SF32LB52x User Manual

V0.8.4

UM5201-SF32LB52x-EN

SiFli Technologies (Nanjing) Co., Ltd.
http://www.sifli.com

Revision History

Document Status

Document Status	Version Range	Description
Draft	0.0.0 ~0.9.9	Initial draft, informal release. The information is preliminary data, reflecting the specifications and performance of the product before mass production. No warranty is made as to the accuracy and the content is subject to change at any time without notice.
Release	1.0.0 ~1.9.9	Official release, and minor amendments might be made to the information to more accurately reflect the specifications and performance of mass-produced products; SiFli reserves the right to make changes to the document at any time without notice.

Revision History

Date	Version	Release Notes
2025-06-04	0.8.4	Added the base address of register table and the overview of some modules.
2025-05-28	0.8.3	Added Default PD/PU Setting in Table 5-1 and updated Figure 3-1
2025-03-10	0.8.2	Corrected the frame format content in the Debug Interface section and the endpoint description for USB. Corrected the description of the data bit width supported by SPI. Added access descriptions for 0x10000000 and 0x60000000.
2025-03-05	0.8.1	Modified the description of the Cordic coprocessor.
2025-03-03	0.8	Added a Debug Interface section, detailing the relationship between various modules and the system clock in the clock and reset section. Included the USART module and RTC register table.
2025-01-10	0.7	Add a description of the combined IO and an overview of each IP.
2024-12-20	0.6	Update LCDC, EFUSE and other modules
2024-12-06	0.5	Update the sections on clock and reset, low power consumption, etc.
2024-11-06	0.4	Update relevant information on GPADC
2024-06-11	0.3	Updated information related to the small core
2023-07-06	0.2	Correct power consumption values and update the AON register table and DMA section content
2023-06-15	0.1	Draft

Overview

SF32LB52x is a family of highly integrated high-performance MCUs desgined for ultra-low power Artifical Intelligence of Things (AIoT) scenarios. SF32LB52x adopts the big.LITTLE architecture with Arm Cortex-M33 STAR-MC1 processors, and is embedded with 2D/2.5D GPU, dual-mode BT5.3, and audio codec. SF32LB52x can be used for a wide variety of applications such as wearables, smart HMI devices, and smart homes.

The high-performance processor ("big core") of SF32LB52x can operate at up to 240MHz for 984 CoreMark. It supports dynamic frequency power adjustment, can also serve as sensor hub and Bluetooth controller at high power efficiency of 4.8uA/CoreMark, thus delivering no-compromise user experience of both high computational performance required for feature-rich graphical HMI (Human Machine Interface) and ultra-low power sensor hub operation.

The 2D/2.5D GPU, at up to 240MHz, supports 2-layer alpha blending, hardware accelerated rotation and scaling, and conversion of common graphic formats. eZip™2.0 supports lossless compressed graphics file, saving memory bandwidth and storage capacity. The LCD controller can support interfaces of 8080/QSPI at a full-screen refresh frame rate up to 60fps, and support Always-On Display.

The dual-mode BT5.3 transceiver has a maximum Tx power of 13dBm at EDR2 mode and Rx power of 2.4mA@3.8V, and the sensitivity reaches -100dBm (1Mbps) for BLE and -95.5dBm for EDR2. SF32LB52x is embedded with high-fidelity audio ADC/DAC, supporting Bluetooth call and connecting headphones for MP3 playback.

Functional Block Diagram

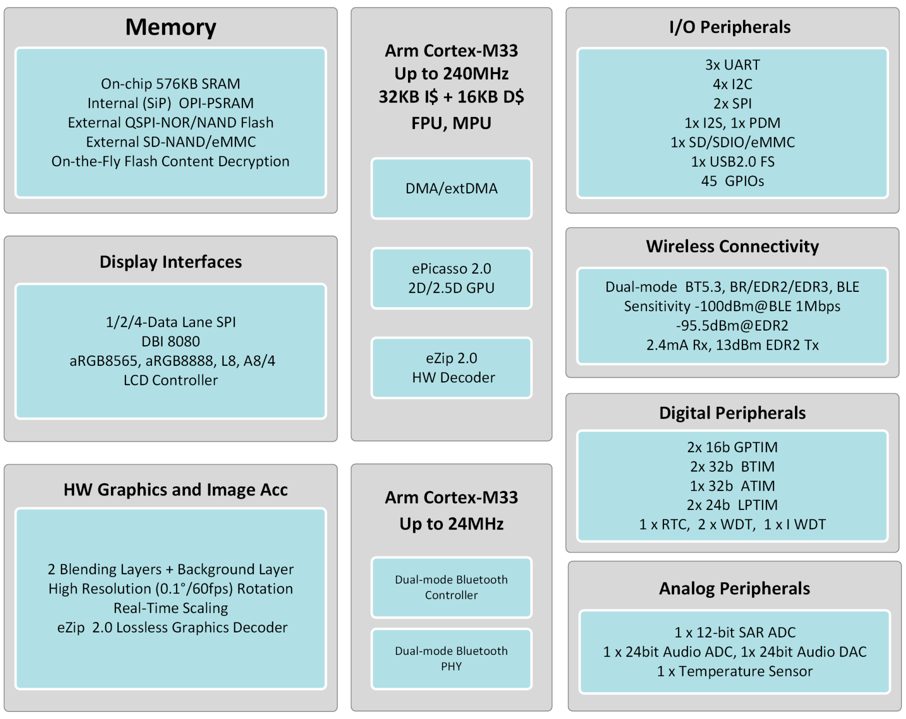


Figure 0-1: **Functional Block Diagram**

Features

CPU and Memory

- High Performance Processor (HCPU)
  - Arm Cortex-M33 STAR-MC1
  - Up to 240MHz clock frequency, adjustable
  - Up to 370DMIPS, 984 EEMBC CoreMark
  - I-Cache + D-Cache: 32KB(2-way)+16KB(4-way)
  - SRAM: 512KB (All Retention SRAM)
  - CoreMark power: 23uA/MHz @3.8V
  - Single Precision Floating Point Unit (FPU)
  - Memory Protection Unit (MPU)
- Ultra Low-Power Processor (LCPU)
  - Arm Cortex-M33 STAR-MC1
  - Up to 24MHz clock frequency, adjustable
  - SRAM: 64KB (All Retention SRAM)

Wireless Connectivity

- Dual-mode BT5.3, support BLE Audio
- Sensitivity: -100dBm(BLE/1Mbps), -96.3dBm(BR), -95.5dBm(EDR2)
- Max. Tx power: 13dBm（EDR2/3）, 19dBm（BR/BLE）
- Rx peak current (BR): 2.4mA@3.8V

Audio

- 1×HiFi 24-bit Audio DAC
  - Noise floor: 3.7uVrms
  - SNR(with 10kohm load and A-Weighted): 109dB, Dynamic Range: 109dB
  - Sample rate: 8k/ 16k/ 11.025k/ 22.05k/ 24k/ 32k/ 44.1k/ 48kHz
  - Support digital volume of 192 steps with zero-crossing detection
- 1×HiFi 24-bit Audio ADC
  - SNR(A-Weighted): 99dB, Dynamic Range: 99dB
  - Sample rate: 8k/ 11.025k/ 12k/ 16k/ 22.05k/ 24k/ 32k/ 44.1k/ 48kHz
  - A digital high-pass filter can be used to remove dc offsets of ADC
  - Support single-ended and fully differential input microphones
  - Micbias LDO with 1.4V~2.8V output voltage and 0~2mA output current

Graphics and Display

- 2D/2.5D GPU—ePicasso™2.0
  - Hardware-accelerated rotation, scaling, and mirroring
  - Max. resolution: 512×512
  - Support aRGB8565, aRGB8888, L8, A8/4/2,YUV, support alpha blending
- Lossless Decompression Accelerator – eZip™2.0
  - Lossless graphics decompression
  - support native animation eZip-A
  - Concatenated operation with ePicasso™2.0
- LCD Controller
  - Support 8080, SPI, Dual-SPI, Quad-SPI
  - 1 layer + 1 background layer alpha blending
  - Independent LCD controller, Always-On Display

Memory Interface

- Support（SiP）NOR-Flash, interface frequency up to 96MHz
- Support（SiP）OPI-PSRAM, interface frequency up to 144MHz
- 1×MPI(QSPI), support NOR, NAND, QPI-PSRAM
- 1×SD/SDIO, support SD3.0, SDIO3.0, eMMC

DMA

- General DMA: high efficiency data transfer between internal memory and peripherals
- extDMA: high efficiency data transfer between internal memory and external memory

Security

- AES,HASH and CRC hardware accelerators
- True random number generator (TRNG)
- PSA Certified Level 1

Timers

- 2×16b GPTIM, 2×32b BTIM, 1×32b ATIM, 2×24b LPTIM
- 1×RTC
- 2×24b WDT, 1×IWDT

Analog Peripherals

- 1×12-bit general purpose SAR ADC, 8 channels
- 1×Temperature sensor
- 1×24-bit audio ADC，1×24-bit audio DAC

I/O Peripherals

- Up to 45 GPIOs
- 3×UART, 4× I^2C, 2×SPI
- 1× I^2S, 1× PDM
- 1×USB2.0 FS
- Peripheral Task Controller (PTC)

Power Management

- High-efficiency buck and low-power LDO
- 2 external 3.3V power supply LDOs, Max. current 150mA×2
- Sleep current: 2uA
- Built-in 560mA lithium battery linear charger, supporting 4.2V-4.45V full voltage
- VBAT voltage range: 3.2V-4.7V
- VBUS voltage range: 4.6V-5.5V

Others

- Operating Temperature Range: -40~85°C
- Package: QFN68L, 44 GPIOs, 7×7×0.85mm

Applications

Smart Wearable

- Smart watch
- Smart wristband
- Wearable medical device
- Fitness equipment

Industrial Device

- Cost-effective display solution
- Graphical Human-Machine Interface (HMI) device
- Industrial sensor hub
- Industrial equipment monitoring
- Industrial instrumentation

Vehicle Device

- Electric vehicle control center
- Car key
- Wearable car remote controls

Home Automation

- Smart home appliance
- Smart door lock

Generic Scenario

- Low-power sensor hub
- Bluetooth mesh

Contents

List of Figures

List of Tables

1 Introduction

1.1 System Architecture

SF32LB52x is a family of highly integrated high-performance MCUs desgined for ultra-low-power Artifical Intelligence of Things (AIoT) scenarios. SF32LB52x adopts the big.LITTLE architecture with the Arm Cortex-M33 STAR-MC1 processor.

- High-Performance Processor/Big Core (HCPU): 32KB instruction cache (I-Cache) and 16KB data cache (D-Cache), 512KB SRAM (All Retention SRAM); Up to 240MHz clock frequency, it can dynamically switch between basic-working mode and enhance-working mode for efficient access to on-chip and off-chip memory. As the system master, enhance-working mode is mainly used for system control, Human-Machine Interaction, and high-performance computing; Meanwhile, as the low-power sensor hub, basic-working mode can be used for all kinds of data acquisition and processing in low-power scenarios.
- Ultra-low Power Processor/LITTLE Core (LCPU): Up to 24MHz clock frequency, 64KB SRAM (all Retention SRAM); it is mainly used for transmission control and basic data processing of Bluetooth Low Energy.

1.2 Cortex-M33 STAR-MC1 Processor

Cortex-M33 STAR-MC1 processor is the first processor of the ”Star” series from Arm China. It has the key features of Cortex-M33, supporting the full functionality of the existing Arm v8-M architecture, and with an in-order three-stage pipeline, it can significantly reduce the power consumption of the system. It also has partial dual-issue 16-bit instruction capability, the coprocessor interface is further improved and support the Cache.

With the performance reaching 1.5DMIPS/MHz and 4.02Coremark/MHz, Cortex-M33 STAR-MC1 delivers a 20% performance improvement over previous-generation Arm processors at the same clock speed.

Cortex-M33 STAR-MC1 has a coprocessor interface which can further enhance the capability of customized calculation to meet the requirements of different scenarios. The MCR (Move from Coprocessor to Register) and MRC (Move from Register to Coprocessor) instructions enable the transfer of register data and computation results between Cortex-M33 STAR-MC1 and the coprocessor, making it ideal for operations with small data volumes, complex calculations but relatively fragmented and low latency. While the coprocessor computes, Cortex-M33 STAR-MC1 processor can still execute other instructions in parallel, thus significantly improving execution efficiency.

In addition, the processor supports Digital Signal Processing (DSP) instruction sets and Floating Point Unit (FPU).

Tightly Coupled Memory (TCM) and Cache technologies are adopted in Cortex-M33 STAR-MC1 processor to enhance flexibility in the use of internal and external memory systems with different characteristics, ensuring the real-time response and computational efficiency of the processor in a variety of scenarios.

1.3 High-Performance Processor (Big Core) System (HPSYS)

1.3.1 Bus Architecture

The HPSYS provides an internal bus matrix based on the AHB protocol, which supports multiple master devices to access the address spaces of multiple slave devices in parallel.

As shown in Figure 1-1, the master devices of the bus are located on the top side and the address spaces of the slave devices are located on the right side, and the black dots at the intersection represent bus connectivity.

The HCPU and DMAC1 has access to all address spaces of the HPSYS.

128KB address space is shared between DTCM and HPSYS_RAM0, and can be accessed by the HCPU and other master devices.

HP_PERI includes APB-related peripherals and AHB-related peripherals, and can be accessed by HCPU, DMAC1 and PTC1.

When multiple master devices access the address space of the same slave device at the same time, the access order will be determined based on the Round-Robin Arbitration Principle.

As shown in the figure, when multiple master devices with unconnected borders access the address spaces of different slave devices at the same time, they will not be affected by each other. When two master devices with connected borders initiate access at the same time, the access order will be decided based on the Round-Robin Arbitration Principle.

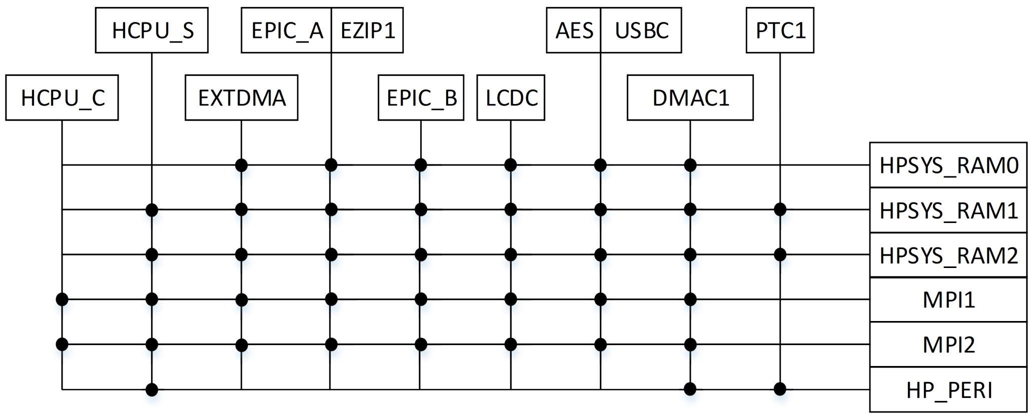


Figure 1-1: **Bus Architecture of HPSYS**

1.3.2 Memory Type

1.3.2.1 Cache

The HCPU has 32KB 2-way I-Cache (Level 1 instruction cache) and 16KB 4-way D-Cache (Level 1 data cache), which can greatly improve CPU execution efficiency during XIP. The MPU (Memory Protection Unit) should be configured appropriately to set the cache address segment and non-cache address segment to balance efficiency and ease of use.

1.3.2.2 TCM

The HCPU has 128KB zero-wait-cycle D-TCM with address space 0x2000_0000-0x2001_FFFF, which can be used to place codes and data with high real-time requirements. This TCM memory is connected to the bus and can be accessed by other AHB masters.

1.3.2.3 SRAM

There is a total of 512KB SRAM on the HPSYS bus, which includes:

- 0x2000_0000-0x2001_FFFF, 128KB zero-wait-cycle SRAM (shared with D-TCM), accessible to all AHB masters. Maximum frequency is 240MHz.
- 0x2002_0000-0x2007_FFFF, 384KB zero-wait-cycle SRAM, accessible to all AHB masters. Maximum frequency is 240MHz.

1.3.2.4 Off-chip RAM

The HPSYS supports combined 4-wire and 8-wire pSRAM with address space 0x6000_0000 - 0x61FF_FFFF, the actual accessible address is determined by the capacity of the external particles. The maximum interface frequency is DDR 144MHz and the data bit width is 8-bit.

1.3.2.5 Off-chip Flash

The HPSYS supports external NOR/NAND FLASHs, in which

- 0x6000_0000–0x61FF_FFFF address segment can be combined with FLASH, recommended frequency is 96MHz
- 0x6200_0000–0x9FFF_FFFF address segment can be connected to external FLASH, recommended frequency is 60MHz

1.3.3 Address Mapping

Table 1-1: **Address Mapping of HPSYS**

Category	Memory /IP	Address Space	HCPU				LCPU	
			Starting Address		Ending Address		Starting Address	Ending Address
HPSYS_ITCM		**64KB**	**0x0000_0000**		**0x0000_FFFF**		**NA**	**NA**
	ROM	64KB	0x0000_0000		0x0000_FFFF		-	-
	Reserved	-	-		-		-	-
External Memory		**1024MB**	***0x1000_0000**	**0x6000_0000**	***0x1FFF_FFFF**	**0x9FFF_FFFF**	**0x6000_0000**	**0x9FFF_FFFF**
	MPI1 Memory	32MB	*0x1000_0000	0x6000_0000	*0x11FF_FFFF	0x61FF_FFFF	0x6000_0000	0x61FF_FFFF
	MPI2 Memory	224MB/992MB	*0x1200_0000	0x6200_0000	*0x1FFF_FFFF	0x9FFF_FFFF	0x6200_0000	0x9FFF_FFFF
HPSYS_RAM		**512KB**	**0x2000_0000**		**0x2007_FFFF**		**0x2A00_0000**	**0x2A07_FFFF**
	RAM0 (DTCM)	128KB	0x2000_0000		0x2001_FFFF		0x2A00_0000	0x2A01_FFFF
	RAM1	128KB	0x2002_0000		0x2003_FFFF		0x2A02_0000	0x2A03_FFFF
	RAM2	256KB	0x2004_0000		0x2007_FFFF		0x2A04_0000	0x2A07_FFFF
HPSYS_APB1		**256KB**	**0x5000_0000**		**0x5003_FFFF**		**0x5000_0000**	**0x5003_FFFF**
	RCC1	4KB	0x5000_0000		0x5000_0FFF		0x5000_0000	0x5000_0FFF
	EXTDMA	4KB	0x5000_1000		0x5000_1FFF		0x5000_1000	0x5000_1FFF
	SECU1	4KB	0x5000_2000		0x5000_2FFF		0x5000_2000	0x5000_2FFF
	PINMUX1	4KB	0x5000_3000		0x5000_3FFF		0x5000_3000	0x5000_3FFF
	ATIM1	4KB	0x5000_4000		0x5000_4FFF		0x5000_4000	0x5000_4FFF
	AUDPRC	4KB	0x5000_5000		0x5000_5FFF		0x5000_5000	0x5000_5FFF
	EZIP1	4KB	0x5000_6000		0x5000_6FFF		0x5000_6000	0x5000_6FFF
	EPIC	4KB	0x5000_7000		0x5000_7FFF		0x5000_7000	0x5000_7FFF
	LCDC1	4KB	0x5000_8000		0x5000_8FFF		0x5000_8000	0x5000_8FFF
	I2S1	4KB	0x5000_9000		0x5000_9FFF		0x5000_9000	0x5000_9FFF
	Reserved	4KB	0x5000_A000		0x5000_AFFF		0x5000_A000	0x5000_AFFF
	SYSCFG1	4KB	0x5000_B000		0x5000_BFFF		0x5000_B000	0x5000_BFFF
	EFUSEC	4KB	0x5000_C000		0x5000_CFFF		0x5000_C000	0x5000_CFFF
	AES	4KB	0x5000_D000		0x5000_DFFF		0x5000_D000	0x5000_DFFF
	Reserved	4KB	0x5000_E000		0x5000_EFFF		0x5000_E000	0x5000_EFFF
	TRNG	4KB	0x5000_F000		0x5000_FFFF		0x5000_F000	0x5000_FFFF
	Reserved	4KB	0x5001_0000		0x5001_0FFF		0x5001_0000	0x5001_0FFF
	Reserved	4KB	0x5001_1000		0x5001_1FFF		0x5001_1000	0x5001_1FFF
	Reserved	4KB	0x5001_2000		0x5001_2FFF		0x5001_2000	0x5001_2FFF
	Reserved	4KB	0x5001_3000		0x5001_3FFF		0x5001_3000	0x5001_3FFF
	Reserved	4KB	0x5001_4000		0x5001_4FFF		0x5001_4000	0x5001_4FFF
	Reserved	4KB	0x5001_5000		0x5001_5FFF		0x5001_5000	0x5001_5FFF
	Reserved	4KB	0x5001_6000		0x5001_6FFF		0x5001_6000	0x5001_6FFF
	Reserved	4KB	0x5001_7000		0x5001_7FFF		0x5001_7000	0x5001_7FFF

Continued on the next page

Table 1-1: Address Mapping of HPSYS (continued)

