIO0-17 interrupt clear register. For every bit that is one in the value that is written here, the corresponding bit in RTCIO\_RTC\_GPIO\_STATUS\_INT will be cleared. (WO)

**Register 4.43: RTCIO\_RTC\_GPIO\_IN\_REG (0x0009)**

**RTCIO\_RTC\_GPIO\_IN\_NEXT** GPIO0-17 input value. Bit14 is GPIO[0], bit15 is GPIO[1], etc. Each bit represents pad input value, 1 for high level and 0 for low level. (RO)

**Register 4.44: RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_REG (*n*: 0-17) (0xA+1\**n*)**

(reserved)																												RTCIO_RTC_GPIO_PIN <sub>n</sub> _WAKEUP_ENABLE				RTCIO_RTC_GPIO_PIN <sub>n</sub> _INT_TYPE				(reserved)				RTCIO_RTC_GPIO_PIN <sub>n</sub> _PAD_DRIVER				(reserved)							
31																												11		10	9	7		6	3		2	3	2												
0 0																												x	x	x	x	0 0 0 0				x	0	0	Reset												

**RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_WAKEUP\_ENABLE** GPIO wake up enable. This will only wake up the ESP32 from Light-sleep. (R/W)

**RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_INT\_TYPE** GPIO interrupt type selection. (R/W)

- 0: GPIO interrupt disable;
- 1: rising edge trigger;
- 2: falling edge trigger;
- 3: any edge trigger;
- 4: low level trigger;
- 5: high level trigger.

**RTCIO\_RTC\_GPIO\_PIN<sub>n</sub>\_PAD\_DRIVER** Pad driver selection. 0: normal output; 1: open drain. (R/W)

**Register 4.45: RTCIO\_DIG\_PAD\_HOLD\_REG (0x001d)**

31																															0	
0																																Reset

**RTCIO\_DIG\_PAD\_HOLD\_REG** Select which digital pads are on hold. 0 allows normal operation, 1 holds the pad. (R/W)

**Register 4.46: RTCIO\_HALL\_SENS\_REG (0x001e)**

RTCIO_HALL_XPD_HALL																															(reserved)																												RTCIO_HALL_PHASE																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31	30	59																																																							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	

**RTCIO\_HALL\_XPD\_HALL** Power on hall sensor and connect to VP and VN. (R/W)

**RTCIO\_HALL\_PHASE** Reverse the polarity of the hall sensor. (R/W)

Register 4.47: RTCIO\_SENSOR\_PADS\_REG (0x001f)

<div> <div>RTCIO_SENSOR_SENSE1_HOLD</div> <div>RTCIO_SENSOR_SENSE2_HOLD</div> <div>RTCIO_SENSOR_SENSE3_HOLD</div> <div>RTCIO_SENSOR_SENSE4_HOLD</div> <div>RTCIO_SENSOR_SENSE1_MUX_SEL</div> <div>RTCIO_SENSOR_SENSE2_MUX_SEL</div> <div>RTCIO_SENSOR_SENSE3_MUX_SEL</div> <div>RTCIO_SENSOR_SENSE4_MUX_SEL</div> <div>RTCIO_SENSOR_SENSE1_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE2_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE3_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE4_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE1_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE2_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE3_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE4_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE1_SLP_IE</div> <div>RTCIO_SENSOR_SENSE2_SLP_IE</div> <div>RTCIO_SENSOR_SENSE3_SLP_IE</div> <div>RTCIO_SENSOR_SENSE4_SLP_IE</div> <div>RTCIO_SENSOR_SENSE1_FUN_IE</div> <div>RTCIO_SENSOR_SENSE2_FUN_IE</div> <div>RTCIO_SENSOR_SENSE3_FUN_IE</div> <div>RTCIO_SENSOR_SENSE4_FUN_IE</div> <div>RTCIO_SENSOR_SENSE1_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE2_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE3_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE4_FUN_SEL</div> <div>RTCIO_SENSOR_SENSE1_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE2_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE3_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE4_SLP_SEL</div> <div>RTCIO_SENSOR_SENSE1_SLP_IE</div> <div>RTCIO_SENSOR_SENSE2_SLP_IE</div> <div>RTCIO_SENSOR_SENSE3_SLP_IE</div> <div>RTCIO_SENSOR_SENSE4_SLP_IE</div> <div>RTCIO_SENSOR_SENSE1_FUN_IE</div> <div>RTCIO_SENSOR_SENSE2_FUN_IE</div> <div>RTCIO_SENSOR_SENSE3_FUN_IE</div> <div>RTCIO_SENSOR_SENSE4_FUN_IE</div> <div>(reserved)</div> </div>																																					
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	7									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	0	0	0	0	Reset

**RTCIO\_SENSOR\_SENSE $n$ \_HOLD** Set to 1 to hold the output value on sense $n$ , 0 for normal operation. (R/W)

**RTCIO\_SENSOR\_SENSE $n$ \_MUX\_SEL** 1: route sense $n$  to the RTC block; 0: route sense $n$  to the digital IO\_MUX. (R/W)

**RTCIO\_SENSOR\_SENSE $n$ \_FUN\_SEL** Select the RTC IO\_MUX function for this pad. 0: select Function 0; 1: select Function 1. (R/W)

**RTCIO\_SENSOR\_SENSE $n$ \_SLP\_SEL** Sleep mode selection signal of the pad. Set to 1 to put the pad in sleep mode. (R/W)

**RTCIO\_SENSOR\_SENSE $n$ \_SLP\_IE** Input enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_SENSOR\_SENSE $n$ \_FUN\_IE** Input enable of the pad. 1: enabled; 0: disabled. (R/W)

#### Register 4.48: RTCIO\_ADC\_PAD\_REG (0x0020)

**RTCIO\_ADC\_ADC<sub>n</sub>\_HOLD** Set to 1 to hold the output value on the pad, 0 for normal operation.  
(R/W)

**RTCIO\_ADC\_ADC $n$ \_MUX\_SEL** 1: route pad to the digital IO\_MUX; (R/W)  
0: route pad to the RTC block.

**RTCIO\_ADC\_ADCn\_FUN\_SEL** Select the RTC function for this pad. 0: select Function 0; 1: select Function 1. (R/W)

**RTCIO\_ADC\_ADC $n$ \_SLP\_SEL** Sleep mode selection signal of the pad. Set this bit to 1 to put the pad to sleep. (R/W)

**RTCIO\_ADC\_ADC<sub>n</sub>\_SLP\_IE** Input enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_ADC\_ADC<sub>n</sub>\_FUN\_IE** Input enable of the pad. 1: enabled; 0: disabled. (R/W)



### Register 4.50: RTCIO PAD DAC2 REG (0x0022)

[illegible]

**RTCIO\_PAD\_PDAC2\_DRV** Select the drive strength of the pad. (R/W)

**RTCIO\_PAD\_PDAC2\_HOLD** Set to 1 to hold the output value on the pad, 0 for normal operation.  
(R/W)

**RTCIO\_PAD\_PDAC2\_RDE** 1: Pull-down on pad enabled; 0: Pull-down disabled. (R/W)

**RTCIO\_PAD\_PDAC2\_RUE** 1: Pull-up on pad enabled; 0: Pull-up disabled. (R/W)

**RTCIO\_PAD\_PDAC2\_DAC** PAD DAC2 output value. (R/W)

**RTCIO\_PAD\_PDAC2\_XPD\_DAC** Power on DAC2. PDAC2 needs to be tristated if we power on the DAC, i.e. IE=0, OE=0, RDE=0, RUE=0. (R/W)

**RTCIO\_PAD\_PDAC2\_MUX\_SEL** 1: route pad to the digital IO\_MUX; (R/W)  
0: route sense 4 to the RTC block.

**RTCIO\_PAD\_PDAC2\_FUN\_SEL** Select the RTC function for this pad. 0: select Function 0; 1: select Function 1. (R/W)

**RTCIO\_PAD\_PDAC2\_SLP\_SEL** Sleep mode selection signal of the pad. Set this bit to 1 to put the pad to sleep. (R/W)

**RTCIO\_PAD\_PDAC2\_SLP\_IE** Input enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_PAD\_PDAC2\_SLP\_OE** Output enable of the pad. 1: enabled; 0: disabled. (R/W)

**RTCIO\_PAD\_PDAC2\_FUN\_IE** Input enable of the pad. 1: enabled; 0: disabled. (R/W)

**RTCIO\_PAD\_PDAC2\_DAC\_XPD\_FORCE** Power on DAC2. Usually, we need to tristate PDAC2 if we power on the DAC, i.e. IE=0, OE=0, RDE=0, RUE=0. (R/W)

Register 4.51: RTCIO\_XTAL\_32K\_PAD\_REG (0x0023)

RTCIO_XTAL_X32N_DRV		RTCIO_XTAL_X32N_HOLD		RTCIO_XTAL_X32N_RDE		RTCIO_XTAL_X32N_RUE		RTCIO_XTAL_X32P_DRV		RTCIO_XTAL_X32P_HOLD		RTCIO_XTAL_X32P_RDE		RTCIO_XTAL_X32P_RUE		RTCIO_XTAL_DAC_XTAL_32K		RTCIO_XTAL_XPD_XTAL_32K		RTCIO_XTAL_X32N_MUX_SEL		RTCIO_XTAL_X32P_MUX_SEL		RTCIO_XTAL_X32N_FUN_SEL		RTCIO_XTAL_X32N_SLP_SEL		RTCIO_XTAL_X32N_SLP_IE		RTCIO_XTAL_X32N_SLP_OE		RTCIO_XTAL_X32P_FUN_SEL		RTCIO_XTAL_X32P_SLP_SEL		RTCIO_XTAL_X32P_SLP_IE		RTCIO_XTAL_X32P_SLP_OE		RTCIO_XTAL_DRES_XTAL_32K		RTCIO_XTAL_DBIAS_XTAL_32K		(reserved)			
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	1																
2		0		0		0		2		0		0		0		0		1		0		0		0		0		0		0		0		0		1		0		0		0		0		Reset	

**RTCIO\_XTAL\_X32N\_DRV** Select the drive strength of the pad. (R/W)

**RTCIO\_XTAL\_X32N\_HOLD** Set to 1 to hold the output value on the pad, 0 for normal operation. (R/W)

**RTCIO\_XTAL\_X32N\_RDE** 1: Pull-down on pad enabled; 0: Pull-down disabled. (R/W)

**RTCIO\_XTAL\_X32N\_RUE** 1: Pull-up on pad enabled; 0: Pull-up disabled. (R/W)

**RTCIO\_XTAL\_X32P\_DRV** Select the drive strength of the pad. (R/W)

**RTCIO\_XTAL\_X32P\_HOLD** Set to 1 to hold the output value on the pad, 0 for normal operation. (R/W)

**RTCIO\_XTAL\_X32P\_RDE** 1: Pull-down on pad enabled; 0: Pull-down disabled. (R/W)

**RTCIO\_XTAL\_X32P\_RUE** 1: Pull-up on pad enabled; 0: Pull-up disabled. (R/W)

**RTCIO\_XTAL\_DAC\_XTAL\_32K** 32K XTAL bias current DAC value. (R/W)

**RTCIO\_XTAL\_XPD\_XTAL\_32K** Power up 32 KHz crystal oscillator. (R/W)

**RTCIO\_XTAL\_X32N\_MUX\_SEL** 1: route X32N pad to the digital IO\_MUX, 0: route to RTC block. (R/W)

**RTCIO\_XTAL\_X32P\_MUX\_SEL** 1: route X32P pad to the digital IO\_MUX, 0: route to RTC block. (R/W)

**RTCIO\_XTAL\_X32N\_FUN\_SEL** Select the RTC function. 0: select function0; 1: select function1. (R/W)

**RTCIO\_XTAL\_X32N\_SLP\_SEL** Sleep mode selection. Set this bit to 1 to put the pad to sleep. (R/W)

**RTCIO\_XTAL\_X32N\_SLP\_IE** Input enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_X32N\_SLP\_OE** Output enable of the pad. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_X32N\_FUN\_IE** Input enable of the pad. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_X32P\_FUN\_SEL** Select the RTC function. 0: select Function 0; 1: select Function 1. (R/W)

**RTCIO\_XTAL\_X32P\_SLP\_SEL** Sleep mode selection. Set this bit to 1 to put the pad to sleep. (R/W)

**RTCIO\_XTAL\_X32P\_SLP\_IE** Input enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_X32P\_SLP\_OE** Output enable of the pad in sleep mode. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_X32P\_FUN\_IE** Input enable of the pad. 1: enabled; 0: disabled. (R/W)

**RTCIO\_XTAL\_DRES\_XTAL\_32K** 32K XTAL resistor bias control. (R/W)

**RTCIO\_XTAL\_DBIAS\_XTAL\_32K** 32K XTAL self-bias reference control. (R/W)



Register 4.52: RTCIO\_TOUCH\_CFG\_REG (0x0024)

RTCIO_TOUCH_XPD_BIAS			RTCIO_TOUCH_DREFH			RTCIO_TOUCH_DREFL			RTCIO_TOUCH_DRANGE			RTCIO_TOUCH_DCUR			(reserved)																					
31	30	29	28	27	26	25	24	23	45																											23
0	1	1	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	0	0	0	Reset			

**RTCIO\_TOUCH\_XPD\_BIAS** Touch sensor bias power on bit. 1: power on; 0: disabled. (R/W)

**RTCIO\_TOUCH\_DREFH** Touch sensor saw wave top voltage. (R/W)

**RTCIO\_TOUCH\_DREFL** Touch sensor saw wave bottom voltage. (R/W)

**RTCIO\_TOUCH\_DRANGE** Touch sensor saw wave voltage range. (R/W)

**RTCIO\_TOUCH\_DCUR** Touch sensor bias current. When BIAS\_SLEEP is enabled, this setting is available. (R/W)

Register 4.53: RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_REG (*n*: 0-9) (0x25+1\**n*)

(reserved)										RTCIO_TOUCH_PAD <sub>n</sub> _DAC										RTCIO_TOUCH_PAD <sub>n</sub> _START										RTCIO_TOUCH_PAD <sub>n</sub> _TIE_OPT										RTCIO_TOUCH_PAD <sub>n</sub> _XPD										RTCIO_TOUCH_PAD <sub>n</sub> _TO_GPIO										(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
31						26						25	23						22	21	20	19	37																																					19																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
0						0						0						0						0x4	0	0	0	0	0																																					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0</

**RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_DAC** Touch sensor slope control. 3-bit for each touch pad, defaults to 100. (R/W)

**RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_START** Start touch sensor. (R/W)

**RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_TIE\_OPT** Default touch sensor tie option. 0: tie low; 1: tie high. (R/W)

**RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_XPD** Touch sensor power on. (R/W)

**RTCIO\_TOUCH\_PAD<sub>*n*</sub>\_TO\_GPIO** Connect the RTC pad input to digital pad input, 0 is available. (R/W)

**Register 4.54: RTCIO\_EXT\_WAKEUP0\_REG (0x002f)**

RTCIO_EXT_WAKEUP0_SEL																											(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
31	27				53																									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

Reset

**RTCIO\_EXT\_WAKEUP0\_SEL** GPIO[0-17] can be used to wake up the chip when the chip is in sleep mode. This register selects the pad source to wake up the chip in deep/light sleep mode. 0: select GPIO0; 1: select GPIO2, etc. (R/W)

**Register 4.55: RTCIO\_XTL\_EXT\_CTR\_REG (0x0030)**

RTCIO_XTL_EXT_CTR_SEL																(reserved)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
31	27				53																									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

Reset

**RTCIO\_XTL\_EXT\_CTR\_SEL** Select the external crystal power down enable source in sleep mode. 0: select GPIO0; 1: select GPIO2, etc. The input value on this pin XOR RTCIO\_RTC\_EXT\_XTAL\_CONF\_REG[30] is the crystal power down enable signal. (R/W)

**Register 4.56: RTCIO\_SAR\_I2C\_IO\_REG (0x0031)**

RTCIO_SAR_I2C_SDA_SEL																RTCIO_SAR_I2C_SCL_SEL															
31	30	29	28	55																											28
0	0	0	0	0	0	0	0	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

**RTCIO\_SAR\_I2C\_SDA\_SEL** Selects a different pad as the RTC I2C SDA signal. 0: use pad TOUCH\_PAD[1]; 1: use pad TOUCH\_PAD[3]. (R/W)

**RTCIO\_SAR\_I2C\_SCL\_SEL** Selects a different pad as the RTC I2C SCL signal. 0: use pad TOUCH\_PAD[1]; 1: use pad TOUCH\_PAD[3]. (R/W)

## 5. I2C Controller

### 5.1 Overview

An I2C (Inter-Integrated Circuit) bus can be used for communication with several external devices connected to the same bus as ESP32. The ESP32 has dedicated hardware to communicate with peripherals on the I2C bus.

### 5.2 Features

The I2C controller has the following features:

- Supports both master mode and slave mode
- Supports multi-master and multi-slave communication
- Supports standard mode (100 kbit/s)
- Supports fast mode (400 kbit/s)
- Supports 7-bit addressing and 10-bit addressing
- Supports continuous data transmission with disabled Serial Clock Line (SCL)
- Supports programmable digital noise filter

### 5.3 Functional Description

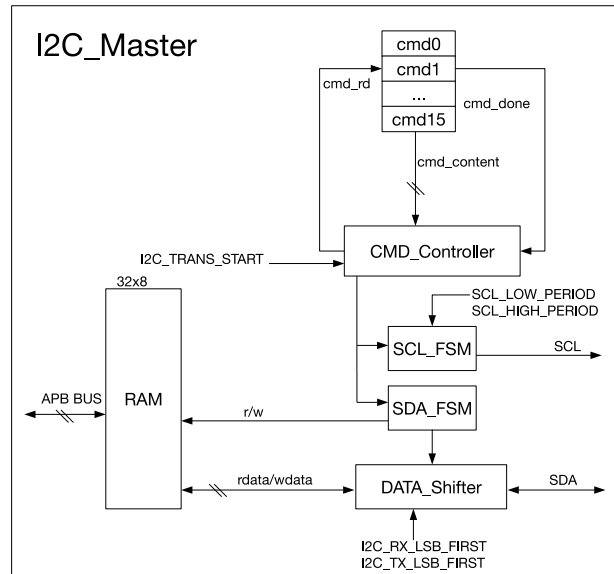
#### 5.3.1 Introduction

I2C is a two-wire bus, consisting of an SDA and an SCL line. These lines are configured to open drain output. The lines are shared by two or more devices, usually one or more masters and one or more slaves.

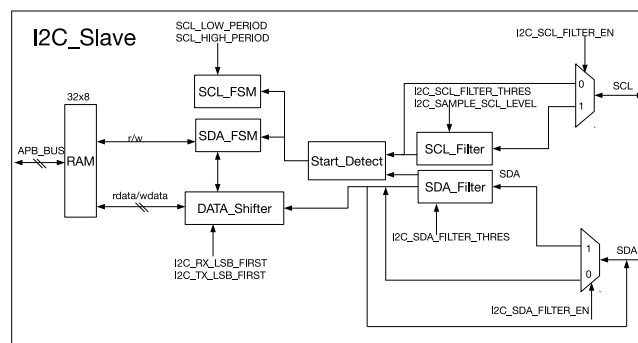
Communication starts when a master sends out a start condition: it will pull the SDA line low followed by pulling the SCL line high. It will send out nine clock pulses over the SCL line. The first eight pulses are used to shift out a byte, consisting of a 7-bit address and a read/write bit. If a slave with this address is active on the bus, the slave can answer by pulling the SDA low on the 9th clock. The master can now send out more 9-bit clock pulse clusters, and depending on the read/write bit sent, either the device or the master will shift out data on the SDA line, with the other side acknowledging the transfer by pulling SDA low on the 9th clock pulse. During data transfer, the SDA line only changes when the SCL line is low. When the master has finished the communication, it will send a stop condition on the bus by raising SDA while SCL already is high.

The ESP32 I2C peripheral can handle the I2C protocol, freeing up the processor cores for other tasks.

### 5.3.2 Architecture



### Figure 10: I2C Master Architecture



### Figure 11: I2C Slave Architecture

An I2C controller can operate either in master mode or slave mode. The I2C\_MS\_MODE register is used to select the mode. Figure 10 shows the I2C Master architecture, Figure 11 shows the I2C Slave architecture. The I2C controller contains the following units:

- **RAM:** The RAM size is 32 x 8 bit and it is directly mapped into the address space of the CPU cores, starting at address REG\_I2C\_BASE+0x100. Each byte of I2C data is stored in a 32-bit word of memory (so the first byte is at +0x100, the second byte at +0x104, the third byte at +0x108, etc.)
- **16 command registers (cmd0 ~ cmd15) and one CMD\_Controller:** They are used by I2C Master to control data transmission. Commands are executed by the I2C controller one at a time.
- **SCL\_FSM:** A state machine that controls the SCL clock. The I2C\_SCL\_HIGH\_PERIOD\_REG and I2C\_SCL\_LOW\_PERIOD\_REG registers are used to configure the frequency and duty cycle of the signal on the SCL line.
- **SDA\_FSM:** A state machine that controls the SDA data line.
- **DATA\_Shifter:** Converts the byte data to an outgoing bitstream, or converts an incoming bitstream to byte data. I2C\_RX\_LSB\_FIRST and I2C\_TX\_LSB\_FIRST can be used for configuring whether the LSB or MSB is stored or transmitted first.

- SCL\_Filter and SDA\_Filter: Input noise filter for the I2C\_Slave. The filter can be enabled or disabled by configuring I2C\_SCL\_FILTER\_EN and I2C\_SDA\_FILTER\_EN. The filter can remove line glitches with pulse width less than I2C\_SCL\_FILTER\_THRES and I2C\_SDA\_FILTER\_THRES ABP clock cycles.

### 5.3.3 I2C Bus Timing

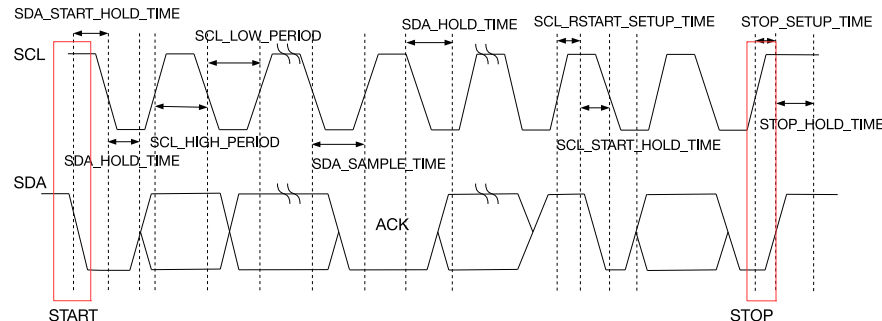


Figure 12: I2C Sequence Chart

Figure 12 is an I2C sequence chart. When the I2C controller works in master mode, SCL is an output signal. In contrast, when the I2C controller works in slave mode, SCL is an input signal.

According to the I2C protocol, each transmission of data begins with a START condition and ends with a STOP condition. Data is transmitted one byte at a time and each byte has an ACK bit. The receiver informs the transmitter to continue transmission by pulling down SDA, which indicates an ACK. The receiver can also indicate it wants to stop the transmission by not pulling down the SDA line, thereby not giving an ACK.

Figure 12 also shows the registers that can configure the START bit, STOP bit, SDA hold time and SDA sample time.

### 5.3.4 I2C cmd Structure

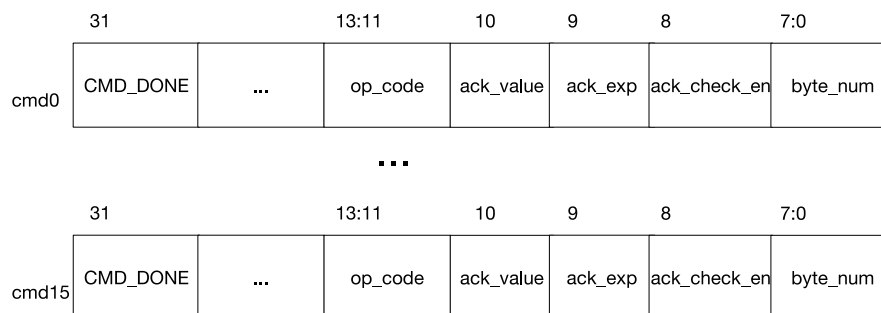


Figure 13: Structure of The I2C Command Register

The Command register is only active in I2C master mode with its internal structure shown in Figure 13.

**CMD\_DONE:** The CMD\_DONE bit of every command can be read by software to tell if the command has been handled by hardware.

**op\_code:** op\_code is used to indicate the command. The I2C controller supports four commands:

- RSTART: op\_code = 0 is the RSTART command to control the transmission of a START or RESTART I2C condition.
- WRITE: op\_code = 1 is the WRITE command which means the I2C Master will transmit data.

- READ: op\_code = 2 is the READ command which means the I2C Master will receive data.
- STOP: op\_code = 3 is the STOP command to control the transmission of a STOP I2C condition.
- END: op\_code = 4 is the END command for continuous data transmission. When the END command occurs, SCL is temporarily disabled to allow software to reload the command and data registers for subsequent events before resuming. Transmission will seamlessly continue.

A complete data transmission process begins with a RSTART command and ends with a STOP command.

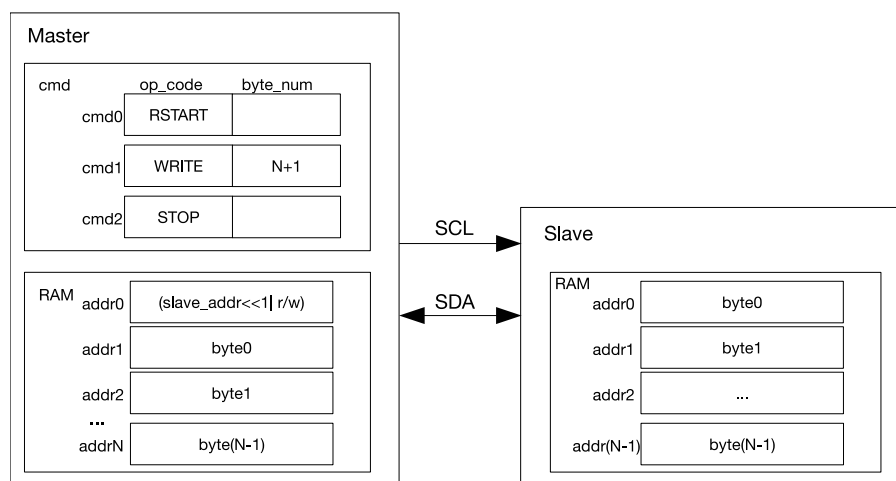
ack\_value: When receiving data, this bit is used to indicate if the receiver will send an ACK after this byte has been received.

ack\_exp: This bit is to set an expected ACK value for the transmitter.

ack\_check\_en: When transmitting a byte, this bit enables checking the ACK value received against the ack\_exp value. 1 enables checking, 0 disables it.

byte\_num: This register specifies the length of data to be read or written. When the op\_code is RSTART, STOP or END, this value has no meaning.