Category	Memory /IP	Address Space	HCPU		LCPU	
			Starting Address	Ending Address	Starting Address	Ending Address
	Reserved	4KB	0x5001_8000	0x5001_8FFF	0x5001_8000	0x5001_8FFF
	Reserved	4KB	0x5001_9000	0x5001_9FFF	0x5001_9000	0x5001_9FFF
	Reserved	4KB	0x5001_A000	0x5001_AFFF	0x5001_A000	0x5001_AFFF
	Reserved	4KB	0x5001_B000	0x5001_BFFF	0x5001_B000	0x5001_BFFF
	Reserved	4KB	0x5001_C000	0x5001_CFFF	0x5001_C000	0x5001_CFFF
	Reserved	4KB	0x5001_D000	0x5001_DFFF	0x5001_D000	0x5001_DFFF
	Reserved	4KB	0x5001_E000	0x5001_EFFF	0x5001_E000	0x5001_EFFF
	Reserved	4KB	0x5001_F000	0x5001_FFFF	0x5001_F000	0x5001_FFFF
	Reserved	128KB	0x5002_0000	0x5003_FFFF	0x5002_0000	0x5003_FFFF
HPSYS_AHB1		**256KB**	**0x5004_0000**	**0x5007_FFFF**	**0x5004_0000**	**0x5007_FFFF**
	Reserved	4KB	0x5004_0000	0x5004_0FFF	0x5004_0000	0x5004_0FFF
	MPI1	4KB	0x5004_1000	0x5004_1FFF	0x5004_1000	0x5004_1FFF
	MPI2	4KB	0x5004_2000	0x5004_2FFF	0x5004_2000	0x5004_2FFF
	Reserved	4KB	0x5004_3000	0x5004_3FFF	0x5004_3000	0x5004_3FFF
	Reserved	4KB	0x5004_4000	0x5004_4FFF	0x5004_4000	0x5004_4FFF
	SDMMC1	4KB	0x5004_5000	0x5004_5FFF	0x5004_5000	0x5004_5FFF
	Reserved	4KB	0x5004_6000	0x5004_6FFF	0x5004_6000	0x5004_6FFF
	USBC	4KB	0x5004_7000	0x5004_7FFF	0x5004_7000	0x5004_7FFF
	CRC1	4KB	0x5004_8000	0x5004_8FFF	0x5004_8000	0x5004_8FFF
	Reserved	28KB	0x5004_9000	0x5004_FFFF	0x5004_9000	0x5004_FFFF
	GFX_RAM	64KB	0x5005_0000	0x5005_FFFF	0x5005_0000	0x5005_FFFF
	Reserved	128KB	0x5006_0000	0x5007_FFFF	0x5006_0000	0x5007_FFFF
HPSYS_APB2		**128KB**	**0x5008_0000**	**0x5009_FFFF**	**0x5008_0000**	**0x5009_FFFF**
	PTC1	4KB	0x5008_0000	0x5008_0FFF	0x5008_0000	0x5008_0FFF
	DMAC1	4KB	0x5008_1000	0x5008_1FFF	0x5008_1000	0x5008_1FFF
	MAILBOX1	4KB	0x5008_2000	0x5008_2FFF	0x5008_2000	0x5008_2FFF
	Reserved	4KB	0x5008_3000	0x5008_3FFF	0x5008_3000	0x5008_3FFF
	USART1	4KB	0x5008_4000	0x5008_4FFF	0x5008_4000	0x5008_4FFF
	USART2	4KB	0x5008_5000	0x5008_5FFF	0x5008_5000	0x5008_5FFF
	USART3	4KB	0x5008_6000	0x5008_6FFF	0x5008_6000	0x5008_6FFF
	GPADC	4KB	0x5008_7000	0x5008_7FFF	0x5008_7000	0x5008_7FFF
	AUDCODEC	4KB	0x5008_8000	0x5008_8FFF	0x5008_8000	0x5008_8FFF
	TSEN	4KB	0x5008_9000	0x5008_9FFF	0x5008_9000	0x5008_9FFF
	Reserved	4KB	0x5008_A000	0x5008_AFFF	0x5008_A000	0x5008_AFFF
	Reserved	4KB	0x5008_B000	0x5008_BFFF	0x5008_B000	0x5008_BFFF
	Reserved	4KB	0x5008_C000	0x5008_CFFF	0x5008_C000	0x5008_CFFF
	Reserved	4KB	0x5008_D000	0x5008_DFFF	0x5008_D000	0x5008_DFFF
	Reserved	4KB	0x5008_E000	0x5008_EFFF	0x5008_E000	0x5008_EFFF
	Reserved	4KB	0x5008_F000	0x5008_FFFF	0x5008_F000	0x5008_FFFF
	GPTIM1	4KB	0x5009_0000	0x5009_0FFF	0x5009_0000	0x5009_0FFF
	Reserved	4KB	0x5009_1000	0x5009_1FFF	0x5009_1000	0x5009_1FFF
	BTIM1	4KB	0x5009_2000	0x5009_2FFF	0x5009_2000	0x5009_2FFF
	Reserved	4KB	0x5009_3000	0x5009_3FFF	0x5009_3000	0x5009_3FFF
	WDT1	4KB	0x5009_4000	0x5009_4FFF	0x5009_4000	0x5009_4FFF
	SPI1	4KB	0x5009_5000	0x5009_5FFF	0x5009_5000	0x5009_5FFF
	SPI2	4KB	0x5009_6000	0x5009_6FFF	0x5009_6000	0x5009_6FFF
	Reserved	4KB	0x5009_7000	0x5009_7FFF	0x5009_7000	0x5009_7FFF
	Reserved	4KB	0x5009_8000	0x5009_8FFF	0x5009_8000	0x5009_8FFF
	Reserved	4KB	0x5009_9000	0x5009_9FFF	0x5009_9000	0x5009_9FFF
	PDM1	4KB	0x5009_A000	0x5009_AFFF	0x5009_A000	0x5009_AFFF
	Reserved	4KB	0x5009_B000	0x5009_BFFF	0x5009_B000	0x5009_BFFF
	I2C1	4KB	0x5009_C000	0x5009_CFFF	0x5009_C000	0x5009_CFFF
	I2C2	4KB	0x5009_D000	0x5009_DFFF	0x5009_D000	0x5009_DFFF
	I2C3	4KB	0x5009_E000	0x5009_EFFF	0x5009_E000	0x5009_EFFF
	I2C4	4KB	0x5009_F000	0x5009_FFFF	0x5009_F000	0x5009_FFFF
HPSYS_AHB2		**64KB**	**0x500A_0000**	**0x500A_FFFF**	**0x500A_0000**	**0x500A_FFFF**
	GPIO1	4KB	0x500A_0000	0x500A_0FFF	0x500A_0000	0x500A_0FFF
	Reserved	60KB	0x500A_1000	0x500A_FFFF	0x500A_1000	0x500A_FFFF
HPSYS_APB3		**64KB**	**0x500B_0000**	**0x500B_FFFF**	**0x500B_0000**	**0x500B_FFFF**
	GPTIM2	4KB	0x500B_0000	0x500B_0FFF	0x500B_0000	0x500B_0FFF
	BTIM2	4KB	0x500B_1000	0x500B_1FFF	0x500B_1000	0x500B_1FFF
	Reserved	56KB	0x500B_2000	0x500B_FFFF	0x500B_2000	0x500B_FFFF
HPSYS_APB4		**256KB**	**0x500C_0000**	**0x500F_FFFF**	**0x500C_0000**	**0x500F_FFFF**
	HPSYS_AON	4KB	0x500C_0000	0x500C_0FFF	0x500C_0000	0x500C_0FFF
	LPTIM1	4KB	0x500C_1000	0x500C_1FFF	0x500C_1000	0x500C_1FFF
	LPTIM2	4KB	0x500C_2000	0x500C_2FFF	0x500C_2000	0x500C_2FFF
	Reserved	4KB	0x500C_3000	0x500C_3FFF	0x500C_3000	0x500C_3FFF
	Reserved	24KB	0x500C_4000	0x500C_9FFF	0x500C_4000	0x500C_9FFF
	PMUC	4KB	0x500C_A000	0x500C_AFFF	0x500C_A000	0x500C_AFFF
	RTC	4KB	0x500C_B000	0x500C_BFFF	0x500C_B000	0x500C_BFFF
	IWDT	4KB	0x500C_C000	0x500C_CFFF	0x500C_C000	0x500C_CFFF
	Reserved	12KB	0x500C_D000	0x500C_FFFF	0x500C_D000	0x500C_FFFF
	Reserved	64KB	0x500D_0000	0x500D_FFFF	0x500D_0000	0x500D_FFFF
	Reserved	64KB	0x500E_0000	0x500E_FFFF	0x500E_0000	0x500E_FFFF
	EUROPA	4KB	0x500F_0000	0x500F_0FFF	0x500F_0000	0x500F_0FFF
	Reserved	60KB	0x500F_1000	0x500F_FFFF	0x500F_1000	0x500F_FFFF

*Only HCPU can access MPI through the starting address 0x10000000 or 0x60000000 ; other controllers can only access through 0x60000000.

1.3.4 Interrupt List

Table 1-2: **HCPU Interrupt List**

IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source
NMI	**WDT1**	IRQ[26]	**rsvd**	IRQ[53]	**DMAC1_CH4**	IRQ[80]	**MAILBOX2_CH2**
IRQ[0]	**AON**	IRQ[27]	**rsvd**	IRQ[54]	**DMAC1_CH5**	IRQ[81]	**rsvd**
IRQ[1]	**LCPU_IRQ[1]**	IRQ[28]	**rsvd**	IRQ[55]	**DMAC1_CH6**	IRQ[82]	**PDM1**
IRQ[2]	**LCPU_IRQ[2]**	IRQ[29]	**rsvd**	IRQ[56]	**DMAC1_CH7**	IRQ[83]	**rsvd**
IRQ[3]	**LCPU_IRQ[3]**	IRQ[30]	**rsvd**	IRQ[57]	**DMAC1_CH8**	IRQ[84]	**GPIO1**
IRQ[4]	**LCPU_IRQ[4]**	IRQ[31]	**rsvd**	IRQ[58]	**MAILBOX2_CH1**	IRQ[85]	**MPI1**
IRQ[5]	**LCPU_IRQ[5]**	IRQ[32]	**rsvd**	IRQ[59]	**USART1**	IRQ[86]	**MPI2**
IRQ[6]	**LCPU_IRQ[6]**	IRQ[33]	**rsvd**	IRQ[60]	**SPI1**	IRQ[87]	**rsvd**
IRQ[7]	**LCPU_IRQ[7]**	IRQ[34]	**rsvd**	IRQ[61]	**I2C1**	IRQ[88]	**rsvd**
IRQ[8]	**LCPU_IRQ[8]**	IRQ[35]	**rsvd**	IRQ[62]	**EPIC**	IRQ[89]	**EZIP1**
IRQ[9]	**LCPU_IRQ[9]**	IRQ[36]	**rsvd**	IRQ[63]	**LCDC1**	IRQ[90]	**AUDPRC**
IRQ[10]	**LCPU_IRQ[10]**	IRQ[37]	**rsvd**	IRQ[64]	**I2S1**	IRQ[91]	**TSEN**
IRQ[11]	**LCPU_IRQ[11]**	IRQ[38]	**rsvd**	IRQ[65]	**GPADC**	IRQ[92]	**USBC**
IRQ[12]	**LCPU_IRQ[12]**	IRQ[39]	**rsvd**	IRQ[66]	**EFUSEC**	IRQ[93]	**I2C3**
IRQ[13]	**LCPU_IRQ[13]**	IRQ[40]	**rsvd**	IRQ[67]	**AES**	IRQ[94]	**ATIM1**
IRQ[14]	**LCPU_IRQ[14]**	IRQ[41]	**rsvd**	IRQ[68]	**PTC1**	IRQ[95]	**USART3**
IRQ[15]	**LCPU_IRQ[15]**	IRQ[42]	**rsvd**	IRQ[69]	**TRNG**	IRQ[96]	**AUD_HP**
IRQ[16]	**LCPU_IRQ[16]**	IRQ[43]	**rsvd**	IRQ[70]	**GPTIM1**	IRQ[97]	**rsvd**
IRQ[17]	**LCPU_IRQ[17]**	IRQ[44]	**rsvd**	IRQ[71]	**GPTIM2**	IRQ[98]	**SECU1**
IRQ[18]	**LCPU_IRQ[18]**	IRQ[45]	**rsvd**	IRQ[72]	**BTIM1**	IRQ[99]	**rsvd**
IRQ[19]	**LCPU_IRQ[19]**	IRQ[46]	**LPTIM1**	IRQ[73]	**BTIM2**	\	\
IRQ[20]	**LCPU_IRQ[20]**	IRQ[47]	**LPTIM2**	IRQ[74]	**USART2**	\	\
IRQ[21]	**LCPU_IRQ[21]**	IRQ[48]	**PMUC**	IRQ[75]	**SPI2**	\	\
IRQ[22]	**LCPU_IRQ[22]**	IRQ[49]	**RTC**	IRQ[76]	**I2C2**	\	\
IRQ[23]	**LCPU_IRQ[23]**	IRQ[50]	**DMAC1_CH1**	IRQ[77]	**EXTDMA**	\	\
IRQ[24]	**rsvd**	IRQ[51]	**DMAC1_CH2**	IRQ[78]	**I2C4**	\	\
IRQ[25]	**rsvd**	IRQ[52]	**DMAC1_CH3**	IRQ[79]	**SDMMC1**	\	\

1.4 Bus Access Permissions

Table 1-3: **Bus Access Permissions**

	AHB Slave Device							
AHB Master Control	HP_ITCM	HP_RAM0	HP_RAM 1~2	MPI1~2	HP_AHB HP_APB	LP_ITCM	LP_RAM 0~1	LP_AHB LP_APB
HCPU	√	√	√	√(1)	√	√(2)	√	√
DMAC1	√(3)	√	√	√	√	√(2)	√	√
EXTDMA	x	√	√	√	x	x	√	√
AES	x	√	√	√	x	x	x	x
LCDC1	x	√	√	√	x	x	x	x
EZIP	x	√	√	√	x	x	x	x
EPIC	x	√	√	√	x	x	x	x
USBC	x	√	√	√	x	x	x	x
PTC1	x	x	√	x	√	x	x	√
LCPU	x	√(4)	√(4)	√	√	√	√(5)	√
DMAC2	x	√(4)	√(4)	√	√	√	√	√
PTC2	x	x	x	√	√	x	x	√

* (1)HCPU can access MPI content via the starting address 0x10000000 or the starting address 0x60000000. Other master controls can only access MPI through 0x60000000.
(2)HPSYS's master control accesses LPSYS's ITCM , starting at the address 0x20800000 , which differs from the address accessed by LCPU.
(3)Other master controls access HCPU's ROM, starting from the address 0xa0000000, indicating an offset of 0xa0000000 is applied.
(4)The main control of LPSYS accesses the SRAM of HPSYS, starting from the address 0x2a000000, which indicates an offset of 0x0a000000 has been added.

(5)LCPU accesses the SRAM of LPSYS, starting from either 0x00400000 or 0x20400000.

2 Clock and Reset

2.1 Introduction

The clock and reset module is utilized to control the chip's clock and reset functions, enabling features such as clock selection, clock division, module enabling, and module resetting.

2.2 Reset Sources

The chip's reset sources are primarily categorized into four types: board-level reset, watchdog reset, software reset, and wake-up reset. Each category contains several types of reset sources, each with distinct scopes of action.

2.2.1 Board-Level Reset Sources

Board-level reset sources primarily include:

Power-On Reset POR (Power On Reset). A reset that is automatically generated when the chip is powered on, initializing the entire chip and resetting all module states to their default values.

Brown-Out Reset BOR (Brown-Out Reset). A reset that is automatically generated when the chip's VSYS voltage falls below a specifed threshold, initializing the entire chip and resetting all module states to their default values.

Power Key (PWRKEY) Reset. If the chip's power key PA34 remains at a high level for more than 10 seconds, a PWRKEY reset will occur, resetting all modules except for RTC and IWDT. The PMUC WSR_PWRKEY flag can be used to determine if a PWRKEY reset has occurred, and the PMUC WCR_PWRKEY can be used to clear this flag.

VBAT Under-voltage reset. The chip VBAT automatically generates a reset when the supply voltage falls below a certain threshold, which can reset all modules except for RTC and IWDT. The occurrence of a VBAT under-voltage reset can be queried through the PMUC WSR_LOWBAT flag, and this flag can be cleared using the PMUC WCR_LOWBAT.

2.2.2 Watchdog Reset Source

The watchdog reset sources primarily include:

Global watchdog IWDT . If this watchdog times out, it can generate an IWDT reset, which resets all modules except for RTC and IWDT. The occurrence of an IWDT reset can be queried through the PMUC WSR_IWDT flag, and this flag can be cleared using the IWDT WDT_ICR.

HPSYS Watchdog WDT1 . If this watchdog times out, it can trigger a WDT1 reset, which resets HCPU and HPSYS excluding HPSYS_AON and other peripherals. When the PMUC 's WER_WDT1 register is set to 1 , the reset scope can also be expanded to include all modules except for PMUC , RTC , and IWDT . After the reset scope is expanded, it can be verifed whether a WDT1 reset has occurred through the PMUC 's WSR_WDT1 flag, which can be cleared using PMUC 's WCR_WDT.

LPSYS Watchdog WDT2 . If this watchdog times out, it can trigger a WDT2 reset, resetting LCPU and all peripherals except for LPSYS_AON. When the PMUC ’s WER_WDT2 register is set to 1 , it can also expand the reset scope to include all peripherals except for PMUC , RTC , and IWDT. A module reset has occurred. Once the reset scope is expanded, the occurrence of a WDT2 reset can be checked using the WSR_WDT2 flag of the PMUC, which can be cleared by the WCR_WDT2 of the PMUC.

2.2.3 Software Reset Source

Software Reboot. The software can initiate a reboot by writing a 1 to the CR_REBOOT of the PMUC. Following the reboot, all modules, except for the PMUC, RTC, and IWDT, will be reset. The CR_REBOOT of the PMUC remains at 1 after a software reboot, indicating that a software reboot has taken place. To trigger a software reboot again, the CR_REBOOT must first be set to 0.

HCPU System Reset. The software sends SYSRESETREQ by configuring the HCPU internal registers, allowing it to reset all modules within HPSYS except for HPSYS_AON , including HCPU , EPIC , DMAC1 , and others. A system reset initiated by an external debugger connected to HCPU is equivalent to a HCPU system reset.

LCPU System Reset. The software sends SYSRESETREQ by configuring the LCPU internal registers, allowing it to reset all modules within LPSYS except for LPSYS_AON , including LCPU , DMAC2 , MAC , and others. A system reset initiated by an external debugger connected to LCPU is equivalent to a LCPU system reset.

Module RCC Reset. The reset of a single module can be achieved through the HPSYS_RCC or LPSYS_RCC within the RSTRx register.

2.2.4 Wake-Up Reset Sources

Hibernate Wake-Up. When the chip enters hibernate mode, all modules except PMUC, RTC, and IWDT will be reset upon wake-up.

HPSYS Standby Wake-Up. When HPSYS enters standby mode, all HPSYS internal modules except HPSYS_AON will be reset upon wake-up, including HCPU, EPIC, DMAC1, and others.

LPSYS Standby Wake-Up. When LPSYS enters standby mode, all LPSYS internal modules except LPSYS_AON will be reset upon wake-up, including LCPU, DMAC2, MAC, and others.

Table 2-1: **Main Reset Sources of the Chip**

Reset Scope		Board-Level Reset				Watchdog Reset		
		POR	BOR	PWRKEY	VBAT Under Voltage	IWDT	WDT1	WDT2
HPSYS	HCPU	√	√	√	√	√	√	√(2)
	SRAM ret	√	√	√	√	√	√(1)	√(2)
	SRAM noret	√	√	√	√	√	√(1)	√(2)
	HPSYS Peripheral	√	√	√	√	√	√	√(2)
	HPAON	√	√	√	√	√	√(1)	√(2)
LPSYS	LCPU	√	√	√	√	√	√(1)	√
	SRAM ret	√	√	√	√	√	√(1)	√(2)
	SRAM noret	√	√	√	√	√	√(1)	√(2)
	LPSYS Peripheral	√	√	√	√	√	√(1)	√
	LPAON	√	√	√	√	√	√(1)	√(2)
AON	PMUC	√	√	√	√	√	x	x
	RTC	√	√	x	x	x	x	x
	IWDT	√	√	x	x	x	x	x

* (1)When the WER_WDT1 register of PMUC is set to 1, the reset scope is expanded.
(2)When the WER_WDT2 register of PMUC is set to 1, the reset scope is expanded.

Table 2-2: **Main Reset Sources of the Chip-Continued**

Reset Scope		Software Reset			Wake-Up Reset		
		Reboot	HCPU sysrst	LCPU sysrst	hibernate Wake-Up	HPSYS standby Wake-Up	LPSYS standby Wake-Up
HPSYS	HCPU	√	√	x	√	√	x
	SRAM ret	√	x	x	√	x	x
	SRAM noret	√	x	x	√	√	x
	HPSYS Peripheral	√	√	x	√	√	x
	HPAON	√	x	x	√	x	x
LPSYS	LCPU	√	x	√	√	x	√
	SRAM ret	√	x	x	√	x	x
	SRAM noret	√	x	x	√	x	√
	LPSYS Peripheral	√	x	√	√	x	√
	LPAON	√	x	x	√	x	x
AON	PMUC	x	x	x	x	x	x
	RTC	x	x	x	x	x	x
	IWDT	x	x	x	x	x	x

2.3 Clock Source

The main internal clock sources of the chip are listed in the table below. The clocks for each functional module are generated based on these clock sources.

Table 2-3: **Clock Source**

Clock	Frequency	Dependency
clk_lrc10	~10kHz	None
clk_lrc32	~32kHz	None
clk_lxt32	32.768kHz	32k Crystal Oscillator
clk_hrc48	~48MHz	None
clk_hxt48	48MHz	48M Crystal Oscillator
dll1/2	48~384MHz	clk_hxt48
clk_audpll	49.152MHz	clk_hxt48

clk_lrc10 is a low-power RC oscillator clock generated internally by the chip, with a frequency of approximately 10kHz . This clock frequency may be influenced by external environmental factors, and the current frequency can be obtained through measurement processes during operation. After the chip is powered on, this clock is automatically enabled as the default working clock for low-power modules (such as PMUC) . The configuration register associated with clk_lrc10 is PMUC 's LRC10_CR .

clk_lrc32 is a low-power RC oscillator clock generated internally by the chip, with a frequency of approximately 32kHz . clk_lrc32 is enabled by default, and the associated configuration register is PMUC 's LRC32_CR .

clk_lxt32 is a low-power clock generated by an external 32k crystal oscillator, with a frequency of 32.768kHz . This clock is optional and is recommended for scenarios requiring precise timing. clk_lxt32 configuration is disabled by default, and the register is PMUC 's LXT_CR .

clk_hrc48 is the clock generated by the chip's internal RC oscillator. Upon startup, this clock is automatically enabled as the default working clock for HCPU ; however, at this point, the frequency is uncalibrated and is an unknown value less than 48MHz . Before calibration, the working clock of HCPU should be switched to another clock (for example, clk_hxt48) , after which the calibration process should be executed. After calibration, the frequency of clk_hrc48 is 48MHz . The configuration register for clk_hrc48 is PMUC 's HRC_CR , and the related calibration registers are HRCCAL1 and HRCCAL2 in HPSYS_RCC . When both HPSYS and LPSYS are in low power mode, this clock is disabled by default.

clk_hxt48 is the clock generated by an external 48M crystal oscillator, with a frequency of 48MHz . This clock is automatically enabled upon chip startup and serves as the base clock for generating higher frequency clocks, as well as the base clock required for Bluetooth operation. When the system does not require a higher frequency clock and Bluetooth is in sleep mode, this clock can be turned off. When both the chip HPSYS and LPSYS are in low power mode, this clock is turned off by default. The configuration registers related to clk_hrc48 are PMUC's HXT_CR1/2/3 .

clk_dll1/2 is the high-frequency clock generated by the internal DLL module based on clk_hxt48 . These clocks are turned off by default and can be independently enabled to generate different frequencies when needed. The clock frequency generated by the DLL module can be configured in increments of 24MHz , with the configuration registers being DLL1CR and DLL2CR in HPSYS_RCC . When the chip HPSYS is in low power mode, these clocks are turned off by default.

clk_audpll is the clock generated by the PLL module within the chip, based on clk_hxt48, and serves as the operational clock for audio-related modules, with an adjustable frequency typically set to 44.1MHz . This clock is disabled by default, and the configuration register is located in the AUDCODEC module.