### 5.3.5 I2C Master Writes to Slave

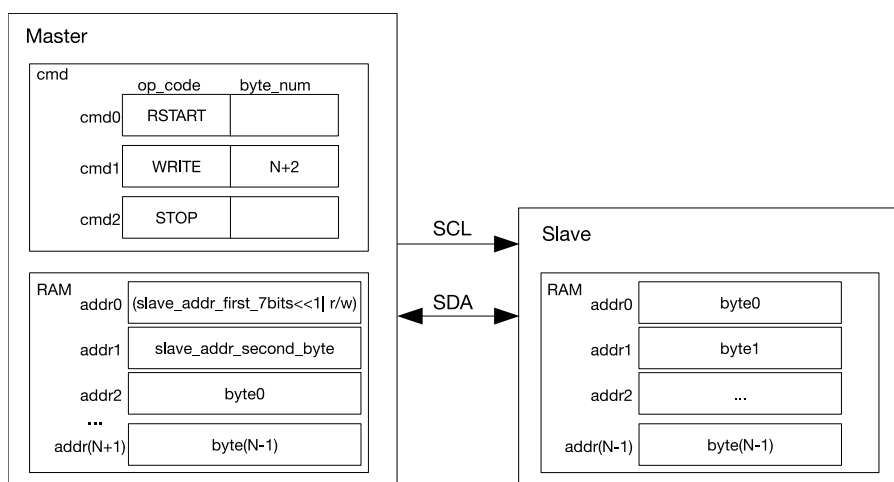


**Figure 14: I2C Master Writes to Slave with 7-bit Address**

Figure 14 shows the I2C Master writing N bytes of data to an external I2C Slave; both are assumed to be ESP32 I2C controllers. According to the I2C protocol, the first byte is the Slave address. As shown in the diagram, the first byte of the RAM unit has been populated with the Slave's 7-bit address plus the 1-bit read/write flag. In this case, the flag is zero, indicating a write operation. The rest of the RAM unit stores N bytes of data that is ready for transmission. The cmd unit has been populated with the sequence of commands for the operation.

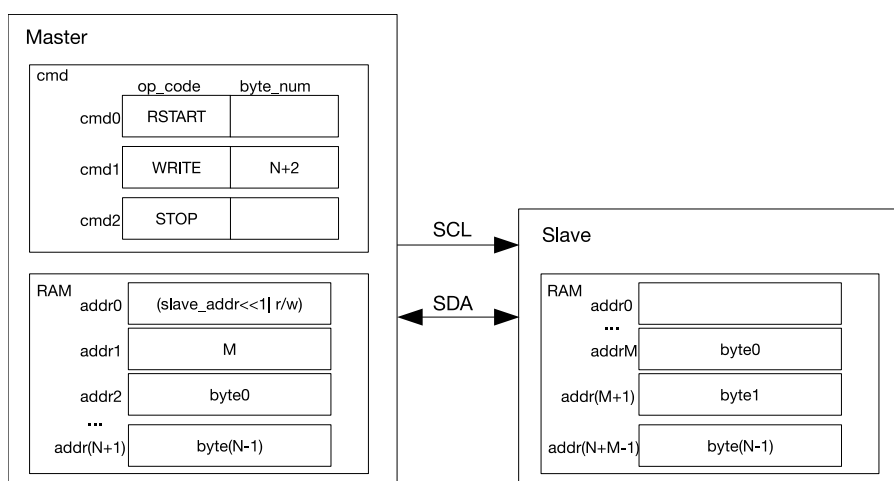
The FIFO offset in RAM can be configured via the TXFIFO\_START\_ADDR field in the RXFIFO\_ST\_REG register.

When all registers are ready, the I2C\_TRANS\_START bit in I2C\_CTR\_REG is set to start transmission. The I2C Master then initiates a START condition to activate the slave devices. I2C Master will then progress to the WRITE command, which will cause N+1 bytes to be fetched from RAM and sent to the Slave. The first of these bytes is the address byte. Each slave device will compare this to its own. If the address does not match, the slave will ignore the rest of the transmission. If it does match, the slave will ACK the initial byte and the I2C master will continue to send the rest of the data; when ack\_check\_en is set to 1, Master will check ACK value.



**Figure 15: I2C Master Writes to Slave with 10-bit Address**

The I2C controller uses 7-bit addressing by default. 10-bit addressing can also be used. In the master, this is done by sending a second I2C address byte after the first address byte. In the slave, the I2C\_SLAVE\_ADDR\_10BIT\_EN register bit can be set to activate 10-bit addressing. I2C\_SLAVE\_ADDR is used to configure I2C Slave's address, as usual. Figure 15 shows the equivalent I2C Master operation writing N bytes of data to an I2C Slave with a 10-bit address. Since 10-bit Slave addresses require one more address byte, both the byte\_num field of the WRITE command and the number of total bytes in RAM increase by 1.



**Figure 16: I2C Master Writes to addrM in RAM of Slave with 7-bit Address**

One way many I2C Slave devices are designed is by exposing a register block containing various settings. The I2C Master can write one or more of these registers by sending the Slave a register address. The ESP32 I2C Slave controller has hardware support for such a scheme.

Specifically, on the Slave, I2C\_FIFO\_ADDR\_CFG\_EN can be set so that the I2C Master can write to a specified register address inside the I2C Slave memory block. Figure 16 shows the I2C Master writing N bytes of data byte0 ~ byte(N-1) from the RAM unit to register address M (determined by addrM in RAM unit) in the Slave.

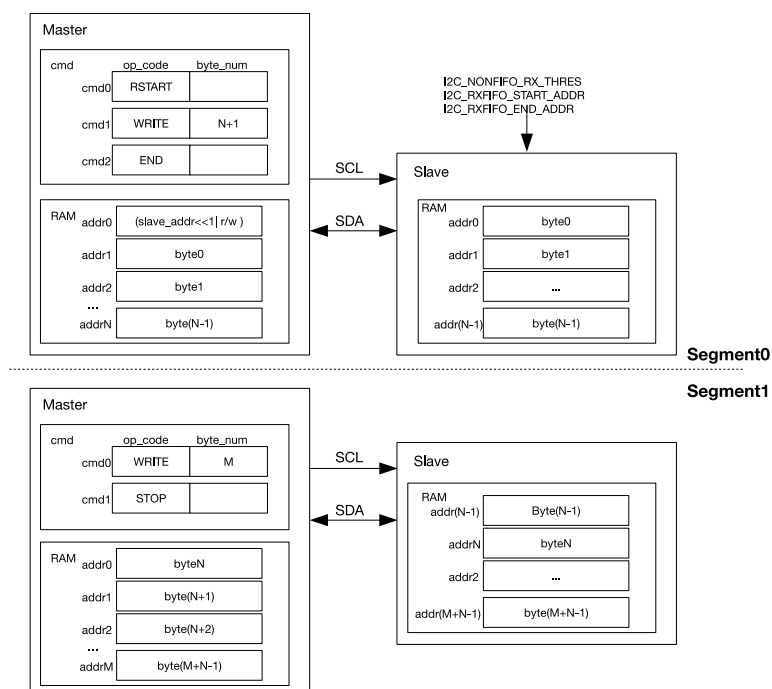


Figure 17: I2C Master Writes to Slave with 7-bit Address in Two Segments

If the data size exceeds the RAM unit capacity of 32 bytes, the END command can be called to enable segmented transmission. Figure 17 shows I2C Master writing data in two segments to Slave. The upper part of the figure shows the configuration of the first sequence of bytes in the transfer. I2C Master will turn off SCL clock after executing END command and the controller generates the I2C\_END\_DETECT\_INT interrupt.

On receiving I2C\_END\_DETECT\_INT (or polling the CMD\_DONE bit of the command register the END was placed into), software should refresh the contents of the cmd and RAM units as shown in the lower part of the figure. Subsequently, it should clear the I2C\_END\_DETECT\_INT interrupt and resume the transaction by setting the I2C\_TRANS\_START bit in CTR\_CTR\_REG.

### 5.3.6 I2C Master Reads from Slave

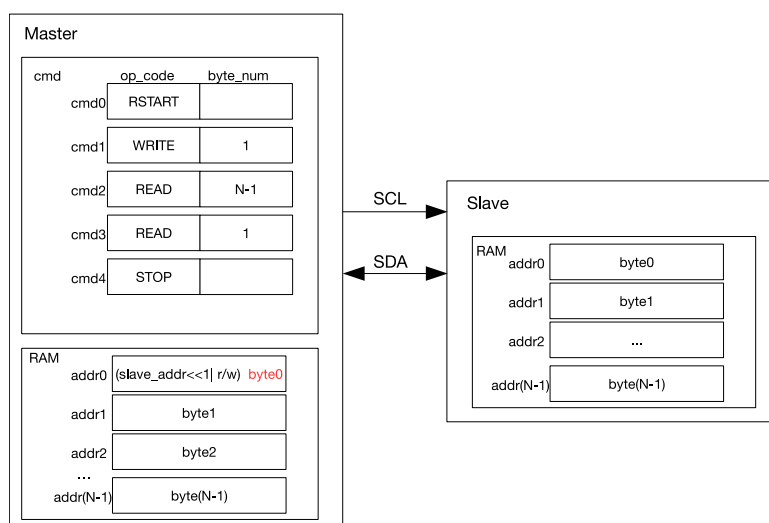


Figure 18: I2C Master Reads from Slave with 7-bit Address

Figure 18 shows the I2C Master reading N bytes of data from an I2C Slave with a 7-bit address. At first, the I2C Master needs to send the address of the I2C Slave, so cmd1 is a WRITE command. The byte that this command



sends is the I2C slave address plus the R/W flag, which in this case is 1 to indicate this is going to be a read operation. According to the I2C protocol, I2C Master will not return ACK on receiving the last byte of data read from the slave, consequently READ is divided into two segments. The I2C Master replies ACK to N-1 bytes in cmd2 and does not reply ACK to the single byte READ command in cmd3, i.e., the last transmitted data.

When storing the received data, I2C Master will start from the first address in RAM. Byte0 (Slave address + 1-bit R/W marker bit) will be overwritten. The FIFO RAM offsets for reading and writing data can be configured via the RXFIFO\_START\_ADDR and TXFIFO\_START\_ADDR fields in the RXFIFO\_ST\_REG register.

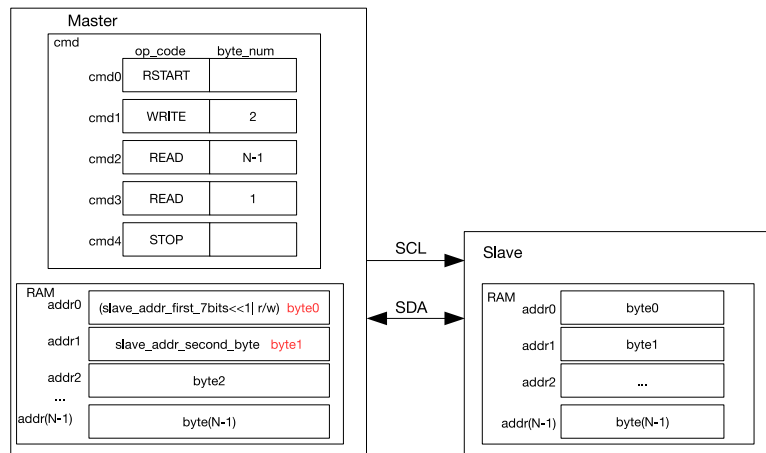


Figure 19: I2C Master Reads from Slave with 10-bit Address

Figure 19 shows the I2C Master reading data from a slave with a 10-bit address. In the Slave, this mode is enabled by setting I2C\_SLAVE\_ADDR\_10BIT\_EN register. In the Master, two bytes of RAM are used for a 10-bit address.

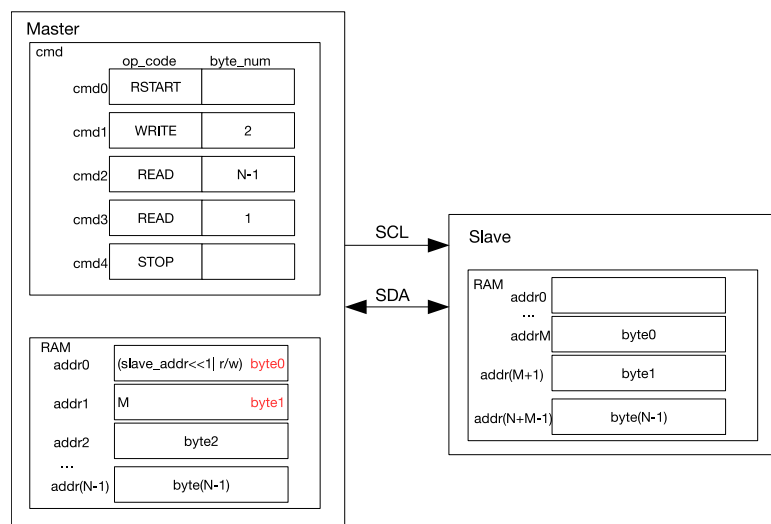
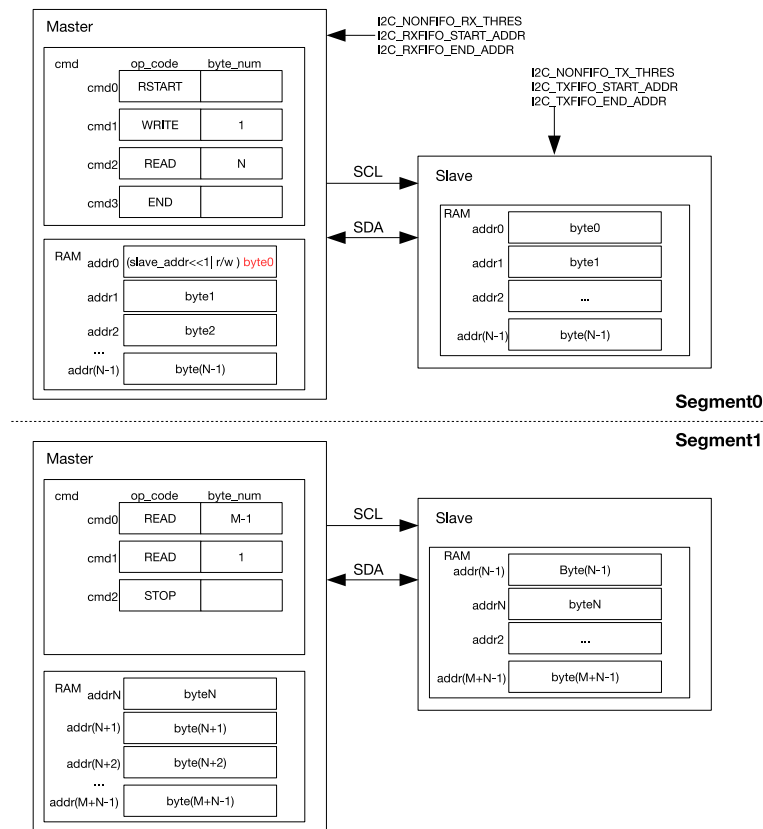


Figure 20: I2C Master Reads N Bytes of Data from addrM in Slave with 7-bit Address

Figure 20 shows the I2C Master selecting a register address inside the I2C Slave device and then reading data from it and subsequent addresses. This mode is enabled by setting the I2C\_FIFO\_ADDR\_CFG\_EN register in the Slave. The internal register address of the Slave, M, is stored in the RAM byte following the address. The WRITE command has a length of two data bytes to compensate.



**Figure 21: I2C Master Reads from Slave with 7-bit Address in Two Segments**

Figure 21 shows the I2C Master reading N+M bytes of data in two segments from I2C Slave by using the END command. This allows more data to be read than fits into RAM. The upper part of the figure shows the configuration of Segment0. The Master will update the configuration of cmd after executing the END command, as shown in the lower part of the figure. I2C Slave will refresh the data before its RAM is empty.

### 5.3.7 Interrupts

- I2C\_TX\_SEND\_EMPTY\_INT: Triggered when I2C sends more data than nonfifo\_tx\_thres.
- I2C\_RX\_REC\_FULL\_INT: Triggered when I2C receives more data than nonfifo\_rx\_thres.
- I2C\_ACK\_ERR\_INT: Triggered when I2C receives a wrong ACK bit..
- I2C\_TRANS\_START\_INT: Triggered when I2C sends the START bit.
- I2C\_TIME\_OUT\_INT: Triggered when I2C takes too long to receive data.
- I2C\_TRANS\_COMPLETE\_INT: Triggered when I2C Master has finished STOP command.
- I2C\_MASTER\_TRAN\_COMP\_INT: Triggered when I2C Master sends or receives a byte.
- I2C\_ARBITRATION\_LOST\_INT: Triggered when I2C Master has lost the usage right of I2C BUS.
- I2C\_SLAVE\_TRAN\_COMP\_INT: Triggered when I2C Slave detects the STOP bit.
- I2C\_END\_DETECT\_INT: Triggered when I2C deals with the END command.

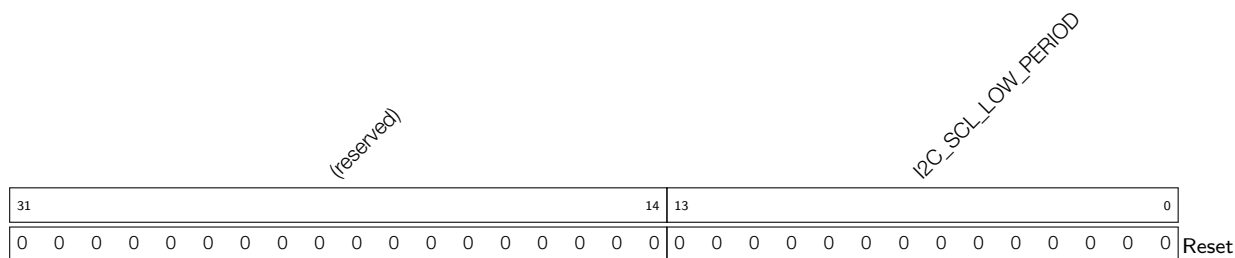
## 5.4 Register summary

Name	Description	I2C0	I2C1	Acc
<b>Configuration registers</b>				
<a href="#">I2C_SLAVE_ADDR_REG</a>	Configures the I2C slave address	0x3FF53010	0x3FF67010	R/W
<a href="#">I2C_RXFIFO_ST_REG</a>	FIFO status register	0x3FF53014	0x3FF67014	RO
<a href="#">I2C_FIFO_CONF_REG</a>	FIFO configuration register	0x3FF53018	0x3FF67018	R/W
<b>Timing registers</b>				
<a href="#">I2C_SDA_HOLD_REG</a>	Configures the hold time after a negative SCL edge	0x3FF53030	0x3FF67030	R/W
<a href="#">I2C_SDA_SAMPLE_REG</a>	Configures the sample time after a positive SCL edge	0x3FF53034	0x3FF67034	R/W
<a href="#">I2C_SCL_LOW_PERIOD_REG</a>	Configures the low level width of the SCL clock	0x3FF53000	0x3FF67000	R/W
<a href="#">I2C_SCL_HIGH_PERIOD_REG</a>	Configures the high level width of the SCL clock	0x3FF53038	0x3FF67038	R/W
<a href="#">I2C_SCL_START_HOLD_REG</a>	Configures the delay between the SDA and SCL negative edge for a start condition	0x3FF53040	0x3FF67040	R/W
<a href="#">I2C_SCL_RSTART_SETUP_REG</a>	Configures the delay between the positive edge of SCL and the negative edge of SDA	0x3FF53044	0x3FF67044	R/W
<a href="#">I2C_SCL_STOP_HOLD_REG</a>	Configures the delay after the SCL clock edge for a stop condition	0x3FF53048	0x3FF67048	R/W
<a href="#">I2C_SCL_STOP_SETUP_REG</a>	Configures the delay between the SDA and SCL positive edge for a stop condition	0x3FF5304C	0x3FF6704C	R/W
<b>Filter registers</b>				
<a href="#">I2C_SCL_FILTER_CFG_REG</a>	SCL filter configuration register	0x3FF53050	0x3FF67050	R/W
<a href="#">I2C_SDA_FILTER_CFG_REG</a>	SDA filter configuration register	0x3FF53054	0x3FF67054	R/W
<b>Interrupt registers</b>				
<a href="#">I2C_INT_RAW_REG</a>	Raw interrupt status	0x3FF53020	0x3FF67020	RO
<a href="#">I2C_INT_ENA_REG</a>	Interrupt enable bits	0x3FF53028	0x3FF67028	R/W
<a href="#">I2C_INT_CLR_REG</a>	Interrupt clear bits	0x3FF53024	0x3FF67024	WO
<b>Command registers</b>				
<a href="#">I2C_COMD0_REG</a>	I2C command register 0	0x3FF53058	0x3FF67058	R/W
<a href="#">I2C_COMD1_REG</a>	I2C command register 1	0x3FF5305C	0x3FF6705C	R/W
<a href="#">I2C_COMD2_REG</a>	I2C command register 2	0x3FF53060	0x3FF67060	R/W
<a href="#">I2C_COMD3_REG</a>	I2C command register 3	0x3FF53064	0x3FF67064	R/W
<a href="#">I2C_COMD4_REG</a>	I2C command register 4	0x3FF53068	0x3FF67068	R/W
<a href="#">I2C_COMD5_REG</a>	I2C command register 5	0x3FF5306C	0x3FF6706C	R/W
<a href="#">I2C_COMD6_REG</a>	I2C command register 6	0x3FF53070	0x3FF67070	R/W
<a href="#">I2C_COMD7_REG</a>	I2C command register 7	0x3FF53074	0x3FF67074	R/W
<a href="#">I2C_COMD8_REG</a>	I2C command register 8	0x3FF53078	0x3FF67078	R/W
<a href="#">I2C_COMD9_REG</a>	I2C command register 9	0x3FF5307C	0x3FF6707C	R/W
<a href="#">I2C_COMD10_REG</a>	I2C command register 10	0x3FF53080	0x3FF67080	R/W
<a href="#">I2C_COMD11_REG</a>	I2C command register 11	0x3FF53084	0x3FF67084	R/W
<a href="#">I2C_COMD12_REG</a>	I2C command register 12	0x3FF53088	0x3FF67088	R/W

Name	Description	I2C0	I2C1	Acc
<a href="#">I2C_COMD13_REG</a>	I2C command register 13	0x3FF5308C	0x3FF6708C	R/W
<a href="#">I2C_COMD14_REG</a>	I2C command register 14	0x3FF53090	0x3FF67090	R/W
<a href="#">I2C_COMD15_REG</a>	I2C command register 15	0x3FF53094	0x3FF67094	R/W

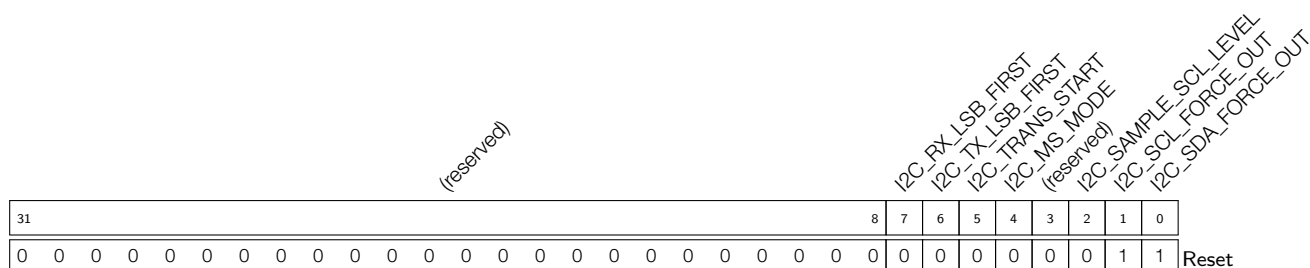
## 5.5 Registers

**Register 5.1: I2C\_SCL\_LOW\_PERIOD\_REG (0x0000)**



**I2C\_SCL\_LOW\_PERIOD** This register is used to configure the low level width of the SCL clock signal, in APB clock cycles. (R/W)

**Register 5.2: I2C\_CTR\_REG (0x0004)**



**I2C\_RX\_LSB\_FIRST** This bit is used to control the storage mode for received data. (R/W)

- 1: receive data from the most significant bit;
- 0: receive data from the least significant bit.

**I2C\_TX\_LSB\_FIRST** This bit is used to control the sending mode for data needing to be sent. (R/W)

- 1: send data from the least significant bit;
- 0: send data from the most significant bit.