2.4 System Clock Structure

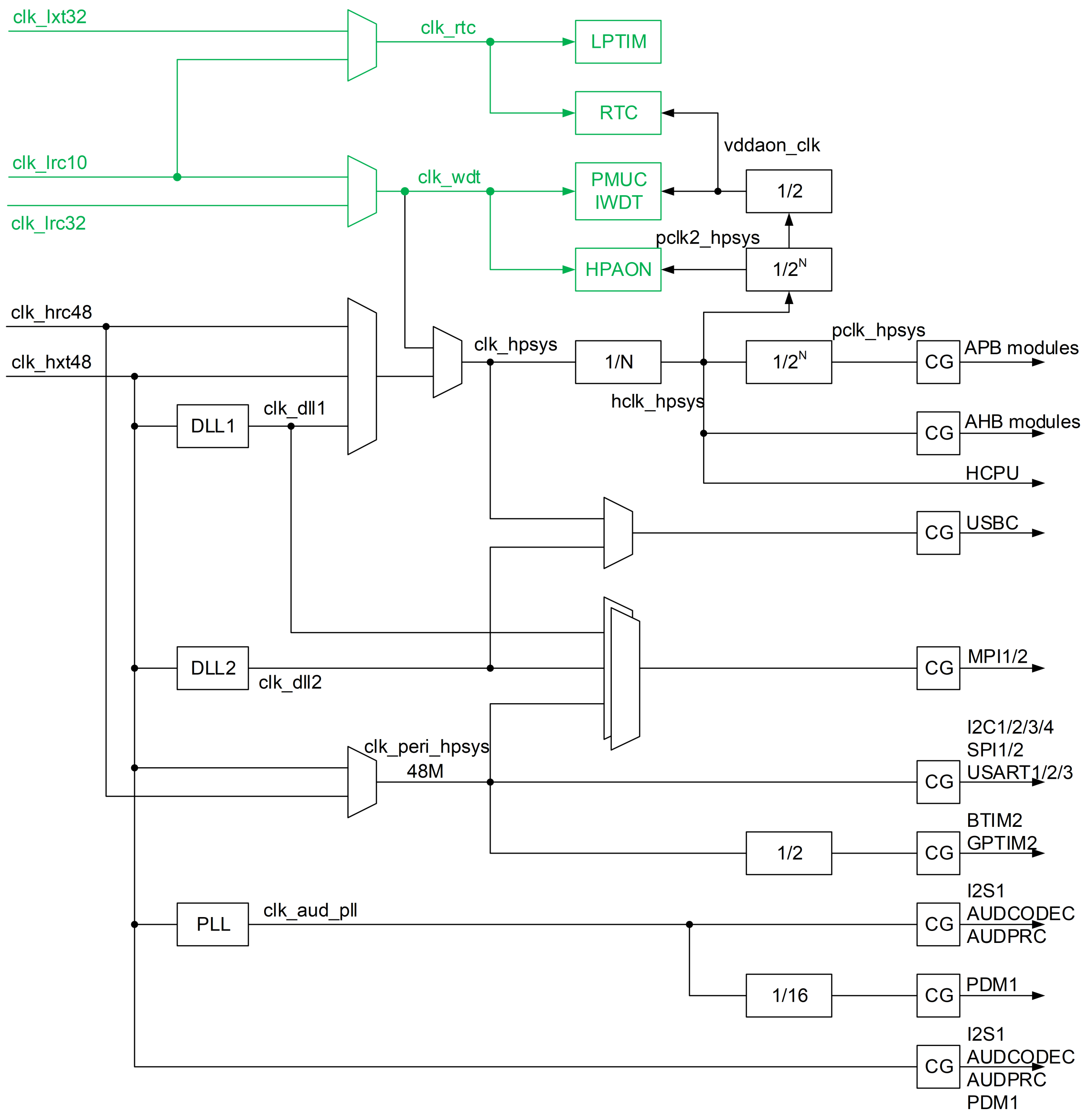


Figure 2-1: **HPSYS Clock Structure**

The system clock clk_hpsys of HPSYS can be selected from clk_hrc48, clk_hxt48 and clk_dll1. The selection register is found in HPSYS_RCC under CSR_SEL_SYS . The maximum frequency supported by clk_hpsys is 48MHz (basic mode) or 240MHz (enhanced mode) .

hclk_hpsys is generated by clk_hpsys with a division ratio of N, where the division ratio is specifed in HPSYS_RCC under CFGR_HDIV. hclk supports a maximum frequency of 48MHz (basic mode) or 240MHz (enhanced mode) and serves as the operating clock for AHB modules such as HCPU, EPIC, DMAC1, the AHB bus, and SRAM.

pclk_hpsys is generated by hclk_hpsys with a division ratio of 2^N, where the division ratio is 2^{CFGR_PDIV1}. pclk_hpsys supports a maximum frequency of 48MHz (basic mode) or 120MHz (enhanced mode) and serves as the operating clock for APB modules such as GPTIM1 and BTIM1, as well as the APB bus clock. When the frequency of hclk_hpsys or pclk_hpsys changes, the operating clock frequency of modules such as GPTIM1 and BTIM1 will also change, which may affect their functionality. Therefore, it is essential to maintain a constant frequency for both hclk_hpsys and pclk_hpsys while the relevant modules are in operation.

pclk2_hpsys is generated by hclk_hpsys with a division ratio of 2^N , where the division ratio is 2^{CFGR_PDIV2} . The maximum frequency supported by pclk2_hpsys is 6 MHz (in basic mode) or 7.5 MHz (in enhanced mode) , and it serves as the register access clock for the HP_AON module.

The working clock for MPI1/2 can be selected from clk_dll1, clk_dll2 , and clk_peri_hpsys . The selection register is CSR_SEL_MPI1/2 in HPSYS_RCC .

The working clock for USB can be selected from clk_hpsys and clk_dll2 , with the selection register being CSR_SEL_USB in HPSYS_RCC . It is generated by a division ratio of 1 to N , where the division ratio is USBCR_DIV . It is essential to ensure that the working clock for USB after division is 60 MHz ; otherwise, USB will not function properly.

The operating clock for peripherals such as USART1/2/3, SPI1/2, and I2C1/2/3/4, known as clk_peri_hpsys, can be selected from clk_hrc 48 and clk_hxt48, with a frequency of 48MHz. The selection register is located in HPSYS_RCC under CSR_SEL_PERI. clk_peri_hpsys is independent of the system clock, ensuring it remains unaffected when the system dynamically adjusts the frequency.

The operating clock for BTIM2 and GPTIM2 is a divided frequency of clk_peri_hpsys, independent of the system clock, ensuring that counting is not impacted when the system dynamically adjusts the frequency.

The audio modules I2S1, AUDPRC, and AUDCODEC can select one of two operating clocks, with the selection register located within these modules. One operating clock is clk_hxt48, while the other is clk_audpll.

PDM1 can select one of the two operational clocks, with the selection register located within PDM. One operational clock is generated by clk_hxt48; the other is produced by clk_audpll 16 through frequency division.。

The low power clock clk_rtc can be selected from clk_lrc10 and clk_lxt32, serving as the operational clock for low power modules such as LPTIM1/2 and RTC, as well as the sleep clock for Bluetooth. The selection register for clk_rtc is CR_LPCKSEL of the RTC module.。

Table 2-4: clk_rtc related module operational clock

Module	RTC->CR_LPCKSEL=0	RTC->CR_LPCKSEL=1
RTC	lrc10	lxt32
LPTIM1	lrc10	lxt32
LPTIM2	lrc10	lxt32
GTIM	lrc10	lxt32
Bluetooth sleep clock	lrc10	lxt32

The low power clock clk_wdt can be selected from clk_lrc10 and clk_lrc32, serving as the operational clock for low power modules such as HPAON, PMUC, and IWDT clk_wdt Select the register for the PMUC module's CR_SEL_LPCLK.

Table 2-5: clk_wdt Related module operating clock

Module	PMUC->CR_SEL_LPCLK=0	PMUX->CR_SEL_LPCLK=1
WDT1	lrc10	lrc32
IWDT	lrc10	lrc32
PMUC	lrc10	lrc32

2.5 Module clock

This chapter lists the operating clocks and bus clocks for each module.

The operating clock serves as the runtime clock for the internal logic; a higher clock frequency leads to greater operational effciency. For modules with IO interfaces, such as LCDC/I2C , the interface clock is often derived from the operating clock through division.。

The bus clock is the clock used to access module registers via the AHB or APB bus; a higher clock frequency enhances the effciency of reading and writing module registers.

Module	Operating Clock	Bus Clock
HCPU	hclk_hpsys	hclk_hpsys
DMAC1	hclk_hpsys	pclk_hpsys
EXTDMA	hclk_hpsys	pclk_hpsys
EPIC	hclk_hpsys	pclk_hpsys
EZIP1	hclk_hpsys	pclk_hpsys
LCDC1	hclk_hpsys	pclk_hpsys
AES	hclk_hpsys	pclk_hpsys
USBC	clk_hpsys or clk_dll2,and divided	hclk_hpsys
SDMMC1	hclk_hpsys	hclk_hpsys
MPI1	clk_dll1 or clk_dll2 or clk_peri_hpsys	hclk_hpsys
MPI2	clk_dll1 or clk_dll2 or clk_peri_hpsys	hclk_hpsys
CRC1	hclk_hpsys	hclk_hpsys
GPIO1	hclk_hpsys	hclk_hpsys
PTC1	hclk_hpsys	pclk_hpsys
SECU1	hclk_hpsys	pclk_hpsys
PINMUX1	/	pclk_hpsys
AUDPRC	clk_hxt48 or clk_audpll	pclk_hpsys
AUDCODEC	clk_hxt48 or clk_audpll	pclk_hpsys
I2S1	clk_hxt48 or clk_audpll	pclk_hpsys
PDM1	clk_hxt48 or clk_audpll/16	pclk_hpsys
I2C1/2/3/4	clk_peri_hpsys	pclk_hpsys
SPI1/2	clk_peri_hpsys	pclk_hpsys
USART1/2/3/4	clk_peri_hpsys	pclk_hpsys
GPADC	pclk_hpsys	pclk_hpsys
TSEN	pclk_hpsys	pclk_hpsys
WDT1	clk_wdt	pclk_hpsys
TRNG	pclk_hpsys	pclk_hpsys
EFUSEC	pclk_hpsys	pclk_hpsys
MAILBOX1	pclk_hpsys	pclk_hpsys
ATIM1	pclk_hpsys	pclk_hpsys
GPTIM1	pclk_hpsys	pclk_hpsys
GPTIM2	clk_peri_hpsys/2	pclk_hpsys
BTIM1	pclk_hpsys	pclk_hpsys
BTIM2	clk_peri_hpsys/2	pclk_hpsys
RTC	clk_rtc	vddaon_clk
LPTIM1	clk_rtc	pclk2_hpsys
LPTIM2	clk_rtc	pclk2_hpsys
PMUC	clk_wdt	vddaon_clk
IWDT	clk_wdt	vddaon_clk

2.6 Module Enablement

The ENR1 and ENR2 registers in HPSYS_RCC control the enablement of each module. When the corresponding bit for a module is 1, the module's register is accessible, and the module can operate. When the corresponding bit is 0, the working clock and bus clock for that module are both turned off, the module ceases operation, and the register becomes inaccessible; however, the register value will not be reset.

The ESR1 and ESR2 registers in HPSYS_RCC can be utilized for bitwise operations to enable the module. Writing a 1 to the corresponding bit of the module will enable that module, while other modules remain unaffected.

The ECR1 and ECR2 registers in HPSYS_RCC can be utilized for bitwise operations to disable the module. Writing a 1 to the corresponding bit of the module will disable that module, while other modules remain unaffected.

2.7 Module Reset

The RSTR1 and RSTR2 registers in HPSYS_RCC control the reset of each module. When the corresponding bit of the module is 1, the module's register and internal state are both reset. When the corresponding bit is 0, the module ceases to reset.

2.8 HPSYS_RCC Register

HPSYS_RCC base address is 0x50000000.

Table 2-6: **HPSYS_RCC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**RSTR1**	Reset Register 1
[31]	rw	1'h0	PTC1	0 - no reset; 1 - reset
[30:29]			RSVD	
[28]	rw	1'h0	I2C2	0 - no reset; 1 - reset
[27]	rw	1'h0	I2C1	0 - no reset; 1 - reset
[26]			RSVD	
[25]	rw	1'h0	PDM1	0 - no reset; 1 - reset
[24:23]			RSVD	
[22]	rw	1'h0	EXTDMA	0 - no reset; 1 - reset
[21]	rw	1'h0	SPI2	0 - no reset; 1 - reset
[20]	rw	1'h0	SPI1	0 - no reset; 1 - reset
[19]			RSVD	
[18]	rw	1'h0	BTIM2	0 - no reset; 1 - reset
[17]	rw	1'h0	BTIM1	0 - no reset; 1 - reset
[16]	rw	1'h0	GPTIM2	0 - no reset; 1 - reset
[15]	rw	1'h0	GPTIM1	0 - no reset; 1 - reset
[14]	rw	1'h0	TRNG	0 - no reset; 1 - reset
[13]	rw	1'h0	CRC1	0 - no reset; 1 - reset
[12]	rw	1'h0	AES	0 - no reset; 1 - reset
[11]	rw	1'h0	EFUSEC	0 - no reset; 1 - reset
[10]	rw	1'h0	SYSCFG1	0 - no reset; 1 - reset
[9]			RSVD	
[8]	rw	1'h0	I2S1	0 - no reset; 1 - reset
[7]	rw	1'h0	LCDC1	0 - no reset; 1 - reset
[6]	rw	1'h0	EPIC	0 - no reset; 1 - reset
[5]	rw	1'h0	EZIP1	0 - no reset; 1 - reset
[4]	rw	1'h0	USART2	0 - no reset; 1 - reset
[3]	rw	1'h0	USART1	0 - no reset; 1 - reset
[2]	rw	1'h0	PINMUX1	0 - no reset; 1 - reset
[1]	rw	1'h0	MAILBOX1	0 - no reset; 1 - reset

Continued on the next page...

Table 2-6: HPSYS_RCC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	DMAC1	0 - no reset; 1 - reset
0x04			**RSTR2**	Reset Register 2
[31:26]			RSVD	
[25]	rw	1'h0	I2C4	0 - no reset; 1 - reset
[24]			RSVD	
[23]	rw	1'h0	TSEN	0 - no reset; 1 - reset
[22]	rw	1'h0	GPADC	0 - no reset; 1 - reset
[21]			RSVD	
[20]	rw	1'h0	AUDPRC	0 - no reset; 1 - reset
[19]	rw	1'h0	AUDCODEC	0 - no reset; 1 - reset
[18:13]			RSVD	
[12]	rw	1'h0	USART3	0 - no reset; 1 - reset
[11:10]			RSVD	
[9]	rw	1'h0	ATIM1	0 - no reset; 1 - reset
[8]	rw	1'h0	I2C3	0 - no reset; 1 - reset
[7]			RSVD	
[6]	rw	1'h0	USBC	0 - no reset; 1 - reset
[5]			RSVD	
[4]	rw	1'h0	SDMMC1	0 - no reset; 1 - reset
[3]			RSVD	
[2]	rw	1'h0	MPI2	0 - no reset; 1 - reset
[1]	rw	1'h0	MPI1	0 - no reset; 1 - reset
[0]	rw	1'h0	GPIO1	0 - no reset; 1 - reset
0x08			**ENR1**	Enable Register 1
[31]	rw	1'h0	PTC1	write 1 to set module enable, write 0 to disable module
[30:29]			RSVD	
[28]	rw	1'h1	I2C2	write 1 to set module enable, write 0 to disable module
[27]	rw	1'h1	I2C1	write 1 to set module enable, write 0 to disable module
[26]			RSVD	
[25]	rw	1'h0	PDM1	write 1 to set module enable, write 0 to disable module
[24]			RSVD	
[23]	rw	1'h1	SECU1	write 1 to set module enable, write 0 to disable module
[22]	rw	1'h1	EXTDMA	write 1 to set module enable, write 0 to disable module
[21]	rw	1'h0	SPI2	write 1 to set module enable, write 0 to disable module
[20]	rw	1'h0	SPI1	write 1 to set module enable, write 0 to disable module
[19]			RSVD	
[18]	rw	1'h1	BTIM2	write 1 to set module enable, write 0 to disable module
[17]	rw	1'h1	BTIM1	write 1 to set module enable, write 0 to disable module
[16]	rw	1'h1	GPTIM2	write 1 to set module enable, write 0 to disable module
[15]	rw	1'h1	GPTIM1	write 1 to set module enable, write 0 to disable module
[14]	rw	1'h1	TRNG	write 1 to set module enable, write 0 to disable module
[13]	rw	1'h1	CRC1	write 1 to set module enable, write 0 to disable module
[12]	rw	1'h1	AES	write 1 to set module enable, write 0 to disable module
[11]	rw	1'h1	EFUSEC	write 1 to set module enable, write 0 to disable module
[10]	rw	1'h1	SYSCFG1	write 1 to set module enable, write 0 to disable module
[9]			RSVD	
[8]	rw	1'h0	I2S1	write 1 to set module enable, write 0 to disable module
[7]	rw	1'h0	LCDC1	write 1 to set module enable, write 0 to disable module
[6]	rw	1'h0	EPIC	write 1 to set module enable, write 0 to disable module
[5]	rw	1'h0	EZIP1	write 1 to set module enable, write 0 to disable module
[4]	rw	1'h1	USART2	write 1 to set module enable, write 0 to disable module
[3]			RSVD	

Continued on the next page...

Table 2-6: **HPSYS_RCC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h1	PINMUX1	write 1 to set module enable, write 0 to disable module
[1]	rw	1'h1	MAILBOX1	write 1 to set module enable, write 0 to disable module
[0]	rw	1'h1	DMAC1	write 1 to set module enable, write 0 to disable module
0x0C			**ENR2**	Enable Register 2
[31:26]			RSVD	
[25]	rw	1'h1	I2C4	write 1 to set module enable, write 0 to disable module
[24]			RSVD	
[23]	rw	1'h0	TSEN	write 1 to set module enable, write 0 to disable module
[22]	rw	1'h1	GPADC	write 1 to set module enable, write 0 to disable module
[21]			RSVD	
[20]	rw	1'h0	AUDPRC	write 1 to set module enable, write 0 to disable module
[19]	rw	1'h0	AUDCODEC	write 1 to set module enable, write 0 to disable module
[18:13]			RSVD	
[12]	rw	1'h1	USART3	write 1 to set module enable, write 0 to disable module
[11:10]			RSVD	
[9]	rw	1'h0	ATIM1	write 1 to set module enable, write 0 to disable module
[8]	rw	1'h1	I2C3	write 1 to set module enable, write 0 to disable module
[7]			RSVD	
[6]	rw	1'h0	USBC	write 1 to set module enable, write 0 to disable module
[5]			RSVD	
[4]	rw	1'h0	SDMMC1	write 1 to set module enable, write 0 to disable module
[3]			RSVD	
[2]	rw	1'h1	MPI2	write 1 to set module enable, write 0 to disable module
[1]	rw	1'h1	MPI1	write 1 to set module enable, write 0 to disable module
[0]	rw	1'h1	GPIO1	write 1 to set module enable, write 0 to disable module
0x10			**ESR1**	Enable Set Register 1
[31]	w	1'h0	PTC1	write 1 to set module enable, write 0 has no effect
[30:29]			RSVD	
[28]	w	1'h0	I2C2	write 1 to set module enable, write 0 has no effect
[27]	w	1'h0	I2C1	write 1 to set module enable, write 0 has no effect
[26]			RSVD	
[25]	w	1'h0	PDM1	write 1 to set module enable, write 0 has no effect
[24]			RSVD	
[23]	w	1'h0	SECU1	write 1 to set module enable, write 0 has no effect
[22]	w	1'h0	EXTDMA	write 1 to set module enable, write 0 has no effect
[21]	w	1'h0	SPI2	write 1 to set module enable, write 0 has no effect
[20]	w	1'h0	SPI1	write 1 to set module enable, write 0 has no effect
[19]			RSVD	
[18]	w	1'h0	BTIM2	write 1 to set module enable, write 0 has no effect
[17]	w	1'h0	BTIM1	write 1 to set module enable, write 0 has no effect
[16]	w	1'h0	GPTIM2	write 1 to set module enable, write 0 has no effect
[15]	w	1'h0	GPTIM1	write 1 to set module enable, write 0 has no effect
[14]	w	1'h0	TRNG	write 1 to set module enable, write 0 has no effect
[13]	w	1'h0	CRC1	write 1 to set module enable, write 0 has no effect
[12]	w	1'h0	AES	write 1 to set module enable, write 0 has no effect
[11]	w	1'h0	EFUSEC	write 1 to set module enable, write 0 has no effect
[10]	w	1'h0	SYSCFG1	write 1 to set module enable, write 0 has no effect
[9]			RSVD	
[8]	w	1'h0	I2S1	write 1 to set module enable, write 0 has no effect
[7]	w	1'h0	LCDC1	write 1 to set module enable, write 0 has no effect
[6]	w	1'h0	EPIC	write 1 to set module enable, write 0 has no effect
[5]	w	1'h0	EZIP1	write 1 to set module enable, write 0 has no effect

Continued on the next page...

Table 2-6: HPSYS_RCC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	w	1'h0	USART2	write 1 to set module enable, write 0 has no effect
[3]			RSVD	
[2]	w	1'h0	PINMUX1	write 1 to set module enable, write 0 has no effect
[1]	w	1'h0	MAILBOX1	write 1 to set module enable, write 0 has no effect
[0]	w	1'h0	DMAC1	write 1 to set module enable, write 0 has no effect
0x14			**ESR2**	Enable Set Register 2
[31:26]			RSVD	
[25]	w	1'h0	I2C4	write 1 to set module enable, write 0 has no effect
[24]			RSVD	
[23]	w	1'h0	TSEN	write 1 to set module enable, write 0 has no effect
[22]	w	1'h0	GPADC	write 1 to set module enable, write 0 has no effect
[21]			RSVD	
[20]	w	1'h0	AUDPRC	write 1 to set module enable, write 0 has no effect
[19]	w	1'h0	AUDCODEC	write 1 to set module enable, write 0 has no effect
[18:13]			RSVD	
[12]	w	1'h0	USART3	write 1 to set module enable, write 0 has no effect
[11:10]			RSVD	
[9]	w	1'h0	ATIM1	write 1 to set module enable, write 0 has no effect
[8]	w	1'h0	I2C3	write 1 to set module enable, write 0 has no effect
[7]			RSVD	
[6]	w	1'h0	USBC	write 1 to set module enable, write 0 has no effect
[5]			RSVD	
[4]	w	1'h0	SDMMC1	write 1 to set module enable, write 0 has no effect
[3]			RSVD	
[2]	w	1'h0	MPI2	write 1 to set module enable, write 0 has no effect
[1]	w	1'h0	MPI1	write 1 to set module enable, write 0 has no effect
[0]	w	1'h0	GPIO1	write 1 to set module enable, write 0 has no effect
0x18			**ECR1**	Enable Clear Register 1
[31]	w	1'h0	PTC1	write 1 to clear module enable, write 0 has no effect
[30:29]			RSVD	
[28]	w	1'h0	I2C2	write 1 to clear module enable, write 0 has no effect
[27]	w	1'h0	I2C1	write 1 to clear module enable, write 0 has no effect
[26]			RSVD	
[25]	w	1'h0	PDM1	write 1 to clear module enable, write 0 has no effect
[24]			RSVD	
[23]	w	1'h0	SECU1	write 1 to clear module enable, write 0 has no effect
[22]	w	1'h0	EXTDMA	write 1 to clear module enable, write 0 has no effect
[21]	w	1'h0	SPI2	write 1 to clear module enable, write 0 has no effect
[20]	w	1'h0	SPI1	write 1 to clear module enable, write 0 has no effect
[19]			RSVD	
[18]	w	1'h0	BTIM2	write 1 to clear module enable, write 0 has no effect
[17]	w	1'h0	BTIM1	write 1 to clear module enable, write 0 has no effect
[16]	w	1'h0	GPTIM2	write 1 to clear module enable, write 0 has no effect
[15]	w	1'h0	GPTIM1	write 1 to clear module enable, write 0 has no effect
[14]	w	1'h0	TRNG	write 1 to clear module enable, write 0 has no effect
[13]	w	1'h0	CRC1	write 1 to clear module enable, write 0 has no effect
[12]	w	1'h0	AES	write 1 to clear module enable, write 0 has no effect
[11]	w	1'h0	EFUSEC	write 1 to clear module enable, write 0 has no effect
[10]	w	1'h0	SYSCFG1	write 1 to clear module enable, write 0 has no effect
[9]			RSVD	
[8]	w	1'h0	I2S1	write 1 to clear module enable, write 0 has no effect
[7]	w	1'h0	LCDC1	write 1 to clear module enable, write 0 has no effect

Continued on the next page...

Table 2-6: HPSYS_RCC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	w	1'h0	EPIC	write 1 to clear module enable, write 0 has no effect
[5]	w	1'h0	EZIP1	write 1 to clear module enable, write 0 has no effect
[4]	w	1'h0	USART2	write 1 to clear module enable, write 0 has no effect
[3]			RSVD	
[2]	w	1'h0	PINMUX1	write 1 to clear module enable, write 0 has no effect
[1]	w	1'h0	MAILBOX1	write 1 to clear module enable, write 0 has no effect
[0]	w	1'h0	DMAC1	write 1 to clear module enable, write 0 has no effect
0x1C			**ECR2**	Enable Clear Register 2
[31:26]			RSVD	
[25]	w	1'h0	I2C4	write 1 to clear module enable, write 0 has no effect
[24]			RSVD	
[23]	w	1'h0	TSEN	write 1 to clear module enable, write 0 has no effect
[22]	w	1'h0	GPADC	write 1 to clear module enable, write 0 has no effect
[21]			RSVD	
[20]	w	1'h0	AUDPRC	write 1 to clear module enable, write 0 has no effect
[19]	w	1'h0	AUDCODEC	write 1 to clear module enable, write 0 has no effect
[18:13]			RSVD	
[12]	w	1'h0	USART3	write 1 to clear module enable, write 0 has no effect
[11:10]			RSVD	
[9]	w	1'h0	ATIM1	write 1 to clear module enable, write 0 has no effect
[8]	w	1'h0	I2C3	write 1 to clear module enable, write 0 has no effect
[7]			RSVD	
[6]	w	1'h0	USBC	write 1 to clear module enable, write 0 has no effect
[5]			RSVD	
[4]	w	1'h0	SDMMC1	write 1 to clear module enable, write 0 has no effect
[3]			RSVD	
[2]	w	1'h0	MPI2	write 1 to clear module enable, write 0 has no effect
[1]	w	1'h0	MPI1	write 1 to clear module enable, write 0 has no effect
[0]	w	1'h0	GPIO1	write 1 to clear module enable, write 0 has no effect
0x20			**CSR**	Clock Select Register
[31:16]			RSVD	
[15]	rw	1'b0	SEL_USBC	select USB source clock 0 - clk_hpsys; 1 - clk_dll2
[14:13]	rw	2'h0	SEL_TICK	select clock source for systick reference 0 - clk_rtc; 1 - reserved; 2 - clk_hrc48; 3 - clk_hxt48
[12]	rw	1'h1	SEL_PERI	select clk_peri_hpsys source used by USART/SPI/I2C/GPTIM2/BTIM2 0 - clk_hrc48; 1 - clk_hxt48
[11:10]			RSVD	
[9:8]			RSVD	
[7:6]	rw	2'h0	SEL_MPI2	selet MPI2 function clock 0 - clk_peri_hpsys; 1 - clk_dll1; 2 - clk_dll2; 3 - reserved
[5:4]	rw	2'h0	SEL_MPI1	selet MPI1 function clock 0 - clk_peri_hpsys; 1 - clk_dll1; 2 - clk_dll2; 3 - reserved
[3]			RSVD	
[2]	rw	1'h0	SEL_SYS_LP	select clk_hpsys source 0 - selected by SEL_SYS; 1 - clk_wdt
[1:0]	rw	2'h0	SEL_SYS	select clk_hpsys source 0 - clk_hrc48; 1 - clk_hxt48; 2 - reserved; 3 - clk_dll1
0x24			**CFGR**	Clock Configuration Register
[31:22]			RSVD	
[21:16]	rw	6'h2	TICKDIV	systick reference clock is systick reference clock source (selected by SEL_TICK) devided by TICKDIV

Continued on the next page...

Table 2-6: HPSYS_RCC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15]			RSVD	
[14:12]	rw	3’b100	PDIV2	pclk2_hpsys = hclk_hpsys / (2^{PDIV2}), by default divided by 16
[11]			RSVD	
[10:8]	rw	3’b001	PDIV1	pclk_hpsys = hclk_hpsys / (2^{PDIV1}), by default divided by 2

[7:0]	rw	8’h1	HDIV	hclk_hpsys = clk_hpsys / HDIV
				if HDIV=0, hclk_hpsys = clk_hpsys

0x28			**USBCR**	USBC Control Register
[31:3]			RSVD	
[2:0]	rw	3’h4	DIV	USB function clock is USB source clock divided by DIV. After divider, USB function clock must be 60MHz. For example, if USBC clock source is 240MHz clk_dll2, DIV should be 4.
0x2C			**DLL1CR**	DLL1 Control Register
[31]	r	1’b0	READY	0: dll not ready 1: dll ready
[30:28]	rw	3’b0	LOCK_DLY	
[27:25]	rw	3’b0	PU_DLY	
[24:21]	rw	4’b0	DTEST_TR	
[20]	rw	1’b0	DTEST_EN	
[19]	rw	1’b0	BYPASS	
[18]	rw	1’b0	VST_SEL	
[17]	rw	1’b0	PRCHG_EXT	
[16]	rw	1’b1	PRCHG_EN	
[15]	rw	1’b1	MCU_PRCHG	
[14]	rw	1’b1	MCU_PRCHG_EN	
[13]	rw	1’b1	OUT_DIV2_EN	0: dll output not divided 1: dll output divided by 2
[12]	rw	1’b1	IN_DIV2_EN	
[11:8]	rw	4’ha	LDO_VREF	
[7]	rw	1’b0	MODE48M_EN	
[6]	rw	1’b1	XTALIN_EN	
[5:2]	rw	4’h0	STG	DLL lock freqency is decided by STG. DLL output frequency is (STG+1)*24MHz if OUT_DIV2_EN=0 e.g. STG=9,DLL output is 240M
[1]	rw	1’b0	SW	
[0]	rw	1’b0	EN	0: dll disabled 1: dll enabled
0x30			**DLL2CR**	DLL2 Control Register
[31]	r	1’b0	READY	0: dll not ready 1: dll ready
[30:28]	rw	3’b0	LOCK_DLY	
[27:25]	rw	3’b0	PU_DLY	
[24:21]	rw	4’b0	DTEST_TR	
[20]	rw	1’b0	DTEST_EN	
[19]	rw	1’b0	BYPASS	
[18]	rw	1’b0	VST_SEL	
[17]	rw	1’b0	PRCHG_EXT	
[16]	rw	1’b1	PRCHG_EN	
[15]	rw	1’b1	MCU_PRCHG	
[14]	rw	1’b1	MCU_PRCHG_EN	
[13]	rw	1’b1	OUT_DIV2_EN	0: dll output not divided 1: dll output divided by 2
[12]	rw	1’b1	IN_DIV2_EN	

Continued on the next page...

Table 2-6: **HPSYS_RCC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[11:8]	rw	4'ha	LDO_VREF	
[7]	rw	1'b0	MODE48M_EN	
[6]	rw	1'b1	XTALIN_EN	
[5:2]	rw	4'h0	STG	DLL lock freqency is decided by STG. DLL output frequency is (STG+1)*24MHz if OUT_DIV2_EN=0 e.g. STG=9,DLL output is 240M
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x34			**HRCCAL1**	HRC Calibration Register 1
[31]	r	1'b0	CAL_DONE	Calibration done. After a new calibration started, results should be processed only when cal_done asserted.
[30]	rw	1'b0	CAL_EN	Calibration enble. Set to 0 to clear result, then set to 1 to start a new calibration
[29:16]			RSVD	
[15:0]	rw	16'h8000	CAL_LENGTH	Target clk_hxt48 cycles during calibration
0x38			**HRCCAL2**	HRC Calibration Register 2
[31:16]	r	16'h0	HXT_CNT	Total clk_hxt48 cycles during calibration
[15:0]	r	16'h0	HRC_CNT	Total clk_hrc48 cycles during calibration
0x3C			**DBGCLKR**	Debug Clock Register
[31:20]			RSVD	
[19:18]	rw	2'h1	DLL2_OUT_STR	for debug only
[17]	rw	1'b0	DLL2_CG_EN	for debug only
[16]	rw	1'b0	DLL2_OUT_RSTB	for debug only
[15]	rw	1'b0	DLL2_LOOP_EN	for debug only
[14]	rw	1'b0	DLL2_OUT_EN	for debug only
[13]	rw	1'b0	DLL2_LDO_EN	for debug only
[12]	rw	1'b0	DLL2_DBG	for debug only
[11:10]	rw	2'h1	DLL1_OUT_STR	for debug only
[9]	rw	1'b0	DLL1_CG_EN	for debug only
[8]	rw	1'b0	DLL1_OUT_RSTB	for debug only
[7]	rw	1'b0	DLL1_LOOP_EN	for debug only
[6]	rw	1'b0	DLL1_OUT_EN	for debug only
[5]	rw	1'b0	DLL1_LDO_EN	for debug only
[4]	rw	1'b0	DLL1_DBG	for debug only
[3]			RSVD	
[2]	rw	1'b0	CLK_EN	for debug only
[1:0]	rw	2'b0	CLK_SEL	for debug only
0x40			**DBGR**	Debug Register
[31:5]			RSVD	
[4]	rw	1'h0	FORCE_HP	for debug only
[3]	rw	1'h0	FORCE_GPIO	for debug only
[2]	rw	1'h0	FORCE_BUS	for debug only
[1]	rw	1'h0	SYSCLK_SWLP	for debug only
[0]	rw	1'h0	SYSCLK_AON	for debug only
0x44			**DWCFGR**	Deep WFI mode Clock Configuration Register
[31:28]			RSVD	
[27]	rw	1'b0	DLL2_OUT_RSTB	for debug only
[26]	rw	1'b0	DLL2_OUT_EN	for debug only
[25]	rw	1'b0	DLL1_OUT_RSTB	for debug only
[24]	rw	1'b0	DLL1_OUT_EN	for debug only

Continued on the next page...

Table 2-6: HPSYS_RCC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:19]			RSVD	
[18]	rw	1'h1	SEL_SYS_LP	select clk_hpsys source during deep WFI 0 - selected by SEL_SYS; 1 - clk_wdt
[17:16]	rw	2'h0	SEL_SYS	select clk_hpsys source during deep WFI 0 - clk_hrc48; 1 - clk_hxt48; 2 - RSVD; 3 - clk_dll1
[15]	rw	1'h1	DIV_EN	enable PDIV1, PDIV2 and HDIV reconfiguration during deep wfi
[14:12]	rw	3'b001	PDIV2	pclk2_hpsys = hclk_hpsys / (2^{PDIV2}) during deep wfi
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk_hpsys = hclk_hpsys / (2^{PDIV1}) during deep wfi
[7:0]	rw	8'h1	HDIV	hclk_hpsys = clk_hpsys / HDIV during deep wfi

3 Power Management

3.1 Introduction

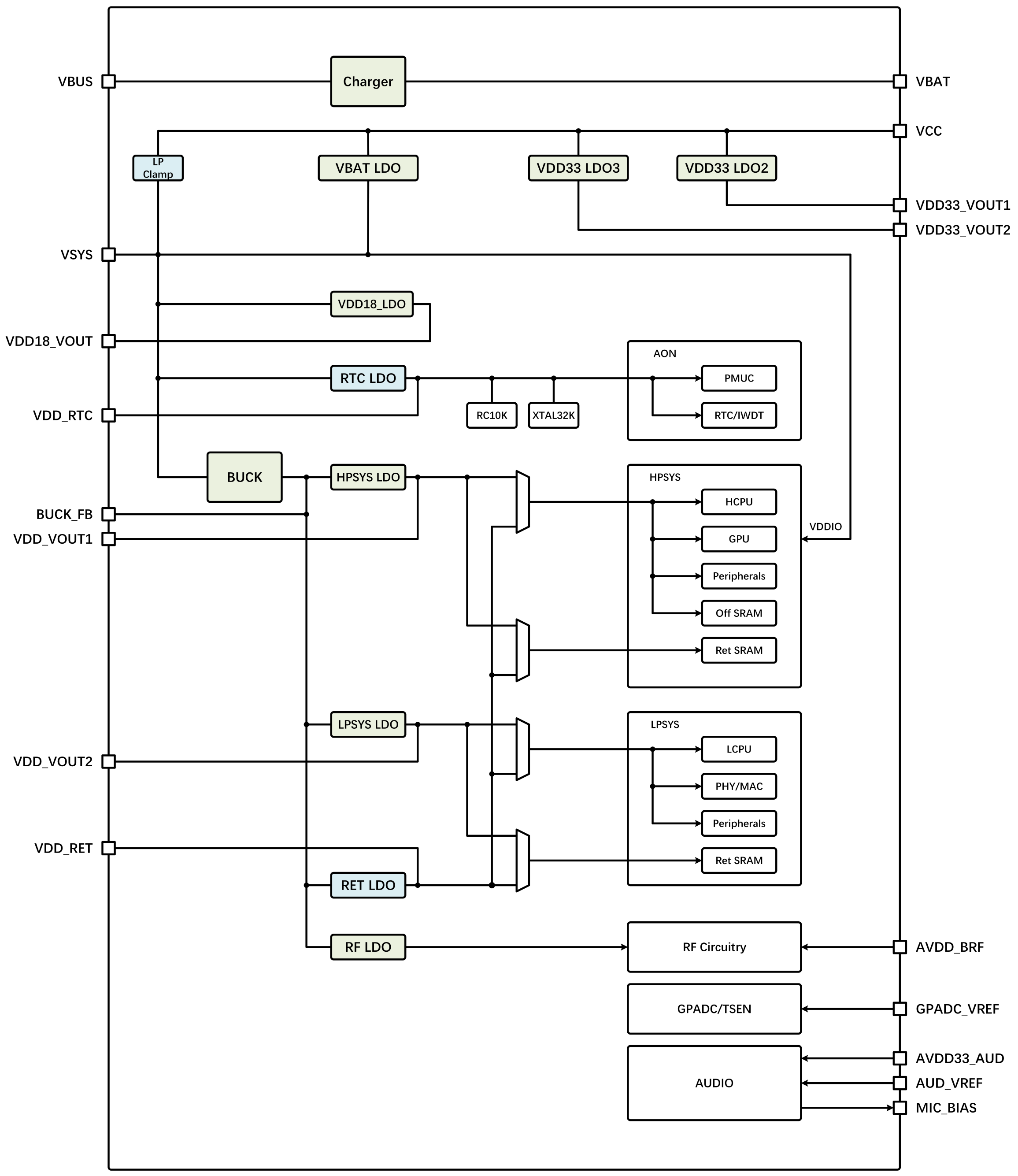


Figure 3-1: QFN Package Power Management Architecture

3.2 Charging Module

The chip integrates a lithium battery charging module. The charging current and full voltage can be adjusted, and the charging current supports up to 560 mA. Customers can set the corresponding parameters according to the battery specifications and the wire resistance of VBUS.

The following figure 3-2 is the charging curve of the battery. When the battery voltage is lower than V$_{cc}$, the charging module is in Trickle Charge mode, which will charge the battery with a lower current I$_{tri}$. When the battery voltage is higher than V$_{cc}$, the charging module is in Constant Charge mode and charged with constant current I$_{cc}$ until the battery voltage is close to the set full voltage V$_{cv}$. After that, the charging module enters the Constant Voltage mode. In this mode, the charging current will slowly drop until the current is less than the cut-off charging current I$_{end}$, the charging loop is automatically disconnected and enters the Charger Full mode. After the battery is fully charged, if the power adapter is not disconnected, the battery voltage is reduced to the Re-Charge Threshold after a period of time, and the charging program will automatically start until the battery is fully charged.

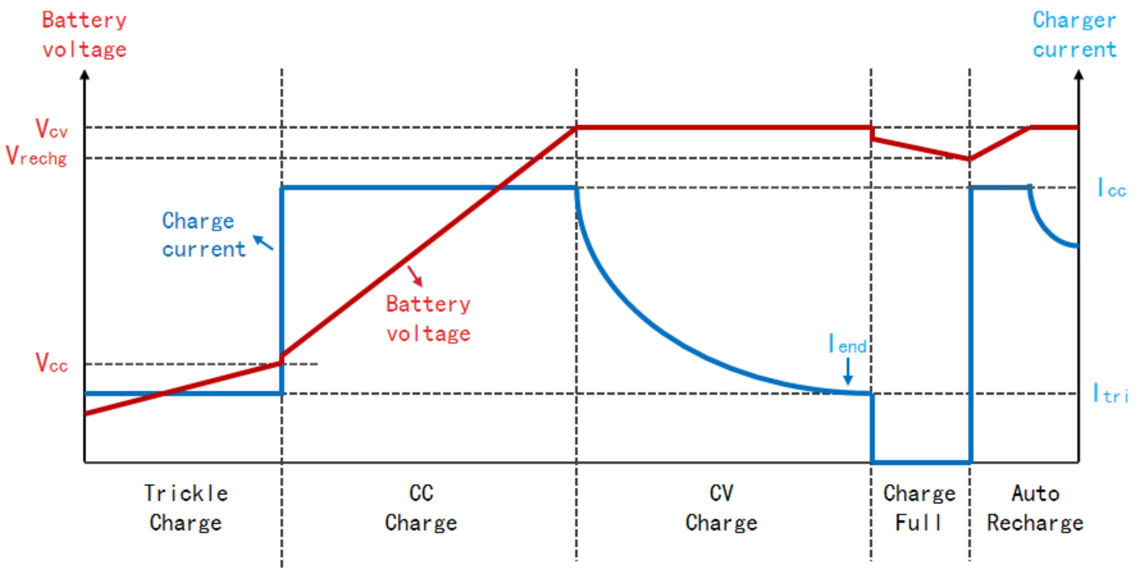


Figure 3-2: **Charging Curve**

3.3 PMUC Register

PMUC base address is 0x500CA000.

Table 3-1: **PMUC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31:20]			RSVD	
[19:15]	rw	5’h1	PIN1_SEL	
[14:10]	rw	5’h0	PIN0_SEL	select one out of PA[44:24]. 0 - PA24, 1 - PA25, 20 - PA44, etc.
[9:7]	rw	3’h0	PIN1_MODE	0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge
[6:4]	rw	3’h0	PIN0_MODE	4/6 - both edge (high-active detection), 5/7 - both edge (low-active detection)
[3]	rw	1’b0	PIN_RET	If set to 1, IO retained during hibernate mode; otherwise, high-Z
[2]	rw	1’h0	REBOOT	Write 1 to reboot; write 0 to clear after boot up

Continued on the next page...

Table 3-1: **PMUC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	HIBER_EN	Write 1 to enter hibernate mode; write 0 to clear when exit from hibernate
[0]	rw	1'h0	SEL_LPCLK	LP clock for watchdog and FSM. 0 - LRC10, 1 - LRC32
0x04			**WER**	Wakeup Enable register
[31:9]			RSVD	
[8]	rw	1'b0	CHG	
[7]	rw	1'b0	LOWBAT	If enabled, auto shut down upon battery low; and will power up if battery ready
[6:5]			RSVD	
[4]	rw	1'b0	PIN1	Set 1 to enable PIN1 as wakeup source
[3]	rw	1'b0	PIN0	Set 1 to enable PIN0 as wakeup source
[2]	rw	1'b0	WDT2	Set 1 to enable WDT2 as reboot cause
[1]	rw	1'b0	WDT1	Set 1 to enable WDT1 as reboot cause
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x08			**WSR**	Wakeup Status register
[31:9]			RSVD	
[8]	r	1'b0	CHG	
[7]	r	1'b0	LOWBAT	Indicates auto reboot due to battery low
[6]	r	1'b0	PWRKEY	
[5]	r	1'b0	IWDT	
[4]	r	1'b0	PIN1	
[3]	r	1'b0	PIN0	
[2]	r	1'b0	WDT2	Indicates reboot by WDT2
[1]	r	1'b0	WDT1	Indicates reboot by WDT1
[0]	r	1'b0	RTC	Indicates the wakeup status from RTC. Note: the status is masked by WER
0x0C			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:8]			RSVD	
[7]	w1c	1'b0	LOWBAT	Write 1 to clear LOWBAT flag
[6]	w1c	1'b0	PWRKEY	Write 1 to clear PWRKEY reset flag
[5]			RSVD	
[4]	w1c	1'b0	PIN1	Write 1 to clear PIN1 wakeup flag.
[3]	w1c	1'b0	PIN0	Write 1 to clear PIN0 wakeup flag. Only valid if PIN wakeup is configured as edge trigger
[2]	w1c	1'b0	WDT2	Write 1 to clear WDT2 reboot flag
[1]	w1c	1'b0	WDT1	Write 1 to clear WDT1 reboot flag
[0]			RSVD	
0x10			**VRTC_CR**	VRTC Control Register
[31:13]			RSVD	
[12:9]	rw	4'h0	BOR_VT_TRIM	
[8]	rw	1'h1	BOR_EN	Brownout Reset Enable
[7:4]	rw	4'h7	VRTC_TRIM	
[3:0]	rw	4'hc	VRTC_VBIT	
0x14			**VRET_CR**	VRET Control Register
[31]	r	1'h0	RDY	
[30:22]			RSVD	
[21:16]	rw	6'h20	DLY	VRET_LDO power up delay in number of CLK_LP cycles
[15:14]			RSVD	
[13:10]	rw	4'h7	TRIM	
[9:6]			RSVD	
[5:2]	rw	4'h7	VBIT	
[1]	rw	1'h0	BM	
[0]	rw	1'h1	EN	
0x18			**LRC10_CR**	RC10K Control Register

Continued on the next page...

Table 3-1: PMUC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31]	r	1'b1	RDY	
[30:9]			RSVD	
[8]	rw	1'h0	REFRES	
[7:6]	rw	2'h3	CHGCAP	
[5:4]	rw	2'h3	CHGCRT	
[3]	rw	1'h0	CMPBM2	
[2:1]	rw	2'h0	CMPBM1	
[0]	rw	1'h1	EN	Enabled by default
0x1C			**LRC32_CR**	RC32K Control Register
[31]	r	1'b0	RDY	
[30:10]			RSVD	
[9:6]	rw	4'h6	RSEL	
[5:4]	rw	2'h0	CHGCRT	
[3]	rw	1'h0	CMPBM2	
[2:1]	rw	2'h0	CMPBM1	
[0]	rw	1'h1	EN	Disabled by default
0x20			**LXT_CR**	XTAL32K Control Register
[31]	r	1'h0	RDY	
[30:16]			RSVD	
[15]	rw	1'h0	EXT_EN	use external 32K from Pin
[14]	rw	1'h1	CAP_SEL	
[13:10]	rw	4'hf	BMSTART	
[9]	rw	1'h1	BMSEL	
[8]	rw	1'h0	AMPCTRL_ENB	
[7:6]	rw	2'h2	AMP_BM	
[5:2]	rw	4'h2	BM	
[1]	rw	1'h0	RSN	
[0]	rw	1'h0	EN	
0x24			**AON_BG**	AON Bandgap Register
[31:6]			RSVD	
[5]	rw	1'h0	BUF_VOS_POLAR	
[4:3]	rw	2'h3	BUF_VOS_STEP	
[2:0]	rw	3'h0	BUF_VOS_TRIM	
0x28			**AON_LDO**	AON LDO Register
[31:7]			RSVD	
[6:4]	rw	3'h1	VBAT_POR_TH	
[3:0]	rw	4'h6	VBAT_LDO_SET_VOUT	
0x2C			**BUCK_CR1**	BUCK Control Register 1
[31]	r	1'h1	SS_DONE	
[30]	rw	1'h0	BG_BUF_VOS_POLAR	
[29:28]	rw	2'h3	BG_BUF_VOS_STEP	
[27:25]	rw	3'h0	BG_BUF_VOS_TRIM	
[24]	rw	1'b0	UVLO_X_BIAS	
[23]	rw	1'b0	ZCD_AON	
[22]	rw	1'b0	OCP_AON	
[21]	rw	1'b0	SEL_LX22	
[20]	rw	1'b0	SEL_IOCP_HI	
[19:17]	rw	3'h4	IOCP_TUNE	
[16:15]	rw	2'h2	COMP_IDYN_TUNE	
[14:13]	rw	2'h1	COMP_IQ_TUNE	
[12]	rw	1'h0	COMP_BM_AHI	
[11:9]	rw	3'h4	COT_CTUNE	

Continued on the next page...