**I2C\_TRANS\_START** Set this bit to start sending the data in txfifo. (R/W)

**I2C\_MS\_MODE** Set this bit to configure the module as an I2C Master. Clear this bit to configure the module as an I2C Slave. (R/W)

**I2C\_SAMPLE\_SCL\_LEVEL** 1: sample SDA data on the SCL low level; 0: sample SDA data on the SCL high level. (R/W)

**I2C\_SCL\_FORCE\_OUT** 1: direct output; 0: open drain output. (R/W)

**I2C\_SDA\_FORCE\_OUT** 1: direct output; 0: open drain output. (R/W)

### Register 5.3: I2C\_SR\_REG (0x0008)

[illegible]

**I2C\_TXFIFO\_CNT** This field stores the amount of received data in RAM. (RO)

**I2C\_RXFIFO\_CNT** This field represents the amount of data needing to be sent. (RO)

**I2C\_BYTE\_TRANS** This field changes to 1 when one byte is transferred. (RO)

**I2C\_SLAVE\_ADDRESSED** When configured as an I2C Slave, and the address sent by master is equal to slave's address, then this bit will be high level. (RO)

**I2C\_BUS\_BUSY** 1: The I2C bus is busy transferring data; 0: I2C bus is in idle state. (RO)

**I2C\_ARB\_LOST** When the I2C controller loses control of SCL line, this register changes to 1. (RO)

**I2C\_TIME\_OUT** When the I2C controller takes more than I2C\_TIME\_OUT clocks to receive a data bit, this field changes to 1. (RO)

**I2C\_SLAVE\_RW** When in slave mode, 1: master reads from slave; 0: master writes to slave. (RO)

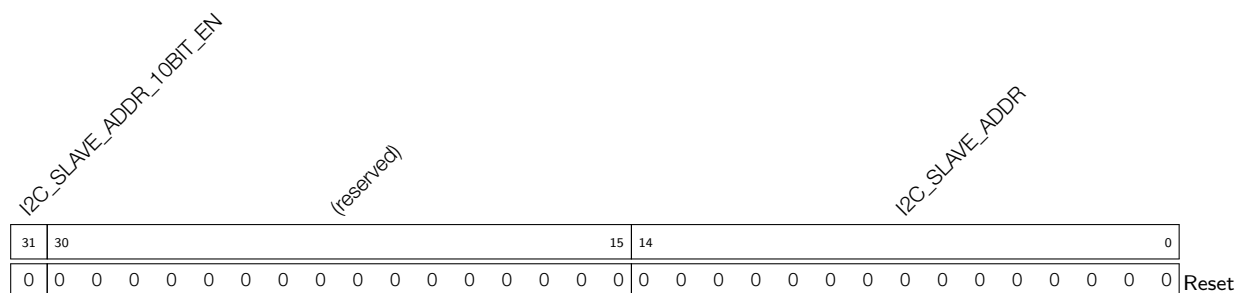
**I2C\_ACK\_REC** This register stores the value of the received ACK bit. (RO)

### Register 5.4: I2C\_TO\_REG (0x000c)

Diagram illustrating the structure of the I2C\_TIME\_OUT register:

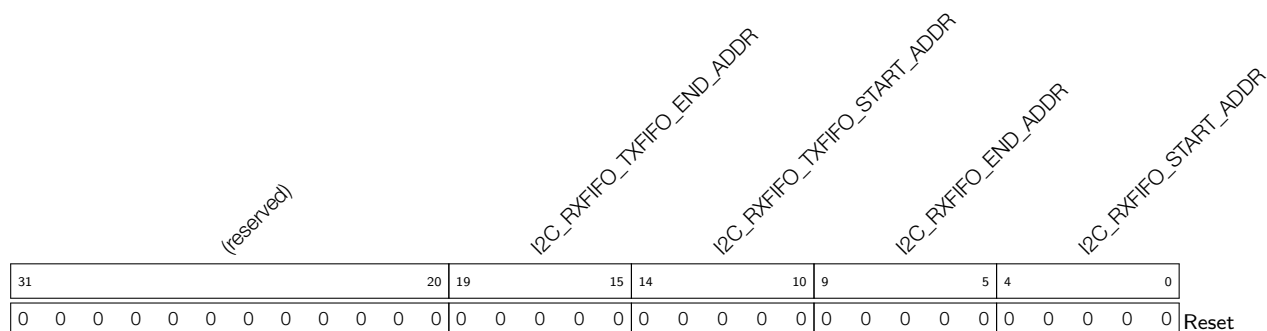
- Bits 31 to 20: (reserved)
- Bits 19 to 0: I2C\_TIME\_OUT
- Reset

**I2C\_TIME\_OUT** This register is used to configure the timeout, for receiving a data bit, in APB clock cycles. (R/W)

**Register 5.5: I2C\_SLAVE\_ADDR\_REG (0x0010)**

**I2C\_SLAVE\_ADDR\_10BIT\_EN** This field is used to enable the slave 10-bit addressing mode. (R/W)

**I2C\_SLAVE\_ADDR** When configured as an I2C Slave, this field is used to configure the slave address. (R/W)

**Register 5.6: I2C\_RXFIFO\_ST\_REG (0x0014)**

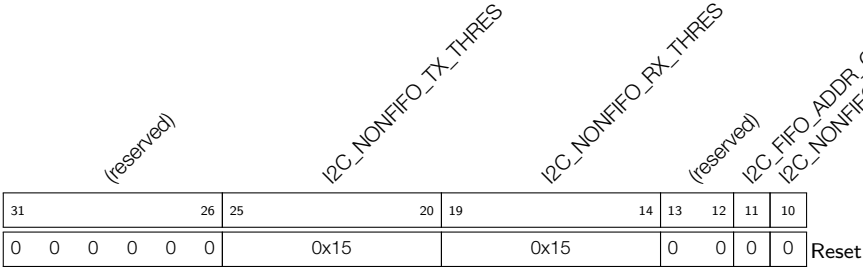
**I2C\_TXFIFO\_END\_ADDR** This is the offset address of the last sent data as described in non-fifo\_tx\_thres register. (RO)

**I2C\_TXFIFO\_START\_ADDR** This is the offset address of the first sent data as described in non-fifo\_tx\_thres register. (RO)

**I2C\_RXFIFO\_END\_ADDR** This is the offset address of the first received data as described in non-fifo\_rx\_thres\_register. (RO)

**I2C\_RXFIFO\_START\_ADDR** This is the offset address of the last received data as described in non-fifo\_rx\_thres\_register. (RO)

Register 5.7: I2C\_FIFO\_CONF\_REG (0x0018)



**I2C\_NONFIFO\_TX\_THRES** When I2C sends more than nonfifo\_tx\_thres bytes of data, it will generate a tx\_send\_empty\_int\_raw interrupt and update the current offset address of the sent data. (R/W)

**I2C\_NONFIFO\_RX\_THRES** When I2C receives more than nonfifo\_rx\_thres bytes of data, it will generate a rx\_send\_full\_int\_raw interrupt and update the current offset address of the received data. (R/W)

**I2C\_FIFO\_ADDR\_CFG\_EN** When this bit is set to 1, the byte received after the I2C address byte represents the offset address in the I2C Slave RAM. (R/W)

**I2C\_NONFIFO\_EN** Set this bit to enable APB nonfifo access. (R/W)



Register 5.8: I2C\_INT\_RAW\_REG (0x0020)

(reserved)																								I2C_TX_SEND_EMPTY_INT_RAW I2C_RX_REC_FULL_INT_RAW I2C_ACK_ERR_INT_RAW I2C_TRANS_START_INT_RAW I2C_TIME_OUT_INT_RAW I2C_TRANS_COMPLETE_INT_RAW I2C_MASTER_TRAN_COMP_INT_RAW I2C_SLAVE_TRAN_COMP_INT_RAW I2C_END_DETECT_INT_RAW											
31													13											12	11	10	9	8	7	6	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	0	0	0	0	0	0	0	0	Reset				

**I2C\_TX\_SEND\_EMPTY\_INT\_RAW** The raw interrupt status bit for the [I2C\\_TX\\_SEND\\_EMPTY\\_INT](#) interrupt. (RO)

**I2C\_RX\_REC\_FULL\_INT\_RAW** The raw interrupt status bit for the [I2C\\_RX\\_REC\\_FULL\\_INT](#) interrupt. (RO)

**I2C\_ACK\_ERR\_INT\_RAW** The raw interrupt status bit for the [I2C\\_ACK\\_ERR\\_INT](#) interrupt. (RO)

**I2C\_TRANS\_START\_INT\_RAW** The raw interrupt status bit for the [I2C\\_TRANS\\_START\\_INT](#) interrupt. (RO)

**I2C\_TIME\_OUT\_INT\_RAW** The raw interrupt status bit for the [I2C\\_TIME\\_OUT\\_INT](#) interrupt. (RO)

**I2C\_TRANS\_COMPLETE\_INT\_RAW** The raw interrupt status bit for the [I2C\\_TRANS\\_COMPLETE\\_INT](#) interrupt. (RO)

**I2C\_MASTER\_TRAN\_COMP\_INT\_RAW** The raw interrupt status bit for the [I2C\\_MASTER\\_TRAN\\_COMP\\_INT](#) interrupt. (RO)

**I2C\_ARBITRATION\_LOST\_INT\_RAW** The raw interrupt status bit for the [I2C\\_ARBITRATION\\_LOST\\_INT](#) interrupt. (RO)

**I2C\_SLAVE\_TRAN\_COMP\_INT\_RAW** The raw interrupt status bit for the [I2C\\_SLAVE\\_TRAN\\_COMP\\_INT](#) interrupt. (RO)

**I2C\_END\_DETECT\_INT\_RAW** The raw interrupt status bit for the [I2C\\_END\\_DETECT\\_INT](#) interrupt. (RO)

Register 5.9: I2C\_INT\_CLR\_REG (0x0024)

(reserved)																								I2C_TX_SEND_EMPTY_INT_CLR I2C_RX_REC_FULL_INT_CLR I2C_ACK_ERR_INT_CLR I2C_TRANS_START_INT_CLR I2C_TIME_OUT_INT_CLR I2C_TRANS_COMPLETE_INT_CLR I2C_MASTER_TRAN_COMP_INT_CLR I2C_SLAVE_TRAN_COMP_INT_CLR I2C_END_DETECT_INT_CLR																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																							
31																								13		12	11	10	9	8	7	6	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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\_TX\_SEND\_EMPTY\_INT\_CLR** Set this bit to clear the [I2C\\_TX\\_SEND\\_EMPTY\\_INT](#) interrupt. (WO)

**I2C\_RX\_REC\_FULL\_INT\_CLR** Set this bit to clear the [I2C\\_RX\\_REC\\_FULL\\_INT](#) interrupt. (WO)

**I2C\_ACK\_ERR\_INT\_CLR** Set this bit to clear the [I2C\\_ACK\\_ERR\\_INT](#) interrupt. (WO)

**I2C\_TRANS\_START\_INT\_CLR** Set this bit to clear the [I2C\\_TRANS\\_START\\_INT](#) interrupt. (WO)

**I2C\_TIME\_OUT\_INT\_CLR** Set this bit to clear the [I2C\\_TIME\\_OUT\\_INT](#) interrupt. (WO)

**I2C\_TRANS\_COMPLETE\_INT\_CLR** Set this bit to clear the [I2C\\_TRANS\\_COMPLETE\\_INT](#) interrupt. (WO)

**I2C\_MASTER\_TRAN\_COMP\_INT\_CLR** Set this bit to clear the [I2C\\_MASTER\\_TRAN\\_COMP\\_INT](#) interrupt. (WO)

**I2C\_ARBITRATION\_LOST\_INT\_CLR** Set this bit to clear the [I2C\\_ARBITRATION\\_LOST\\_INT](#) interrupt. (WO)

**I2C\_SLAVE\_TRAN\_COMP\_INT\_CLR** Set this bit to clear the [I2C\\_SLAVE\\_TRAN\\_COMP\\_INT](#) interrupt. (WO)

**I2C\_END\_DETECT\_INT\_CLR** Set this bit to clear the [I2C\\_END\\_DETECT\\_INT](#) interrupt. (WO)

Register 5.10: I2C\_INT\_ENA\_REG (0x0028)

(reserved)																								I2C_TX_SEND_EMPTY_INT_ENA I2C_RX_REC_FULL_INT_ENA I2C_ACK_ERR_INT_ENA I2C_TRANS_START_INT_ENA I2C_TIME_OUT_INT_ENA I2C_TRANS_COMPLETE_INT_ENA I2C_MASTER_TRAN_COMP_INT_ENA I2C_ARBITRATION_LOST_INT_ENA I2C_SLAVE_TRAN_COMP_INT_ENA I2C_END_DETECT_INT_ENA																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
31																								13								12	11	10	9	8	7	6	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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\_TX\_SEND\_EMPTY\_INT\_ENA** The interrupt enable bit for the [I2C\\_TX\\_SEND\\_EMPTY\\_INT](#) interrupt. (R/W)

**I2C\_RX\_REC\_FULL\_INT\_ENA** The interrupt enable bit for the [I2C\\_RX\\_REC\\_FULL\\_INT](#) interrupt. (R/W)

**I2C\_ACK\_ERR\_INT\_ENA** The interrupt enable bit for the [I2C\\_ACK\\_ERR\\_INT](#) interrupt. (R/W)

**I2C\_TRANS\_START\_INT\_ENA** The interrupt enable bit for the [I2C\\_TRANS\\_START\\_INT](#) interrupt. (R/W)

**I2C\_TIME\_OUT\_INT\_ENA** The interrupt enable bit for the [I2C\\_TIME\\_OUT\\_INT](#) interrupt. (R/W)

**I2C\_TRANS\_COMPLETE\_INT\_ENA** The interrupt enable bit for the [I2C\\_TRANS\\_COMPLETE\\_INT](#) interrupt. (R/W)

**I2C\_MASTER\_TRAN\_COMP\_INT\_ENA** The interrupt enable bit for the [I2C\\_MASTER\\_TRAN\\_COMP\\_INT](#) interrupt. (R/W)

**I2C\_ARBITRATION\_LOST\_INT\_ENA** The interrupt enable bit for the [I2C\\_ARBITRATION\\_LOST\\_INT](#) interrupt. (R/W)

**I2C\_SLAVE\_TRAN\_COMP\_INT\_ENA** The interrupt enable bit for the [I2C\\_SLAVE\\_TRAN\\_COMP\\_INT](#) interrupt. (R/W)

**I2C\_END\_DETECT\_INT\_ENA** The interrupt enable bit for the [I2C\\_END\\_DETECT\\_INT](#) interrupt. (R/W)

Register 5.11: I2C\_INT\_STATUS\_REG (0x002c)

(reserved)																								I2C_TX_SEND_EMPTY_INT_ST I2C_RX_REC_FULL_INT_ST I2C_ACK_ERR_INT_ST I2C_TRANS_START_INT_ST I2C_TIME_OUT_INT_ST I2C_TRANS_COMPLETE_INT_ST I2C_MASTER_TRAN_COMP_INT_ST I2C_ARBITRATION_LOST_INT_ST I2C_SLAVE_TRAN_COMP_INT_ST I2C_END_DETECT_INT_ST												
31																								13	12	11	10	9	8	7	6	5	4	3		
0													0											0	0	0	0	0	0	0	0	0	0	0	0	Reset

**I2C\_TX\_SEND\_EMPTY\_INT\_ST** The masked interrupt status bit for the [I2C\\_TX\\_SEND\\_EMPTY\\_INT](#) interrupt. (RO)

**I2C\_RX\_REC\_FULL\_INT\_ST** The masked interrupt status bit for the [I2C\\_RX\\_REC\\_FULL\\_INT](#) interrupt. (RO)

**I2C\_ACK\_ERR\_INT\_ST** The masked interrupt status bit for the [I2C\\_ACK\\_ERR\\_INT](#) interrupt. (RO)

**I2C\_TRANS\_START\_INT\_ST** The masked interrupt status bit for the [I2C\\_TRANS\\_START\\_INT](#) interrupt. (RO)

**I2C\_TIME\_OUT\_INT\_ST** The masked interrupt status bit for the [I2C\\_TIME\\_OUT\\_INT](#) interrupt. (RO)

**I2C\_TRANS\_COMPLETE\_INT\_ST** The masked interrupt status bit for the [I2C\\_TRANS\\_COMPLETE\\_INT](#) interrupt. (RO)

**I2C\_MASTER\_TRAN\_COMP\_INT\_ST** The masked interrupt status bit for the [I2C\\_MASTER\\_TRAN\\_COMP\\_INT](#) interrupt. (RO)

**I2C\_ARBITRATION\_LOST\_INT\_ST** The masked interrupt status bit for the [I2C\\_ARBITRATION\\_LOST\\_INT](#) interrupt. (RO)

**I2C\_SLAVE\_TRAN\_COMP\_INT\_ST** The masked interrupt status bit for the [I2C\\_SLAVE\\_TRAN\\_COMP\\_INT](#) interrupt. (RO)

**I2C\_END\_DETECT\_INT\_ST** The masked interrupt status bit for the [I2C\\_END\\_DETECT\\_INT](#) interrupt. (RO)

Register 5.12: I2C\_SDA\_HOLD\_REG (0x0030)

(reserved)																						I2C_SDA_HOLD_TIME													
31										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	Reset		

**I2C\_SDA\_HOLD\_TIME** This register is used to configure the time to hold the data after the negative edge of SCL, in APB clock cycles. (R/W)

**Register 5.13: I2C\_SDA\_SAMPLE\_REG (0x0034)**

(reserved)										I2C_SDA_SAMPLE_TIME																			
31										9										0									
0 0										0 0										Reset									

**I2C\_SDA\_SAMPLE\_TIME** This register is used to configure the delay between the positive edge of SCL and the I2C controller sampling SDA, in APB clock cycles. (R/W)

**Register 5.14: I2C\_SCL\_HIGH\_PERIOD\_REG (0x0038)**

(reserved)														I2C_SCL_HIGH_PERIOD													
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													
														Reset													

**I2C\_SCL\_HIGH\_PERIOD** This register is used to configure how long SCL is kept high, in APB clock cycles. (R/W)

**Register 5.15: I2C\_SCL\_START\_HOLD\_REG (0x0040)**

(reserved)																						I2C_SCL_START_HOLD_TIME																																																																															
31																						10										9										0																																																											
0																						0										0										0										1										0										0										0										Reset									

**I2C\_SCL\_START\_HOLD\_TIME** This register is used to configure the time between the negative edge of SDA and the negative edge of SCL for a START condition, in APB clock cycles. (R/W)

**Register 5.16: I2C\_SCL\_RSTART\_SETUP\_REG (0x0044)**

(reserved)																						I2C_SCL_RSTART_SETUP_TIME																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
31																						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	0	0	0	0	0	0	0	0	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\_RSTART\_SETUP\_TIME** This register is used to configure the time between the positive edge of SCL and the negative edge of SDA for a RESTART condition, in APB clock cycles. (R/W)

**Register 5.17: I2C\_SCL\_STOP\_HOLD\_REG (0x0048)**

(reserved)																	I2C_SCL_STOP_HOLD_TIME																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																
31																	14																	13																	0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	0																	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\_STOP\_HOLD\_TIME** This register is used to configure the delay after the STOP condition's positive edge, in APB clock cycles. (R/W)

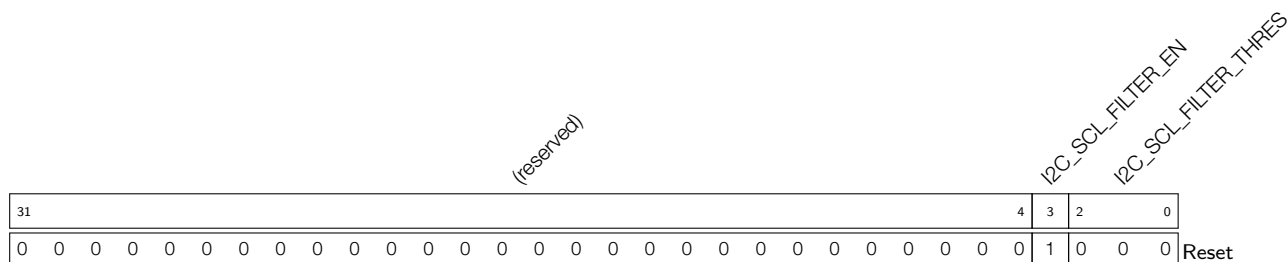
**Register 5.18: I2C\_SCL\_STOP\_SETUP\_REG (0x004C)**

(reserved)																						I2C_SCL_STOP_SETUP_TIME																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																						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									

Reset

**I2C\_SCL\_STOP\_SETUP\_TIME** This register is used to configure the time between the positive edge of SCL and the positive edge of SDA, in APB clock cycles. (R/W)

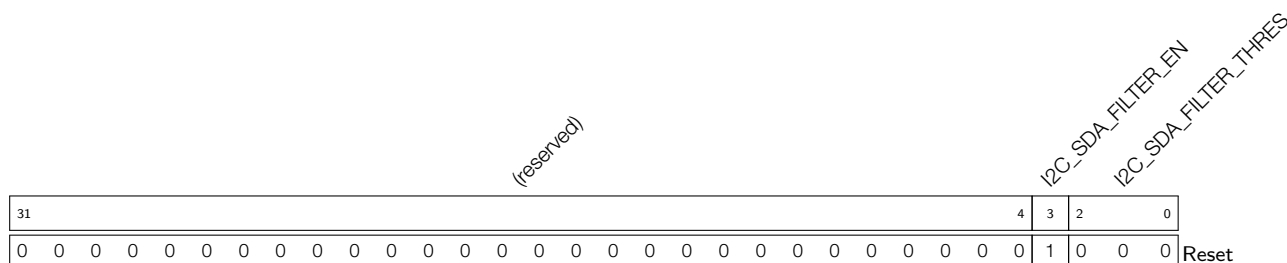
Register 5.19: I2C\_SCL\_FILTER\_CFG\_REG (0x0050)



**I2C\_SCL\_FILTER\_EN** This is the filter enable bit for SCL. (R/W)

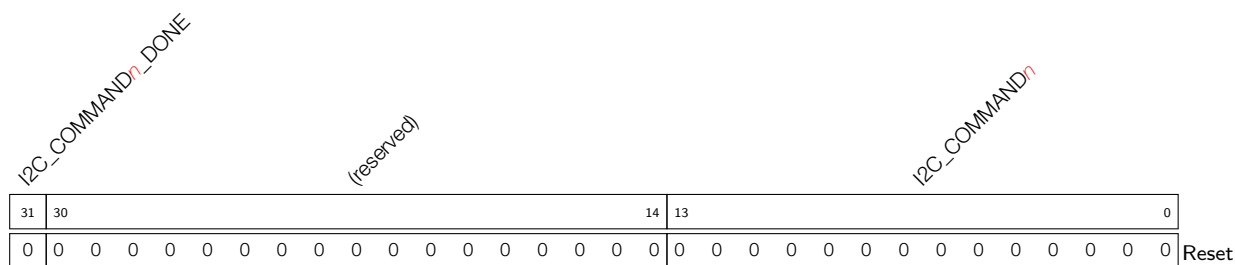
**I2C\_SCL\_FILTER\_THRES** When a pulse on the SCL input has a width smaller than this register value, in APB clock cycles, the I2C controller will ignore that pulse. (R/W)

Register 5.20: I2C\_SDA\_FILTER\_CFG\_REG (0x0054)



**I2C\_SDA\_FILTER\_EN** This is the filter enable bit for SDA. (R/W)

**I2C\_SDA\_FILTER\_THRES** When a pulse on the SDA input has a width smaller than this register value, in APB clock cycles, the I2C controller will ignore that pulse. (R/W)

Register 5.21: I2C\_CMD<sub>n</sub>\_REG (*n*: 0-15) (0x58+4\**n*)

**I2C\_COMMAND<sub>n</sub>\_DONE** When command *n* is done in I2C Master mode, this bit changes to high level. (R/W)

**I2C\_COMMAND<sub>n</sub>** This is the content of command *n*. It consists of three parts: (R/W)

op\_code is the command, 0: RSTART; 1: WRITE; 2: READ; 3: STOP; 4: END.

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](#) for more information.

## 6. LED\_PWM

### 6.1 Introduction

The LED\_PWM controller is primarily designed to control the intensity of LEDs, although it can be used to generate PWM signals for other purposes as well. It has 16 channels which can generate independent waveforms that can be used to drive e.g. RGB LED devices. For maximum flexibility, the high-speed as well as the low-speed channels can be driven from one of four high-speed/low-speed timers. The PWM controller also has the ability to automatically increase or decrease the duty cycle gradually, allowing for fades without any processor interference. To increase resolution, the LED\_PWM controller is also able to dither between two values when a fractional PWM value is configured.

The LED\_PWM controller has eight high-speed and eight low-speed PWM generators. In this document, they will be referred to as `hschn` and `lschn` respectively. These channels can be driven from four timers, which will be indicated by `h_timerx` and `l_timerx`.

### 6.2 Functional Description

#### 6.2.1 Architecture

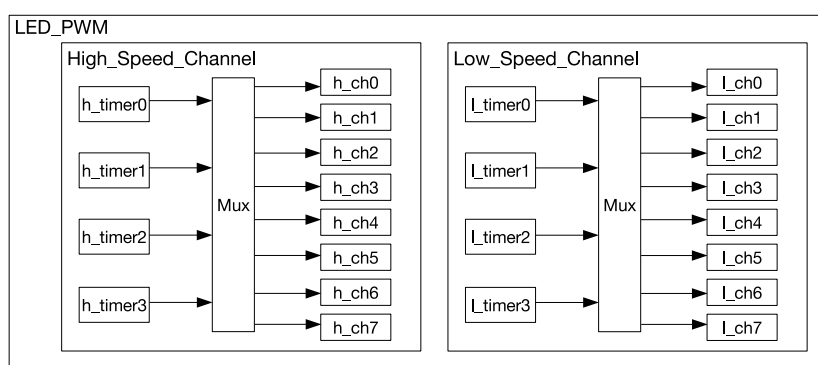


Figure 22: LED\_PWM Architecture

Figure 22 shows the architecture of the LED\_PWM controller. As can be seen from the figure, the LED\_PWM controller contains 8 high-speed and 8 low-speed channels. There are 4 high-speed clock modules for the high-speed channels, from which one `h_timerx` can be selected. There are also 4 low-speed clock modules for the low-speed channels, from which one `l_timerx` can be selected.

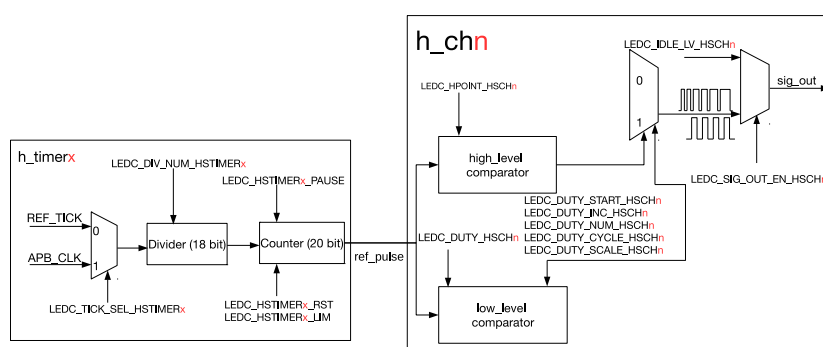


Figure 23: LED\_PWM High-speed Channel Diagram



Figure 23 illustrates a PWM channel with its selected timer; in this instance a high-speed channel and associated high-speed timer.

### 6.2.2 Timers

A high-speed timer consists of a multiplexer to select one of two clock sources: either REF\_TICK or APB\_CLK. For more information on the clock sources, please refer to Chapter [Reset And Clock](#). The input clock is divided down by a divider first. The division factor is specified by LEDC\_DIV\_NUM\_HSTIMER<sub>x</sub> which contains a fixed point number: the highest 10 bits represent the integer portion while the lowest 8 bits contain the fractional portion.

The divided clock signal is then fed into a 20-bit counter. This counter will count up to the value specified in LEDC\_HSTIMER<sub>x</sub>\_LIM. An overflow interrupt will be generated once the counting value reaches this value, at which point the counter restarts counting from 0. It is also possible to reset, suspend and read the values of the counter by software.

The output signal of the timer is the 20-bit value generated by the counter. The cycle period of this signal defines the frequency of the signals of any PWM channels connected to this timer. This frequency depends on both the division factor of the divider as well as the range of the counter:

$$f_{\text{sig\_out}} = \frac{f_{\text{REF\_TICK}} \cdot (!\text{LEDC\_TICK\_SEL\_HSTIMER}_x) + f_{\text{APB\_CLK}} \cdot \text{LEDC\_TICK\_SEL\_HSTIMER}_x}{\text{LEDC\_DIV\_NUM\_HSTIMER}_x \cdot 2^{\text{LEDC\_HSTIMER}_x\_LIM}}$$

The low-speed timers l\_timer<sub>x</sub> on the low-speed channel differ from the high-speed timers h\_timer<sub>x</sub> in two aspects:

1. Where the high-speed timer clock source can be clocked from REF\_TICK or APB\_CLK, the low speed timers are sourced from either REF\_TICK or SLOW\_CLOCK. The SLOW\_CLOCK source can be either APB\_CLK (80 MHz) or 8 MHz, selectable using LEDC\_APB\_CLK\_SEL.
2. The high-speed counter and divider are glitch-free, meaning that if software modifies the maximum counter or divisor value, the update will come into effect after the next overflow interrupt. In contrast, the low-speed counter and divider will update these values only when LEDC\_LSTIMER<sub>x</sub>\_PARA\_UP is set.

### 6.2.3 Channels

A channel takes the 20-bit value from the counter of the selected high-speed timer and compares it to a set of two values in order to set the channel output. The first value it is compared to is the contents of LEDC\_HPOINT\_HSCH<sub>n</sub>; if these two match, the output will be latched high. The second value is the sum of LEDC\_HPOINT\_HSCH<sub>n</sub> and LEDC\_DUTY\_HSCH<sub>n</sub>[24..4]. When this value is reached, the output is latched low. By using these two values, the relative phase and the duty cycle of the PWM output can be set. Figure 24 illustrates this.

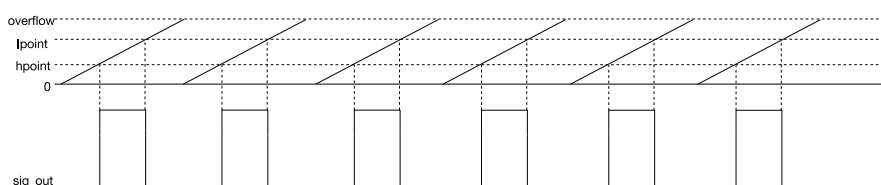


Figure 24: LED PWM Output Signal Diagram

LEDC\_DUTY\_HSCH $n$  is a fixed-point register with 4 fractional bits. As stated before, LEDC\_DUTY\_HSCH $n$ [24..4] is used in the PWM calculation directly, LEDC\_DUTY\_HSCH $n$ [3..0] can be used to dither the output. If this value is non-zero, with a statistical chance of LEDC\_DUTY\_HSCH $n$ [3..0]/16, the actual PWM pulse will be one cycle longer. This effectively increases the resolution of the PWM generator to 24 bits, but at the cost of a slight jitter in the duty cycle.