Table 3-1: PMUC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8:6]	rw	3'h4	MOT_CTUNE	
[5:2]			RSVD	
[1]	rw	1'h0	CTRL	
[0]	rw	1'h1	EN	
0x30			**BUCK_CR2**	BUCK Control Register 2
[31:28]	rw	4'h4	TDIS	Discharge for TDIS*4 LP clock cycles during reboot
[27:24]	rw	4'h3	SET_VOUT_L	0.75V
[23:20]	rw	4'ha	SET_VOUT_M	1.1V
[19]	rw	1'h0	FORCE_RDY	
[18]	rw	1'h0	BYPASS_UVLO	
[17]	rw	1'h0	BYPASS_OCP	
[16]	rw	1'h0	BYPASS_PG	
[15:12]	rw	4'h0	L2M_CNT	
[11:8]	rw	4'h0	L2H_CNT	
[7:4]	rw	4'h0	M2H_CNT	
[3]	rw	1'h0	L2M_EN	
[2]	rw	1'h0	M2L_EN	
[1]	rw	1'h0	H2L_EN	
[0]	rw	1'h0	H2M_EN	
0x34			**CHG_CR1**	Charger Control Register 1
[31:26]	rw	6'h2a	CV_VCTRL	
[25:24]	rw	2'h0	CC_RANGE	
[23:19]	rw	5'hf	CC_MN	
[18:14]	rw	5'hf	CC_MP	
[13:8]	rw	6'h5	CC_VCTRL	
[7:2]	rw	6'hc	CC_ICTRL	
[1]	rw	1'b0	LOOP_EN	only available when CR3 FORCE_CTRL bit is set
[0]	rw	1'b0	EN	only available when CR3 FORCE_CTRL bit is set
0x38			**CHG_CR2**	Charger Control Register 2
[31:28]	rw	4'h8	VBAT_RANGE	
[27]	rw	1'b0	RANGE_EOC	
[26:24]	rw	3'h3	BM_EOC	
[23:18]	rw	6'h36	HIGH_VCTRL	
[17:12]	rw	6'h23	REP_VCTRL	
[11:6]	rw	6'h2f	PRECC_ICTRL	
[5:4]	rw	2'h2	PRECC_RANGE	
[3:0]	rw	4'h7	BG_PROG_V1P2	
0x3C			**CHG_CR3**	Charger Control Register 3
[31]	rw	1'b0	FORCE_CTRL	When charger plugged out, this bit will auto reset
[30]	rw	1'b0	FORCE_RST	When charger plugged out, this bit will auto reset
[29:11]			RSVD	
[10:6]	rw	5'hf	DLY2	
[5:0]	rw	6'h3f	DLY1	
0x40			**CHG_CR4**	Charger Control Register 4
[31:29]	rw	3'h2	IM_EOC_MODE	
[28:26]	rw	3'h2	IM_CV_MODE	
[25:23]	rw	3'h2	IM_CC_MODE	
[22:20]	rw	3'h2	IM_ABOVE_CC	
[19:17]	rw	3'h3	IM_ABOVE_REP	
[16:14]	rw	3'h2	IM_VBAT_HIGH	
[13:11]	rw	3'h2	IM_VBUS_RDY	0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, others - both edge
[10:8]			RSVD	

Continued on the next page...

Table 3-1: PMUC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'b0	IE_EOC	
[6]	rw	1'b0	IE_EOC_MODE	
[5]	rw	1'b0	IE_CV_MODE	
[4]	rw	1'b0	IE_CC_MODE	
[3]	rw	1'b0	IE_ABOVE_CC	
[2]	rw	1'b0	IE_ABOVE_REP	
[1]	rw	1'h0	IE_VBAT_HIGH	
[0]	rw	1'h0	IE_VBUS_RDY	
0x44			**CHG_CR5**	Charger Control Register 5
[31:24]			RSVD	
[23]	r	1'b0	IS_EOC	
[22]	r	1'b0	IS_EOC_MODE	
[21]	r	1'b0	IS_CV_MODE	
[20]	r	1'b0	IS_CC_MODE	
[19]	r	1'b0	IS_ABOVE_CC	
[18]	r	1'b0	IS_ABOVE_REP	
[17]	r	1'b0	IS_VBAT_HIGH	
[16]	r	1'b0	IS_VBUS_RDY	
[15:8]			RSVD	
[7]	w1c	1'b0	IC_EOC	
[6]	w1c	1'b0	IC_EOC_MODE	
[5]	w1c	1'b0	IC_CV_MODE	
[4]	w1c	1'b0	IC_CC_MODE	
[3]	w1c	1'b0	IC_ABOVE_CC	
[2]	w1c	1'b0	IC_ABOVE_REP	
[1]	w1c	1'b0	IC_VBAT_HIGH	
[0]	w1c	1'b0	IC_VBUS_RDY	
0x48			**CHG_SR**	Charger Status Register
[31:15]			RSVD	
[14:8]	r	7'h0	CHG_STATE	Charger finite state machine
[7]			RSVD	
[6]	r	1'b0	EOC_MODE	
[5]	r	1'b0	CV_MODE	
[4]	r	1'b0	CC_MODE	
[3]	r	1'b0	VBAT_ABOVE_CC_OUT	
[2]	r	1'b0	VBAT_ABOVE_REP_OUT	
[1]	r	1'b0	VBAT_HIGH_OUT	
[0]	r	1'b0	VBUS_RDY_OUT	
0x4C			**HPSYS_LDO**	HPSYS LDO Control Register
[31:17]			RSVD	
[16]	r	1'b0	RDY	
[15:10]	rw	6'h8	DLY	HPSYS_LDO power up delay in CLK_LP cycles
[9:6]	rw	4'h0	VREF2	Lower voltage for deep sleep mode (0.6V)
[5:2]	rw	4'h5	VREF	optional voltage (0.9V)
[1]	rw	1'b0	BP	
[0]	rw	1'b1	EN	
0x50			**LPSYS_LDO**	LPSYS LDO Control Register
[31:17]			RSVD	
[16]	r	1'b0	RDY	
[15:10]	rw	6'h8	DLY	LPSYS_LDO power up delay in CLK_LP cycles
[9:6]	rw	4'h0	VREF2	Lower voltage for deep sleep mode (0.6V)
[5:2]	rw	4'h7	VREF	optional voltage (1.0V)

Continued on the next page...

Table 3-1: PMUC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1’b0	BP	
[0]	rw	1’b1	EN	
0x54			**HPSYS_SWR**	HPSYS Switch Register
[31]	r	1’b0	RDY	
[30:8]			RSVD	
[7]	rw	1’b0	NORET	Cut off VHPMEM entirely during standby. No retention
[6:4]	rw	3’h3	DLY	wait for N cycles before asserting RDY
[3:2]	rw	2’b01	PSW_RET	PSW value during DS/SB
[1:0]	rw	2’b10	PSW	[0] - RET_LDO; [1] - HPSYS_LDO
0x58			**LPSYS_SWR**	LPSYS Switch Register
[31]	r	1’b0	RDY	
[30:8]			RSVD	
[7]	rw	1’b0	NORET	Cut off VLPMEM entirely during standby. No retention
[6:4]	rw	3’h3	DLY	wait for N cycles before asserting RDY
[3:2]	rw	2’b01	PSW_RET	PSW value during DS/SB
[1:0]	rw	2’b10	PSW	[0] - RET_LDO; [1] - LPSYS_LDO
0x5C			**PERI_LDO**	Peripherals LDO
[31:22]			RSVD	
[21]	rw	1’b0	VDD33_LDO3_PD	
[20:17]	rw	4’h6	VDD33_LDO3_SET_VOUT	
[16]	rw	1’b0	EN_VDD33_LDO3	
[15:14]			RSVD	
[13]	rw	1’b0	VDD33_LDO2_PD	
[12:9]	rw	4’h6	VDD33_LDO2_SET_VOUT	
[8]	rw	1’b0	EN_VDD33_LDO2	
[7:6]			RSVD	
[5]	rw	1’b0	LDO18_PD	
[4:1]	rw	4’hc	LDO18_VREF_SEL	
[0]	rw	1’b0	EN_LDO18	
0x60			**PMU_TR**	PMU Test Register
[31:6]			RSVD	
[5:3]	rw	3’h0	PMU_DC_MR	macro select
[2:0]	rw	3’h0	PMU_DC_TR	test point select
0x64			**PMU_RSVD**	PMU Reserved Register
[31:24]	r	8’h0	RESERVE3	
[23:16]	rw	8’h0	RESERVE2	
[15:8]	rw	8’h0	RESERVE1	
[7:0]	rw	8’h0	RESERVE0	
0x68			**HXT_CR1**	HXT48 Control Register 1
[31:30]			RSVD	
[29:20]	rw	10’h1ca	CBANK_SEL	
[19]	rw	1’b1	GM_EN	
[18:17]	rw	2’h3	LDO_FLT_RSEL	
[16:13]	rw	4’ha	LDO_VREF	
[12:11]	rw	2’h1	BUF_RF_STR	
[10:9]	rw	2’h1	BUF_AUD_STR	
[8]	rw	1’b0	BUF_AUD_EN	
[7:6]	rw	2’h1	BUF_DLL_STR	
[5]	rw	1’b0	BUF_DLL_EN	
[4:3]	rw	2’h1	BUF_DIG_STR	
[2]	rw	1’b1	BUF_DIG_EN	
[1]	rw	1’b1	BUF_EN	

Continued on the next page...

Table 3-1: **PMUC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1’b1	EN	
0x6C			**HXT_CR2**	HXT48 Control Register 2
[31]	rw	1’b0	SLEEP_EN	
[30:29]	rw	2’h2	SDADC_CLKDIV2_SEL	
[28:27]	rw	2’h2	SDADC_CLKDIV1_SEL	
[26]	rw	1’b0	SDADC_CLKIN_EN	
[25:16]	rw	10’h32	IDAC	
[15]	rw	1’b0	IDAC_EN	
[14:13]	rw	2’h2	BUF_SEL3	
[12:11]	rw	2’h2	BUF_SEL2	
[10]	rw	1’h0	ACBUF_RSEL	
[9:8]	rw	2’h2	ACBUF_SEL	
[7:6]	rw	2’h1	AGC_VINDC	
[5:2]	rw	4’h8	AGC_VTH	
[1]	rw	1’b0	AGC_ISTART_SEL	
[0]	rw	1’b1	AGC_EN	
0x70			**HXT_CR3**	HXT48 Control Register 3
[31:10]			RSVD	
[9:4]	rw	6’d31	DLY	
[3:2]	rw	2’h1	BUF_OSLO_STR	
[1:0]	rw	2’h1	BUF_DAC_STR	
0x74			**HRC_CR**	HRC48 Control Register
[31]	rw	1’b0	DLY	number of cycles for BG ready. 0 - one cycle of CLK_LP; 1 - two cycles of CLK_LP
[30]			RSVD	
[29:28]	rw	2’h0	CLKLP_STR	
[27:26]	rw	2’h2	CLKLP_SEL	
[25]	rw	1’b1	CLKLP_EN	
[24:23]	rw	2’h0	CLKHP_STR	
[22:21]	rw	2’h2	CLKHP_SEL	
[20]	rw	1’b1	CLKHP_EN	
[19]			RSVD	
[18]	rw	1’b0	CLK96M_EN	
[17:15]	rw	3’h1	TEMP_TRIM	
[14:5]	rw	10’h200	FREQ_TRIM	
[4:1]	rw	4’ha	LDO_VREF	
[0]	rw	1’b1	EN	
0x78			**DBL96_CR**	DBL96 Control Register
[31:29]			RSVD	
[28:18]	rw	11’h1a6	DLY_SEL_EXT	
[17]	rw	1’b0	DLY_SEL_EXT_EN	
[16]	rw	1’b0	DLY_EXT_EN	
[15:12]	rw	4’h1	DLY_EN	
[11:8]	rw	4’b0	PH_EN	
[7]	rw	1’b0	LOOP_RSTB	
[6]	rw	1’b0	TOOSLO_EN	
[5]	rw	1’b0	TORF_EN	
[4:3]	rw	2’h1	TODIG_STR	
[2]	rw	1’b0	TODIG_EN	
[1]	rw	1’b0	OUT_EN	
[0]	rw	1’b0	EN	
0x7C			**DBL96_CALR**	DBL96 Calibration Register
[31:14]			RSVD	

Continued on the next page...

Table 3-1: PMUC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13]	r	1’h0	CAL_LOCK	
[12:2]	r	11’h0	CAL_OP	
[1]	rw	1’b0	CAL_CLOSE_EXT_EN	
[0]	rw	1’b0	CAL_EN	
0x80			**CAU_BGR**	CAU Bandgap Register
[31:11]			RSVD	
[10:7]	rw	4’ha	LPBG_VREF12	
[6:3]	rw	4’ha	LPBG_VREF06	
[2]	rw	1’b0	LPBG_EN	
[1]	rw	1’b0	HPBG_EN	
[0]	rw	1’b0	HPBG_VDDPSW_EN	
0x84			**CAU_TR**	CAU Test Register
[31:9]			RSVD	
[8:6]	rw	3’h0	CAU_DC_MR	
[5:3]	rw	3’h0	CAU_DC_BR	
[2:0]	rw	3’h0	CAU_DC_TR	
0x88			**CAU_RSVD**	CAU Reserved Register
[31:24]			RSVD	
[23:16]	r	8’h0	RESERVE2	
[15:8]	rw	8’h0	RESERVE1	
[7:0]	rw	8’h0	RESERVE0	
0x8C			**WKUP_CNT**	Wakeup Count Register
[31:16]	rw	16’hffff	PIN1_CNT	
[15:0]	rw	16’hffff	PIN0_CNT	
0x90			**PWRKEY_CNT**	PowerKey Count Register
[31:20]			RSVD	
[19:4]	rw	16’h186a	RST_CNT	press high for RST_CNT*16 CLK_WDT cycles to reset the whole chip
[3:0]			RSVD	
0x94			**HPSYS_VOUT**	
[31:4]			RSVD	
[3:0]	rw	4’hb	VOUT	0xD - 1.2V, 0xA - 1.1V, 0x8 - 1.0V, 0x5 - 0.9V
0x98			**LPSYS_VOUT**	
[31:4]			RSVD	
[3:0]	rw	4’h5	VOUT	0x8 - 1.0V, 0x5 - 0.9V
0x9C			**BUCK_VOUT**	
[31:4]			RSVD	
[3:0]	rw	4’hd	VOUT	0xF - 1.35V, 0xD - 1.25V, 0x9 - 1.05V, 0x6 - 0.9V, 0x2 - 0.7V

4 Low Power Mode

4.1 Introduction

The chip supports multiple low power modes to meet the low power requirements across various scenarios.

4.2 Summary of Main Operating Mode

The main operating modes of the chip HPSYSsubsystem are detailed in the table below. Except for the activemode, all other modes are classifed as low power modes.

Table 4-1: **Chip Operating Modes**

Mode	CPU	Peripheral	SRAM	IO	LPTIM	Wake-Up Source	Wake-Up Time
Active	run	run	Accessible	Normal	run	/	/
Sleep	stop	run	Accessible	Normal	run	Any Interrupt	<1us
Deepsleep	stop	stop	Not Accessible, Fully Retained	Level Holding	run	RTC,Wake-Up PIN, IO(PA), LPTIM1,Bluetooth	~250us
Standby	reset	reset	Not Accessible, Retain 384KB	Level Holding	run	RTC,Wake-Up PIN, LPTIM1,Bluetooth	~1ms
Hibernate	reset	reset	Not Accessible, Not Retained	High Impedance// Pull-Up/Pull-Down	reset	RTC,Wake-Up PIN	>2ms

The LPSYS subsystem is specifically designed for Bluetooth communication, automatically entering and exiting low power mode based on Bluetooth activity, and will automatically wake up the HPSYS when necessary.

4.2.1 Active Mode

In Active mode, the CPU and peripherals operate normally, SRAM is accessible, and IO can toggle as expected. To reduce runtime power consumption, appropriate clock sources and clock frequencies can be selected for the CPU and peripherals, and the peripheral module enable should only be activated when the peripherals are in use. In Active mode, HPSYS can operate in either basic mode or enhanced mode to meet the requirements of low power or high-performance scenarios.

In the basic operating mode, the HCPU has a maximum operating frequency of 48MHz, while the MPI also has a maximum operating frequency of 48MHz, primarily utilized for sensor data collection, simple algorithm computations, and other low power consumption scenarios. In the enhanced operating mode, the HCPU can reach a maximum operating frequency of 240MHz, mainly for image and audio processing, complex algorithm computations, and other high-performance applications.

Upon startup, the chip defaults to the basic operating mode. The HCPU must first initialize all power-related configurations, after which it can switch between basic mode and enhanced mode. The HCPU can perform the following operations to switch to enhanced operating mode:

1. Set the SYSCR_LDO_VSEL in HPSYS_CFG to 0;
2. Delay 1 ms; during the delay process, high-speed clocks such as clk_hxt48 and clk_dll1 can be enabled in parallel.
3. Set the ULPMCR register of HPSYS_CFG to 0x00130213.
4. Switch clk_hpsys to clk_dll1 ; the MPI operating clock can also be switched to a high-speed clock.

HCPU can perform the following operations to switch to the basic operating mode:

1. Switch clk_hpsys to clk_hrc48 ; switch the MPI operating clock to clk_peri_hpsys ; and turn off the high-speed clock.
2. Set the ULPMCR register of HPSYS_CFG to 0x00110331.
3. Set the SYSCR_LDO_VSEL register of HPSYS_CFG to 1 ;

The HCPU can enter Low Power Mode when idle. All Low Power Modes can only be entered from activemode, and upon exit, it returns to active mode.

4.2.2 Sleep Mode

In sleep mode, the CPU pauses instruction execution, while peripherals continue to operate normally, SRAM is accessible, and IO can toggle normally.

When the PMR_MODE register of HPSYS_AON is set to 0, the HCPU can enter sleep mode by executing the WFI or WFE instruction. The HCPU exits sleep mode upon receiving any enabled interrupt request and continues execution from the position it was at before entering sleep mode.

4.2.3 Deepsleep Mode

In deepsleep mode, most clocks in the system are turned off, leaving only the low power clock active. The CPU and peripherals cease operation. SRAM is inaccessible, but its contents can be retained. The configured output IO (pins excluding PA24~PA27) cannot toggle and must maintain the level prior to entering deepsleep mode.

The process for HPSYS to enter deepsleep mode is as follows:

1. Ensure that the tasks of all peripherals in HPSYS (except LPTIM1) are completed;
2. HCPU completes the currently reported interrupts;
3. HCPU disables all interrupt enables through the NVIC register to prevent generating exceptions such as SysTick, SVCall and PendSV, and sets PRIMASK to 1;
4. Clock clk_hpsys switch to clk_hrc48, disable clk_dll1/2/3and other high-speed clocks;
5. In HPSYS_AON set the wake-up source;
6. Set ISSR_HP_ACTIVE of HPSYS_AON to 0 ;
7. Set Configure the PMR_MODE register of HPSYS_AON to 2 ;
8. HCPU executes the WFI instruction.

After executing the above process, if there are still interrupts that have not been closed and an interrupt or exception request occurs, the attempt to enter deepsleep mode fails, and the CPU will continue to operate in active mode. If this occurs, you can retry entering deepsleepmode after handling the current interrupt.

For debugging convenience, write 0x80000002 to the PMR_MODE register in step 7 to ignore the current interrupt status and force entry into deepsleep mode.

The deepsleep mode has primary wake-up sources, including RTC, wake-up PIN, IO(PA), LPTIM1, as well as wake-up requests from LPSYS and MAILBOX2. The wake-up sources are configured via the HPSYS_AON WER register.

When the wake-up sources are activated, the subsystem exits the deepsleep mode after a brief initialization period (approximately 250us) and returns to active mode. The states of the CPU , peripherals, and memory are preserved, allowing the CPU to continue running from the point it entered deepsleep (after the WFI instruction).

Upon exiting the deepsleep mode, the subsystem generates an AON interrupt. The CPU should first execute the corresponding recovery process.

After exiting the deepsleep mode, the recommended recovery process for the HCPU is as follows:

1. Set the PMR_MODE register of HPSYS_AON to 0;
2. Enable the AON interrupt via the NVIC register and set PRIMASK to 0;
3. In the AON interrupt, query the wake-up source using the WSR register of HPSYS_AON and perform the corresponding processing, then clear the AON interrupt flag using the WCR register of HPSYS_AON;
4. Set the ISSR_HP_ACTIVE of HPSYS_AON to 1;
5. Enable other interrupts for the HCPU through the NVIC register;
6. If necessary, re-enable the high-speed clock.

When the HCPU clears the wake-up source flag, it should be noted that only the PIN wake-up flag is cleared through the WCR register of HPSYS_AON, while other wake-up source flags must be cleared by configuring the corresponding module that generates the wake-up source.

After exiting the deepsleep mode, the subsystem clock source is clk_hrc48 . If a switch to another clock is required, the necessary clock source should be re-enabled.

After exiting the deepsleep mode, the register contents of each peripheral in the subsystem will not be reset, and each peripheral will maintain its state prior to entering the deepsleep mode.

4.2.4 Standby Mode

In standby mode, most clocks and power supplies within the system are turned off, retaining only the clocks and power supplies for low power consumption modules. The CPU and peripherals are both reset. SRAM is inaccessible but can retain 384KB (0x20020000 to 0x2007FFFF). IO configured as output (except for PA24~PA27) cannot toggle and will maintain the level prior to entering standby mode.

The process for HPSYS entering standby mode is as follows:

1. Ensure that the tasks of all peripherals in HPSYS (except LPTIM1) are completed;
2. Backup the contents of SRAM and the status of peripherals in the retained SRAM of HPSYS.
3. HCPU completes the processing of the currently reported interrupts.
4. HCPU disables all interrupt through the NVIC register to avoid generating SysTick, SVCall and PendSV exceptions, and sets PRIMASK to 1.
5. The clock clk_hpsys is switched to clk_hrc48.
6. In HPSYS_AON set the wake-up source;
7. Set ISSR_HP_ACTIVE of HPSYS_AON to 0 ;
8. Set the ANACR register of HPSYS_AON is configured to 3.

9. Set the PMR_MODE register of HPSYS_AON to 3.
10. HCPU executes the WFI instruction.

After executing the above process, if there are still interrupts that have not been closed and an interrupt or exception request is generated, the attempt to enter standby mode will fail, and theCPU will remain in active mode. If this occurs, you may attempt to re-enter standby mode after handling the current interrupt.

For debugging convenience, in step 9 above, writing 0x80000003 to the PMR_MODE register will disregard the current interrupt status and force entry into standby mode.

When HPSYS is in standby mode, the primary wake-up sources include RTC, wake-up PIN,LPTIM1, and wake-up requests from LPSYS and MAILBOX2. The wake-up sources are configured via the HPSYS_AON WER register.

When the wake-up source is activated, the system exits standby mode after a brief initialization period (approximately 1ms) and returns to active mode. The status of the CPU and peripherals has been reset. The SRAM retains 384KB data; the CPU begins execution from address 0 in ROM and queries the PMR_MODE register of HPSYS_AON or LPSYS_AON to select different execution branches.

Upon exiting standby mode, the subsystem generates an AON interrupt. After the CPU exits the ROMprogram, it should first execute the corresponding recovery process.

After exiting standby mode, the recommended recovery process for the HCPU is as follows:

1. Set Configure the PMR_MODE of HPSYS_AON and the ANACR register to 0;
2. Enable the AON interrupt through the NVIC register;
3. In the AON interrupt, query the wake-up source using the WSR register of HPSYS_AON and perform the corresponding processing, then clear the AON interrupt flag using the WCR register of HPSYS_AON;
4. Set the ISSR_HP_ACTIVE of HPSYS_AON to 1;
5. Re-enable the high-speed clock;
6. Restore the peripheral configuration and SRAM content from the HPSYS retained SRAM;
7. Enable other interrupts for the HCPU through the NVIC register.

When the HCPU clears the wake-up source flag, it should be noted that only the PIN wake-up flag is cleared through the WCR register of HPSYS_AON, while other wake-up source flags must be cleared by configuring the corresponding module that generates the wake-up source.

After exiting standby mode, the subsystem’s system clock source is clk_hrc48; if a switch to another clock is required, the necessary clock source should be re-enabled.

After exiting standby mode, all peripherals in the subsystem are reset except for low power consumption peripherals (such as LPTIM); configurations should be restored or re-initialized before use. The peripheral state can be backed up in SRAM or PSRAM with retention functionality before entering standby mode, and the backed-up content will not be cleared after entering and exiting standby mode.

4.2.5 Hibernate Mode

The Hibernate Mode represents the chip’s lowest power consumption state. In this mode, most of the chip’s clocks and power supply are disabled, retaining only the clocks and power supply for specific low power modules. The CPU, peripherals,HPSYS (including HPSYS_AON), and LPSYS (including LPSYS_AON) registers are all reset. All SRAM contents are not preserved. The wake-up PIN can be configured as high impedance, pull-up, or pull-down, while other IOs enter a high impedance state.

The procedure for entering hibernate mode is as follows:

1. Configure the wake-up source in the PMUC WER register;
2. Set PMUC's CR_HIBER_EN to 1.

After entering hibernate mode, only the PMUC, RTC, and IWDT modules are operational, and the registers of these modules will not be lost.