The channels also have ability to automatically fade from one duty cycle value to another. This feature is enabled by setting LEDC\_DUTY\_START\_HSCH $n$ . When this bit is set, the PWM controller will automatically increment or decrement the value in LEDC\_DUTY\_HSCH $n$ , depending on whether the bit LEDC\_DUTY\_INC\_HSCH $n$  is set or cleared, respectively. The speed the duty cycle changes is defined as such: every LEDC\_DUTY\_CYCLE\_HSCH $n$  cycles, the content of LEDC\_DUTY\_SCALE\_HSCH $n$  is added to or subtracted from LEDC\_DUTY\_HSCH $n$ [24..4]. The length of the fade can be limited by setting LEDC\_DUTY\_NUM\_HSCH $n$ : the fade will only last that number of cycles before finishing. A finished fade also generates an interrupt.

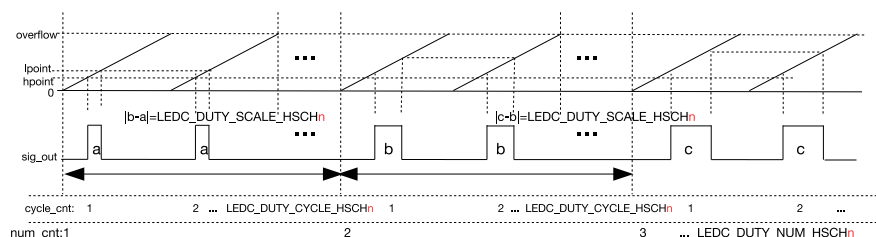


Figure 25: Output Signal Diagram of Gradient Duty Cycle

Figure 25 is an illustration of this. In this configuration, LEDC\_DUTY\_NUM\_HSCH $n$ \_R increases by LEDC\_DUTY\_SCALE\_HSCH $n$  every LEDC\_DUTY\_CYCLE\_HSCH $n$  clock cycles, which is reflected in the duty cycle of the output signal.

#### 6.2.4 Interrupts

- LEDC\_DUTY\_CHNG\_END\_LSCH $n$ \_INT: Triggered when a fade on a low-speed channel has finished.
- LEDC\_DUTY\_CHNG\_END\_HSCH $n$ \_INT: Triggered when a fade on a high-speed channel has finished.
- LEDC\_HS\_TIMER $x$ \_OVF\_INT: Triggered when a high-speed timer has reached its maximum counter value.
- LEDC\_LS\_TIMER $x$ \_OVF\_INT: Triggered when a low-speed timer has reached its maximum counter value.

## 6.3 Register Summary

Name	Description	Address	Access
<b>Configuration registers</b>			
LEDC_CONF_REG	Global ledc configuration register	0x3FF59190	R/W
LEDC_HSCH0_CONF0_REG	Configuration register 0 for high-speed channel 0	0x3FF59000	R/W
LEDC_HSCH1_CONF0_REG	Configuration register 0 for high-speed channel 1	0x3FF59014	R/W
LEDC_HSCH2_CONF0_REG	Configuration register 0 for high-speed channel 2	0x3FF59028	R/W
LEDC_HSCH3_CONF0_REG	Configuration register 0 for high-speed channel 3	0x3FF5903C	R/W
LEDC_HSCH4_CONF0_REG	Configuration register 0 for high-speed channel 4	0x3FF59050	R/W
LEDC_HSCH5_CONF0_REG	Configuration register 0 for high-speed channel 5	0x3FF59064	R/W
LEDC_HSCH6_CONF0_REG	Configuration register 0 for high-speed channel 6	0x3FF59078	R/W

Name	Description	Address	Access
<a href="#">LEDC_HSCH7_CONF0_REG</a>	Configuration register 0 for high-speed channel 7	0x3FF5908C	R/W
<a href="#">LEDC_HSCH0_CONF1_REG</a>	Configuration register 1 for high-speed channel 0	0x3FF5900C	R/W
<a href="#">LEDC_HSCH1_CONF1_REG</a>	Configuration register 1 for high-speed channel 1	0x3FF59020	R/W
<a href="#">LEDC_HSCH2_CONF1_REG</a>	Configuration register 1 for high-speed channel 2	0x3FF59034	R/W
<a href="#">LEDC_HSCH3_CONF1_REG</a>	Configuration register 1 for high-speed channel 3	0x3FF59048	R/W
<a href="#">LEDC_HSCH4_CONF1_REG</a>	Configuration register 1 for high-speed channel 4	0x3FF5905C	R/W
<a href="#">LEDC_HSCH5_CONF1_REG</a>	Configuration register 1 for high-speed channel 5	0x3FF59070	R/W
<a href="#">LEDC_HSCH6_CONF1_REG</a>	Configuration register 1 for high-speed channel 6	0x3FF59084	R/W
<a href="#">LEDC_HSCH7_CONF1_REG</a>	Configuration register 1 for high-speed channel 7	0x3FF59098	R/W
<a href="#">LEDC_LSCH0_CONF0_REG</a>	Configuration register 0 for low-speed channel 0	0x3FF590A0	R/W
<a href="#">LEDC_LSCH1_CONF0_REG</a>	Configuration register 0 for low-speed channel 1	0x3FF590B4	R/W
<a href="#">LEDC_LSCH2_CONF0_REG</a>	Configuration register 0 for low-speed channel 2	0x3FF590C8	R/W
<a href="#">LEDC_LSCH3_CONF0_REG</a>	Configuration register 0 for low-speed channel 3	0x3FF590DC	R/W
<a href="#">LEDC_LSCH4_CONF0_REG</a>	Configuration register 0 for low-speed channel 4	0x3FF590F0	R/W
<a href="#">LEDC_LSCH5_CONF0_REG</a>	Configuration register 0 for low-speed channel 5	0x3FF59104	R/W
<a href="#">LEDC_LSCH6_CONF0_REG</a>	Configuration register 0 for low-speed channel 6	0x3FF59118	R/W
<a href="#">LEDC_LSCH7_CONF0_REG</a>	Configuration register 0 for low-speed channel 7	0x3FF5912C	R/W
<a href="#">LEDC_LSCH0_CONF1_REG</a>	Configuration register 1 for low-speed channel 0	0x3FF590AC	R/W
<a href="#">LEDC_LSCH1_CONF1_REG</a>	Configuration register 1 for low-speed channel 1	0x3FF590C0	R/W
<a href="#">LEDC_LSCH2_CONF1_REG</a>	Configuration register 1 for low-speed channel 2	0x3FF590D4	R/W
<a href="#">LEDC_LSCH3_CONF1_REG</a>	Configuration register 1 for low-speed channel 3	0x3FF590E8	R/W
<a href="#">LEDC_LSCH4_CONF1_REG</a>	Configuration register 1 for low-speed channel 4	0x3FF590FC	R/W
<a href="#">LEDC_LSCH5_CONF1_REG</a>	Configuration register 1 for low-speed channel 5	0x3FF59110	R/W
<a href="#">LEDC_LSCH6_CONF1_REG</a>	Configuration register 1 for low-speed channel 6	0x3FF59124	R/W
<a href="#">LEDC_LSCH7_CONF1_REG</a>	Configuration register 1 for low-speed channel 7	0x3FF59138	R/W
<b>Duty-cycle registers</b>			
<a href="#">LEDC_HSCH0_DUTY_REG</a>	Initial duty cycle for high-speed channel 0	0x3FF59008	R/W
<a href="#">LEDC_HSCH1_DUTY_REG</a>	Initial duty cycle for high-speed channel 1	0x3FF5901C	R/W
<a href="#">LEDC_HSCH2_DUTY_REG</a>	Initial duty cycle for high-speed channel 2	0x3FF59030	R/W
<a href="#">LEDC_HSCH3_DUTY_REG</a>	Initial duty cycle for high-speed channel 3	0x3FF59044	R/W
<a href="#">LEDC_HSCH4_DUTY_REG</a>	Initial duty cycle for high-speed channel 4	0x3FF59058	R/W
<a href="#">LEDC_HSCH5_DUTY_REG</a>	Initial duty cycle for high-speed channel 5	0x3FF5906C	R/W
<a href="#">LEDC_HSCH6_DUTY_REG</a>	Initial duty cycle for high-speed channel 6	0x3FF59080	R/W
<a href="#">LEDC_HSCH7_DUTY_REG</a>	Initial duty cycle for high-speed channel 7	0x3FF59094	R/W
<a href="#">LEDC_HSCH0_DUTY_R_REG</a>	Current duty cycle for high-speed channel 0	0x3FF59010	RO
<a href="#">LEDC_HSCH1_DUTY_R_REG</a>	Current duty cycle for high-speed channel 1	0x3FF59024	RO
<a href="#">LEDC_HSCH2_DUTY_R_REG</a>	Current duty cycle for high-speed channel 2	0x3FF59038	RO
<a href="#">LEDC_HSCH3_DUTY_R_REG</a>	Current duty cycle for high-speed channel 3	0x3FF5904C	RO
<a href="#">LEDC_HSCH4_DUTY_R_REG</a>	Current duty cycle for high-speed channel 4	0x3FF59060	RO
<a href="#">LEDC_HSCH5_DUTY_R_REG</a>	Current duty cycle for high-speed channel 5	0x3FF59074	RO
<a href="#">LEDC_HSCH6_DUTY_R_REG</a>	Current duty cycle for high-speed channel 6	0x3FF59088	RO
<a href="#">LEDC_HSCH7_DUTY_R_REG</a>	Current duty cycle for high-speed channel 7	0x3FF5909C	RO
<a href="#">LEDC_LSCH0_DUTY_REG</a>	Initial duty cycle for low-speed channel 0	0x3FF590A8	R/W
<a href="#">LEDC_LSCH1_DUTY_REG</a>	Initial duty cycle for low-speed channel 1	0x3FF590BC	R/W

Name	Description	Address	Access
<a href="#">LEDC_LSCH2_DUTY_REG</a>	Initial duty cycle for low-speed channel 2	0x3FF590D0	R/W
<a href="#">LEDC_LSCH3_DUTY_REG</a>	Initial duty cycle for low-speed channel 3	0x3FF590E4	R/W
<a href="#">LEDC_LSCH4_DUTY_REG</a>	Initial duty cycle for low-speed channel 4	0x3FF590F8	R/W
<a href="#">LEDC_LSCH5_DUTY_REG</a>	Initial duty cycle for low-speed channel 5	0x3FF5910C	R/W
<a href="#">LEDC_LSCH6_DUTY_REG</a>	Initial duty cycle for low-speed channel 6	0x3FF59120	R/W
<a href="#">LEDC_LSCH7_DUTY_REG</a>	Initial duty cycle for low-speed channel 7	0x3FF59134	R/W
<a href="#">LEDC_LSCH0_DUTY_R_REG</a>	Current duty cycle for low-speed channel 0	0x3FF590B0	RO
<a href="#">LEDC_LSCH1_DUTY_R_REG</a>	Current duty cycle for low-speed channel 1	0x3FF590C4	RO
<a href="#">LEDC_LSCH2_DUTY_R_REG</a>	Current duty cycle for low-speed channel 2	0x3FF590D8	RO
<a href="#">LEDC_LSCH3_DUTY_R_REG</a>	Current duty cycle for low-speed channel 3	0x3FF590EC	RO
<a href="#">LEDC_LSCH4_DUTY_R_REG</a>	Current duty cycle for low-speed channel 4	0x3FF59100	RO
<a href="#">LEDC_LSCH5_DUTY_R_REG</a>	Current duty cycle for low-speed channel 5	0x3FF59114	RO
<a href="#">LEDC_LSCH6_DUTY_R_REG</a>	Current duty cycle for low-speed channel 6	0x3FF59128	RO
<a href="#">LEDC_LSCH7_DUTY_R_REG</a>	Current duty cycle for low-speed channel 7	0x3FF5913C	RO
<b>Timer registers</b>			
<a href="#">LEDC_HSTIMER0_CONF_REG</a>	High-speed timer 0 configuration	0x3FF59140	R/W
<a href="#">LEDC_HSTIMER1_CONF_REG</a>	High-speed timer 1 configuration	0x3FF59148	R/W
<a href="#">LEDC_HSTIMER2_CONF_REG</a>	High-speed timer 2 configuration	0x3FF59150	R/W
<a href="#">LEDC_HSTIMER3_CONF_REG</a>	High-speed timer 3 configuration	0x3FF59158	R/W
<a href="#">LEDC_HSTIMER0_VALUE_REG</a>	High-speed timer 0 current counter value	0x3FF59144	RO
<a href="#">LEDC_HSTIMER1_VALUE_REG</a>	High-speed timer 1 current counter value	0x3FF5914C	RO
<a href="#">LEDC_HSTIMER2_VALUE_REG</a>	High-speed timer 2 current counter value	0x3FF59154	RO
<a href="#">LEDC_HSTIMER3_VALUE_REG</a>	High-speed timer 3 current counter value	0x3FF5915C	RO
<a href="#">LEDC_HSTIMER0_CONF_REG</a>	Low-speed timer 0 configuration	0x3FF59140	R/W
<a href="#">LEDC_HSTIMER1_CONF_REG</a>	Low-speed timer 1 configuration	0x3FF59148	R/W
<a href="#">LEDC_HSTIMER2_CONF_REG</a>	Low-speed timer 2 configuration	0x3FF59150	R/W
<a href="#">LEDC_HSTIMER3_CONF_REG</a>	Low-speed timer 3 configuration	0x3FF59158	R/W
<a href="#">LEDC_HSTIMER0_VALUE_REG</a>	Low-speed timer 0 current counter value	0x3FF59144	RO
<a href="#">LEDC_HSTIMER1_VALUE_REG</a>	Low-speed timer 1 current counter value	0x3FF5914C	RO
<a href="#">LEDC_HSTIMER2_VALUE_REG</a>	Low-speed timer 2 current counter value	0x3FF59154	RO
<a href="#">LEDC_HSTIMER3_VALUE_REG</a>	Low-speed timer 3 current counter value	0x3FF5915C	RO
<b>Interrupt registers</b>			
<a href="#">LEDC_INT_RAW_REG</a>	Raw interrupt status	0x3FF59180	RO
<a href="#">LEDC_INT_ST_REG</a>	Masked interrupt status	0x3FF59184	RO
<a href="#">LEDC_INT_ENA_REG</a>	Interrupt enable bits	0x3FF59188	R/W
<a href="#">LEDC_INT_CLR_REG</a>	Interrupt clear bits	0x3FF5918C	WO

## 6.4 Registers

Register 6.1: LEDC\_HSCH $n$ \_CONF0\_REG ( $n$ : 0-7) (0x1C+0x10\* $n$ )

(reserved)					LEDC_IDLE_LV_HSCH <sup>n</sup> LEDC_SIG_OUT_EN_HSCH <sup>n</sup> LEDC_TIMER_SEL_HSCH <sup>n</sup>				
31				4	3	2	1	0	
0x00000000					0	0	0	Reset	

**LEDC\_IDLE\_LV\_HSCH $n$**  This bit is used to control the output value when high-speed channel  $n$  is inactive. (R/W)

**LEDC\_SIG\_OUT\_EN\_HSCH $n$**  This is the output enable control bit for high-speed channel  $n$ . (R/W)

**LEDC\_TIMER\_SEL\_HSCH $n$**  There are four high-speed timers. These two bits are used to select one of them for high-speed channel  $n$ : (R/W)

- 0: select htimer0;
- 1: select htimer1;
- 2: select htimer2;
- 3: select htimer3.

Register 6.2: LEDC\_HSCH $n$ \_HPOINT\_REG ( $n$ : 0-7) (0x20+0x10\* $n$ )

(reserved)																LEDC_HPOINT_HSCH <sup>n</sup>																																															
31																20																19																0															
0x0000																0x000000																Reset																															

**LEDC\_HPOINT\_HSCH $n$**  The output value changes to high when htimer $x$ ( $x$ =[0,3]) selected by high-speed channel  $n$  has reached reg\_hpoint\_hsch $n$ [19:0]. (R/W)

**Register 6.3: LEDC\_HSCH<sub>*n*</sub>\_DUTY\_REG (*n*: 0-7) (0x24+0x10\**n*)**

**LEDC\_DUTY\_HSCH $n$**  The register is used to control output duty. When hstimerx(x=[0,3]) selected by high-speed channel  $n$  has reached reg\_lpoint\_hsch $n$ , the output signal changes to low. (R/W)  
 reg\_lpoint\_hsch $n$ =(reg\_hpoint\_hsch $n$ [19:0]+reg\_duty\_hsch $n$ [24:4]) (1)  
 reg\_lpoint\_hsch $n$ =(reg\_hpoint\_hsch $n$ [19:0]+reg\_duty\_hsch $n$ [24:4] +1) (2)  
 See the [Functional Description](#) for more information on when (1) or (2) is chosen.

**Register 6.4: LEDC\_HSCH<sub>*n*</sub>\_CONF1\_REG (*n*: 0-7) (0x28+0x10\**n*)**

Bit Range	Field Name	Reset Value
31:24	LEDC_DUTY_START_HSCH	0x0000
23:16	LEDC_DUTY_NUM_HSCH	0x0000
15:8	LEDC_DUTY_CYCLE_HSCH	0x0000
7:0	LEDC_DUTY_SCALE_HSCH	0x0000

**LEDC\_DUTY\_START\_HSCH $n$**  When REG\_DUTY\_NUM\_HSCH $n$ , REG\_DUTY\_CYCLE\_HSCH $n$  and REG\_DUTY\_SCALE\_HSCH $n$  has been configured, these register won't take effect until REG\_DUTY\_START\_HSCH $n$  is set. This bit is automatically cleared by hardware. (R/W)

**LEDC\_DUTY\_INC\_HSCH $n$**  This register is used to increase or decrease the duty of output signal for high-speed channel  $n$ . (R/W)

**LEDC\_DUTY\_NUM\_HSCH $n$**  This register is used to control the number of times the duty cycle is increased or decreased for high-speed channel  $n$ . (R/W)

**LEDC\_DUTY\_CYCLE\_HSCH<sub>*n*</sub>** This register is used to increase or decrease the duty cycle every REG\_DUTY\_CYCLE\_HSCH<sub>*n*</sub> cycles for high-speed channel *n*. (R/W)

**LEDC\_DUTY\_SCALE\_HSCH $n$**  This register is used to increase or decrease the step scale for high-speed channel  $n$ . (R/W)

**Register 6.5: LEDC\_HSCH $n$ \_DUTY\_R\_REG ( $n$ : 0-7) (0x2C+0x10\* $n$ )**

(reserved)																LEDC_DUTY_HSCH <sub>n</sub> _R															
31											25	24																		0	
0x00												0x0000000																		Reset	

**LEDC\_DUTY\_HSCH $n$ \_R** This register represents the current duty cycle of the output signal for high-speed channel  $n$ . (RO)

**Register 6.6: LEDC\_LSCH $n$ \_CONF0\_REG ( $n$ : 0-7) (0xBC+0x10\* $n$ )**

(reserved)																LEDC_PARA_UP_LSCH <sup>n</sup> LEDC_IDLE_LV_LSCH <sup>n</sup> LEDC_SIG_OUT_EN_LSCH <sup>n</sup> LEDC_TIMER_SEL_LSCH <sup>n</sup>																					
31																5	4	3	2	1	0																
0x00000000																0	0	0	0	0	0	Reset															

**LEDC\_PARA\_UP\_LSCH $n$**  This bit is used to update register LEDC\_LSCH $n$ \_HPOINT and LEDC\_LSCH $n$ \_DUTY for low-speed channel  $n$ . (R/W)

**LEDC\_IDLE\_LV\_LSCH $n$**  This bit is used to control the output value when low-speed channel  $n$  is inactive. (R/W)

**LEDC\_SIG\_OUT\_EN\_LSCH $n$**  This is the output enable control bit for low-speed channel  $n$ . (R/W)

**LEDC\_TIMER\_SEL\_LSCH $n$**  There are four low speed timers, the two bits are used to select one of them for low-speed channel  $n$ . (R/W)

- 0: select Istimer0;
- 1: select Istimer1;
- 2: select Istimer2;
- 3: select Istimer3.

**Register 6.7: LEDC\_LSCH<sub>*n*</sub>\_HPOINT\_REG (*n*: 0-7) (0xC0+0x10\**n*)**

(reserved)																LEDC_HPOINT_LSCH <sup>n</sup>																																															
31																20																19																0															
0x0000																0x000000																Reset																															

**LEDC\_HPOINT\_LSCH<sub>*n*</sub>** The output value changes to high when Istimerx(*x*=[0,3]) selected by low-speed channel *n* has reached reg\_hpoint\_lschn[19:0]. (R/W)

**Register 6.8: LEDC\_LSCH<sub>*n*</sub>\_DUTY\_REG (*n*: 0-7) (0xC4+0x10\**n*)**

(reserved)																LEDC_DUTY_LSCH <sup>n</sup>																
31																				25	24											0
0x00										0x00000000																						Reset

**LEDC\_DUTY\_LSCH<sub>*n*</sub>** The register is used to control output duty. When Istimerx(*x*=[0,3]) chosen by low-speed channel *n* has reached reg\_lpoint\_lschn, the output signal changes to low. (R/W)

reg\_lpoint\_lschn=(reg\_hpoint\_lschn[19:0]+reg\_duty\_lschn[24:4]) (1)

reg\_lpoint\_lschn=(reg\_hpoint\_lschn[19:0]+reg\_duty\_lschn[24:4] +1) (2)

See the [Functional Description](#) for more information on when (1) or (2) is chosen.



**Register 6.9: LEDC\_LSCH<sub>*n*</sub>\_CONF1\_REG (*n*: 0-7) (0xC8+0x10\**n*)**

LEDC_DUTY_START_LSCH <sup>n</sup>			LEDC_DUTY_NUM_LSCH <sup>n</sup>																	LEDC_DUTY_CYCLE_LSCH <sup>n</sup>											LEDC_DUTY_SCALE_LSCH <sup>n</sup>												
31	30	29	20																	19	10											9	0										
0	1	0x000																	0x000											0x000											Reset		

**LEDC\_DUTY\_START\_LSCH<sub>*n*</sub>** When `reg_duty_num_hschn`, `reg_duty_cycle_hschn` and `reg_duty_scale_hschn` has been configured, these settings won't take effect until set `reg_duty_start_hschn`. This bit is automatically cleared by hardware. (R/W)

**LEDC\_DUTY\_INC\_LSCH<sub>*n*</sub>** This register is used to increase or decrease the duty of output signal for low-speed channel *n*. (R/W)

**LEDC\_DUTY\_NUM\_LSCH<sub>*n*</sub>** This register is used to control the number of times the duty cycle is increased or decreased for low-speed channel *n*. (R/W)

**LEDC\_DUTY\_CYCLE\_LSCH<sub>*n*</sub>** This register is used to increase or decrease the duty every `reg_duty_cycle_lschn` cycles for low-speed channel *n*. (R/W)

**LEDC\_DUTY\_SCALE\_LSCH<sub>*n*</sub>** This register is used to increase or decrease the step scale for low-speed channel *n*. (R/W)

**Register 6.10: LEDC\_LSCH<sub>*n*</sub>\_DUTY\_R\_REG (*n*: 0-7) (0xCC+0x10\**n*)**

(reserved)																LEDC_DUTY_LSCH <sub><i>n</i></sub> _R																
31								25								24								0								
0x00								0x00000000																								Reset

**LEDC\_DUTY\_LSCH<sub>*n*</sub>\_R** This register represents the current duty of the output signal for low-speed channel *n*. (RO)

**Register 6.11: LEDC\_HSTIMER<sub>x</sub>\_CONF\_REG (x: 0-3) (0x140+8\*x)**

(reserved)						LEDC_TICK_SEL_HSTIMER <del>x</del>																LEDC_HSTIMER <del>x</del> _RST																LEDC_HSTIMER <del>x</del> _PAUSE																LEDC_DIV_NUM_HSTIMER <del>x</del>																LEDC_HSTIMER <del>x</del> _LIM															
31						26						25		24		23		22																		5		4		0																																													
0x00						0						1		0		0x00000																		0x00		Reset																																																	

Reset

**LEDC\_TICK\_SEL\_HSTIMER<sub>x</sub>** This bit is used to select APB\_CLK or REF\_TICK for high-speed timer <sub>x</sub>. (R/W)

1: APB\_CLK;

0: REF\_TICK.

**LEDC\_HSTIMER<sub>x</sub>\_RST** This bit is used to reset high-speed timer <sub>x</sub>. The counter value will be 0 after reset. (R/W)

**LEDC\_HSTIMER<sub>x</sub>\_PAUSE** This bit is used to suspend the counter in high-speed timer <sub>x</sub>. (R/W)

**LEDC\_DIV\_NUM\_HSTIMER<sub>x</sub>** This register is used to configure the division factor for the divider in high-speed timer <sub>x</sub>. The least significant eight bits represent the fractional part. (R/W)

**LEDC\_HSTIMER<sub>x</sub>\_LIM** This register is used to control the range of the counter in high-speed timer <sub>x</sub>. The counter range is [0, 2\*\*reg\_hstimer<sub>x</sub>\_lim], the maximum bit width for counter is 20. (R/W)

**Register 6.12: LEDC\_HSTIMER<sub>x</sub>\_VALUE\_REG (x: 0-3) (0x144+8\*x)**

(reserved)										LEDC_HSTIMER <sub>x</sub> _CNT																							
31																			20	19													0
0x0000										0 0												Reset											

Reset

**LEDC\_HSTIMER<sub>x</sub>\_CNT** Software can read this register to get the current counter value of high-speed timer <sub>x</sub>. (RO)

**Register 6.13: LEDC\_LSTIMER<sub>x</sub>\_CONF\_REG (x: 0-3) (0x160+8\*x)**

(reserved)																							LEDC_LSTIMER <del>x</del> _PARA_UP																LEDC_DIV_NUM_LSTIMER <del>x</del>																LEDC_LSTIMER <del>x</del> _LIM							
																							LEDC_TICK_SEL_LSTIMER <del>x</del> _RST																																							
																							LEDC_LSTIMER <del>x</del> _PAUSE																																							
31																										5	4			0																																
0x00				0	0	1	0	0x00000																0x00				Reset																																		

**LEDC\_LSTIMER<sub>x</sub>\_PARA\_UP** Set this bit to update REG\_DIV\_NUM\_LSTIME<sub>x</sub> and REG\_LSTIMER<sub>x</sub>\_LIM. (R/W)

**LEDC\_TICK\_SEL\_LSTIMER<sub>x</sub>** This bit is used to select SLOW\_CLK or REF\_TICK for low-speed timer <sub>x</sub>. (R/W)  
 1: SLOW\_CLK;  
 0: REF\_TICK.