The wake-up sources for hibernate mode include the RTC and the wake-up PIN. The IO that supports the wake-up PIN function can still maintain effective pull-up and pull-down resistors (configuration is located in the RTC register, so it will not be lost).

When the wake-up source is activated, the chip exits hibernate mode after a period of initialization time (>2ms) and returns to active mode. The HCPU starts from 0 in ROM. The address begins execution. At this point, the PMUC's CR_HIBER_EN register remains set to 1, allowing the CPU to recognize that this startup is exiting from hibernate mode, enabling it to perform the necessary processing and reset CR_HIBER_EN to 0.

After exiting hibernate mode, the subsystem's system clock source is clk_hrc48. If a switch to another clock is necessary, the required clock source should be re-enabled.

After exiting hibernate mode, all modules except for PMUC, RTC, and IWDT are reset and must be reinitialized before use.

4.2.6 Debugger Behavior in Low Power Mode

Once the chip enters low power mode, debugging becomes more difficult, primarily evidenced by the debugger disconnecting or being unable to connect. The debugger is connected by default through the PA18 and PA19 IO pins.

When connecting the debugger to HCPU,the following situations will cause disconnection and prevent reconnection:

1. The IO function of PA18 or PA19 has been altered;
2. The USART1 or DMAC1 module has been reset, or the clock has been disabled;
3. The ANACR_PA_ISO register of HPSYS_AON is set to 1 (configured when the CPU enters deepsleep or standby mode);
4. HPSYS is in deepsleep or standbymode;
5. The chip is in hibernatemode.

After HPSYS wakes up from deepsleep or standby mode, and the ISSR_HP_ACTIVE of HPSYS_AON is set to 1, the debugger can reconnect to HCPU and access the HPSYS address space.

4.2.7 Determining the Current Low Power Mode

By measuring the voltage at the chip's power pins, the current low power mode can be determined. When HPSYS is in active or sleep mode, the LDO1_VOUT voltage remains at1.0V (basic mode) or1.2V (enhanced mode). When HPSYS is in deepsleep mode, and when HPSYS is in standby mode, the LDO1_VOUT voltage cannot be maintained and will gradually decrease to 0V. When the chip enters hibernate mode, the LDO1_VOUT, LDO2_VOUT, and VDD_RET all drop to 0V.

Table 4-2: Power Pin Voltage in Low Power Mode

HPSYS	VDD_VOUT1	VDD_RET	VDD_RTC
active/ sleep/	1.0V/1.2V	0.7V	1.1V
deepsleep	0.7V	0.7V	1.1V
standby	0V	0.7V	1.1V
hibernate	0V	0V	1.1V
power off	0V	0V	0V

4.3 HPSYS_AON Register

HPSYS_AON base address is 0x500C0000.

Table 4-3: HPSYS_AON Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**PMR**	Power Mode Register
[31]	w1s	1’b0	FORCE_SLEEP	Set 1 to force enter low power mode. Will be cleared automatically
[30]	rw	1’b0	FORCE_LCPU	for debug only
[29:2]			RSVD	
[1:0]	rw	2’h0	MODE	Power Mode: 2’h0 - active; 2’h1 - light sleep; 2’h2 - deep sleep; 2’h3 - standby
0x04			**CR1**	Control Register 1
[31]	rw	1’h0	GTIM_EN	Enable global timer
[30:28]	rw	3’h0	PINOUT_SEL1	select output to PBR 2: inverted LPTIM1 PWM output 3: inverted LPTIM2 PWM output others: reserved
[27:25]	rw	3’h0	PINOUT_SEL0	select output to PBR 2: LPTIM1 PWM output 3: LPTIM2 PWM output others: reserved
[24:12]			RSVD	
[11:9]	rw	3’h0	PIN3_MODE	mode for wakeup PIN3 (PA27)
[8:6]	rw	3’h0	PIN2_MODE	mode for wakeup PIN2 (PA26)
[5:3]	rw	3’h0	PIN1_MODE	mode for wakeup PIN1 (PA25)
[2:0]	rw	3’h0	PIN0_MODE	mode for wakeup PIN0 (PA24) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x08			**CR2**	Control Register 2
[31:24]			RSVD	
[23:21]	rw	3’h0	PIN15_MODE	mode for wakeup PIN15 (PA39)
[20:18]	rw	3’h0	PIN14_MODE	mode for wakeup PIN14 (PA38)
[17:15]	rw	3’h0	PIN13_MODE	mode for wakeup PIN13 (PA37)
[14:12]	rw	3’h0	PIN12_MODE	mode for wakeup PIN12 (PA36)
[11:9]	rw	3’h0	PIN11_MODE	mode for wakeup PIN11 (PA35)
[8:6]	rw	3’h0	PIN10_MODE	mode for wakeup PIN10 (PA34) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
[5:0]			RSVD	
0x0C			**CR3**	Control Register 3
[31:15]			RSVD	
[14:12]	rw	3’h0	PIN20_MODE	mode for wakeup PIN20 (PA44)
[11:9]	rw	3’h0	PIN19_MODE	mode for wakeup PIN19 (PA43)
[8:6]	rw	3’h0	PIN18_MODE	mode for wakeup PIN18 (PA42)

Continued on the next page...

Table 4-3: **HPSYS_AON Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[5:3]	rw	3'h0	PIN17_MODE	mode for wakeup PIN17 (PA41)
[2:0]	rw	3'h0	PIN16_MODE	mode for wakeup PIN16 (PA40) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x10			**ACR**	Active Mode Control register
[31]	r	1'b0	HXT48_RDY	Indicate hxt48 is ready
[30]	r	1'b0	HRC48_RDY	Indicate hrc48 is ready
[29:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	for debug only
[2]	rw	1'b1	PWR_REQ	Request power during Active mode
[1]	rw	1'b1	HXT48_REQ	Request hxt48 in active mode
[0]	rw	1'b1	HRC48_REQ	Request hrc48 in active mode
0x14			**LSCR**	Light Sleep Ctrl Register
[31:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	for debug only
[2]	rw	1'b1	PWR_REQ	Request power during Light Sleep mode
[1]	rw	1'b1	HXT48_REQ	Request hxt48 in Light Sleep mode
[0]	rw	1'b1	HRC48_REQ	Request hrc48 in Light Sleep mode
0x18			**DSCR**	Deep Sleep Ctrl Register
[31:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	for debug only
[2]	rw	1'b1	PWR_REQ	Request power during Deep Sleep mode
[1]	rw	1'b0	HXT48_REQ	Request hxt48 in Deep Sleep mode
[0]	rw	1'b0	HRC48_REQ	Request hrc48 in Deep Sleep mode
0x1C			**SBCR**	Standby Mode Ctrl Register
[31:9]			RSVD	
[8]	rw	1'h0	PD_RAM2	for debug only
[7]	rw	1'h0	PD_RAM1	for debug only
[6]	rw	1'h0	PD_RAM0	for debug only
[5:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	for debug only
[2]	rw	1'b0	PWR_REQ	Request power during Standby mode
[1]	rw	1'b0	HXT48_REQ	Request hxt48 in Standby mode
[0]	rw	1'b0	HRC48_REQ	Request hrc48 in Standby mode
0x20			**WER**	Wakeup Enable register
[31:29]			RSVD	
[28]	rw	1'b0	PIN20	Set 1 to enable PA44 as wakeup source
[27]	rw	1'b0	PIN19	Set 1 to enable PA43 as wakeup source
[26]	rw	1'b0	PIN18	Set 1 to enable PA42 as wakeup source
[25]	rw	1'b0	PIN17	Set 1 to enable PA41 as wakeup source
[24]	rw	1'b0	PIN16	Set 1 to enable PA40 as wakeup source
[23]	rw	1'b0	PIN15	Set 1 to enable PA39 as wakeup source
[22]	rw	1'b0	PIN14	Set 1 to enable PA38 as wakeup source
[21]	rw	1'b0	PIN13	Set 1 to enable PA37 as wakeup source
[20]	rw	1'b0	PIN12	Set 1 to enable PA36 as wakeup source
[19]	rw	1'b0	PIN11	Set 1 to enable PA35 as wakeup source
[18]	rw	1'b0	PIN10	Set 1 to enable PA34 as wakeup source
[17:12]			RSVD	
[11]	rw	1'b0	PIN3	Set 1 to enable PA27 as wakeup source
[10]	rw	1'b0	PIN2	Set 1 to enable PA26 as wakeup source
[9]	rw	1'b0	PIN1	Set 1 to enable PA25 as wakeup source
[8]	rw	1'b0	PIN0	Set 1 to enable PA24 as wakeup source
[7]	rw	1'b0	LP2HP_IRQ	Set 1 to enable MAILBOX2 as wakeup source

Continued on the next page...

Table 4-3: **HPSYS_AON Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw	1’b0	LP2HP_REQ	Set 1 to enable LPSYS request as wakeup source
[5:4]			RSVD	
[3]	rw	1’b0	PMUC	Set 1 to enable PMUC as wakeup source
[2]	rw	1’b0	LPTIM1	Set 1 to enable LPTIM1 as wakeup source
[1]	rw	1’b0	GPIO1	Set 1 to enable IO(PA) as wakeup source
[0]	rw	1’b0	RTC	Set 1 to enable RTC as wakeup source
0x24			**WSR**	Wakeup Status register
[31:29]			RSVD	
[28]	r	1’b0	PIN20	Indicates the wakeup status from PA44 request. Note: the status is masked by WER
[27]	r	1’b0	PIN19	Indicates the wakeup status from PA43 request. Note: the status is masked by WER
[26]	r	1’b0	PIN18	Indicates the wakeup status from PA42 request. Note: the status is masked by WER
[25]	r	1’b0	PIN17	Indicates the wakeup status from PA41 request. Note: the status is masked by WER
[24]	r	1’b0	PIN16	Indicates the wakeup status from PA40 request. Note: the status is masked by WER
[23]	r	1’b0	PIN15	Indicates the wakeup status from PA39 request. Note: the status is masked by WER
[22]	r	1’b0	PIN14	Indicates the wakeup status from PA38 request. Note: the status is masked by WER
[21]	r	1’b0	PIN13	Indicates the wakeup status from PA37 request. Note: the status is masked by WER
[20]	r	1’b0	PIN12	Indicates the wakeup status from PA36 request. Note: the status is masked by WER
[19]	r	1’b0	PIN11	Indicates the wakeup status from PA35 request. Note: the status is masked by WER
[18]	r	1’b0	PIN10	Indicates the wakeup status from PA34 request. Note: the status is masked by WER
[17:12]			RSVD	
[11]	r	1’b0	PIN3	Indicates the wakeup status from PA27 request. Note: the status is masked by WER
[10]	r	1’b0	PIN2	Indicates the wakeup status from PA26 request. Note: the status is masked by WER
[9]	r	1’b0	PIN1	Indicates the wakeup status from PA25 request. Note: the status is masked by WER
[8]	r	1’b0	PIN0	Indicates the wakeup status from PA24 request. Note: the status is masked by WER
[7]	r	1’b0	LP2HP_IRQ	Indicates the wakeup status from MAILBOX2. Note: the status is masked by WER
[6]	r	1’b0	LP2HP_REQ	Indicates the wakeup status from LPSYS request. Note: the status is masked by WER
[5:4]			RSVD	
[3]	r	1’b0	PMUC	Indicates the wakeup status from PMUC. Note: the status is masked by WER
[2]	r	1’b0	LPTIM1	Indicates the wakeup status from LPTIM1. Note: the status is masked by WER
[1]	r	1’b0	GPIO1	Indicates the wakeup status from IO(PA). Note: the status is masked by WER
[0]	r	1’b0	RTC	Indicates the wakeup status from RTC. Note: the status is masked by WER
0x28			**WCR**	Wakeup Clear register
[31]	w1c	1’b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:29]			RSVD	
[28]	w1c	1’b0	PIN20	Write 1 to clear PA44 wakeup source. Only valid if PIN wakeup is configured as edge trigger

Continued on the next page...

Table 4-3: HPSYS_AON Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[27]	w1c	1’b0	PIN19	Write 1 to clear PA43 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[26]	w1c	1’b0	PIN18	Write 1 to clear PA42 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[25]	w1c	1’b0	PIN17	Write 1 to clear PA41 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[24]	w1c	1’b0	PIN16	Write 1 to clear PA40 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[23]	w1c	1’b0	PIN15	Write 1 to clear PA39 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[22]	w1c	1’b0	PIN14	Write 1 to clear PA38 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[21]	w1c	1’b0	PIN13	Write 1 to clear PA37 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[20]	w1c	1’b0	PIN12	Write 1 to clear PA36 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[19]	w1c	1’b0	PIN11	Write 1 to clear PA35 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[18]	w1c	1’b0	PIN10	Write 1 to clear PA34 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[17:12]			RSVD	
[11]	w1c	1’b0	PIN3	Write 1 to clear PA27 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[10]	w1c	1’b0	PIN2	Write 1 to clear PA26 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[9]	w1c	1’b0	PIN1	Write 1 to clear PA25 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[8]	w1c	1’b0	PIN0	Write 1 to clear PA24 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[7:0]			RSVD	
0x2C			**ISSR**	Inter System Wakeup Register
[31:6]			RSVD	
[5]	r	1’h1	LP_ACTIVE	Read 1 indicates LPSYS is active
[4]	rw	1’h1	HP_ACTIVE	Write 1 to indicates HPSYS is active
[3:2]			RSVD	
[1]	r	1’b0	LP2HP_REQ	Indicate LPSYS request exists
[0]	rw	1’b0	HP2LP_REQ	Write 1 to request LPSYS to stay in active mode
0x30			**ANACR**	Analog Control Register
[31:2]			RSVD	
[1]	rw	1’b0	VHP_ISO	Set 1 to force off all HPSYS related analog modules
[0]	rw	1’b0	PA_ISO	Set 1 to force IO(PA) into retention mode
0x34			**GTIMR**	Global Timer Register
[31:0]	r	32’h0	CNT	Global timer value
0x38			**RESERVE0**	Reserved Register 0
[31:0]	rw	32’b0	DATA	for debug only
0x3C			**RESERVE1**	Reserved Register 1
[31:0]	rw	32’b0	DATA	for debug only

5 Input and Output

5.1 Introduction

The SF32LB52B/E/G/J chip supports up to 45 configurable general-purpose IO pins (PA00~PA44) . The SF32LB520/3/5/7 chip supports up to 44 configurable general-purpose IO pins (PA00~PA20, PA22~PA44) . Each general-purpose IO can independently select input or output functions and configure drive strength as well as pull-up/pull-down resistors. When configured as GPIO function, each general-purpose IO can trigger interrupts based on high or low levels or transitions, and can wake the system from certain low power modes. Additionally, PA24~PA27 can be configured as low power IO for output in low power mode.

5.2 IO Structure

The structure of a single general-purpose IOis illustrated in the fgure below.

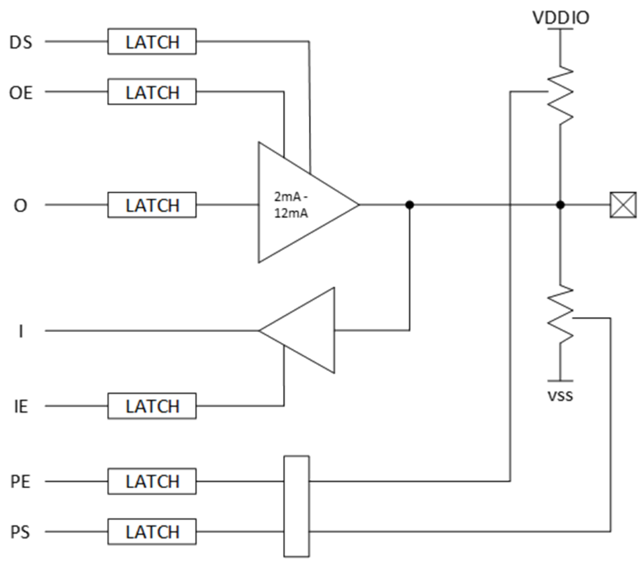


Figure 5-1: **IO Structure**

The DS register is used to configure the drive strength, which is specifed by the DS0 and DS1 registers corresponding to the IO via PINMUX. The values of DS1,DS0 range from 0 to 3, increasing the drive strength incrementally.

The IO output is determined by O and OE, with different functions selected based on the FSEL register corresponding to the IO via PINMUX, automatically mapping to the output of the respective function.

The IO input enable is controlled by the IE register corresponding to the IO via PINMUX. When IE is high, the input level I is automatically mapped to the input of the corresponding function based on the FSEL register corresponding to the IO; when IE is low, the corresponding function cannot access the input level of the IO.

PE and PS are used to control the pull-up and pull-down resistors of IO, which are specifed by the PE and PS registers corresponding to IO in PINMUX. When PE is 0, both pull-up and pull-down resistors are inactive. When PE is 1 and PS is 0, the pull-down resistor is active. When PE is 1 and PS is 1, the pull-up resistor is active. The resistance value of the pull-up and pull-down resistors is approximately 10k~40k ohms, which is related to the IO external circuit and interface level.

5.3 Input and Output Selection

By configuring the FSEL register corresponding to IO in PINMUX, configurable IO can be mapped to one of several functions. The functions that each IO can map to can be found in the GPIO pin list. If the function is an input or bidirectional input/output, the corresponding IO 's IE register must also be set to 1.

Each general-purpose IO can be mapped to a GPIO function, at which point the output of the IO is controlled by the GPIO module. Regardless of the function to which the IO is mapped, theIO input can be read from the GPIO module and can generate GPIO interrupts.

When a specific PA is mapped to the PA_I2C_UART function, that IO can serve as any interface signal for either I2C or USART. The specific interface signal must be specifed in the HPSYS_CFG register. For instance, to use PA07 as the TXD for USART2, the PINMUX register's PAD_PA07_FSEL must be set to 4 (corresponding to PA_I2C_UART) , and the HPSYS_CFG register's USART2_PINR_TXD_PIN must be set to 7 (corresponding to PA07) . It is important to note that assigning multiple interface signals to the same IO can lead to functional errors.

When a specific PA is mapped to the PA_TIM function, this IO can serve as an interface signal for any ATIM/GPTIM/LPTIM. The specific interface signal must be defined in the HPSYS_CFG register. For instance, to use PA38 as CH3 of GPTIM2, the PINMUX register's PAD_PA38_FSEL must be set to 5 (corresponding to PA_TIM) , and the HPSYS_CFG register's GPTIM2_PINR_CH3_PIN must be set to 38 (corresponding to PA38) . It is important to note that assigning multiple interface signals to the same IO can lead to functional errors.

5.4 IO High Impedance

Once the chip is powered on, the IO defaults to having a pull-up or pull-down resistor (PE default value is 1) , which must be configured by software to the desired state. To set the IO to high impedance, set PE to 0 , configure the IO to GPIO function, and ensure that the GPIO output enable is turned off.

5.5 GPIO Output

When the IO is configured as a GPIO function, theO and OE control signals of the IO are managed by the GPIO Register, thus generating the output of the IO. The general IO (PA00~PA44) is controlled by HPSYS_GPIO. Each bit of the GPIO Register corresponds to a generalIO; for instance, bit 0 of DOER0 in HPSYS_GPIO corresponds to PA00, while bit 31 corresponds to PA31; bit 0 of DOER1 corresponds to PA32, and bit 12 corresponds to PA44.

The DOERx register directly controls the OE signal of the IO; when set to 1, the IO output is enabled, and when set to 0, the IO output is disabled. Software can directly configure the DOERx register, or it can configure the DOESRx or DOECRx for bit manipulation to avoid affecting other IO.

The DORx Register directly controls the O signal of the IO. When set to 1, the output of the IO is high, and when set to 0, the output of the IO is low. The software can directly configure the DORx Register, or it can configure the DOSRx or

DOCRx for bit manipulation to prevent affecting other IOs.

The configuration method for GPIO push-pull output is as follows:

For push-pull output 0, IO's OE must be set to 1, O must be set to 0, which means that the corresponding bit of DOERx is configured to 1, while the corresponding bit of DORx is configured to 0.

Push-pull output 1 , IO's OE must be 1 ， O must be 1 , which means that the corresponding bit of DOERx is configured to 1， DORx corresponding bit configuration is 1。

When an open-drain output is required, the internal pull-up resistor of IO should first be enabled according to the toggle rate requirements (PE=1, PS=1) , or an external pull-up resistor should be connected to the chip.

The configuration method for GPIO open-drain output is as follows:

For open-drain output 0, the OE of IO must be set to 1, and O must be 0, which means that the corresponding bit configuration of DOERx is 1, and the corresponding bit configuration of DORx is 0.

For open-drain output 1, the OE of IO must be set to 0, which means that the corresponding bit configuration of DOERx is 0.

5.6 GPIO Input

Regardless of the function to which IO is mapped, the IO input can be read from the GPIO register DIRx and can generate GPIOinterrupts based on the configuration.

The IO interrupt enable should only be activated when necessary. The software can directly configure the IERx Register or configure IESRx or IECRx for bit manipulation to avoid impacting other IO .

Even if IO is not configured for GPIO functionality, IOinterrupts can still be generated. Therefore, when IO interrupts are not needed, it is essential to ensure that the interrupt enable is turned off.

The conditions for generating GPIO interrupts include IOinput signals being at high level/low level/rising edge/falling edge/both edges, as shown in the table below.

IPH	IPL	ITR=0	ITR=1
0	0	No trigger	No trigger
0	1	Low level	Falling edge
1	0	High level	Rising edge
1	1	Invalid configuration	Dual Edge

The ITR is used to select the type of interrupt condition. The software can directly configure the ITRx Register, or it can configure the ITSRx or ITCRx for bit manipulation to prevent affecting other IOs.

IPH is used to enable high-level or rising edge interrupt conditions. The software can directly configure the IPHRx Register or configure IPHSRx or IPHCRx for bit manipulation to avoid affecting other IO .

IPL is used to enable low-level or falling edge interrupt conditions. The software can directly configure the IPLRx Register or configure IPLSRx or IPLCRx for bit manipulation to avoid affecting other IO .

The IO that generates the interrupt can be queried through ISRx. Writing 1to the corresponding bit of ISRx can clear the interrupt status.

GPIO interrupts can wake the system from certain low power modes. After waking up, the software should query ISRx to identify the wake-up source and clear the relevant flags.

5.7 IO Function List

Table 5-1: Big Core Domain (PA) PIN Function List

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
51	51	PA00	I/O	PD	0	GPIO_A0
					1	LCDC1_SPI_RSTB
					4	PA_I2C_UART
					5	PA_TIM
					7	LCDC1_8080_RSTB
					Others	Reserved
50	50	PA01	I/O	PD	0	GPIO_A1
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
49	49	PA02	I/O	PD	0	GPIO_A2
					1	LCDC1_SPI_TE
					3	I2S1_MCLK
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_B2
					7	LCDC1_8080_TE
					Others	Reserved
48	48	PA03	I/O	PU	0	GPIO_A3
					1	LCDC1_SPI_CS
					3	I2S1_SDO
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_B1
					7	LCDC1_8080_CS
					Others	Reserved
47	47	PA04	I/O	PD	0	GPIO_A4
					1	LCDC1_SPI_CLK
					3	I2S1_SDI
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_G1
					7	LCDC1_8080_WR
					Others	Reserved
46	46	PA05	I/O	PD	0	GPIO_A5
					1	LCDC1_SPI_DIO0
					3	I2S1_BCK
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_R1
					7	LCDC1_8080_RD
					Others	Reserved

Continued on the next page...

Table 5-1: **GPIO (PA) Pin List (Continued)**

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
45	45	PA06	I/O	PD	0	GPIO_A6
					1	LCDC1_SPI_DIO1
					3	I2S1_LRCK
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_HST
					7	LCDC1_8080_DC
					Others	Reserved
44	44	PA07	I/O	PD	0	GPIO_A7
					1	LCDC1_SPI_DIO2
					3	PDM1_CLK
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_ENB
					7	LCDC1_8080_DIO0
					Others	Reserved
43	43	PA08	I/O	PD	0	GPIO_A8
					1	LCDC1_SPI_DIO3
					3	PDM1_DATA
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_VST
					7	LCDC1_8080_DIO1
					Others	Reserved
42	42	PA09	I/O	PD	0	GPIO_A9
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
35	35	PA10	I/O	PD	0	GPIO_A10
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
34	34	PA11	I/O	PU	0	GPIO_A11
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
33	33	PA12	I/O	PU	0	GPIO_A12
					1	MPI2_CS
					2	SD1_DIO2
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved

Continued on the next page...