**LEDC\_LSTIMER<sub>x</sub>\_RST** This bit is used to reset low-speed timer <sub>x</sub>. The counter will be 0 after reset. (R/W)

**LEDC\_LSTIMER<sub>x</sub>\_PAUSE** This bit is used to suspend the counter in low-speed timer <sub>x</sub>. (R/W)

**LEDC\_DIV\_NUM\_LSTIMER<sub>x</sub>** This register is used to configure the division factor for the divider in low-speed timer <sub>x</sub>. The least significant eight bits represent the fractional part. (R/W)

**LEDC\_LSTIMER<sub>x</sub>\_LIM** This register is used to control the range of the counter in low-speed timer <sub>x</sub>. The counter range is  $[0, 2^{**}reg\_lstimer\_lim]$ , the max bit width for counter is 20. (R/W)

**Register 6.14: LEDC\_LSTIMER<sub>x</sub>\_VALUE\_REG (x: 0-3) (0x164+8\*x)**

(reserved)																LEDC_LSTIMER <sub>x</sub> _CNT																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
31																20																19																0																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
0x0000																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																0																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\_LSTIMER<sub>x</sub>\_CNT** Software can read this register to get the current counter value of low-speed timer <sub>x</sub>. (RO)

Register 6.15: LEDC\_INT\_RAW\_REG (0x0180)

(reserved)																								LEDC_DUTY_CHNG_END_LSC7_INT_RAW LEDC_DUTY_CHNG_END_LSC6_INT_RAW LEDC_DUTY_CHNG_END_LSC5_INT_RAW LEDC_DUTY_CHNG_END_LSC4_INT_RAW LEDC_DUTY_CHNG_END_LSC3_INT_RAW LEDC_DUTY_CHNG_END_LSC2_INT_RAW LEDC_DUTY_CHNG_END_LSC1_INT_RAW LEDC_DUTY_CHNG_END_HSC7_INT_RAW LEDC_DUTY_CHNG_END_HSC6_INT_RAW LEDC_DUTY_CHNG_END_HSC5_INT_RAW LEDC_DUTY_CHNG_END_HSC4_INT_RAW LEDC_DUTY_CHNG_END_HSC3_INT_RAW LEDC_DUTY_CHNG_END_HSC2_INT_RAW LEDC_DUTY_CHNG_END_HSC1_INT_RAW LEDC_LSTIMER3_OVF_INT_RAW LEDC_LSTIMER2_OVF_INT_RAW LEDC_LSTIMER1_OVF_INT_RAW LEDC_HSTIMER3_OVF_INT_RAW LEDC_HSTIMER2_OVF_INT_RAW LEDC_HSTIMER1_OVF_INT_RAW															
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	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												

**LEDC\_DUTY\_CHNG\_END\_LSCH<sub>n</sub>\_INT\_RAW** The raw interrupt status bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_LSCH<sub>n</sub>\\_INT](#) interrupt. (RO)

**LEDC\_DUTY\_CHNG\_END\_HSCH<sub>n</sub>\_INT\_RAW** The raw interrupt status bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_HSCH<sub>n</sub>\\_INT](#) interrupt. (RO)

**LEDC\_LSTIMER<sub>x</sub>\_OVF\_INT\_RAW** The raw interrupt status bit for the [LEDC\\_LSTIMER<sub>x</sub>\\_OVF\\_INT](#) interrupt. (RO)

**LEDC\_HSTIMER<sub>x</sub>\_OVF\_INT\_RAW** The raw interrupt status bit for the [LEDC\\_HSTIMER<sub>x</sub>\\_OVF\\_INT](#) interrupt. (RO)

Register 6.16: LEDC\_INT\_ST\_REG (0x0184)

(reserved)																								LEDC_DUTY_CHNG_END_LSC7_INT_ST LEDC_DUTY_CHNG_END_LSC6_INT_ST LEDC_DUTY_CHNG_END_LSC5_INT_ST LEDC_DUTY_CHNG_END_LSC4_INT_ST LEDC_DUTY_CHNG_END_LSC3_INT_ST LEDC_DUTY_CHNG_END_LSC2_INT_ST LEDC_DUTY_CHNG_END_LSC1_INT_ST LEDC_DUTY_CHNG_END_HSCH7_INT_ST LEDC_DUTY_CHNG_END_HSCH6_INT_ST LEDC_DUTY_CHNG_END_HSCH5_INT_ST LEDC_DUTY_CHNG_END_HSCH4_INT_ST LEDC_DUTY_CHNG_END_HSCH3_INT_ST LEDC_DUTY_CHNG_END_HSCH2_INT_ST LEDC_DUTY_CHNG_END_HSCH1_INT_ST LEDC_LSTIMER3_OVF_INT_ST LEDC_LSTIMER2_OVF_INT_ST LEDC_LSTIMER1_OVF_INT_ST LEDC_HSTIMER3_OVF_INT_ST LEDC_HSTIMER2_OVF_INT_ST LEDC_HSTIMER1_OVF_INT_ST LEDC_HSTIMER0_OVF_INT_ST															
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	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												

**LEDC\_DUTY\_CHNG\_END\_LSCH<sub>n</sub>\_INT\_ST** The masked interrupt status bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_LSCH<sub>n</sub>\\_INT](#) interrupt. (RO)

**LEDC\_DUTY\_CHNG\_END\_HSCH<sub>n</sub>\_INT\_ST** The masked interrupt status bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_HSCH<sub>n</sub>\\_INT](#) interrupt. (RO)

**LEDC\_LSTIMER<sub>x</sub>\_OVF\_INT\_ST** The masked interrupt status bit for the [LEDC\\_LSTIMER<sub>x</sub>\\_OVF\\_INT](#) interrupt. (RO)

**LEDC\_HSTIMER<sub>x</sub>\_OVF\_INT\_ST** The masked interrupt status bit for the [LEDC\\_HSTIMER<sub>x</sub>\\_OVF\\_INT](#) interrupt. (RO)

Register 6.17: LEDC\_INT\_ENA\_REG (0x0188)

(reserved)																								LEDC_DUTY_CHNG_END_LSCH7_INT_ENA LEDC_DUTY_CHNG_END_LSCH6_INT_ENA LEDC_DUTY_CHNG_END_LSCH5_INT_ENA LEDC_DUTY_CHNG_END_LSCH4_INT_ENA LEDC_DUTY_CHNG_END_LSCH3_INT_ENA LEDC_DUTY_CHNG_END_LSCH2_INT_ENA LEDC_DUTY_CHNG_END_LSCH1_INT_ENA LEDC_DUTY_CHNG_END_LSCH0_INT_ENA LEDC_DUTY_CHNG_END_HSCH7_INT_ENA LEDC_DUTY_CHNG_END_HSCH6_INT_ENA LEDC_DUTY_CHNG_END_HSCH5_INT_ENA LEDC_DUTY_CHNG_END_HSCH4_INT_ENA LEDC_DUTY_CHNG_END_HSCH3_INT_ENA LEDC_DUTY_CHNG_END_HSCH2_INT_ENA LEDC_DUTY_CHNG_END_HSCH1_INT_ENA LEDC_DUTY_CHNG_END_HSCH0_INT_ENA LEDC_LSTIMER3_OVF_INT_ENA LEDC_LSTIMER2_OVF_INT_ENA LEDC_LSTIMER1_OVF_INT_ENA LEDC_LSTIMER0_OVF_INT_ENA LEDC_HSTIMER3_OVF_INT_ENA LEDC_HSTIMER2_OVF_INT_ENA LEDC_HSTIMER1_OVF_INT_ENA LEDC_HSTIMER0_OVF_INT_ENA																							
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	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																					

**LEDC\_DUTY\_CHNG\_END\_LSCH $n$ \_INT\_ENA** The interrupt enable bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_LSCH \$n\$ \\_INT](#) interrupt. (R/W)

**LEDC\_DUTY\_CHNG\_END\_HSCH $n$ \_INT\_ENA** The interrupt enable bit for the [LEDC\\_DUTY\\_CHNG\\_END\\_HSCH \$n\$ \\_INT](#) interrupt. (R/W)

**LEDC\_LSTIMER $x$ \_OVF\_INT\_ENA** The interrupt enable bit for the [LEDC\\_LSTIMER \$x\$ \\_OVF\\_INT](#) interrupt. (R/W)

**LEDC\_HSTIMER $x$ \_OVF\_INT\_ENA** The interrupt enable bit for the [LEDC\\_HSTIMER \$x\$ \\_OVF\\_INT](#) interrupt. (R/W)

Register 6.18: LEDC\_INT\_CLR\_REG (0x018C)

(reserved)																								LEDC_DUTY_CHNG_END_LSCH7_INT_CLR												LEDC_DUTY_CHNG_END_LSCH6_INT_CLR												LEDC_DUTY_CHNG_END_LSCH5_INT_CLR												LEDC_DUTY_CHNG_END_LSCH4_INT_CLR												LEDC_DUTY_CHNG_END_LSCH3_INT_CLR												LEDC_DUTY_CHNG_END_LSCH2_INT_CLR												LEDC_DUTY_CHNG_END_LSCH1_INT_CLR												LEDC_DUTY_CHNG_END_LSCH0_INT_CLR												LEDC_DUTY_CHNG_END_HSCH7_INT_CLR												LEDC_DUTY_CHNG_END_HSCH6_INT_CLR												LEDC_DUTY_CHNG_END_HSCH5_INT_CLR												LEDC_DUTY_CHNG_END_HSCH4_INT_CLR												LEDC_DUTY_CHNG_END_HSCH3_INT_CLR												LEDC_DUTY_CHNG_END_HSCH2_INT_CLR												LEDC_DUTY_CHNG_END_HSCH1_INT_CLR												LEDC_DUTY_CHNG_END_HSCH0_INT_CLR												LEDC_LSTIMER3_OVF_INT_CLR												LEDC_LSTIMER2_OVF_INT_CLR												LEDC_LSTIMER1_OVF_INT_CLR												LEDC_LSTIMER0_OVF_INT_CLR												LEDC_HSTIMER3_OVF_INT_CLR												LEDC_HSTIMER2_OVF_INT_CLR												LEDC_HSTIMER1_OVF_INT_CLR												LEDC_HSTIMER0_OVF_INT_CLR																																																																																																																																																																																																																						
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	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

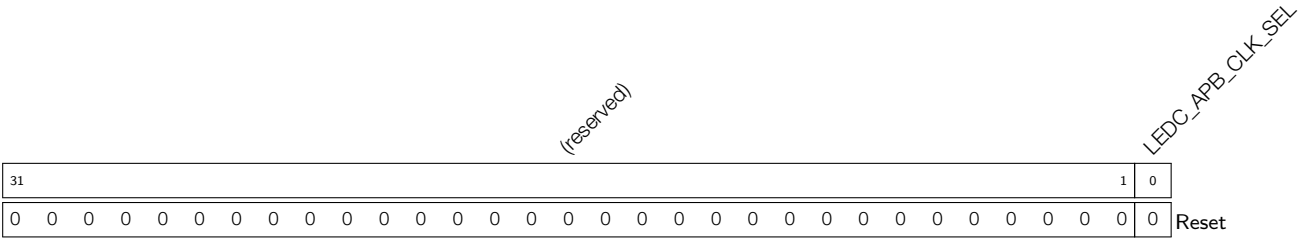
**LEDC\_DUTY\_CHNG\_END\_LSCH $n$ \_INT\_CLR** Set this bit to clear the [LEDC\\_DUTY\\_CHNG\\_END\\_LSCH \$n\$ \\_INT](#) interrupt. (WO)

**LEDC\_DUTY\_CHNG\_END\_HSCH $n$ \_INT\_CLR** Set this bit to clear the [LEDC\\_DUTY\\_CHNG\\_END\\_HSCH \$n\$ \\_INT](#) interrupt. (WO)

**LEDC\_LSTIMER $x$ \_OVF\_INT\_CLR** Set this bit to clear the [LEDC\\_LSTIMER \$x\$ \\_OVF\\_INT](#) interrupt. (WO)

**LEDC\_HSTIMER $x$ \_OVF\_INT\_CLR** Set this bit to clear the [LEDC\\_HSTIMER \$x\$ \\_OVF\\_INT](#) interrupt. (WO)

Register 6.19: LEDC\_CONF\_REG (0x0190)



**LEDC\_APB\_CLK\_SEL** This bit is used to set the frequency of SLOW\_CLK. (R/W)  
0: 8 MHz;  
1: 80 MHz.

## 7. Remote Controller Peripheral

### 7.1 Introduction

The RMT (Remote Control) module is primarily designed to send and receive infrared remote control signals that use on-off-keying of a carrier frequency, but due to its design it can be used to generate various types of signals. An RMT transmitter does this by reading consecutive duration values for an active and inactive output from the built-in RAM block, optionally modulating it with a carrier wave. A receiver will inspect its input signal, optionally filtering it, and will place the lengths of time the signal is active and inactive in the RAM block.

The RMT module has eight channels, numbered 0 to 7; registers, signals and blocks that are duplicated every channel are indicated by an *n* as a placeholder for the channel number.

### 7.2 Functional Description

#### 7.2.1 RMT Architecture

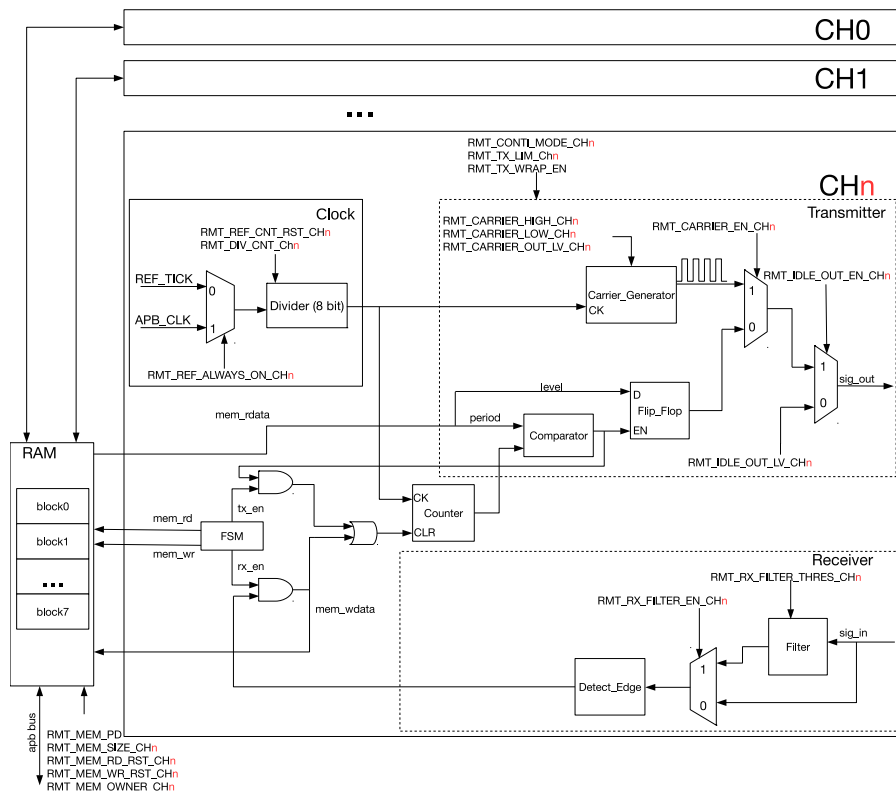


Figure 26: RMT Architecture

The RMT module contains eight channels; each channel has a transmitter and receiver, of which one can be active per channel. The 8 channels share a 512x32-bit RAM block which can be read and written by the processor cores over the APB bus, as well as read by the transmitters and written by the receivers. The transmitted signal can optionally be modulated by a carrier wave. Each channel is clocked by a divided-down signal derived from either the APB bus clock or REF\_TICK.

## 7.2.2 RMT RAM

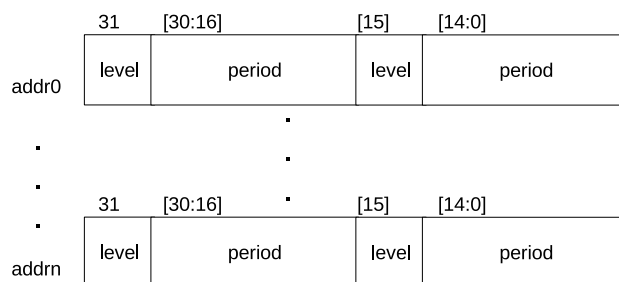


Figure 27: Data Structure

The data structure in RAM is shown in Figure 27. Each 32-bit value contains two 16-bit entries, containing two fields each: "level" indicates whether a high-level or a low-level value is to be sent or was received, "period" is the duration (in channel clock periods) for which the level lasts. A period of zero is interpreted as an end marker: the transmitter will stop transmitting once it has read this and the receiver will write this once it has detected that the signal it received has gone idle.

Normally, only one block of 64x32-bit worth of data can be sent or received. If the data size is larger than this block size, blocks can either be extended or the channel can be configured for wraparound mode.

The RMT RAM can be accessed via APB bus. The initial address is RMT base address + 0x800. The RAM block is divided into eight 64x32-bit blocks. By default, each channel uses one block (block 0 for channel 0, block 1 for channel 1, and so on). Users can extend the memory for a specific channel by configuring RMT\_MEM\_SIZE\_CH $n$  register; setting this to >1 will make the channel also use the memory of subsequent channels. The RAM address range for channel  $n$  is start\_addr\_CH $n$  to end\_addr\_CH $n$ , which are defined by:

start\_addr\_ch $n$  = RMT base address + 0x800 + 64 \* 4 \*  $n$ , and

end\_addr\_ch $n$  = RMT base address + 0x800 + 64 \* 4 \*  $n$  + 64 \* 4 \* RMT\_MEM\_SIZE\_CH $n$  mod 512 \* 4

To protect a receiver from overwriting the blocks a transmitter is about to transmit, RMT\_MEM\_OWNER\_CH $n$  can be configured to assign the owner, i.e. transmitter or receiver, of channel  $n$ 's RAM block. If this ownership is violated, the RMT\_CH $n$ \_ERR interrupt will be generated.

## 7.2.3 Clock

The main clock for a channel is generated by taking either the 80 MHz APB clock or REF\_TICK (usually 1MHz), according to the state of RMT\_REF\_ALWAYS\_ON\_CH $n$ . For more information on the clock sources, please refer to Chapter [Reset And Clock](#). This gets then scaled down using a configurable 8-bit divider to create the channel clock, to be used by both the carrier wave generator as well as by the counter. The divider value can be set by configuring RMT\_DIV\_CNT\_CH $n$ .

## 7.2.4 Transmitter

When the RMT\_TX\_START\_CH $n$  register is 1, the transmitter of channel  $n$  will start reading data from RAM and sending it. The transmitter will receive a 32-bits value each time it reads from RAM. Of these 32 bits, the low 16-bit entry is sent first, the high entry second.

To transmit more data than fits in the channels RAM, wraparound mode can be enabled. In this mode, when the transmitter has reached the last entry in the channels memory, it will loop back to the first byte. To use this mechanism to send more data than fits in the channels RAM, fill the RAM with the initial events and set RMT\_CH $n$ \_TX\_LIM\_REG to cause an RMT\_CH $n$ \_TX\_THR\_EVENT\_INT interrupt before the wraparound



happens. Then, when the interrupt happens, the data that was already sent should be replaced with subsequent events: when the wraparound happens the transmitter will seamlessly continue sending the new events.

With or without wraparound mode enabled, transmission ends when an entry with a length of 0 is encountered. When this happens, the transmitter will generate a `RMT_CH $n$ _TX_END_INT` interrupt, and return to the idle state. When a transmitter is in the idle state users can configure `RMT_IDLE_OUT_EN_CH $n$`  and `RMT_IDLE_OUT_LV_CH $n$`  to control the transmitter output manually.

The output of the transmitter can be modulated using a carrier wave by setting `RMT_CARRIER_EN_CH $n$` . The carrier frequency and duty cycle can be configured by configuring its high and low durations, in channel clock cycles, in `RMT_CARRIER_HIGH_CH $n$`  and `RMT_CARRIER_LOW_CH $n$` .

### 7.2.5 Receiver

When `RMT_RX_EN_CH $n$`  is set to 1, the receiver in channel  $n$  becomes active, measuring the duration between input signal edges. These will be written as period/level value pairs to the channel RAM in the same fashion as the transmitter sends them. Receiving ends when the receiver detects no change in signal level for more than `RMT_IDLE_THRES_CH $n$`  channel clock ticks; the receiver will write a final entry with period 0, generate an `RMT_CH $n$ _RX_END_INT_RAW` interrupt and return to the idle state.

The receiver has an input signal filter which can be configured using `RMT_RX_FILTER_EN_CH $n$` : The filter will remove pulses with of length less than `RMT_RX_FILTER_THRES_CH $n$`  APB clock periods.

When the RMT module is inactive, the RAM can be put into low-power mode by setting the `RMT_MEM_PD` register to 1.

### 7.2.6 Interrupts

- `RMT_CH $n$ _TX_THR_EVENT_INT`: Triggered when the number of events the transmitter has sent matches the contents of the `RMT_CH $n$ _TX_LIM_REG` register.
- `RMT_CH $n$ _TX_END_INT`: Triggered when the transmitter has finished transmitting the signal.
- `RMT_CH $n$ _RX_END_INT`: Triggered when the receiver has finished receiving a signal.

## 7.3 Register Summary

Name	Description	Address	Access
<b>Configuration registers</b>			
<a href="#">RMT_CH0CONF0_REG</a>	Channel 0 config register 0	0x3FF56020	R/W
<a href="#">RMT_CH0CONF1_REG</a>	Channel 0 config register 1	0x3FF56024	R/W
<a href="#">RMT_CH1CONF0_REG</a>	Channel 1 config register 0	0x3FF56028	R/W
<a href="#">RMT_CH1CONF1_REG</a>	Channel 1 config register 1	0x3FF5602C	R/W
<a href="#">RMT_CH2CONF0_REG</a>	Channel 2 config register 0	0x3FF56030	R/W
<a href="#">RMT_CH2CONF1_REG</a>	Channel 2 config register 1	0x3FF56034	R/W
<a href="#">RMT_CH3CONF0_REG</a>	Channel 3 config register 0	0x3FF56038	R/W
<a href="#">RMT_CH3CONF1_REG</a>	Channel 3 config register 1	0x3FF5603C	R/W
<a href="#">RMT_CH4CONF0_REG</a>	Channel 4 config register 0	0x3FF56040	R/W
<a href="#">RMT_CH4CONF1_REG</a>	Channel 4 config register 1	0x3FF56044	R/W
<a href="#">RMT_CH5CONF0_REG</a>	Channel 5 config register 0	0x3FF56048	R/W
<a href="#">RMT_CH5CONF1_REG</a>	Channel 5 config register 1	0x3FF5604C	R/W
<a href="#">RMT_CH6CONF0_REG</a>	Channel 6 config register 0	0x3FF56050	R/W
<a href="#">RMT_CH6CONF1_REG</a>	Channel 6 config register 1	0x3FF56054	R/W
<a href="#">RMT_CH7CONF0_REG</a>	Channel 7 config register 0	0x3FF56058	R/W
<a href="#">RMT_CH7CONF1_REG</a>	Channel 7 config register 1	0x3FF5605C	R/W
<b>Interrupt registers</b>			
<a href="#">RMT_INT_RAW_REG</a>	Raw interrupt status	0x3FF560A0	RO
<a href="#">RMT_INT_ST_REG</a>	Masked interrupt status	0x3FF560A4	RO
<a href="#">RMT_INT_ENA_REG</a>	Interrupt enable bits	0x3FF560A8	R/W
<a href="#">RMT_INT_CLR_REG</a>	Interrupt clear bits	0x3FF560AC	WO
<b>Carrier wave duty cycle registers</b>			
<a href="#">RMT_CH0CARRIER_DUTY_REG</a>	Channel 0 duty cycle configuration register	0x3FF560B0	R/W
<a href="#">RMT_CH1CARRIER_DUTY_REG</a>	Channel 1 duty cycle configuration register	0x3FF560B4	R/W
<a href="#">RMT_CH2CARRIER_DUTY_REG</a>	Channel 2 duty cycle configuration register	0x3FF560B8	R/W
<a href="#">RMT_CH3CARRIER_DUTY_REG</a>	Channel 3 duty cycle configuration register	0x3FF560BC	R/W
<a href="#">RMT_CH4CARRIER_DUTY_REG</a>	Channel 4 duty cycle configuration register	0x3FF560C0	R/W
<a href="#">RMT_CH5CARRIER_DUTY_REG</a>	Channel 5 duty cycle configuration register	0x3FF560C4	R/W
<a href="#">RMT_CH6CARRIER_DUTY_REG</a>	Channel 6 duty cycle configuration register	0x3FF560C8	R/W
<a href="#">RMT_CH7CARRIER_DUTY_REG</a>	Channel 7 duty cycle configuration register	0x3FF560CC	R/W
<b>Tx event configuration registers</b>			
<a href="#">RMT_CH0_TX_LIM_REG</a>	Channel 0 Tx event configuration register	0x3FF560D0	R/W
<a href="#">RMT_CH1_TX_LIM_REG</a>	Channel 1 Tx event configuration register	0x3FF560D4	R/W
<a href="#">RMT_CH2_TX_LIM_REG</a>	Channel 2 Tx event configuration register	0x3FF560D8	R/W
<a href="#">RMT_CH3_TX_LIM_REG</a>	Channel 3 Tx event configuration register	0x3FF560DC	R/W
<a href="#">RMT_CH4_TX_LIM_REG</a>	Channel 4 Tx event configuration register	0x3FF560E0	R/W
<a href="#">RMT_CH5_TX_LIM_REG</a>	Channel 5 Tx event configuration register	0x3FF560E4	R/W
<a href="#">RMT_CH6_TX_LIM_REG</a>	Channel 6 Tx event configuration register	0x3FF560E8	R/W
<a href="#">RMT_CH7_TX_LIM_REG</a>	Channel 7 Tx event configuration register	0x3FF560EC	R/W
<b>Other registers</b>			
<a href="#">RMT_APB_CONF_REG</a>	RMT-wide configuration register	0x3FF560F0	R/W

## 7.4 Registers

**Register 7.1: RMT\_CH $n$ CONF0\_REG ( $n$ : 0-7) (0x0058+8\* $n$ )**

(reserved)				RMT_MEM_PD				RMT_CARRIER_OUT_LV_CH $n$				RMT_CARRIER_EN_CH $n$				RMT_MEM_SIZE_CH $n$				RMT_IDLE_THRES_CH $n$								RMT_DIV_CNT_CH $n$			
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
0x0	0	1	1	0x01				0x01000								0x002															

Reset

**RMT\_MEM\_PD** This bit is used power down the entire RMT RAM block (Only exists in RMT\_CH0CONF0). 1: memory is powered down 0: memory is powered up (R/W)

**RMT\_CARRIER\_OUT\_LV\_CH $n$**  This bit is used to configure when the carrier wave is transmitted. 1: transmit on low output level, 0: transmit on high output level. (R/W)

**RMT\_CARRIER\_EN\_CH $n$**  This is the carrier modulation enable control bit for channel $n$ . 1: carrier modulation enabled 0: carrier modulation disabled (R/W)

**RMT\_MEM\_SIZE\_CH $n$**  This register is used to configure the amount of memory blocks allocated to channel  $n$  (R/W)

**RMT\_IDLE\_THRES\_CH $n$**  In receive mode, when no edge is detected on the input signal for longer than reg\_idle\_thres\_ch $n$  channel clock cycles, the receive process is finished. (R/W)

**RMT\_DIV\_CNT\_CH $n$**  This register is used to set the divider for the channel clock of channel  $n$ . (R/W)



Register 7.3: RMT\_INT\_RAW\_REG (0x00a0)

RMT_CH7_TX_THR_EVENT_INT_RAW	RMT_CH6_TX_THR_EVENT_INT_RAW	RMT_CH5_TX_THR_EVENT_INT_RAW	RMT_CH4_TX_THR_EVENT_INT_RAW	RMT_CH3_TX_THR_EVENT_INT_RAW	RMT_CH2_TX_THR_EVENT_INT_RAW	RMT_CH1_TX_THR_EVENT_INT_RAW	RMT_CH0_TX_THR_EVENT_INT_RAW	RMT_CH7_ERR_INT_RAW	RMT_CH6_ERR_INT_RAW	RMT_CH5_ERR_INT_RAW	RMT_CH4_ERR_INT_RAW	RMT_CH3_ERR_INT_RAW	RMT_CH2_ERR_INT_RAW	RMT_CH1_ERR_INT_RAW	RMT_CH0_ERR_INT_RAW	RMT_CH7_RX_END_INT_RAW	RMT_CH6_RX_END_INT_RAW	RMT_CH5_RX_END_INT_RAW	RMT_CH4_RX_END_INT_RAW	RMT_CH3_RX_END_INT_RAW	RMT_CH2_RX_END_INT_RAW	RMT_CH1_RX_END_INT_RAW	RMT_CH0_RX_END_INT_RAW	RMT_CH7_TX_END_INT_RAW	RMT_CH6_TX_END_INT_RAW	RMT_CH5_TX_END_INT_RAW	RMT_CH4_TX_END_INT_RAW	RMT_CH3_TX_END_INT_RAW	RMT_CH2_TX_END_INT_RAW	RMT_CH1_TX_END_INT_RAW	RMT_CH0_TX_END_INT_RAW	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	0	0

**RMT\_CH<sub>n</sub>\_TX\_THR\_EVENT\_INT\_RAW** The raw interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_TX\\_THR\\_EVENT\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_ERR\_INT\_RAW** The raw interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_ERR\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_RX\_END\_INT\_RAW** The raw interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_RX\\_END\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_TX\_END\_INT\_RAW** The raw interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_TX\\_END\\_INT](#) interrupt. (RO)

Register 7.4: RMT\_INT\_ST\_REG (0x00a4)

RMT_CH7_TX_THR_EVENT_INT_ST	RMT_CH6_TX_THR_EVENT_INT_ST	RMT_CH5_TX_THR_EVENT_INT_ST	RMT_CH4_TX_THR_EVENT_INT_ST	RMT_CH3_TX_THR_EVENT_INT_ST	RMT_CH2_TX_THR_EVENT_INT_ST	RMT_CH1_TX_THR_EVENT_INT_ST	RMT_CH0_TX_THR_EVENT_INT_ST	RMT_CH7_ERR_INT_ST	RMT_CH6_ERR_INT_ST	RMT_CH5_ERR_INT_ST	RMT_CH4_ERR_INT_ST	RMT_CH3_ERR_INT_ST	RMT_CH2_ERR_INT_ST	RMT_CH1_ERR_INT_ST	RMT_CH0_ERR_INT_ST	RMT_CH7_RX_END_INT_ST	RMT_CH6_RX_END_INT_ST	RMT_CH5_RX_END_INT_ST	RMT_CH4_RX_END_INT_ST	RMT_CH3_RX_END_INT_ST	RMT_CH2_RX_END_INT_ST	RMT_CH1_RX_END_INT_ST	RMT_CH0_RX_END_INT_ST	RMT_CH7_TX_END_INT_ST	RMT_CH6_TX_END_INT_ST	RMT_CH5_TX_END_INT_ST	RMT_CH4_TX_END_INT_ST	RMT_CH3_TX_END_INT_ST	RMT_CH2_TX_END_INT_ST	RMT_CH1_TX_END_INT_ST	RMT_CH0_TX_END_INT_ST	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	0	0

**RMT\_CH<sub>n</sub>\_TX\_THR\_EVENT\_INT\_ST** The masked interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_TX\\_THR\\_EVENT\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_ERR\_INT\_ST** The masked interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_ERR\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_RX\_END\_INT\_ST** The masked interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_RX\\_END\\_INT](#) interrupt. (RO)

**RMT\_CH<sub>n</sub>\_TX\_END\_INT\_ST** The masked interrupt status bit for the [RMT\\_CH<sub>n</sub>\\_TX\\_END\\_INT](#) interrupt. (RO)

Register 7.5: RMT\_INT\_ENA\_REG (0x00a8)

RMT_CH7_TX_THR_EVENT_INT_ENA	RMT_CH6_TX_THR_EVENT_INT_ENA	RMT_CH5_TX_THR_EVENT_INT_ENA	RMT_CH4_TX_THR_EVENT_INT_ENA	RMT_CH3_TX_THR_EVENT_INT_ENA	RMT_CH2_TX_THR_EVENT_INT_ENA	RMT_CH1_TX_THR_EVENT_INT_ENA	RMT_CH0_TX_THR_EVENT_INT_ENA	RMT_CH7_ERR_INT_ENA	RMT_CH6_ERR_INT_ENA	RMT_CH5_ERR_INT_ENA	RMT_CH4_ERR_INT_ENA	RMT_CH3_ERR_INT_ENA	RMT_CH2_ERR_INT_ENA	RMT_CH1_ERR_INT_ENA	RMT_CH0_ERR_INT_ENA	RMT_CH7_RX_END_INT_ENA	RMT_CH6_RX_END_INT_ENA	RMT_CH5_RX_END_INT_ENA	RMT_CH4_RX_END_INT_ENA	RMT_CH3_RX_END_INT_ENA	RMT_CH2_RX_END_INT_ENA	RMT_CH1_RX_END_INT_ENA	RMT_CH0_RX_END_INT_ENA	RMT_CH7_TX_END_INT_ENA	RMT_CH6_TX_END_INT_ENA	RMT_CH5_TX_END_INT_ENA	RMT_CH4_TX_END_INT_ENA	RMT_CH3_TX_END_INT_ENA	RMT_CH2_TX_END_INT_ENA	RMT_CH1_TX_END_INT_ENA	RMT_CH0_TX_END_INT_ENA	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	0	0

**RMT\_CH<sub>n</sub>TX\_THR\_EVENT\_INT\_ENA** The interrupt enable bit for the [RMT\\_CH<sub>n</sub>TX\\_THR\\_EVENT\\_INT](#) interrupt. (R/W)

**RMT\_CH<sub>n</sub>ERR\_INT\_ENA** The interrupt enable bit for the [RMT\\_CH<sub>n</sub>ERROR\\_INT](#) interrupt. (R/W)

**RMT\_CH<sub>n</sub>RX\_END\_INT\_ENA** The interrupt enable bit for the [RMT\\_CH<sub>n</sub>RX\\_END\\_INT](#) interrupt. (R/W)

**RMT\_CH<sub>n</sub>TX\_END\_INT\_ENA** The interrupt enable bit for the [RMT\\_CH<sub>n</sub>TX\\_END\\_INT](#) interrupt. (R/W)

Register 7.6: RMT\_INT\_CLR\_REG (0x00ac)

RMT_CH7_TX_THR_EVENT_INT_CLR	RMT_CH6_TX_THR_EVENT_INT_CLR	RMT_CH5_TX_THR_EVENT_INT_CLR	RMT_CH4_TX_THR_EVENT_INT_CLR	RMT_CH3_TX_THR_EVENT_INT_CLR	RMT_CH2_TX_THR_EVENT_INT_CLR	RMT_CH1_TX_THR_EVENT_INT_CLR	RMT_CH0_TX_THR_EVENT_INT_CLR	RMT_CH7_ERR_INT_CLR	RMT_CH6_ERR_INT_CLR	RMT_CH5_ERR_INT_CLR	RMT_CH4_ERR_INT_CLR	RMT_CH3_ERR_INT_CLR	RMT_CH2_ERR_INT_CLR	RMT_CH1_ERR_INT_CLR	RMT_CH0_ERR_INT_CLR	RMT_CH7_RX_END_INT_CLR	RMT_CH6_RX_END_INT_CLR	RMT_CH5_RX_END_INT_CLR	RMT_CH4_RX_END_INT_CLR	RMT_CH3_RX_END_INT_CLR	RMT_CH2_RX_END_INT_CLR	RMT_CH1_RX_END_INT_CLR	RMT_CH0_RX_END_INT_CLR	RMT_CH7_TX_END_INT_CLR	RMT_CH6_TX_END_INT_CLR	RMT_CH5_TX_END_INT_CLR	RMT_CH4_TX_END_INT_CLR	RMT_CH3_TX_END_INT_CLR	RMT_CH2_TX_END_INT_CLR	RMT_CH1_TX_END_INT_CLR	RMT_CH0_TX_END_INT_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	0	0

**RMT\_CH<sub>n</sub>TX\_THR\_EVENT\_INT\_CLR** Set this bit to clear the [RMT\\_CH<sub>n</sub>TX\\_THR\\_EVENT\\_INT](#) interrupt. (WO)

**RMT\_CH<sub>n</sub>ERR\_INT\_CLR** Set this bit to clear the [RMT\\_CH<sub>n</sub>ERRINT](#) interrupt. (WO)

**RMT\_CH<sub>n</sub>RX\_END\_INT\_CLR** Set this bit to clear the [RMT\\_CH<sub>n</sub>RX\\_END\\_INT](#) interrupt. (WO)

**RMT\_CH<sub>n</sub>TX\_END\_INT\_CLR** Set this bit to clear the [RMT\\_CH<sub>n</sub>TX\\_END\\_INT](#) interrupt. (WO)

**Register 7.7: RMT\_CH $n$ CARRIER\_DUTY\_REG ( $n$ : 0-7) (0x00cc+4\* $n$ )**

RMT_CARRIER_HIGH_CH <sup>n</sup>																RMT_CARRIER_LOW_CH <sup>n</sup>																
31															16	15															0	
0x00040																0x00040																Reset

**RMT\_CARRIER\_HIGH\_CH $n$**  This field is used to configure the carrier wave high level duration (in channel clock periods) for channel  $n$ . (R/W)

**RMT\_CARRIER\_LOW\_CH $n$**  This field is used to configure the carrier wave low level duration (in channel clock periods) for channel  $n$ . (R/W)

**Register 7.8: RMT\_CH $n$ TX\_LIM\_REG ( $n$ : 0-7) (0x00ec+4\* $n$ )**

(reserved)																RMT_TX_LIM_CH <sup>n</sup>														
31																9	8												0	
0x000000																0x080														Reset

**RMT\_TX\_LIM\_CH $n$**  When channel  $n$  sends more entries than specified here, it produces a TX\_THR\_EVENT interrupt. (R/W)

**Register 7.9: RMT\_APB\_CONF\_REG (0x00f0)**

(reserved)																															RMT_MEM_TX_WRAP_EN															
31																															2	1												0		
0x00000000																																													0	Reset

**RMT\_MEM\_TX\_WRAP\_EN** bit enables wraparound mode: when the transmitter of a channel has reached the end of its memory block, it will resume sending at the start of its memory region. (R/W)

## 8. PULSE\_CNT

### 8.1 Introduction

The pulse counter module is designed to count the number of rising and/or falling edges of an input signal. Each pulse counter unit has a 16-bit signed counter register and two channels that can be configured to either increment or decrement the counter. Each channel has a signal input that accepts signal edges to be detected, as well as a control input that can be used to enable or disable the signal input. The inputs have optional filters that can be used to discard unwanted glitches in the signal.

The pulse counter has eight independent units, referred to as PULSE\_CNT\_Un.

### 8.2 Functional Description

#### 8.2.1 Architecture

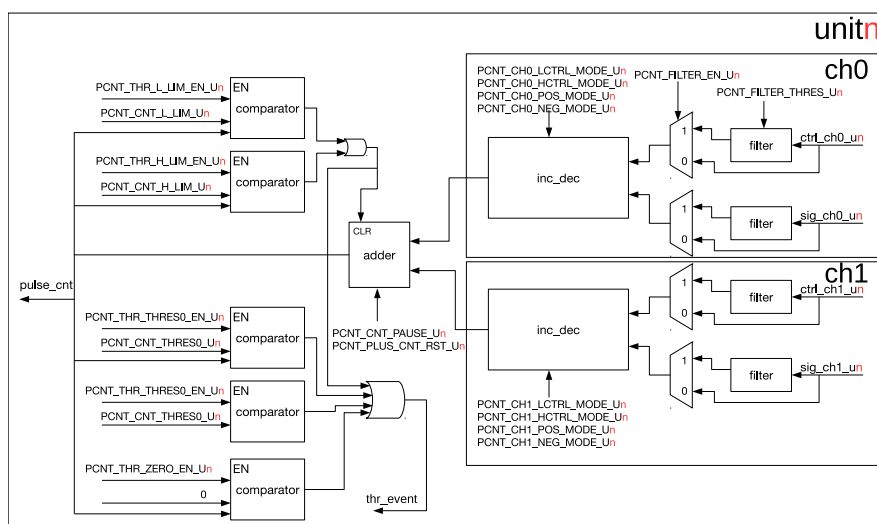


Figure 28: PULSE\_CNT Architecture

The architecture of a pulse counter unit is illustrated in Figure 28. Each unit has two channels: ch0 and ch1, which are functionally equivalent. Each channel has a signal input as well as a control input which can both be connected to I/O pads. The counting behaviour on both positive edge as well as negative edge can be configured separately to increase, decrease or do nothing to the counter value. Separately, for both control signal levels, the hardware can be configured to modify the edge action: invert it, disable it or do nothing. The counter itself is a 16-bit signed up/down counter. Its value can be read by software directly, but is also monitored by a set of comparators which can trigger an interrupt.

#### 8.2.2 Counter Channel Inputs

As stated before, the two inputs of a channel can affect the pulse counter in various ways. The specifics of this behaviour is set by LCTRL\_MODE and HCTRL\_MODE for the case when the control signal is low or high, respectively, and POS\_MODE and NEG\_MODE for positive and negative edges of the input signal. Setting POS\_MODE and NEG\_MODE to 1 will increase the counter when an edge is detected, setting them to 2 will decrease the counter and setting any other value will make the edge not have any effect on the counter. LCTR\_MODE and HCTR\_MODE modify this behaviour as such when the control input has the corresponding low



or high value: 0 does not modify the NEG\_MODE and POS\_MODE behaviour, 1 inverts it (setting POS\_MODE/NEG\_MODE to increase the counter will now decrease the counter and vice versa) and any other value disables counter effects for that signal level.

To summarize, a few examples have been considered. In this table, the effect on the counter for a rising edge is shown for both a low and a high control signal and various configuration options. For clarity, a short description in brackets is added after the values. Note: x denotes 'don't care'.

POS_MODE	LCTRL_MODE	HCTRL_MODE	sig l→h when ctrl=0	sig l→h when ctrl=1
1 (inc)	0 (-)	0 (-)	Inc ctr	Inc ctr
2 (dec)	0 (-)	0 (-)	Dec ctr	Dec ctr
0 (-)	x	x	No action	No action
1 (inc)	0 (-)	1 (inv)	Inc ctr	Dec ctr
1 (inc)	1 (inv)	0 (-)	Dec ctr	Inc ctr
2 (dec)	0 (-)	1 (inv)	Dec ctr	Inc ctr
1 (inc)	0 (-)	2 (dis)	Inc ctr	No action
1 (inc)	2 (dis)	0 (-)	No action	Inc ctr

This table is also valid for negative edges (sig h→l) on substituting NEG\_MODE for POS\_MODE.

Each pulse counter unit also features a filter on each of the four inputs, adding the option to ignore short glitches in the signals. If a PCNT\_FILTER\_EN\_Un can be set to filter the four input signals of the unit. If this filter is enabled, any pulses shorter than REG\_FILTER\_THRES\_Un number of APB\_CLK clock cycles will be filtered out and will have no effect on the counter. With the filter disabled, in theory infinitely small glitches could possibly trigger pulse counter action. However, in practice the signal inputs are sampled on APB\_CLK edges and even with the filter disabled, pulse widths lasting shorter than one APB\_CLK cycle may be missed.

Apart from the input channels, software also has some control over the counter. In particular, the counter value can be frozen to the current value by configuring PCNT\_CNT\_PAUSE\_Un. It can also be reset to zero by configuring PCNT\_PULSE\_CNT\_RST\_Un.

### 8.2.3 Watchpoints

The pulse counters have 5 watchpoints that share one interrupt. Interrupt generation can be enabled or disabled for each individual watchpoint. The watchpoints are:

- Maximum count value: Triggered when  $PULSE\_CNT \geq PCNT\_THR\_H\_LIM\_Un$ . Additionally, this will reset the counter to zero.
- Minimum count value: Triggered when  $PULSE\_CNT \leq PCNT\_THR\_L\_LIM\_Un$ . Additionally, this will reset the counter to zero. This is most useful when  $PCNT\_THR\_L\_LIM\_Un$  is set to a negative number.
- Two threshold values: Triggered when  $PULSE\_CNT = PCNT\_THR\_THRES0\_Un$  or  $PCNT\_THR\_THRES1\_Un$ .
- Zero: Triggered when  $PULSE\_CNT = 0$ .

## 8.2.4 Examples

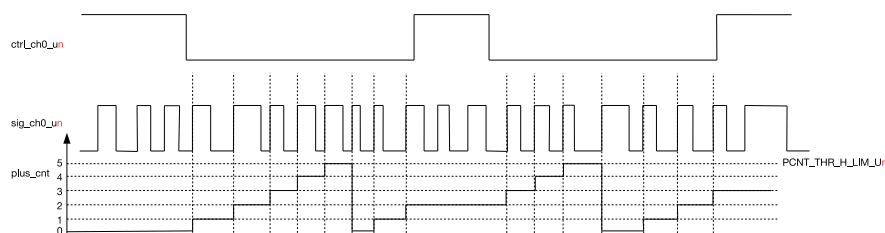


Figure 29: PULSE\_CNT Upcounting Diagram

Figure 29 shows channel 0 being used as an up-counter. The configuration of channel 0 is shown below.

- CNT\_CH0\_POS\_MODE\_Un = 1: increase counter on the rising edge of sig\_ch0\_un.
- PCNT\_CH0\_NEG\_MODE\_Un = 0: no counting on the falling edge of sig\_ch0\_un.
- PCNT\_CH0\_LCTRL\_MODE\_Un = 0: Do not modify counter mode when sig\_ch0\_un is low.
- PCNT\_CH0\_HCTRL\_MODE\_Un = 2: Do not allow counter increments/decrements when sig\_ch0\_un is high.
- PCNT\_THR\_H\_LIM\_Un = 5: PULSE\_CNT resets to 0 when the count value increases to 5.

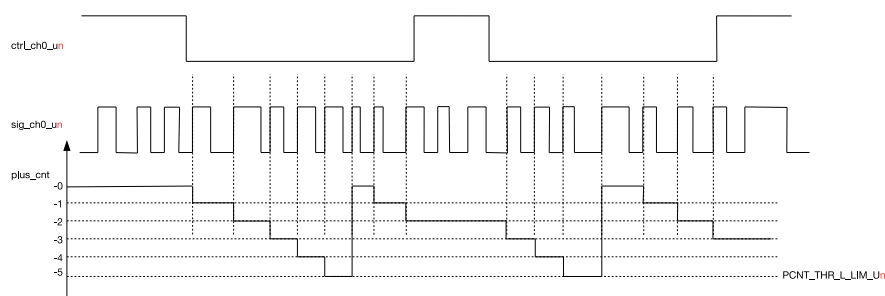


Figure 30: PULSE\_CNT Downcounting Diagram

Figure 30 shows channel 0 decrementing the counter. The configuration of channel 0 differs from that in Figure 29 in the following two aspects:

- PCNT\_CH0\_LCTRL\_MODE\_Un = 1: invert counter mode when ctrl\_ch0\_un is at low level, so it will decrease instead of increase the counter.
- PCNT\_THR\_H\_LIM\_Un = -5: PULSE\_CNT resets to 0 when the count value decreases to -5.

## 8.2.5 Interrupts

PCNT\_CNT\_THR\_EVENT\_Un\_INT: This interrupt gets triggered when one of the five channel comparators detects a match.

## 8.3 Register Summary

Name	Description	Address	Access
<b>Configuration registers</b>			
PCNT_U0_CONF0_REG	Configuration register 0 for unit 0	0x3FF57000	R/W

Name	Description	Address	Access
<a href="#">PCNT_U1_CONF0_REG</a>	Configuration register 0 for unit 1	0x3FF5700C	R/W
<a href="#">PCNT_U2_CONF0_REG</a>	Configuration register 0 for unit 2	0x3FF57018	R/W
<a href="#">PCNT_U3_CONF0_REG</a>	Configuration register 0 for unit 3	0x3FF57024	R/W
<a href="#">PCNT_U4_CONF0_REG</a>	Configuration register 0 for unit 4	0x3FF57030	R/W
<a href="#">PCNT_U5_CONF0_REG</a>	Configuration register 0 for unit 5	0x3FF5703C	R/W
<a href="#">PCNT_U6_CONF0_REG</a>	Configuration register 0 for unit 6	0x3FF57048	R/W
<a href="#">PCNT_U7_CONF0_REG</a>	Configuration register 0 for unit 7	0x3FF57054	R/W
<a href="#">PCNT_U0_CONF1_REG</a>	Configuration register 1 for unit 0	0x3FF57004	R/W
<a href="#">PCNT_U1_CONF1_REG</a>	Configuration register 1 for unit 1	0x3FF57010	R/W
<a href="#">PCNT_U2_CONF1_REG</a>	Configuration register 1 for unit 2	0x3FF5701C	R/W
<a href="#">PCNT_U3_CONF1_REG</a>	Configuration register 1 for unit 3	0x3FF57028	R/W
<a href="#">PCNT_U4_CONF1_REG</a>	Configuration register 1 for unit 4	0x3FF57034	R/W
<a href="#">PCNT_U5_CONF1_REG</a>	Configuration register 1 for unit 5	0x3FF57040	R/W
<a href="#">PCNT_U6_CONF1_REG</a>	Configuration register 1 for unit 6	0x3FF5704C	R/W
<a href="#">PCNT_U7_CONF1_REG</a>	Configuration register 1 for unit 7	0x3FF57058	R/W
<a href="#">PCNT_U0_CONF2_REG</a>	Configuration register 2 for unit 0	0x3FF57008	R/W
<a href="#">PCNT_U1_CONF2_REG</a>	Configuration register 2 for unit 1	0x3FF57014	R/W
<a href="#">PCNT_U2_CONF2_REG</a>	Configuration register 2 for unit 2	0x3FF57020	R/W
<a href="#">PCNT_U3_CONF2_REG</a>	Configuration register 2 for unit 3	0x3FF5702C	R/W
<a href="#">PCNT_U4_CONF2_REG</a>	Configuration register 2 for unit 4	0x3FF57038	R/W
<a href="#">PCNT_U5_CONF2_REG</a>	Configuration register 2 for unit 5	0x3FF57044	R/W
<a href="#">PCNT_U6_CONF2_REG</a>	Configuration register 2 for unit 6	0x3FF57050	R/W
<a href="#">PCNT_U7_CONF2_REG</a>	Configuration register 2 for unit 7	0x3FF5705C	R/W
<b>Counter values</b>			
<a href="#">PCNT_U0_CNT_REG</a>	Counter value for unit 0	0x3FF57060	RO
<a href="#">PCNT_U1_CNT_REG</a>	Counter value for unit 1	0x3FF57064	RO
<a href="#">PCNT_U2_CNT_REG</a>	Counter value for unit 2	0x3FF57068	RO
<a href="#">PCNT_U3_CNT_REG</a>	Counter value for unit 3	0x3FF5706C	RO
<a href="#">PCNT_U4_CNT_REG</a>	Counter value for unit 4	0x3FF57070	RO
<a href="#">PCNT_U5_CNT_REG</a>	Counter value for unit 5	0x3FF57074	RO
<a href="#">PCNT_U6_CNT_REG</a>	Counter value for unit 6	0x3FF57078	RO
<a href="#">PCNT_U7_CNT_REG</a>	Counter value for unit 7	0x3FF5707C	RO
<b>Control registers</b>			
<a href="#">PCNT_CTRL_REG</a>	Control register for all counters	0x3FF570B0	R/W
<b>Interrupt registers</b>			
<a href="#">PCNT_INT_RAW_REG</a>	Raw interrupt status	0x3FF57080	RO
<a href="#">PCNT_INT_ST_REG</a>	Masked interrupt status	0x3FF57084	RO
<a href="#">PCNT_INT_ENA_REG</a>	Interrupt enable bits	0x3FF57088	R/W
<a href="#">PCNT_INT_CLR_REG</a>	Interrupt clear bits	0x3FF5708C	WO

## 8.4 Registers

**Register 8.1: PCNT\_U $n$ \_CONF0\_REG ( $n$ : 0-7) (0x0+0x0C\* $n$ )**

PCNT_CH1_LCTRL_MODE_Un																												PCNT_CH1_HCTRL_MODE_Un																												PCNT_CH1_POS_MODE_Un																												PCNT_CH1_NEG_MODE_Un																												PCNT_CH0_LCTRL_MODE_Un																												PCNT_CH0_HCTRL_MODE_Un																												PCNT_CH0_POS_MODE_Un																												PCNT_CH0_NEG_MODE_Un																												PCNT_THR_THRES1_EN_Un																												PCNT_THR_THRES0_EN_Un																												PCNT_THR_L_LIM_EN_Un																												PCNT_THR_H_LIM_EN_Un																												PCNT_THR_ZERO_EN_Un																												PCNT_FILTER_EN_Un																												PCNT_FILTER_THRES_Un																											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9																	0																																																																																																																																																																																																																																																																																																																																																																																												
0		0		0		0		0		0		0		0		0		1		1		1		1		0x010																Reset																																																																																																																																																																																																																																																																																																																																																																																									

**PCNT\_CH1\_LCTRL\_MODE\_U** This register configures how the CH1\_POS\_MODE/CH1\_NEG\_MODE settings will be modified when the control signal is low. (R/W) 0: No modification; 1: Invert behaviour (increase -> decrease, decrease -> increase); 2, 3: Inhibit counter modification

**PCNT\_CH1\_HCTRL\_MODE\_U** This register configures how the CH1\_POS\_MODE/CH1\_NEG\_MODE settings will be modified when the control signal is low. (R/W) 0: No modification; 1: Invert behaviour (increase -> decrease, decrease -> increase); 2, 3: Inhibit counter modification

**PCNT\_CH1\_POS\_MODE\_U***n* This register sets the behaviour when the signal input of channel 1 detects a positive edge. (R/W) 1: Increment the counter; 2: Decrement the counter; 0, 3: No effect on counter

**PCNT\_CH1\_NEG\_MODE\_U** This register sets the behaviour when the signal input of channel 1 detects a negative edge. (R/W) 1: Increment the counter; 2: Decrement the counter; 0, 3: No effect on counter

**PCNT\_CH0\_LCTRL\_MODE\_U** This register configures how the CH0\_POS\_MODE/CH0\_NEG\_MODE settings will be modified when the control signal is low. (R/W) 0: No modification; 1: Invert behaviour (increase -> decrease, decrease -> increase); 2, 3: Inhibit counter modification

**PCNT\_CH0\_HCTRL\_MODE\_U** This register configures how the CH0\_POS\_MODE/CH0\_NEG\_MODE settings will be modified when the control signal is low. (R/W) 0: No modification; 1: Invert behaviour (increase -> decrease, decrease -> increase); 2, 3: Inhibit counter modification

**PCNT\_CH0\_POS\_MODE\_U***n* This register sets the behaviour when the signal input of channel 0 detects a positive edge. (R/W) 1: Increase the counter; 2: Decrease the counter; 0, 3: No effect on counter

**PCNT\_CH0\_NEG\_MODE\_U** This register sets the behaviour when the signal input of channel 0 detects a negative edge. (R/W) 1: Increase the counter; 2: Decrease the counter; 0, 3: No effect on counter

**PCNT\_THR\_THRES1\_EN** *Un* This is the enable bit for unit *n*'s thres1 comparator. (R/W)

**PCNT\_THR\_THRES0\_EN\_U<sub>n</sub>** This is the enable bit for unit *n*'s thres0 comparator. (R/W)

**PCNT\_THR\_L\_LIM\_EN\_U<sub>*n*</sub>** This is the enable bit for unit *n*'s thr\_l\_lim comparator. (R/W)

**PCNT\_THR\_H\_LIM\_EN\_U<sub>n</sub>** This is the enable bit for unit *n*'s thr\_h\_lim comparator. (R/W)

**PCNT\_THR\_ZERO\_EN\_U<sub>n</sub>** This is the enable bit for unit *n*'s zero comparator. (R/W)

**PCNT\_FILTER\_EN\_U<sub>*n*</sub>** This is the enable bit for unit *n*'s input filter. (R/W)

**PCNT\_FILTER\_THRES\_Un** This sets the maximum threshold, in APB\_CLK cycles, for the filter. Any pulses lasting shorter than this will be ignored when the filter is enabled. (R/W)