Table 5-1: **GPIO (PA) Pin List (Continued)**

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
32	32	PA13	I/O	PD	0	GPIO_A13
					1	MPI2_DIO1
					2	SD1_DIO3
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
31	31	PA14	I/O	PD	0	GPIO_A14
					1	MPI2_DIO2
					2	SD1_CLK
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
30	30	PA15	I/O	PD	0	GPIO_A15
					1	MPI2_DIO0
					2	SD1_CMD
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
29	29	PA16	I/O	PD	0	GPIO_A16
					1	MPI2_CLK
					2	SD1_DIO0
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
28	28	PA17	I/O	PD	0	GPIO_A17
					1	MPI2_DIO3
					2	SD1_DIO1
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
27	27	PA18	I/O	PU	0	GPIO_A18
					2	SWDIO
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
26	26	PA19	I/O	No Pull	0	GPIO_A19
					2	SWCLK
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved

Continued on the next page...

Table 5-1: **GPIO (PA) Pin List (Continued)**

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
25	25	PA20	I/O	PD	0	GPIO_A20
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
24	-	PA21	I/O	PD*	0	GPIO_A21
					4	PA_I2C_UART
					5	PA_TIM
					Others	Reserved
11	11	PA22	I/O	No Pull	0	GPIO_A22
					3	PDM1_CLK
					4	PA_I2C_UART
					5	PA_TIM
					8	#XTAL32K_XI
					Others	Reserved
10	10	PA23	I/O	No Pull	0	GPIO_A23
					3	PDM1_DATA
					4	PA_I2C_UART
					5	PA_TIM
					8	#XTAL32K_XO
					Others	Reserved
9	9	PA24	I/O	PD	0	GPIO_A24
					2	SPI1_DIO
					3	I2S1_MCLK
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN0
					Others	Reserved
8	8	PA25	I/O	PD	0	GPIO_A25
					2	SPI1_DI
					3	I2S1_SDO
					4	PA_I2C_UART
					5	PA_TIM
					7	#XTAL32K_EXT
					8	#WKUP_PIN1
					Others	Reserved
7	7	PA26	I/O	PU	0	GPIO_A26
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN2
					Others	Reserved

Continued on the next page...

Table 5-1: **GPIO (PA) Pin List (Continued)**

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
6	6	PA27	I/O	PU	0	GPIO_A27
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN3
					Others	Reserved
5	5	PA28	I/O	PD	0	GPIO_A28
					2	SPI1_CLK
					3	I2S1_SDI
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH0
					Others	Reserved
4	4	PA29	I/O	PD	0	GPIO_A29
					2	SPI1_CS
					3	I2S1_BCK
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH1
					Others	Reserved
3	3	PA30	I/O	PD	0	GPIO_A30
					2	#EFUSE_PWR
					3	I2S1_LRCK
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH2
					Others	Reserved
2	2	PA31	I/O	PD	0	GPIO_A31
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH3
					Others	Reserved
1	1	PA32	I/O	PD	0	GPIO_A32
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH4
					Others	Reserved

Continued on the next page...

Table 5-1: **GPIO (PA) Pin List (Continued)**

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
67	67	PA33	I/O	PD	0	GPIO_A33
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH5
					Others	Reserved
66	66	PA34	I/O	PD	0	GPIO_A34
					4	PA_I2C_UART
					5	PA_TIM
					7	#GPADC_CH6
					8	#WKUP_PIN10
					Others	Reserved
65	65	PA35	I/O	PD	0	GPIO_A35
					2	#USB11_DP
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN11
					Others	Reserved
64	64	PA36	I/O	PD	0	GPIO_A36
					2	#USB11_DM
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN12
					Others	Reserved
63	63	PA37	I/O	PD	0	GPIO_A37
					2	SPI2_DIO
					4	PA_I2C_UART
					5	PA_TIM
					7	LCDC1_8080_DIO2
					8	#WKUP_PIN13
					Others	Reserved
62	62	PA38	I/O	PD	0	GPIO_A38
					2	SPI2_DI
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN14
					Others	Reserved

Continued on the next page...

Table 5-1: GPIO (PA) Pin List (Continued)

Pin Number		Pin Name	Type	Default PU/PD Setting	Sel #	Functions
SF32LB520 (QFN68L)	SF32LB52x (QFN68L)					
61	61	PA39	I/O	PU	0	GPIO_A39
					2	SPI2_CLK
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_VCK
					7	LCDC1_8080_DIO3
					8	#WKUP_PIN15
					Others	Reserved
60	60	PA40	I/O	PU	0	GPIO_A40
					2	SPI2_CS
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_XRST
					7	LCDC1_8080_DIO4
					8	#WKUP_PIN16
					Others	Reserved
59	59	PA41	I/O	PU	0	GPIO_A41
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_HCK
					7	LCDC1_8080_DIO5
					8	#WKUP_PIN17
					Others	Reserved
58	58	PA42	I/O	PU	0	GPIO_A42
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_R2
					7	LCDC1_8080_DIO6
					8	#WKUP_PIN18
					Others	Reserved
57	57	PA43	I/O	PD	0	GPIO_A43
					4	PA_I2C_UART
					5	PA_TIM
					6	LCDC1_JDI_G2
					7	LCDC1_8080_DIO7
					8	#WKUP_PIN19
					Others	Reserved
56	56	PA44	I/O	PD	0	GPIO_A44
					4	PA_I2C_UART
					5	PA_TIM
					8	#WKUP_PIN20
					Others	Reserved

*In the Boot ROM, the 52B/52D/52G/52J and related 52-series chips assert a high-level signal on the PA21 pin to regulate the co-packaged memory's power supply

5.8 Integrated IO

Integrated IO is not included in the configurable IO count and is specifically designated for MPI access to integrated (SiP) NOR flash or pSRAM. Integrated IO shares the same structure as general IO and allows for individual configuration of each IO's function and pull-up/pull-down settings. Integrated IO (SA) is configured in HPSYS_PINMUX

IO(SA) includes SA00~SA12, which is used for MPI1 to access the encapsulated 8line pSRAM or NOR Flash.

5.9 Low Power IO

PA24~PA27 In addition to the existing general IO input and output pathways, there is an additional set of low power IO input and output pathways that can maintain input and output capabilities while the chip is in low power mode. The input enable, output enable, and pull-up/pull-down resistor functions of low power IO can be configured through the RTC 's PBRxR register. Low power IO can continuously output a fixed level, and the low power PWM which is generated by the low power clock clk_rtc or LPTIM1/2 is unaffected by the system entering or exiting low power mode (hibernate mode stops LPTIM from functioning, thus it cannot output PWM).

When using PA24~PA27, special attention must be given to ensure that the configurations of these IO in the PINMUX register do not conflict with the low power IO configurations in the RTC register. When an IO is utilized as a general-purpose IO or low power IO , it is recommended that the other set of registers be configured to non-input high impedance (IE=0, PE=0, OE=0) . For instance, when PA25 is employed for GPIO output, it is essential to not only configure PAD_PA25 in the PINMUX correctly but also to set the PBR1R register in the RTC to 0 . If PA27 is used as a wake-up PIN , it is crucial to not only configure PBR3R in the RTC correctly but also to set PAD_PA27 in the PINMUX to 0 .

5.10 IO Power Supply

The level standard of IOdepends on its supply voltage.

The power supply for the integrated IO also serves as the power supply for the corresponding combined memory.

The power supply methods for IO vary among different chip models.

Chip Model	IO(PA)Power Supply	IO(SA)Power Supply
SF32LB520	Internal Power Supply 3.3V	Internal Power Supply 3.3V(External Loop of Chip)
SF32LB523/5/7	Internal Power Supply 3.3V	Internal Power Supply 1.8V
SF32LB52BU36	External Supply VDDIOA	External Supply VDD_SIP(1.8V or 3.3V)
SF32LB52BU56	External Supply VDDIOA	External Supply VDD_SIP(3.3V)
SF32LB52E/G/J	External Supply VDDIOA	Internal Power Supply 1.8V (if PVDD=3.3V) External Supply VDD_SIP 1.8V (if PVDD=1.8V)

5.11 Wake-Up PIN

The chip supports up to 15 Wake-Up PINs, including IO PA24~PA27 and PA34~PA44. The Wake-Up PINs can wake the chip from hibernate mode, standby mode, or deepsleep mode. The wake-up trigger methods include high-level wake-up, low-level wake-up, rising edge wake-up, falling edge wake-up, and dual-edge wake-up. The Wake-Up PIN function is independent of the function selection of IO, meaning that the FSEL register does not influence the Wake-Up PIN function.

The hibernate mode supports the selection of up to two of the 15 wake-up PINs to be mapped as actual PIN wake-up sources. The selection registers are PMUC's CR_PIN0_SEL and CR_PIN1_SEL .

The standby mode supports up to 15 wake-up PINs simultaneously as wake-up sources.

The deepsleep mode not only supports up to 15 wake-up PINs simultaneously as wake-up sources but also allows for any IO wake-up. For instance, after HPSYS enters deepsleep mode, it can be awakened by any of the 15 wake-up PINs (the wake-up status bits are HPSYS_AON 's WSR_PIN0~20) , and it can also be awakened by any IO (the wake-up status bit is HPSYS_AON 's WSR_GPIO1 ; the specific IO that triggers the wake-up must be checked in the GPIO register).

In addition to the existing IO input channels from PA34 to PA44, there is another permanently enabled input channel for the wake-up PIN function. Therefore, regardless of whether the chip is in low power mode, it is essential to ensure that these IO pins do not remain in an intermediate voltage state when not toggled, as this may lead to leakage current.

When the chip is in hibernate mode, the pull-up and pull-down resistor configurations for all IO in the PINMUX register are ineffective. Therefore, for PA34~PA44 which support the wake-up PIN function, if the external circuit cannot guarantee that the IO is at a defnite high or low level, additional configuration of the PAWK1R and PAWK2R registers in the RTC is required to enable pull-up or pull-down resistors before entering hibernate mode. It should be noted that the pull-up and pull-down resistor configurations in the PINMUX register and the RTC register will take effect simultaneously, so care should be taken to avoid redundancy and conflicts during configuration. It is recommended to configure solely through the RTC register.

5.12 IO Status in Low Power Mode

In active mode and sleep mode, HPSYS operates normally, with IO functioning correctly, outputs able to toggle, and IO interrupts being generated.

After HPSYS enters Deep Sleep mode, all IO enter sleep mode, with input enable, output enable, and pull-up/pull-down resistor configurations remaining unchanged. Outputs retain their previous values, stop toggling, and cannot generate IO interrupts, but can produce IO Wake-Up. PA24~PA 27 can perform low power output and generate PIN Wake-Up. PA34~PA44 can generate PIN Wake-Up.

After HPSYS enters Standby mode, all IO enter Retention mode, with input enable, output enable, and pull-up/pull-down resistor configurations remaining unchanged. Outputs retain their previous values, stop toggling, and cannot generate IO interrupts or IO Wake-Up. PA24~PA27 can perform low power output and generate PIN Wake-Up. PA34~PA44 can generate PINwake-up signals.

Once the chip enters hibernate mode, the wake-up PIN function is not supported for IO (PA00~PA23, PA28~PA33, SA00~SA12), which will enter shutdown mode. Input enable and output enable are disabled, pull-up and pull-down resistors are turned off, and IO will exhibit high impedance, rendering it unable to generate IO interrupts and wake-ups. PA24~PA27 can perform low power consumption output and can generate PIN wake-up signals, with pull-up and pull-down resistors configured by the RTC's PBRxR register. PA34~PA44 can generate PIN wake-up signals, with pull-up and pull-down resistors configured by the RTC's PAWK1R and PAWK2R registers.

Table 5-2: **IO Working Status**

IO	Function	active	sleep	deepsleep	standby	hibernate
PA00~PA23 PA28~PA33	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Reversible	Reversible	Hold	Hold	High Impedance
	Pull-Up and Pull-Down Resistors	Valid	Valid	Hold	Hold	Invalid
	GPIO Interrupt	Yes	Yes	None	None	None
	IO Wake-Up	Yes	Yes	Yes	None	None
	PIN Wake-Up	None	None	None	None	None
PA34~PA44	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Reversible	Reversible	Hold	Hold	Invalid
	Pull-Up and Pull-Down Resistors	Valid	Valid	Valid	Valid	Valid
	GPIO Interrupt	Yes	Yes	None	None	None
	IO Wake-Up	Yes	Yes	Yes	None	None
	PIN Wake-Up	Yes	Yes	Yes	Yes	Yes
PA24~PA27	Input Enable	Active	Active	Active	Active	Active
	Output Enable	Active	Active	Active(1)	Active(1)	Active(1)
	Output Level	Reversible	Reversible	Reversible(1)	Reversible(1)	Reversible(1)
	Pull-Up and Pull-Down Resistors	Valid	Valid	Valid	Valid	Valid
	GPIO Interrupt	Yes	Yes	None	None	None
	IO Wake-Up	Yes	Yes	Yes	None	None
	PIN Wake-Up	Yes	Yes	Yes	Yes	Yes
SA00~SA12	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Reversible	Reversible	Hold	Hold	High Impedance
	Pull-Up and Pull-Down Resistors	Valid	Valid	Hold	Hold	Invalid
	GPIO Interrupt	None	None	None	None	None
	IO Wake-Up	None	None	None	None	None
	PIN Wake-Up	None	None	None	None	None

* (1)Low Power IOOutput Path Only

5.13 Avoid IO Leakage

IO leakage is a common issue encountered during low power consumption debugging of the chip. The common types of IOleakage are as follows:

- Short circuit leakage. When an output high level IO is shorted with another output low level IO (possibly from other devices) through the chip's external circuits, it can generate a signifcant current, which may severely damage the chip.
- Output conduction leakage. When theIO outputs a high level, there may be a pull-down resistor present either internally or externally in the chip, resulting in conduction current. Conversely, when the IO outputs a low level, there may be a pull-up resistor present either internally or externally in the chip, also resulting in conduction current.
- Pull-up and pull-down resistors can cause leakage current. When a specific IO has both pull-up and pull-down resistors present, either internally or externally within the chip, it generates a conducting current.

- Leakage current occurs during intermediate input levels. When the IO input path is conducting, the input terminal is at an intermediate level, resulting in the IO internally generating conducting current. This type of leakage is relatively subtle and may manifest as fluctuating leakage current.

For the second type of leakage, for IO configured as module output or GPIO push-pull output, the pull-up and pull-down resistors can be disabled (set PE to 0).

For the third type of leakage, the external circuits of the chip should be examined to ensure that there are no conflicts between the pull-up and pull-down resistors. For general IO that supports wake-up PIN functionality, the pull-up and pull-down resistors should be configured through the RTC using the PAWK1R and PAWK2R registers, and the pull-up and pull-down resistor configurations in the PINMUX register should be disabled to avoid configuration conflicts.

For type 4 leakage, in addition to being able to clearly identify whether the input level is high or low, the following configuration methods should be adopted:

- For general IOthat does not require input functionality, disable the input enable (IEset to 0).
- If PA24~PA27 are not used as Wake-Up PINs, disable the input enable (RTC's PBRxR_IE set to 0) .
- If PA24~PA27 are used as Wake-Up PINs, configure the pull-up or pull-down resistors through the RTC's PBRxR register.
- General IO that supports Wake-Up PIN functionality (PA34~PA44) can configure pull-up or pull-down resistors through the RTC's PAWK1R and PAWK2R registers.

When there is suspicion of IO leakage, the following checks can be performed on each IO.

Table 5-3: IO Leakage Inspection Table

IO	If configured for GPIO functionality and output enable is activated, or configured for output signals of other modules.	When input enable is activated.	When both input enable and output enable are deactivated.
PA00~PA23 PA28~PA33	Check if the output value matches the pull-up/pull-down configuration of PINMUX. Verify if the output value corresponds to the pull-up/pull-down of the external circuit. Ensure that a high level is output, while the external device connected to IO is powered off.	Check if the external device connected to IO can provide a defnite level. Verify if PINMUX is configured with appropriate pull-up/pull-down settings.	Check if the pull-up/pull-down configuration of PINMUX conflicts with the levels of the external circuit.
PA34~PA44	Verify whether the output value aligns with the pull-up and pull-down configuration of the RTC. Assess if the PINMUX conflicts with the pull-up and pull-down settings of the RTC. Ensure that the output value corresponds with the external circuit's pull-up and pull-down configuration. Confrm whether a high level is being output while the external device connected to IO is powered off.	Check whether the external devices connected to the IO can provide a defnite level. Verify if the PINMUX and RTC are configured with appropriate pull-up and pull-down resistors.	Check whether the external devices connected to the IO can provide a defnite level. Verify if the PINMUX and RTC are configured with appropriate pull-up and pull-down resistors.
PA24~PA27	Verify whether the output value aligns with the pull-up and pull-down configuration of the RTC. Assess if the PINMUX conflicts with the pull-up and pull-down settings of the RTC. Ensure that the output value corresponds with the external circuit's pull-up and pull-down configuration. Confrm whether a high level is being output while the external device connected to IO is powered off.	Determine if the external device connected to IO can provide a stable level. Check if the PINMUX and RTC are configured with appropriate pull-up and pull-down resistors.	Verify if the pull-up and pull-down configuration of the RTC conflicts with the levels of the external circuit.

The chip offers a multitude of IOs. When it is uncertain which IOs are experiencing current leakage, all general IOs (PA00~PA44) can initially be set in the PINMUX to a non-input high-impedance state (IE=0, PE=0, OE=0). The low-power IOs PA24~PA27 should be configured in the RTC to a non-input high-impedance state (IE=0, PE=0, OE=0), while the other IOs that support wake-up PIN functionality (PA34~PA44) should have appropriate pull-up and pull-down resistors configured in the RTC register. In this state, the chip's IOs will not exhibit current leakage. Subsequently, restore the configuration of the IOs in batches to identify the IOs that are leaking current and eliminate the leakage sources.

In Hibernate mode, the chip's IO leakage occurs only on the wake-up PIN , so only the IO configuration of the wake-up PIN needs to be examined.

5.14 Avoid Leakage in Integrated IO

The power supply for the integrated IO also powers the corresponding combined memory, so the leakage from the combined memory will also be reflected in the power supply of the integrated IO. The common types of leakage in integrated IO are as follows:

1. The chip enters low power mode, but the combined memory does not enter a low power state. The combined NOR flash or pSRAM will consume current (depending on the memory model, typically in the range of 10~200uA) even in standby state when not accessed, and this leakage becomes more pronounced after the chip enters low power mode. Therefore, it is recommended that the chip first puts the combined memory into a low power state (with leakage around 1~10uA) before entering deepsleep or standby mode. After the chip wakes up, the combined memory can exit the low power state. The combined pSRAM can enter half sleep mode via instructions, and the combined NOR flash can enter deep power down mode via instructions; these modes can signifcantly reduce the memory's leakage.
2. The integrated IO may cause uncontrollable leakage in the combined storage when the chip enters hibernate mode. In this mode, the integrated IO exhibits a high-impedance floating level. If the power supply for the integrated IO remains active at this time, it may lead to a high leakage state in the combined storage (for example, if the CS pin is at a floating low level or an intermediate voltage, it may result in leakage currents ranging from hundreds of uA to several mA) . Therefore, before the chip enters hibernate mode, the power supply for the integrated IO should be turned off to avoid leakage in the combined storage. If the power supply for the integrated IO is provided internally by the chip, the software can control the internal power supply switch when entering and exiting hibernate mode, eliminating the need for an external power switch. The SF32LB52B/E/G/J chips reserve PA21 for controlling the combined power supply. When the chip is powered on or wakes up from hibernate , PA21 automatically outputs high and becomes high-impedance after entering hibernate mode. If the power supply for the integrated IO is externally powered, PA21 can be used to enable the external power switch, thereby controlling the power supply's activation and deactivation to prevent leakage. If the power supply for the integrated IO can only maintain constant power without control, it is advisable to avoid using the chip's hibernate mode and instead use standby mode.

5.15 HPSYS_PINMUX Register

HPSYS_PINMUX base address is 0x50003000.

Table 5-4: HPSYS_PINMUX Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x0			**PAD_SA00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x4			**PAD_SA01**	
[31:12]			RSVD	

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x8			**PAD_SA02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC			**PAD_SA03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x10			**PAD_SA04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x14			**PAD_SA05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4'h0	FSEL	Function Select
0x18			**PAD_SA06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x1C			**PAD_SA07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x20			**PAD_SA08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x24			**PAD_SA09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x28			**PAD_SA10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x2C			**PAD_SA11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x30			**PAD_SA12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x34			**PAD_PA00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x38			**PAD_PA01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x3C			**PAD_PA02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x40			**PAD_PA03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x44			**PAD_PA04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x48			**PAD_PA05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x4C			**PAD_PA06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x50			**PAD_PA07**	

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x54			**PAD_PA08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x58			**PAD_PA09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x5C			**PAD_PA10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x60			**PAD_PA11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0x64			**PAD_PA12**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0x68			**PAD_PA13**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0x6C			**PAD_PA14**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0x70			**PAD_PA15**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0x74			**PAD_PA16**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x78			**PAD_PA17**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x7C			**PAD_PA18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h4	FSEL	Function Select
0x80			**PAD_PA19**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h4	FSEL	Function Select
0x84			**PAD_PA20**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x88			**PAD_PA21**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x8C			**PAD_PA22**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x90			**PAD_PA23**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x94			**PAD_PA24**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x98			**PAD_PA25**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0x9C			**PAD_PA26**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA0			**PAD_PA27**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA4			**PAD_PA28**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA8			**PAD_PA29**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xAC			**PAD_PA30**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB0			**PAD_PA31**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB4			**PAD_PA32**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB8			**PAD_PA33**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xBC			**PAD_PA34**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC0			**PAD_PA35**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0xC4			**PAD_PA36**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0xC8			**PAD_PA37**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0xCC			**PAD_PA38**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1’b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1’h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0xD0			**PAD_PA39**	
[31:12]			RSVD	
[11]	rw	1’h0	POE	Reserved. Always set to logic LOW
[10]	rw	1’b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1’h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1’h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1’h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1’h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1’h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4’h0	FSEL	Function Select
0xD4			**PAD_PA40**	
[31:12]			RSVD	

Continued on the next page...

Table 5-4: HPSYS_PINMUX Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xD8			**PAD_PA41**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xDC			**PAD_PA42**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xE0			**PAD_PA43**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xE4			**PAD_PA44**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device

Continued on the next page...

Table 5-4: **HPSYS_PINMUX Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4’h0	FSEL	Function Select

5.16 HPSYS_CFG Register

HPSYS_CFG base address is 0x5000B000.

Table 5-5: **HPSYS_CFG Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**BMR**	Boot Mode Register
[31:1]			RSVD	
[0]	r	1’h0	BOOT_MODE	0 - normal mode, 1 - download mode
0x04			**IDR**	ID Register
[31:24]	r	8’hC2	SID	Series ID
[23:16]	r	8’h04	CID	Chip ID
[15:8]	r	8’h0	PID	Package ID
[7:0]	r	8’h0	REVID	Revision ID
0x08			**SWCR**	SW Control Register
[31:1]			RSVD	
[0]	rw	1’h0	SWSEL	reserved for debug
0x0C			**SCR**	Security Control Register
[31:1]			RSVD	
[0]	rw	1’h1	FKEY_MODE	reserved for debug
0x10			**SYSCR**	System Configure Register
[31:3]			RSVD	
[2]	rw	1’h0	LDO_VSEL	select work mode 0: enhanced mode 1: base mode
[1]	rw	1’h0	SDNAND	0: MPI2 AHB space is allocated to MPI2 1: MPI2 AHB space is allocated to SDMMC1
[0]	rw	1’h0	WDT1_REBOOT	If set to 1, WDT1 reset will reboot the whole chip
0x14			**RTC_TR**	Mirrored RTC Time Register
[31]	r	1’h0	PM	AM/PM notation 0: AM 1: PM
[30:29]	r	2’h0	HT	Hour tens in BCD format
[28:25]	r	4’h0	HU	Hour units in BCD format
[24:22]	r	3’h0	MNT	Minute tens in BCD format
[21:18]	r	4’h0	MNU	Minute units in BCD format
[17:15]	r	3’h0	ST	Second tens in BCD format
[14:11]	r	4’h0	SU	Second units in BCD format
[10]			RSVD	
[9:0]	r	10’h0	SS	Sub-second counter
0x18			**RTC_DR**	Mirrored RTC Date Register
[31]	r	1’h0	ERR	reserved for debug
[30:25]			RSVD	
[24]	r	1’h0	CB	Century flag
[23:20]	r	4’h0	YT	Year tens in BCD format
[19:16]	r	4’h0	YU	Year units in BCD format

Continued on the next page...