**Register 8.2: PCNT\_UN\_CONF1\_REG (*n*: 0-7) (0x4+0x0C\**n*)**

PCNT_CNT_THRES1_UN																PCNT_CNT_THRES0_UN																
31																16	15														0	
0x000																0x000																Reset

**PCNT\_CNT\_THRES1\_UN** This register is used to configure the thres1 value for unit *n*. (R/W)

**PCNT\_CNT\_THRES0\_UN** This register is used to configure the thres0 value for unit *n*. (R/W)

**Register 8.3: PCNT\_UN\_CONF2\_REG (*n*: 0-7) (0x8+0x0C\**n*)**

PCNT_CNT_L_LIM_UN																PCNT_CNT_H_LIM_UN																
31																16	15														0	
0x000																0x000																Reset

**PCNT\_CNT\_L\_LIM\_UN** This register is used to configure the thr\_l\_lim value for unit *n*. (R/W)

**PCNT\_CNT\_H\_LIM\_UN** This register is used to configure the thr\_h\_lim value for unit *n*. (R/W)

**Register 8.4: PCNT\_UN\_CNT\_REG (*n*: 0-7) (0x28+0x0C\**n*)**

(reserved)																PCNT_PLUS_CNT_UN																																															
31																16																15																0															
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																0x00000																Reset																															

**PCNT\_PLUS\_CNT\_UN** This register stores the current pulse count value for unit *n*. (RO)

### Register 8.5: PCNT\_INT\_RAW\_REG (0x0080)

[illegible]

**PCNT\_CNT\_THR\_EVENT\_U<sub>n</sub>INT\_RAW** The raw interrupt status bit for the PCNT CNT\_THR\_EVENT U<sub>n</sub>INT interrupt. (RO)

### Register 8.6: PCNT\_INT\_ST\_REG (0x0084)

[illegible]

**PCNT\_CNT\_THR\_EVENT\_U**<sub>n</sub>**\_INT\_ST** The masked interrupt status bit for the **PCNT\_CNT\_THR\_EVENT\_U**<sub>n</sub>**\_INT** interrupt. (RO)

### Register 8.7: PCNT\_INT\_ENA\_REG (0x0088)

Register 0x00000000 (Reset) bit field diagram:

- Bits 31-8: (reserved)
- Bit 7: PONT\_CNT\_THR\_EVENT\_U7\_INT\_ENA
- Bit 6: PONT\_CNT\_THR\_EVENT\_U6\_INT\_ENA
- Bit 5: PONT\_CNT\_THR\_EVENT\_U5\_INT\_ENA
- Bit 4: PONT\_CNT\_THR\_EVENT\_U4\_INT\_ENA
- Bit 3: PONT\_CNT\_THR\_EVENT\_U3\_INT\_ENA
- Bit 2: PONT\_CNT\_THR\_EVENT\_U2\_INT\_ENA
- Bit 1: PONT\_CNT\_THR\_EVENT\_U1\_INT\_ENA
- Bit 0: PONT\_CNT\_THR\_EVENT\_U0\_INT\_ENA

**PCNT\_CNT\_THR\_EVENT\_U**<sub>n</sub>**\_INT\_ENA** The interrupt enable bit for the **PCNT\_CNT\_THR\_EVENT\_U**<sub>n</sub>**\_INT** interrupt. (R/W)

### Register 8.8: PCNT\_INT\_CLR\_REG (0x008c)

[illegible]

**PCNT\_CNT\_THR\_EVENT\_U<sub>n</sub>INT\_CLR** Set this bit to clear the PCNT\_CNT\_THR\_EVENT\_U<sub>n</sub>INT interrupt. (WO)

### Register 8.9: PCNT\_CTRL\_REG (0x00b0)

[illegible]

**PCNT\_CNT\_PAUSE\_U $n$**  Set this bit to freeze unit  $n$ 's counter. (R/W)

**PCNT\_PLUS\_CNT\_RST\_U<sub>*n*</sub>** Set this bit to clear unit *n*'s counter. (R/W)

## 9. 64-bit Timers

### 9.1 Introduction

There are four general-purpose timers embedded in the ESP32. They are all 64-bit generic timers based on 16-bit prescalers and 64-bit auto-reload-capable up/downcounters.

The ESP32 contains two timer modules, each containing two timers. The two timers in a block are indicated by an *x* in TIMG*n*\_Tx; the blocks themselves are indicated by an *n*.

The timers feature:

- A 16-bit clock prescaler, from 2 to 65536
- A 64-bit time-base counter
- Configurable up/down time-base counter: incrementing or decrementing
- Halt and resume of time-base counter
- Auto-reload at alarm
- Software-controlled instant reload
- Level and edge interrupt generation

### 9.2 Functional Description

#### 9.2.1 16-bit Prescaler

Each timer uses the APB clock (APB\_CLK, normally 80 MHz) as the basic clock. This clock is then divided down by a 16-bit prescaler which generates the time-base counter clock (TB\_clk). Every cycle of TB\_clk causes the time-base counter to increment or decrement by one. The timer must be disabled (TIMG*n*\_Tx\_EN is cleared) before changing the prescaler divisor which is configured by TIMG*n*\_Tx\_DIVIDER register; changing it on an enabled timer can lead to unpredictable results. The prescaler can divide the APB clock by a factor from 2 to 65536. Specifically, when TIMG*n*\_Tx\_DIVIDER is either 1 or 2, the clock divisor is 2; when TIMG*n*\_Tx\_DIVIDER is 0, the clock divisor is 65536. Any other value will cause the clock to be divided by exactly that value.

#### 9.2.2 64-bit Time-base Counter

The 64-bit time-base counter can be configured to count either up or down, dependent on whether TIMG*n*\_Tx\_INCREASE is set or cleared, respectively. It supports both auto-reload and software instant reload. An alarm event can be set to trigger when the counter reaches a value specified by software.

Counting can be enabled and disabled by setting and clearing TIMG*n*\_Tx\_EN. Clearing this bit essentially freezes the counter, causing it to neither count up nor count down, but instead retain its value until TIMG*n*\_Tx\_EN is set again. Reloading the counter when TIMG*n*\_Tx\_EN is cleared will change its value, but counting will not be resumed until TIMG*n*\_Tx\_EN is set.

Software can set a new counter value by setting registers TIMG*n*\_Tx\_LOAD\_LO and TIMG*n*\_Tx\_LOAD\_HI to the intended new value. The hardware will ignore these register settings until a reload; a reload will cause the contents of these registers to be copied to the counter itself. A reload event can be triggered by an alarm (auto-reload at alarm) or by software (software instant reload). To enable auto-reload at alarm, the register



TIMG $n$ \_Tx\_AUTORELOAD should be set. If auto-reload at alarm is not enabled, the time-base counter will continue incrementing or decrementing after the alarm. To trigger a software instant reload, any value can be written to the register TIMG $n$ \_Tx\_LOAD\_REG; this will cause the counter value to change instantaneously. Software can also change the direction of the time-base counter instantaneously by changing the value of TIMG $n$ \_Tx\_INCREASE.

The time-base counter can also be read by software, but because the counter is 64-bit, the CPU can only get the value as two 32-bit values, the counter value needs to be latched into TIMG $n$ \_TxLO\_REG and TIMG $n$ \_TxHI\_REG first. This is done by writing any value to TIMG $n$ \_TxUPDATE\_REG; this will instantly latch the 64-bit timer value to the two registers. Software can then read them at any point of time. This approach stops the timer value being read erroneously when a carry-over happens between reading the low and high word of the timer value.

### 9.2.3 Alarm Generation

The timer can trigger an alarm, which can cause a reload and/or an interrupt to trigger. The alarm triggers when the alarm registers TIMG $n$ \_Tx\_ALARMLO\_REG and TIMG $n$ \_Tx\_ALARMHI\_REG match the current timer value. In order to simplify the scenario where these registers are set 'too late' and the counter already passed these values, the alarm also triggers when the current timer value is higher (for an up-counting timer) or lower (for a down-counting timer) than the current alarm value: if this is the case, the alarm will be triggered immediately upon loading the alarm registers.

### 9.2.4 MWDT

Each timer module also contains a Main System Watchdog Timer and associated registers. While the registers are described here, the functional description can be found in the Chapter [Watchdog Timer](#).

### 9.2.5 Interrupts

- TIMG $n$ \_Tx\_INT\_WDT\_INT: Generated when a watchdog timer interrupt stage times out.
- TIMG $n$ \_Tx\_INT\_T1\_INT: An alarm event on timer 1 generates this interrupt.
- TIMG $n$ \_Tx\_INT\_T0\_INT: An alarm event on timer 0 generates this interrupt.

## 9.3 Register summary

Name	Description	TIMG0	TIMG1	Acc
<b>Timer 0 configuration and control registers</b>				
TIMG $n$ _T0CONFIG_REG	Timer 0 configuration register	0x3FF5F000	0x3FF60000	R/W
TIMG $n$ _T0LO_REG	Timer 0 current value, low 32 bits	0x3FF5F004	0x3FF60004	RO
TIMG $n$ _T0HI_REG	Timer 0 current value, high 32 bits	0x3FF5F008	0x3FF60008	RO
TIMG $n$ _T0UPDATE_REG	Write to copy current timer value to TIMG $n$ _T0_(LO/HI)_REG	0x3FF5F00C	0x3FF6000C	WO
TIMG $n$ _T0ALARMLO_REG	Timer 0 alarm value, low 32 bits	0x3FF5F010	0x3FF60010	R/W
TIMG $n$ _T0ALARMHI_REG	Timer 0 alarm value, high bits	0x3FF5F014	0x3FF60014	R/W
TIMG $n$ _T0LOADLO_REG	Timer 0 reload value, low 32 bits	0x3FF5F018	0x3FF60018	R/W
TIMG $n$ _T0LOAD_REG	Write to reload timer from TIMG $n$ _T0_(LOADLOLOADHI)_REG	0x3FF5F020	0x3FF60020	WO

Name	Description	TIMG0	TIMG1	Acc
<b>Timer 1 configuration and control registers</b>				
<a href="#">TIMG<sub>n</sub>_T1CONFIG_REG</a>	Timer 1 configuration register	0x3FF5F024	0x3FF60024	R/W
<a href="#">TIMG<sub>n</sub>_T1LO_REG</a>	Timer 1 current value, low 32 bits	0x3FF5F028	0x3FF60028	RO
<a href="#">TIMG<sub>n</sub>_T1HI_REG</a>	Timer 1 current value, high 32 bits	0x3FF5F02C	0x3FF6002C	RO
<a href="#">TIMG<sub>n</sub>_T1UPDATE_REG</a>	Write to copy current timer value to TIMG <sub>n</sub> _T1_(LO/HI)_REG	0x3FF5F030	0x3FF60030	WO
<a href="#">TIMG<sub>n</sub>_T1ALARMLO_REG</a>	Timer 1 alarm value, low 32 bits	0x3FF5F034	0x3FF60034	R/W
<a href="#">TIMG<sub>n</sub>_T1ALARMHI_REG</a>	Timer 1 alarm value, high 32 bits	0x3FF5F038	0x3FF60038	R/W
<a href="#">TIMG<sub>n</sub>_T1LOADLO_REG</a>	Timer 1 reload value, low 32 bits	0x3FF5F03C	0x3FF6003C	R/W
<a href="#">TIMG<sub>n</sub>_T1LOAD_REG</a>	Write to reload timer from TIMG <sub>n</sub> _T1_(LOADLOLOADHI)_REG	0x3FF5F044	0x3FF60044	WO
<b>System watchdog timer configuration and control registers</b>				
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG0_REG</a>	Watchdog timer configuration register	0x3FF5F048	0x3FF60048	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG1_REG</a>	Watchdog timer prescaler register	0x3FF5F04C	0x3FF6004C	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG2_REG</a>	Watchdog timer stage 0 timeout value	0x3FF5F050	0x3FF60050	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG3_REG</a>	Watchdog timer stage 1 timeout value	0x3FF5F054	0x3FF60054	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG4_REG</a>	Watchdog timer stage 2 timeout value	0x3FF5F058	0x3FF60058	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTCONFIG5_REG</a>	Watchdog timer stage 3 timeout value	0x3FF5F05C	0x3FF6005C	R/W
<a href="#">TIMG<sub>n</sub>_Tx_WDTFEED_REG</a>	Write to feed the watchdog timer	0x3FF5F060	0x3FF60060	WO
<a href="#">TIMG<sub>n</sub>_Tx_WDTWPROTECT_REG</a>	Watchdog write protect register	0x3FF5F064	0x3FF60064	R/W
<b>Interrupt registers</b>				
<a href="#">TIMG<sub>n</sub>_Tx_INT_RAW_REG</a>	Raw interrupt status	0x3FF5F09C	0x3FF6009C	RO
<a href="#">TIMG<sub>n</sub>_Tx_INT_ST_REG</a>	Masked interrupt status	0x3FF5F0A0	0x3FF600A0	RO
<a href="#">TIMG<sub>n</sub>_Tx_INT_ENA_REG</a>	Interrupt enable bits	0x3FF5F098	0x3FF60098	R/W
<a href="#">TIMG<sub>n</sub>_Tx_INT_CLR_REG</a>	Interrupt clear bits	0x3FF5F0A4	0x3FF600A4	WO

## 9.4 Registers

**Register 9.1: TIMG<sub>n</sub>\_TXCONFIG\_REG (x: 0-1) (0x0+0x24\*x)**

TIMG <sub>n</sub> _TX_EN				TIMG <sub>n</sub> _TX_DIVIDER									TIMG <sub>n</sub> _TX_EDGE_INT_EN			
TIMG <sub>n</sub> _TX_INCREASE													TIMG <sub>n</sub> _TX_LEVEL_INT_EN			
TIMG <sub>n</sub> _TX_AUTORELOAD													TIMG <sub>n</sub> _TX_ALARM_EN			
31	30	29	28										13	12	11	10
0	1	1	0x00001									0	0	0	Reset	

**TIMG<sub>n</sub>\_TX\_EN** When set, the timer *x* time-base counter is enabled. (R/W)

**TIMG<sub>n</sub>\_TX\_INCREASE** When set, the timer *x* time-base counter will increment every clock tick. When cleared, the timer *x* time-base counter will decrement. (R/W)

**TIMG<sub>n</sub>\_TX\_AUTORELOAD** When set, timer *x* auto-reload at alarm is enabled. (R/W)

**TIMG<sub>n</sub>\_TX\_DIVIDER** Timer *x* clock (TX\_clk) prescale value. (R/W)

**TIMG<sub>n</sub>\_TX\_EDGE\_INT\_EN** When set, an alarm will generate an edge type interrupt. (R/W)

**TIMG<sub>n</sub>\_TX\_LEVEL\_INT\_EN** When set, an alarm will generate a level type interrupt. (R/W)

**TIMG<sub>n</sub>\_TX\_ALARM\_EN** When set, the alarm is enabled. (R/W)

**Register 9.2: TIMG<sub>n</sub>\_TXLO\_REG (x: 0-1) (0x4+0x24\*x)**

31																																0	
0x00000000																																	Reset

**TIMG<sub>n</sub>\_TXLO\_REG** After writing to TIMG<sub>n</sub>\_TXUPDATE\_REG, the low 32 bits of the time-base counter of timer *x* can be read here. (RO)

**Register 9.3: TIMG<sub>n</sub>\_TXHI\_REG (x: 0-1) (0x8+0x24\*x)**

31																															0	
0x00000000																																Reset

**TIMG<sub>n</sub>\_TXHI\_REG** After writing to TIMG<sub>n</sub>\_TXUPDATE\_REG, the high 32 bits of the time-base counter of timer *x* can be read here. (RO)

**Register 9.4: TIMG<sub>n</sub>\_TXUPDATE\_REG (x: 0-1) (0xC+0x24\*x)**

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TXUPDATE\_REG** Write any value to trigger a timer *x* time-base counter value update (timer *x* current value will be stored in registers above). (WO)

**Register 9.5: TIMG<sub>n</sub>\_TXALARMLO\_REG (x: 0-1) (0x10+0x24\*x)**

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TXALARMLO\_REG** Timer *x* alarm trigger time-base counter value, low 32 bits. (R/W)

**Register 9.6: TIMG<sub>n</sub>\_TXALARMHI\_REG (x: 0-1) (0x14+0x24\*x)**

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TXALARMHI\_REG** Timer *x* alarm trigger time-base counter value, high 32 bits. (R/W)

**Register 9.7: TIMG<sub>n</sub>\_TXLOADLO\_REG (x: 0-1) (0x18+0x24\*x)**

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TXLOADLO\_REG** Low 32 bits of the value that a reload will load into timer *x* time-base counter. (R/W)

**Register 9.8: TIMG<sub>n</sub>\_TXLOADHI\_REG (x: 0-1) (0x1C+0x24\*x)**

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TXLOADHI\_REG** High 32 bits of the value that a reload will load into timer *x* time-base counter. (R/W)

Register 9.9: TIMG<sub>n</sub>\_TxLOAD\_REG (x: 0-1) (0x20+0x24\*x)

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>\_TxLOAD\_REG** Write any value to trigger a timer x time-base counter reload. (WO)

Register 9.10: TIMG<sub>n</sub>\_Tx\_WDTCONFIG0\_REG (0x0048)

TIMG <sub>n</sub> _Tx_WDT_EN															
TIMG <sub>n</sub> _Tx_WDT_STG0															
TIMG <sub>n</sub> _Tx_WDT_STG1															
TIMG <sub>n</sub> _Tx_WDT_STG2															
TIMG <sub>n</sub> _Tx_WDT_STG3															
TIMG <sub>n</sub> _Tx_WDT_EDGE_INT_EN															
TIMG <sub>n</sub> _Tx_WDT_LEVEL_INT_EN															
TIMG <sub>n</sub> _Tx_WDT_CPU_RESET_LENGTH															
TIMG <sub>n</sub> _Tx_WDT_SYS_RESET_LENGTH															
TIMG <sub>n</sub> _Tx_WDT_FLASHBOOT_MOD_EN															
31	30	29	28	27	26	25	24	23	22	21	20	18	17	15	14
0	0	0	0	0	0	0	0	0	0	0x1		0x1		1	Reset

**TIMG<sub>n</sub>\_Tx\_WDT\_EN** When set, MWDAT is enabled. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_STG0** Stage 0 configuration. 0: off, 1: interrupt, 2: reset CPU, 3: reset system. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_STG1** Stage 1 configuration. 0: off, 1: interrupt, 2: reset CPU, 3: reset system. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_STG2** Stage 2 configuration. 0: off, 1: interrupt, 2: reset CPU, 3: reset system. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_STG3** Stage 3 configuration. 0: off, 1: interrupt, 2: reset CPU, 3: reset system. (R/W)

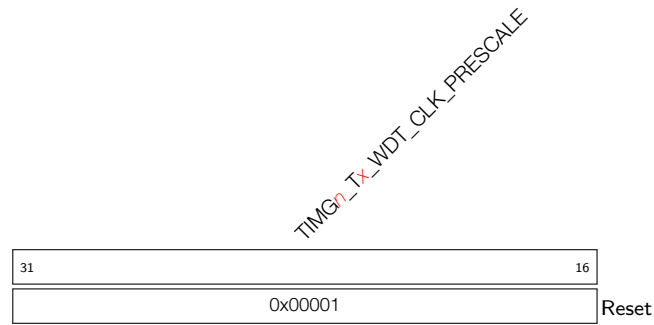
**TIMG<sub>n</sub>\_Tx\_WDT\_EDGE\_INT\_EN** When set, an edge type interrupt will be generated on timeout of a stage configured to generate an interrupt. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_LEVEL\_INT\_EN** When set, a level type interrupt will be generated on timeout of a stage configured to generate an interrupt. (R/W)

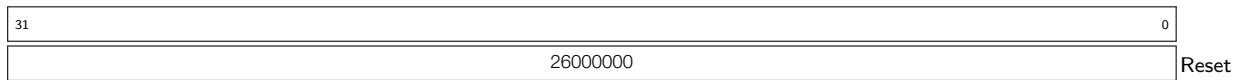
**TIMG<sub>n</sub>\_Tx\_WDT\_CPU\_RESET\_LENGTH** CPU reset signal length selection. 0: 100 ns, 1: 200 ns, 2: 300 ns, 3: 400 ns, 4: 500 ns, 5: 800 ns, 6: 1.6  $\mu$ s, 7: 3.2  $\mu$ s. (R/W)

**TIMG<sub>n</sub>\_Tx\_WDT\_SYS\_RESET\_LENGTH** System reset signal length selection. 0: 100 ns, 1: 200 ns, 2: 300 ns, 3: 400 ns, 4: 500 ns, 5: 800 ns, 6: 1.6  $\mu$ s, 7: 3.2  $\mu$ s. (R/W)

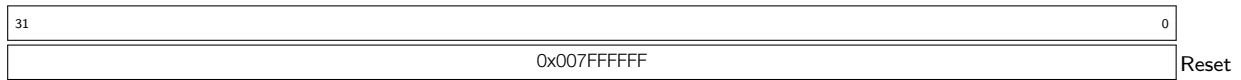
**TIMG<sub>n</sub>\_Tx\_WDT\_FLASHBOOT\_MOD\_EN** When set, Flash boot protection is enabled. (R/W)

**Register 9.11: TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG1\_REG (0x004c)**

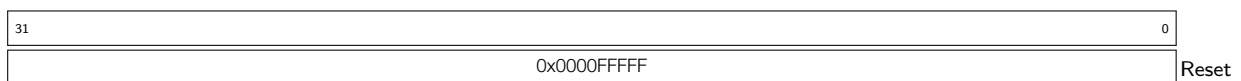
**TIMG<sub>n</sub>T<sub>x</sub>WDT\_CLK\_PRESCALE** MWDT clock prescale value. MWDT clock period = 12.5 ns \* TIMG<sub>n</sub>T<sub>x</sub>WDT\_CLK\_PRESCALE. (R/W)

**Register 9.12: TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG2\_REG (0x0050)**

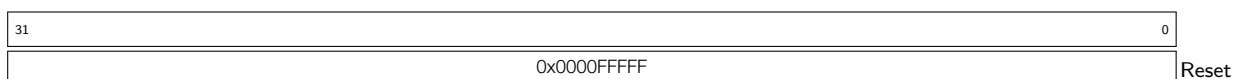
**TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG2\_REG** Stage 0 timeout value, in MWDT clock cycles. (R/W)

**Register 9.13: TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG3\_REG (0x0054)**

**TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG3\_REG** Stage 1 timeout value, in MWDT clock cycles. (R/W)

**Register 9.14: TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG4\_REG (0x0058)**

**TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG4\_REG** Stage 2 timeout value, in MWDT clock cycles. (R/W)

**Register 9.15: TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG5\_REG (0x005c)**

**TIMG<sub>n</sub>T<sub>x</sub>WDTCONFIG5\_REG** Stage 3 timeout value, in MWDT clock cycles. (R/W)

Register 9.16: TIMG<sub>n</sub>T<sub>x</sub>WDTFEED\_REG (0x0060)

31	0
0x00000000	
Reset	

**TIMG<sub>n</sub>T<sub>x</sub>WDTFEED\_REG** Write any value to feed the MWDT. (WO)

Register 9.17: TIMG<sub>n</sub>T<sub>x</sub>WDTWPROTECT\_REG (0x0064)

31	0
0x050D83AA1	
Reset	

**TIMG<sub>n</sub>T<sub>x</sub>WDTWPROTECT\_REG** If the register contains a different value than its reset value, write protection is enabled. (R/W)

Register 9.18: TIMG<sub>n</sub>T<sub>x</sub>INT\_ENA\_REG (0x0098)

(reserved)																																TIMG <sub>n</sub> T <sub>x</sub> INT_WDT_INT_ENA TIMG <sub>n</sub> T <sub>x</sub> INT_T1_INT_ENA TIMG <sub>n</sub> T <sub>x</sub> INT_T0_INT_ENA			
31																															3	2	1	0	Reset
0 0																															0	0	0	0	

**TIMG<sub>n</sub>T<sub>x</sub>INT\_WDT\_INT\_ENA** The interrupt enable bit for the TIMG<sub>n</sub>T<sub>x</sub>INT\_WDT\_INT interrupt. (R/W) (R/W)

**TIMG<sub>n</sub>T<sub>x</sub>INT\_T1\_INT\_ENA** The interrupt enable bit for the TIMG<sub>n</sub>T<sub>x</sub>INT\_T1\_INT interrupt. (R/W) (R/W)

**TIMG<sub>n</sub>T<sub>x</sub>INT\_T0\_INT\_ENA** The interrupt enable bit for the TIMG<sub>n</sub>T<sub>x</sub>INT\_T0\_INT interrupt. (R/W) (R/W)

TIMG\_T1\_INT\_WDT\_INT\_RAW  
TIMG\_T1\_INT\_T1\_INT\_RAW  
TIMG\_T1\_INT\_T0\_INT\_RAW

TIMG\_T1\_INT\_WDT\_INT\_RAW  
TIMG\_T1\_INT\_T1\_INT\_RAW  
TIMG\_T1\_INT\_T0\_INT\_RAW

**TIMG<sub>n</sub>\_Tx\_INT\_T0\_INT\_RAW** The raw interrupt status bit for the TIMG<sub>n</sub>\_Tx\_INT\_T0\_INT interrupt.  
(RO)

TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~W~~ - ~~D~~ - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~  
TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~  
TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~T~~ - ~~O~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~

TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~W~~ - ~~D~~ - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~  
TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~  
TIMG - ~~T~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~T~~ - ~~O~~ - ~~I~~ - ~~N~~ - ~~T~~ - ~~S~~ - ~~T~~

**TIMG<sub>n</sub>\_Tx\_INT\_TO\_INT\_ST** The masked interrupt status bit for the TIMG<sub>n</sub>\_Tx\_INT\_TO\_INT interrupt.  
(RO)





## 10. Watchdog Timers

### 10.1 Introduction

The ESP32 has three watchdog timers: one in each of the two timer modules (called Main System Watchdog Timer, or MWDT) and one in the RTC module (which is called the RTC Watchdog Timer, or RWDT). These watchdog timers are intended to recover from an unforeseen fault, causing the application program to abandon its normal sequence. A watchdog timer has 4 stages. Each stage may take one of three or four actions on expiry of a programmed time period for this stage unless the watchdog is fed or disabled. The actions are: interrupt, CPU reset, core reset and system reset. Only the RWDT can trigger the system reset, and is able to reset the entire chip and the main system, including the RTC itself. A timeout value can be set for each stage individually.

During flash boot the RWDT and the first MWDT start automatically in order to detect and recover from booting problems.

### 10.2 Features

- 4 stages, each can be configured or disabled separately
- Programmable time period for each stage
- One of 3 or 4 possible actions (interrupt, CPU reset, core reset and system reset) on expiration of each stage
- 32-bit expiry counter
- Write protection, to prevent the RWDT and MWDT configuration from being inadvertently altered.
- Flash boot protection  
If the boot process from an SPI flash does not complete within a predetermined time period, the watchdog will reboot the entire main system.

### 10.3 Functional Description

#### 10.3.1 Clock

The RWDT is clocked from the RTC slow clock, which usually will be 32 KHz. The MWDT clock source is derived from the APB clock via a pre-MWDT 16-bit configurable prescaler. For either watchdog, the clock source is fed into the 32-bit expiry counter. When this counter reaches the timeout value of the current stage, the action configured for the stage will execute, the expiry counter will be reset and the next stage will be made active.

### 10.3.1.1 Operating Procedure

When a watchdog timer is enabled, it will proceed in loops from stage 0 to stage 3, then back to stage 0 and start again. The expiry action and time period for each stage can be configured individually.

Every stage can be configured for one of the following actions when the expiry timer reaches the stages timeout value:

- Trigger an interrupt  
When the stage expires an interrupt is triggered.
- Reset a CPU core  
When the stage expires the designated CPU core will be reset. MWDT0 CPU reset only resets the PRO CPU. MWDT1 CPU reset only resets the APP CPU. The RWDT CPU reset can reset either or both or none depending on configuration.
- Reset the main system  
When the stage expires, the main system, including the MWDTs, will be reset. In this article, main system refers to CPU and all peripherals. The RTC is an exception to this and it will not be reset.
- Reset the main system and RTC  
When the stage expires the main system and the RTC will both be reset. This action is only available in the RWDT.
- Disabled  
This stage will have no effects on the system.

When software feeds the the watchdog timer, it returns to stage 0 and its expiry counter restarts from 0.

### 10.3.1.2 Write Protection

Both the MWDTs as well as the RWDT can be protected from accidental writing. To accomplish this, they have a write-key register (TIMERS\_WDT\_WKEY for the MWDT, RTC\_CNTL\_WDT\_WKEY for the RWDT.) On reset, these registers are initialized to the value 0x50D83AA1. When the value in this register is changed from 0x50D83AA1, write protection is enabled. Writes to any WDT register, including the feeding register (but excluding the write-key register itself), are ignored. The recommended procedure for accessing a WDT is:

1. Disable the write protection
2. Make the required modification or feed the watchdog
3. Re-enable the write protection

### 10.3.1.3 Flash Boot Protection

During flash booting, the MWDT in timer group 0 ([TIMG0](#)) as well as the RWDT are automatically enabled. Stage 0 for the enabled MWDT is automatically configured to reset the system upon expiry; stage 0 for the RWDT resets the RTC when it expires. After booting, the register TIMERS\_WDT\_FLASHBOOT\_MOD\_EN should be cleared to stop the flash boot protection procedure for the MWDT, and RTC\_CNTL\_WDT\_FLASHBOOT\_MOD\_EN should be cleared to do the same for the RWDT. After this, the MWDT and RWDT can be configured by software.

#### 10.3.1.4 Registers

The MWDT registers are part of the timer submodule and are described in the [Timer Registers](#) section. The RWDT registers are part of the RTC submodule and are described in the RTC Registers section.

## 11. AES Accelerator

### 11.1 Introduction

The AES Accelerator speeds up AES operations significantly compared to AES algorithms implemented solely in software. The AES Accelerator supports 6 algorithms of FIPS PUB 197, specifically AES-128, AES-192 and AES-256 encryption and decryption.

### 11.2 Features

- Supports AES-128 encryption and decryption
- Supports AES-192 encryption and decryption
- Supports AES-256 encryption and decryption
- Supports 4 variations of key endianness and 4 variations of text endianness

### 11.3 Functional Description

#### 11.3.1 AES Algorithm Operations

The AES Accelerator supports 6 algorithms of FIPS PUB 197, specifically AES-128, AES-192 and AES-256 encryption and decryption. The AES\_MODE\_REG register can be configured to different values to enable different algorithm operations, as is shown in Table 27.

**Table 27: Operation Mode**

AES_MODE_REG[2:0]	Operation
0	AES-128 Encryption
1	AES-192 Encryption
2	AES-256 Encryption
4	AES-128 Decryption
5	AES-192 Decryption
6	AES-256 Decryption

#### 11.3.2 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/decryption, the 128-bit key is stored in AES\_KEY\_0\_REG ~ AES\_KEY\_3\_REG. For AES-192 encryption/decryption, the 192-bit key is stored in AES\_KEY\_0\_REG ~ AES\_KEY\_5\_REG. For AES-256 encryption/decryption, the 256-bit key is stored in AES\_KEY\_0\_REG ~ AES\_KEY\_7\_REG.

Plaintext and ciphertext is stored in the AES\_TEXT\_*m*\_REG registers. There are four 32-bit registers. To enable AES-128/192/256 encryption, initialize the AES\_TEXT\_*m*\_REG registers with plaintext before encryption. When encryption is finished, the AES Accelerator will store back the resulting ciphertext in the AES\_TEXT\_*m*\_REG registers. To enable AES-128/192/256 decryption, initialize the AES\_TEXT\_*m*\_REG registers with ciphertext before decryption. When decryption is finished, the AES Accelerator will store back the resulting plaintext in the AES\_TEXT\_*m*\_REG registers.

### 11.3.3 Endianness

#### Key Endianness

Bit 0 and bit 1 in AES\_ENDIAN\_REG define the key endianness. For detailed information, please refer to Table 29, Table 30 and Table 31.  $w[0] \sim w[3]$  in Table 29,  $w[0] \sim w[5]$  in Table 30 and  $w[0] \sim w[7]$  in Table 31 are “the first Nk words of the expanded key” as specified in “5.2: Key Expansion” of FIPS PUB 197. “Column Bit” specifies the bytes in the word from  $w[0]$  to  $w[7]$ . The bytes of AES\_KEY\_ $n$ \_REG comprise “the first Nk words of the expanded key”.

#### Text Endianness

Bit 2 and bit 3 in AES\_ENDIAN\_REG define the endianness of input text, while Bit 4 and Bit 5 define the endianness of output text. The input text refers to the plaintext in AES-128/192/256 encryption and the ciphertext in decryption. The output text refers to the ciphertext in AES-128/192/256 encryption and the plaintext in decryption. For details, please refer to Table 28. “State” in Table 28 is defined as that in “3.4: The State” of FIPS PUB 197: “The AES algorithm operations are performed on a two-dimensional array of bytes called the State”. The ciphertext or plaintexts stored in each byte of AES\_TEXT\_ $m$ \_REG comprise the State.

**Table 28: AES Text Endianness**

AES_ENDIAN_REG[3]/[5]	AES_ENDIAN_REG[2]/[4]	Plaintext/Ciphertext			
0	0	State		c	
		r	0	1	2
			AES_TEXT_3_REG[31:24]	AES_TEXT_2_REG[31:24]	AES_TEXT_1_REG[31:24]
			AES_TEXT_3_REG[23:16]	AES_TEXT_2_REG[23:16]	AES_TEXT_1_REG[23:16]
			AES_TEXT_3_REG[15:8]	AES_TEXT_2_REG[15:8]	AES_TEXT_1_REG[15:8]
0	1	State		c	
		r	0	1	2
			AES_TEXT_3_REG[7:0]	AES_TEXT_2_REG[7:0]	AES_TEXT_1_REG[7:0]
			AES_TEXT_3_REG[15:8]	AES_TEXT_2_REG[15:8]	AES_TEXT_1_REG[15:8]
			AES_TEXT_3_REG[23:16]	AES_TEXT_2_REG[23:16]	AES_TEXT_1_REG[23:16]
1	0	State		c	
		r	0	1	2
			AES_TEXT_0_REG[31:24]	AES_TEXT_1_REG[31:24]	AES_TEXT_2_REG[31:24]
			AES_TEXT_0_REG[23:16]	AES_TEXT_1_REG[23:16]	AES_TEXT_2_REG[23:16]
			AES_TEXT_0_REG[15:8]	AES_TEXT_1_REG[15:8]	AES_TEXT_2_REG[15:8]
1	1	State		c	
		r	0	1	2
			AES_TEXT_0_REG[7:0]	AES_TEXT_1_REG[7:0]	AES_TEXT_2_REG[7:0]
			AES_TEXT_0_REG[15:8]	AES_TEXT_1_REG[15:8]	AES_TEXT_2_REG[15:8]
			AES_TEXT_0_REG[23:16]	AES_TEXT_1_REG[23:16]	AES_TEXT_2_REG[23:16]



### 11.3.4 Encryption and Decryption Operations

#### Single Operation

1. Initialize AES\_MODE\_REG, AES\_KEY\_*n*\_REG, AES\_TEXT\_*m*\_REG and AES\_ENDIAN\_REG.
2. Write 1 to AES\_START\_REG.
3. Wait until AES\_IDLE\_REG reads 1.
4. Read results from AES\_TEXT\_*m*\_REG.

#### Consecutive Operations

Every time when an operation is completed, only AES\_TEXT\_*m*\_REG is modified by the AES Accelerator. Initialization can be therefore be simplified in a series of consecutive operations.

1. Update contents of AES\_MODE\_REG, AES\_KEY\_*n*\_REG and AES\_ENDIAN\_REG if required.
2. Load AES\_TEXT\_*m*\_REG.
3. Write 1 to AES\_START\_REG.
4. Wait until AES\_IDLE\_REG reads 1.
5. Read results from AES\_TEXT\_*m*\_REG.

### 11.3.5 Speed

The AES Accelerator requires 11 to 15 clock cycles to encrypt a message block and 21 or 22 clock cycles to decrypt a message block.

## 11.4 Register summary

Name	Description	Address	Access
<b>Configuration registers</b>			
<a href="#">AES_MODE_REG</a>	Mode of operation of the AES Accelerator	0x3FF01008	R/W
<a href="#">AES_ENDIAN_REG</a>	Endianness configuration register	0x3FF01040	R/W
<b>Key registers</b>			
<a href="#">AES_KEY_0_REG</a>	AES key material register 0	0x3FF01010	R/W
<a href="#">AES_KEY_1_REG</a>	AES key material register 1	0x3FF01014	R/W
<a href="#">AES_KEY_2_REG</a>	AES key material register 2	0x3FF01018	R/W
<a href="#">AES_KEY_3_REG</a>	AES key material register 3	0x3FF0101C	R/W
<a href="#">AES_KEY_4_REG</a>	AES key material register 4	0x3FF01020	R/W
<a href="#">AES_KEY_5_REG</a>	AES key material register 5	0x3FF01024	R/W
<a href="#">AES_KEY_6_REG</a>	AES key material register 6	0x3FF01028	R/W
<a href="#">AES_KEY_7_REG</a>	AES key material register 7	0x3FF0102C	R/W
<b>Encrypted/decrypted data registers</b>			
<a href="#">AES_TEXT_0_REG</a>	AES encrypted/decrypted data register 0	0x3FF01030	R/W
<a href="#">AES_TEXT_1_REG</a>	AES encrypted/decrypted data register 1	0x3FF01034	R/W
<a href="#">AES_TEXT_2_REG</a>	AES encrypted/decrypted data register 2	0x3FF01038	R/W
<a href="#">AES_TEXT_3_REG</a>	AES encrypted/decrypted data register 3	0x3FF0103C	R/W
<b>Control/status registers</b>			



Name	Description	Address	Access
<a href="#">AES_START_REG</a>	AES operation start control register	0x3FF01000	WO
<a href="#">AES_IDLE_REG</a>	AES idle status register	0x3FF01004	RO

## 11.5 Registers

**Register 11.1: AES\_START\_REG (0x000)**

(reserved)															AES_START	
31														1	0	Reset
0x00000000															x	

**AES\_START** Write 1 to start the AES operation. (WO)

**Register 11.2: AES\_IDLE\_REG (0x004)**

(reserved)																															AES_IDLE	
31																															1	0
0x00000000																															1	Reset

**AES\_IDLE** AES Idle register. Reads 0 while the AES Accelerator is busy processing, 1 otherwise. (RO)

**Register 11.3: AES\_MODE\_REG (0x008)**

(reserved)															AES_MODE		
31														3	2	0	Reset
0x00000000															0		

**AES\_MODE** Selects the AES accelerator mode of operation. See Table 27 for details. (R/W)

**Register 11.4: AES\_KEY\_*n*\_REG (*n*: 0-7) (0x10+4\**n*)**

31																0	Reset
0x00000000																	

**AES\_KEY\_*n*\_REG (*n*: 0-7)** AES key material register. (R/W)

**Register 11.5: AES\_TEXT\_*m*\_REG (*m*: 0-3) (0x30+4\**m*)**

31																0	Reset
0x00000000																	

**AES\_TEXT\_*m*\_REG (*m*: 0-3)** Plaintext and ciphertext register. (R/W)

Register 11.6: AES\_ENDIAN\_REG (0x040)

(reserved)											AES_ENDIAN					
31						6	5					0				
0x0000000							1	1	1	1	1	1	Reset			

**AES\_ENDIAN** Endianness selection register. See Table 28 for details. (R/W)

## 12. SHA Accelerator

### 12.1 Introduction

The SHA Accelerator is included to speed up SHA hashing operations significantly compared to SHA hashing algorithms implemented solely in software. The SHA Accelerator supports 4 algorithms of FIPS PUB 180-4, specifically SHA-1, SHA-256, SHA-384 and SHA-512.

### 12.2 Features

Hardware support for popular secure hashing algorithms:

- SHA-1
- SHA-256
- SHA-384
- SHA-512

### 12.3 Functional Description

#### 12.3.1 Padding and Parsing the Message

The SHA Accelerator can only accept one message block at a time. Software divides the message into blocks according to “5.2 Parsing the Message” in FIPS PUB 180-4 and writes one block to the SHA\_TEXT\_*n*\_REG registers at a time. For SHA-1 and SHA-256, software writes a 512-bit message block to SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_15\_REG each time. For SHA-384 and SHA-512, software writes a 1024-bit message block to SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_31\_REG each time.

The SHA Accelerator is unable to perform the padding operation of “5.1 Padding the Message” in FIPS PUB 180-4; Note that the user software is expected to pad the message before feeding it into the accelerator.

As described in “2.2.1: Parameters” in FIPS PUB 180-4, “ $M_0^{(i)}$  is the left-most word of message block  $i$ ”.  $M_0^{(i)}$  is stored in SHA\_TEXT\_0\_REG. In the same fashion, the SHA\_TEXT\_1\_REG register stores the second left-most word of a message block  $H_1^{(N)}$ , etc.

#### 12.3.2 Message Digest

When the hashing operation is finished, the message digest will be refreshed by SHA Accelerator and will be stored in SHA\_TEXT\_*n*\_REG. SHA-1 produces a 160-bit message digest and stores it in SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_4\_REG. SHA-256 produces a 256-bit message digest and stores it in SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_7\_REG. SHA-384 produces a 384-bit message digest and stores it in SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_11\_REG. SHA-512 produces a 512-bit message digest and stores it in SHA\_TEXT\_0\_REG ~ SHA\_TEXT\_15\_REG.

As is described in “2.2.1 Parameters” in FIPS PUB 180-4, “ $H^{(N)}$  is the final hash value and is used to determine the message digest” and “ $H_0^{(i)}$  is the left-most word of hash value  $i$ ”, so the left-most word  $H_0^{(N)}$  in the message digest is stored in SHA\_TEXT\_0\_REG. In the same fashion, the second left-most word  $H_1^{(N)}$  in the message digest is stored in SHA\_TEXT\_1\_REG, etc.

### 12.3.3 Hash Operation

There is a set of control registers for SHA-1, SHA-256, SHA-384 and SHA-512 respectively; different hashing algorithms use different control registers.

SHA-1 uses SHA\_SHA1\_START\_REG, SHA\_SHA1\_CONTINUE\_REG, SHA\_SHA1\_LOAD\_REG and SHA\_SHA1\_BUSY\_REG.

SHA-256 uses SHA\_SHA256\_START\_REG, SHA\_SHA256\_CONTINUE\_REG, SHA\_SHA256\_LOAD\_REG and SHA\_SHA256\_BUSY\_REG. SHA-384 uses SHA\_SHA384\_START\_REG, SHA\_SHA384\_CONTINUE\_REG, SHA\_SHA384\_LOAD\_REG and SHA\_SHA384\_BUSY\_REG.

SHA-512 uses SHA\_SHA512\_START\_REG, SHA\_SHA512\_CONTINUE\_REG, SHA\_SHA512\_LOAD\_REG and SHA\_SHA512\_BUSY\_REG. For detailed operation, please refer to the following steps.

1. Feed the accelerator the first message block:
  - (a) Use the first message block to initialize SHA\_TEXT\_*n*\_REG.
  - (b) Write 1 to SHA\_*X*\_START\_REG.
  - (c) Wait for SHA\_*X*\_BUSY\_REG to read 0, indicating that operation is completed.
2. Similarly, feed the accelerator with subsequent message blocks:
  - (a) Initialize SHA\_TEXT\_*n*\_REG using the subsequent message block.
  - (b) Write 1 to SHA\_*X*\_CONTINUE\_REG.
  - (c) Wait for SHA\_*X*\_BUSY\_REG to read 0, indicating that operation is completed.
3. Get message digest:
  - (a) Write 1 to SHA\_*X*\_LOAD\_REG.
  - (b) Wait for SHA\_*X*\_BUSY\_REG to read 0, indicating that operation is completed.
  - (c) Read message digest from SHA\_TEXT\_*n*\_REG.

### 12.3.4 Speed

The SHA Accelerator requires 60 to 100 clock cycles to process a message block and 8 to 20 clock cycles to calculate the final digest.

## 12.4 Register Summary

Name	Description	Address	Access
<b>Encrypted/decrypted data registers</b>			
SHA_TEXT_0_REG	SHA encrypted/decrypted data register 0	0x3FF03000	R/W
SHA_TEXT_1_REG	SHA encrypted/decrypted data register 1	0x3FF03004	R/W
SHA_TEXT_2_REG	SHA encrypted/decrypted data register 2	0x3FF03008	R/W
SHA_TEXT_3_REG	SHA encrypted/decrypted data register 3	0x3FF0300C	R/W
SHA_TEXT_4_REG	SHA encrypted/decrypted data register 4	0x3FF03010	R/W
SHA_TEXT_5_REG	SHA encrypted/decrypted data register 5	0x3FF03014	R/W
SHA_TEXT_6_REG	SHA encrypted/decrypted data register 6	0x3FF03018	R/W
SHA_TEXT_7_REG	SHA encrypted/decrypted data register 7	0x3FF0301C	R/W

Name	Description	Address	Access
SHA_TEXT_8_REG	SHA encrypted/decrypted data register 8	0x3FF03020	R/W
SHA_TEXT_9_REG	SHA encrypted/decrypted data register 9	0x3FF03024	R/W
SHA_TEXT_10_REG	SHA encrypted/decrypted data register 10	0x3FF03028	R/W
SHA_TEXT_11_REG	SHA encrypted/decrypted data register 11	0x3FF0302C	R/W
SHA_TEXT_12_REG	SHA encrypted/decrypted data register 12	0x3FF03030	R/W
SHA_TEXT_13_REG	SHA encrypted/decrypted data register 13	0x3FF03034	R/W
SHA_TEXT_14_REG	SHA encrypted/decrypted data register 14	0x3FF03038	R/W
SHA_TEXT_15_REG	SHA encrypted/decrypted data register 15	0x3FF0303C	R/W
SHA_TEXT_16_REG	SHA encrypted/decrypted data register 16	0x3FF03040	R/W
SHA_TEXT_17_REG	SHA encrypted/decrypted data register 17	0x3FF03044	R/W
SHA_TEXT_18_REG	SHA encrypted/decrypted data register 18	0x3FF03048	R/W
SHA_TEXT_19_REG	SHA encrypted/decrypted data register 19	0x3FF0304C	R/W
SHA_TEXT_20_REG	SHA encrypted/decrypted data register 20	0x3FF03050	R/W
SHA_TEXT_21_REG	SHA encrypted/decrypted data register 21	0x3FF03054	R/W
SHA_TEXT_22_REG	SHA encrypted/decrypted data register 22	0x3FF03058	R/W
SHA_TEXT_23_REG	SHA encrypted/decrypted data register 23	0x3FF0305C	R/W
SHA_TEXT_24_REG	SHA encrypted/decrypted data register 24	0x3FF03060	R/W
SHA_TEXT_25_REG	SHA encrypted/decrypted data register 25	0x3FF03064	R/W
SHA_TEXT_26_REG	SHA encrypted/decrypted data register 26	0x3FF03068	R/W
SHA_TEXT_27_REG	SHA encrypted/decrypted data register 27	0x3FF0306C	R/W
SHA_TEXT_28_REG	SHA encrypted/decrypted data register 28	0x3FF03070	R/W
SHA_TEXT_29_REG	SHA encrypted/decrypted data register 29	0x3FF03074	R/W
SHA_TEXT_30_REG	SHA encrypted/decrypted data register 30	0x3FF03078	R/W
SHA_TEXT_31_REG	SHA encrypted/decrypted data register 31	0x3FF0307C	R/W
<b>Control/status registers</b>			
SHA_SHA1_START_REG	Control register to initiate SHA1 operation	0x3FF03080	WO
SHA_SHA1_CONTINUE_REG	Control register to continue SHA1 operation	0x3FF03084	WO
SHA_SHA1_LOAD_REG	Control register to calculate the final SHA1 hash	0x3FF03088	WO
SHA_SHA1_BUSY_REG	Status register for SHA1 operation	0x3FF0308C	RO
SHA_SHA256_START_REG	Control register to initiate SHA256 operation	0x3FF03090	WO
SHA_SHA256_CONTINUE_REG	Control register to continue SHA256 operation	0x3FF03094	WO
SHA_SHA256_LOAD_REG	Control register to calculate the final SHA256 hash	0x3FF03098	WO
SHA_SHA256_BUSY_REG	Status register for SHA256 operation	0x3FF0309C	RO
SHA_SHA384_START_REG	Control register to initiate SHA384 operation	0x3FF030A0	WO
SHA_SHA384_CONTINUE_REG	Control register to continue SHA384 operation	0x3FF030A4	WO
SHA_SHA384_LOAD_REG	Control register to calculate the final SHA384 hash	0x3FF030A8	WO
SHA_SHA384_BUSY_REG	Status register for SHA384 operation	0x3FF030AC	RO
SHA_SHA512_START_REG	Control register to initiate SHA512 operation	0x3FF030B0	WO
SHA_SHA512_CONTINUE_REG	Control register to continue SHA512 operation	0x3FF030B4	WO
SHA_SHA512_LOAD_REG	Control register to calculate the final SHA512 hash	0x3FF030B8	WO
SHA_SHA512_BUSY_REG	Status register for SHA512 operation	0x3FF030BC	RO

## 12.5 Registers

**Register 12.1: SHA\_TEXT\_***n***\_REG (*n*: 0-31) (0x0+4\**n*)**

31	0
0x00000000	
Reset	

**SHA\_TEXT\_***n***\_REG (*n*: 0-31)** SHA Message block and hash result register. (R/W)

**Register 12.2: SHA\_SHA1\_START\_REG (0x080)**

31	1	0
(reserved)		
0x00000000		
Reset		

**SHA\_SHA1\_START** Write 1 to start an SHA-1 operation on the first message block. (WO)

**Register 12.3: SHA\_SHA1\_CONTINUE\_REG (0x084)**

31	1	0
(reserved)		
0x00000000		
Reset		

**SHA\_SHA1\_CONTINUE** Write 1 to continue the SHA-1 operation with subsequent blocks. (WO)

**Register 12.4: SHA\_SHA1\_LOAD\_REG (0x088)**

31	1	0
(reserved)		
0x00000000		
Reset		

**SHA\_SHA1\_LOAD** Write 1 to finish the SHA-1 operation to calculate the final the message hash.  
(WO)

**Register 12.5: SHA\_SHA1\_BUSY\_REG (0x08C)**

(reserved)		SHA_SHA1_BUSY	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA1\_BUSY** SHA-1 operation status: 1 if the SHA accelerator is processing data, 0 if idle. (RO)

**Register 12.6: SHA\_SHA256\_START\_REG (0x090)**

(reserved)		SHA_SHA256_START	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA256\_START** Write 1 to start an SHA-256 operation on the first message block. (WO)

**Register 12.7: SHA\_SHA256\_CONTINUE\_REG (0x094)**

(reserved)		SHA_SHA256_CONTINUE	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA256\_CONTINUE** Write 1 to continue the SHA-256 operation with subsequent blocks. (WO)

**Register 12.8: SHA\_SHA256\_LOAD\_REG (0x098)**

(reserved)		SHA_SHA256_LOAD	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA256\_LOAD** Write 1 to finish the SHA-256 operation to calculate the final the message hash.  
(WO)



Register 12.9: SHA\_SHA256\_BUSY\_REG (0x09C)

(reserved)		SHA_SHA256_BUSY	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA256\_BUSY** SHA-256 operation status: 1 if the SHA accelerator is processing data, 0 if idle.  
(RO)

Register 12.10: SHA\_SHA384\_START\_REG (0x0A0)

(reserved)		SHA_SHA384_START	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA384\_START** Write 1 to start an SHA-384 operation on the first message block. (WO)

Register 12.11: SHA\_SHA384\_CONTINUE\_REG (0x0A4)

(reserved)		SHA_SHA384_CONTINUE	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA384\_CONTINUE** Write 1 to continue the SHA-384 operation with subsequent blocks. (WO)

Register 12.12: SHA\_SHA384\_LOAD\_REG (0x0A8)

(reserved)		SHA_SHA384_LOAD	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA384\_LOAD** Write 1 to finish the SHA-384 operation to calculate the final the message hash.  
(WO)

**Register 12.13: SHA\_SHA384\_BUSY\_REG (0x0AC)**

(reserved)		SHA_SHA384_BUSY	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA384\_BUSY** SHA-384 operation status: 1 if the SHA accelerator is processing data, 0 if idle.  
(RO)

**Register 12.14: SHA\_SHA512\_START\_REG (0x0B0)**

(reserved)		SHA_SHA512_START	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA512\_START** Write 1 to start an SHA-512 operation on the first message block. (WO)

**Register 12.15: SHA\_SHA512\_CONTINUE\_REG (0x0B4)**

(reserved)		SHA_SHA512_CONTINUE	
31	1	0	
0x00000000		0	Reset

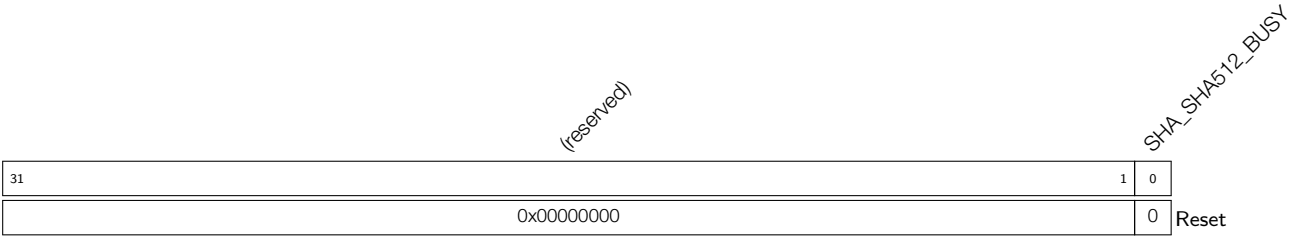
**SHA\_SHA512\_CONTINUE** Write 1 to continue the SHA-512 operation with subsequent blocks. (WO)

**Register 12.16: SHA\_SHA512\_LOAD\_REG (0x0B8)**

(reserved)		SHA_SHA512_LOAD	
31	1	0	
0x00000000		0	Reset

**SHA\_SHA512\_LOAD** Write 1 to finish the SHA-512 operation to calculate the final the message hash.  
(WO)

Register 12.17: SHA\_SHA512\_BUSY\_REG (0x0BC)



**SHA\_SHA512\_BUSY** SHA-512 operation status: 1 if the SHA accelerator is processing data, 0 if idle.  
(RO)