Table 5-5: HPSYS_CFG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:13]	r	3'h1	WD	Week day units 000: forbidden 001: Monday ... 111: Sunday
[12]	r	1'h0	MT	Month tens in BCD format
[11:8]	r	4'h1	MU	Month units in BCD format
[7:6]			RSVD	
[5:4]	r	2'h0	DT	Date tens in BCD format
[3:0]	r	4'h1	DU	Date units in BCD format
0x1C			**ULPMCR**	ULP Memory Control register
[31]	rw	1'h0	FORCE_ON	reserved for debug
[30]	rw	1'h0	ROM_DIS	reserved for debug
[29:21]			RSVD	
[20]	rw	1'b1	ROM_RME	reserved for debug
[19:18]			RSVD	
[17:16]	rw	2'b11	ROM_RM	reserved for debug
[15:13]			RSVD	
[12:10]	rw	3'b000	RAM_WPULSE	reserved for debug
[9:7]	rw	3'b100	RAM_WA	reserved for debug
[6:5]	rw	2'b00	RAM_RA	reserved for debug
[4]	rw	1'b1	RAM_RME	reserved for debug
[3:2]			RSVD	
[1:0]	rw	2'b11	RAM_RM	reserved for debug
0x20			**DBGR**	Debug Select Register
[31]	rw	1'h0	SWAP	reserved for debug
[30]	r	1'h0	LP2HP_NMIF	LP2HP NMI interrupt flag
[29]	rw	1'h0	LP2HP_NMIE	LP2HP NMI interrupt enable
[28]	rw	1'h0	HP2LP_NMI	set 1 to send NMI interrupt to LCPU
[27]	rw	1'b0	CLK_EN	reserved for debug
[26:24]	rw	3'b0	CLK_SEL	reserved for debug
[23:16]	rw	8'h0	BITEN_H	reserved for debug
[15:8]	rw	8'h0	BITEN_L	reserved for debug
[7:4]	rw	4'h0	SEL_H	reserved for debug
[3:0]	rw	4'h0	SEL_L	reserved for debug
0x24			**MDBGR**	Memory Debug Register
[31:5]			RSVD	
[4]	rw	1'b0	PD_ROM	reserved for debug
[3]	rw	1'b0	LS_ROM	reserved for debug
[2]	rw	1'b0	LS_RAM2	reserved for debug
[1]	rw	1'b0	LS_RAM1	reserved for debug
[0]	rw	1'b0	LS_RAM0	reserved for debug
0x3C			**LPIRQ**	Interrupt Selection for LCPU
[31:16]			RSVD	
[15]	rw1c	1'h0	IF1	hp2lp1 interrupt status. Write 1 to clear.
[14]			RSVD	
[13:8]	rw	6'h0	SEL1	select hp2lp1 interrupt source
[7]	rw1c	1'h0	IF0	hp2lp0 interrupt status. Write 1 to clear.
[6]			RSVD	
[5:0]	rw	6'h0	SEL0	select hp2lp0 interrupt source
0x40			**USBCR**	USB Control register
[31:24]			RSVD1	

Continued on the next page...

Table 5-5: **HPSYS_CFG Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[23:16]			RSVD0	
[15:13]	rw	3'h0	DC_TR	reserved for debug
[12]	rw	1'h0	DC_TE	reserved for debug
[11]			RSVD	
[10:8]	rw	3'h0	TX_RTUNE	TX outp impedance tuning 0 = 50 Ohm, 1 = 46 Ohm, 2 = 43 Ohm, 3 = 40 Ohm, 4 = 37.5 Ohm, 5 = 35 Ohm, 6 = 33 Ohm, 7 = 31.5 Ohm
[7]			RSVD	
[6]	rw	1'h0	DP_EN	0:disable dp pull up or pull down 1:enable dp pull or pull down
[5]	rw	1'h0	DM_PD	enable DM 15k Ohm pull down resistor
[4]	rw	1'h0	LDO_LP_EN	2.5V LDO low power mode enable. 0 = 240 uA, 1 = 50 uA
[3:1]	rw	3'h0	LDO_VSEL	2.5V LDO output voltage setting 0 = 2.40 V, 1 = 2.47 V, 2 = 2.53 V, 3 = 2.60 V, 4 = 2.60 V, 5 = 2.67 V, 6 = 2.73 V, 7 = 2.8 V
[0]	rw	1'h0	USB_EN	USB PHY enable, turn on power swith, power up LDO and bias
0x44			**SYS_RSVD**	HPSYS RSVD Register
[31:24]	r	8'h0	RESERVE3	reserved for debug
[23:16]	rw	8'hff	RESERVE2	reserved for debug
[15:8]	rw	8'h0	RESERVE1	reserved for debug
[7:0]	rw	8'h0	RESERVE0	reserved for debug
0x48			**I2C1_PINR**	I2C1 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x4C			**I2C2_PINR**	I2C2 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x50			**I2C3_PINR**	I2C3 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x54			**I2C4_PINR**	I2C4 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.

Continued on the next page...

Table 5-5: **HPSYS_CFG Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x58			**USART1_PINR**	USART1 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'd18	RXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'd19	TXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x5C			**USART2_PINR**	USART2 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	RXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	TXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x60			**USART3_PINR**	USART3 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	RXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	

Continued on the next page...

Table 5-5: **HPSYS_CFG Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[5:0]	rw	6’h3f	TXD_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x64			**GPTIM1_PINR**	GPTIM1 Pin Register
[31:30]			RSVD	
[29:24]	rw	6’h3f	CH4_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6’h3f	CH3_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6’h3f	CH2_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6’h3f	CH1_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x68			**GPTIM2_PINR**	GPTIM2 Pin Register
[31:30]			RSVD	
[29:24]	rw	6’h3f	CH4_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6’h3f	CH3_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6’h3f	CH2_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6’h3f	CH1_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x6C			**ETR_PINR**	GPTIM ETR Pin Register
[31:14]			RSVD	
[13:8]	rw	6’h3f	ETR2_PIN	Connect GPTIM2_ETR to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6’h3f	ETR1_PIN	Connect GPTIM1_ETR to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x70			**LPTIM1_PINR**	LPTIM1 Pin Register
[31:22]			RSVD	
[21:16]	rw	6’h3f	ETR_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	

Continued on the next page...

Table 5-5: HPSYS_CFG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13:8]	rw	6'h3f	OUT_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	IN_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x74			**LPTIM2_PINR**	LPTIM2 Pin Register
[31:22]			RSVD	
[21:16]	rw	6'h3f	ETR_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	OUT_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	IN_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x78			**ATIM1_PINR1**	ATIM1 Pin Register 1
[31:30]			RSVD	
[29:24]	rw	6'h3f	CH4_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	CH3_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	CH2_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	CH1_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x7C			**ATIM1_PINR2**	ATIM1 Pin Register 2
[31:22]			RSVD	
[21:16]	rw	6'h3f	CH3N_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	CH2N_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	CH1N_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x80			**ATIM1_PINR3**	ATIM1 Pin Register 3
[31:22]			RSVD	

Continued on the next page...

Table 5-5: HPSYS_CFG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21:16]	rw	6'h3f	ETR_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	BK2_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	BK_PIN	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x84			**PTA_PINR**	PTA Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	WLAN_ACTIVE	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	BT_PRIORITY	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	BT_COLLISION	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	BT_ACTIVE	Connect function pin to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x88			**ANAU_CR**	ANAU Control Register
[31:7]			RSVD	
[6:4]	rw	3'h0	DC_MR	reserved for debug
[3]	rw	1'b1	EFUSE_VDD_PD	reserved for debug
[2]	rw	1'b0	EFUSE_VDD_EN	reserved for debug
[1]	rw	1'b0	EN_VBAT_MON	reserved for debug
[0]	rw	1'b0	EN_BG	reserved for debug
0x8C			**ANAU_RSVD**	ANAU Reserve Register
[31:24]	r	8'h0	RESERVE3	reserved for debug
[23:16]	rw	8'h0	RESERVE2	reserved for debug
[15:8]	rw	8'h0	RESERVE1	reserved for debug
[7:0]	rw	8'h0	RESERVE0	reserved for debug
0x90			**ANATR**	Analog Test Register
[31:8]			RSVD	
[7:5]	rw	3'h0	DC_UR_ATEST1	reserved for debug
[4]	rw	1'h0	DC_TE_ATEST1	reserved for debug
[3:1]	rw	3'h0	DC_UR_ATEST0	reserved for debug
[0]	rw	1'h0	DC_TE_ATEST0	reserved for debug
0x94			**CAU2_CR**	CAU2 Control Register
[31:13]			RSVD	
[12:10]	rw	3'h0	DC_MR	reserved for debug
[9:7]	rw	3'h0	DC_BR	reserved for debug
[6:4]	rw	3'h0	DC_TR	reserved for debug
[3:2]			RSVD	

Continued on the next page...

Table 5-5: **HPSYS_CFG Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1’b0	HPBG_EN	reserved for debug
[0]	rw	1’b0	HPBG_VDDPSW_EN	reserved for debug
0x98			**CAU2_RSVD**	CAU2 RSVD Register1
[31:24]			RSVD	
[23:16]	r	8’h0	RESERVE2	reserved for debug
[15:8]	r	8’h0	RESERVE1	reserved for debug
[7:0]	rw	8’h0	RESERVE0	reserved for debug

5.17 HPSYS_GPIO Register

HPSYS_GPIO base address is 0x500A0000.

Table 5-6: **HPSYS_GPIO Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**DIR0**	Data Input Register
[31:0]	r	32’h0	IN	GPIO[31:0] input value
0x04			**DOR0**	Data Output Register
[31:0]	rw	32’h0	OUT	GPIO[31:0] output value if output enabled
0x08			**DOSR0**	Data Output Set Register
[31:0]	w	32’h0	DOS	set 1 to pull up output of corresponding GPIO[31:0]
0x0C			**DOCR0**	Data Output Clear Register
[31:0]	w	32’h0	DOC	set 1 to pull down output of corresponding GPIO[31:0]
0x10			**DOER0**	Data Output Enable Register
[31:0]	rw	32’h0	DOE	GPIO[31:0] output enable
0x14			**DOESR0**	Data Output Enable Set Register
[31:0]	w	32’h0	DOES	set 1 to enable output of corresponding GPIO[31:0]
0x18			**DOECR0**	Data Output Enable Clear Register
[31:0]	w	32’h0	DOEC	set 1 to disable output of corresponding GPIO[31:0]
0x1C			**IER0**	Interrupt Enable Register
[31:0]	rw	32’h0	IER	GPIO[31:0] interrupt enable
0x20			**IESR0**	Interrupt Enable Set Register
[31:0]	w	32’h0	IES	set 1 to enable interrupt of corresponding GPIO[31:0]
0x24			**IECR0**	Interrupt Enable Clear Register
[31:0]	w	32’h0	IEC	set 1 to disable interrupt of corresponding GPIO[31:0]
0x28			**ITR0**	Interrupt Type Register
[31:0]	rw	32’h0	ITR	GPIO[31:0] interrupt type
0x2C			**ITSR0**	Interrupt Type Set Register
[31:0]	w	32’h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[31:0]
0x30			**ITCR0**	Interrupt Type Clear Register
[31:0]	w	32’h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[31:0]
0x34			**IPHR0**	Interrupt Polarity High Register
[31:0]	rw	32’h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x38			**IPHSR0**	Interrupt Polarity High Set Register
[31:0]	w	32’h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x3C			**IPHCR0**	Interrupt Polarity High Clear Register
[31:0]	w	32’h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x40			**IPLR0**	Interrupt Polarity Low Register

Continued on the next page...

Table 5-6: **HPSYS_GPIO Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x44			**IPLSR0**	Interrupt Polarity Low Set Register
[31:0]	w	32'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x48			**IPLCR0**	Interrupt Polarity Low Clear Register
[31:0]	w	32'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x4C			**ISR0**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[31:0]
0x60			**OEMR0**	output mode Register
[31:0]	rw	32'h0	OEM	output mode of corresponding GPIO[31:0]
0x64			**OEMSR0**	output mode Set Register
[31:0]	w	32'h0	OEMS	output mode Set of corresponding GPIO[31:0]
0x68			**OEMCR0**	output mode Clear Register
[31:0]	w	32'h0	OEMC	output mode Clear of corresponding GPIO[31:0]
0x80			**DIR1**	Data Input Register
[31:13]			RSVD	
[12:0]	r	13'h0	IN	GPIO[44:32] input value
0x84			**DOR1**	Data Output Register
[31:13]			RSVD	
[12:0]	rw	13'h0	OUT	GPIO[44:32] output value if output enabled
0x88			**DOSR1**	Data Output Set Register
[31:13]			RSVD	
[12:0]	w	13'h0	DOS	set 1 to pull up output of corresponding GPIO[44:32]
0x8C			**DOCR1**	Data Output Clear Register
[31:13]			RSVD	
[12:0]	w	13'h0	DOC	set 1 to pull down output of corresponding GPIO[44:32]
0x90			**DOER1**	Data Output Enable Register
[31:13]			RSVD	
[12:0]	rw	13'h0	DOE	GPIO[44:32] output enable
0x94			**DOESR1**	Data Output Enable Set Register
[31:13]			RSVD	
[12:0]	w	13'h0	DOES	set 1 to enable output of corresponding GPIO[44:32]
0x98			**DOECR1**	Data Output Enable Clear Register
[31:13]			RSVD	
[12:0]	w	13'h0	DOEC	set 1 to disable output of corresponding GPIO[44:32]
0x9C			**IER1**	Interrupt Enable Register
[31:13]			RSVD	
[12:0]	rw	13'h0	IER	GPIO[44:32] interrupt enable
0xA0			**IESR1**	Interrupt Enable Set Register
[31:13]			RSVD	
[12:0]	w	13'h0	IES	set 1 to enable interrupt of corresponding GPIO[44:32]
0xA4			**IECR1**	Interrupt Enable Clear Register
[31:13]			RSVD	
[12:0]	w	13'h0	IEC	set 1 to disable interrupt of corresponding GPIO[44:32]
0xA8			**ITR1**	Interrupt Type Register
[31:13]			RSVD	
[12:0]	rw	13'h0	ITR	GPIO[44:32] interrupt type
0xAC			**ITSR1**	Interrupt Type Set Register
[31:13]			RSVD	
[12:0]	w	13'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[44:32]

Continued on the next page...

Table 5-6: **HPSYS_GPIO Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0xB0			**ITCR1**	Interrupt Type Clear Register
[31:13]			RSVD	
[12:0]	w	13’h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[44:32]
0xB4			**IPHR1**	Interrupt Polarity High Register
[31:13]			RSVD	
[12:0]	rw	13’h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[44:32]
0xB8			**IPHSR1**	Interrupt Polarity High Set Register
[31:13]			RSVD	
[12:0]	w	13’h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[44:32]
0xBC			**IPHCR1**	Interrupt Polarity High Clear Register
[31:13]			RSVD	
[12:0]	w	13’h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[44:32]
0xC0			**IPLR1**	Interrupt Polarity Low Register
[31:13]			RSVD	
[12:0]	rw	13’h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[44:32]
0xC4			**IPLSR1**	Interrupt Polarity Low Set Register
[31:13]			RSVD	
[12:0]	w	13’h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[44:32]
0xC8			**IPLCR1**	Interrupt Polarity Low Clear Register
[31:13]			RSVD	
[12:0]	w	13’h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[44:32]
0xCC			**ISR1**	Interrupt Status Register
[31:13]			RSVD	
[12:0]	rw	13’h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[44:32]
0xE0			**OEMR1**	output mode Register
[31:13]			RSVD	
[12:0]	rw	13’h0	OEM	output mode of corresponding GPIO[44:32]
0xE4			**OEMSR1**	output mode Set Register
[31:13]			RSVD	
[12:0]	w	13’h0	OEMS	output mode Set of corresponding GPIO[44:32]
0xE8			**OEMCR1**	output mode Clear Register
[31:13]			RSVD	
[12:0]	w	13’h0	OEMC	output mode Clear of corresponding GPIO[44:32]

6 DMA

6.1 DMAC

6.1.1 Introduction

The DMAC (Direct Memory Access Controller) facilitates the transfer of data between two different address ranges on the bus. The DMAC features 8 independent channels, each configurable with a source address range and a target address range, which are mapped to the address ranges of various memory or peripherals. This enables high-effciency transfers between memory - memory, memory - peripheral, peripheral - memory, and peripheral - peripheral, effectively alleviating the workload of the CPU. The DMAC supports both peripheral response mode and memory transfer mode: in peripheral response mode, theDMAC performs transfers based on DMA requests from peripherals, adapting to their bandwidth; in memory transfer mode, theDMAC does not wait for DMA requests from peripherals and completes data transfers as quickly as possible. When multiple channels are enabled simultaneously, the DMAC executes transfers in order of priority from high to low; additionally, during the transfer process of lower-priority channels, higher-priority channels can preempt the transfer. An interrupt or PTC trigger can be generated when each channel has transferred half or completed its transfer.

6.1.2 Main Features

- Single AHBmaster control bus, capable of accessing SRAM, PSRAM, FLASH, AHB, and APB peripherals, among others.
- 8 independent configurable channels
- Each channel's DMA request can select 1 from up to 64external peripheral DMA requests, or can be requested by software.
- Each channel supports 4 levels of priority configuration; in the event of equal priority, the decision is based on the channel number.
- Supports data transfers from peripherals to memory, memory to peripherals, memory to memory, and peripherals to peripherals.
- Both source and destination addresses independently support single-byte, double-byte, and four-byte access. The addresses of the source and destination must be aligned according to the size of the transfer data unit and support automatic address increment.
- The number of transmission units per instance can be configured from 0 to 65535.
- It supports circular buffer mode, automatically restarting after a single transmission is completed.
- Each channel supports 3 types of event flags: transmission complete, half transmission, or transmission error, and can independently generate interrupt requests and PTC triggers.
- Each channel supports block transfer mode with configurable block sizes.

6.1.3 Peripheral Requests

Each channel can select any one of the 64 peripheral request sources as its bound request source by configuring CSELR1/2_CxS, and the response signal of that peripheral is also bound to that channel. The peripheral request table for DMAC is as follows.

Table 6-1: DMAC Peripheral Request Table

CSELR1/2_CxS	DMAC1
0	mpi1
1	mpi2
2	/
3	i2c4
4	usart1_tx
5	usart1_rx
6	usart2_tx
7	usart2_rx
8	gptim1_update
9	gptim1_trigger
10	gptim1_cc1
11	gptim1_cc2
12	gptim1_cc3
13	gptim1_cc4
14	btim1
15	btim2
16	atim1_update
17	atim1_trigger
18	atim1_cc1
19	atim1_cc2
20	atim1_cc3
21	atim1_cc4
22	i2c1
23	i2c2
24	i2c3
25	atim1_com
26	usart3_tx
27	usart3_rx
28	spi1_tx
29	spi1_rx
30	spi2_tx
31	spi2_rx
32	i2s1_tx
33	i2s1_rx
34	/
35	/
36	pdm1_l
37	pdm1_r
38	gpadc
39	audadc_ch0
40	audadc_ch1
41	auddac_ch0
42	auddac_ch1
43	gptim2_update
44	gptim2_trigger
45	gptim2_cc1
46	audprc_tx_out_ch1
47	audprc_tx_out_ch0
48	audprc_tx_ch3
49	audprc_tx_ch2

Continued on the next page...

Table 6-1: DMAC Peripheral Request Table (Continued)

CSELR1/2_CxS	DMAC1
50	audprc_tx_ch1
51	audprc_tx_ch0
52	audprc_rx_ch1
53	audprc_rx_ch0
54	gptim2_cc2
55	gptim2_cc3
56	gptim2_cc4
57	sdmmc1
58	/
59	/
60	/
61	/
62	/
63	/

6.1.4 DMAC Function Description

6.1.4.1 DMAC Block Diagram

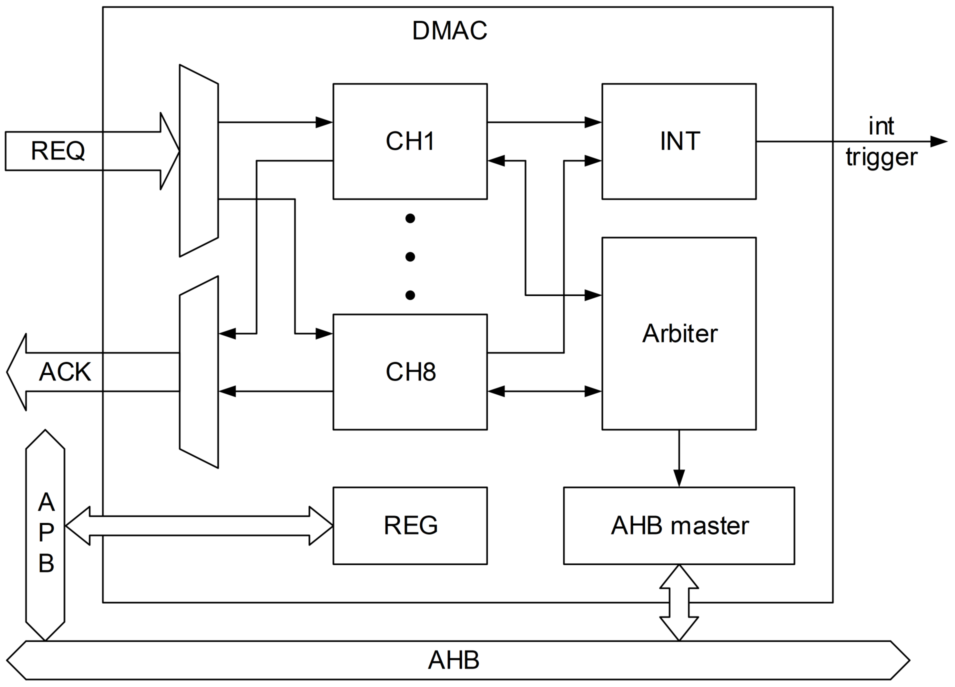


Figure 6-1: DMAC Block Diagram

6.1.4.2 Transfer Effciency

The DMAC shares the AHB bus with the CPU and other master devices. When the CPU and DMAC simultaneously access the same target (memory or peripheral), the DMAC request may pause the CPU access to the system bus, with the bus arbiter performing round-robin scheduling to ensure that the CPU can obtain at least half of the total bus bandwidth. Similarly, when other master devices access the same AHB target as the DMAC, the bandwidth available to the DMAC will also decrease. When there are no other accesses on the bus, the DMAC can complete a single unit transfer in as little as 2 HCLK cycles, depending on the AHB wait cycles of the target being accessed.

6.1.4.3 Transfer Mode

Each channel of the DMAC can be independently configured for either peripheral transfer mode or memory transfer mode, controlled by the CCRx_MEM2MEM register. The primary distinction lies in whether the peripheral request/acknowledge mechanism is enabled. The peripheral transfer mode adapts to the data bandwidth of the peripheral through the request/acknowledge mechanism, initiating transmission only when the peripheral is ready. This mode is typically used in scenarios such as UART transmission and reception, I2S audio input and output, and FLASH programming. The memory transfer mode does not require waiting for a request signal and will transmit at the maximum possible data bandwidth, commonly used for SRAM transfers and CRC checks.

6.1.4.4 Transmission Process

In peripheral transfer mode, the DMAC transfer utilizes a peripheral request/acknowledgment mechanism, initiated by the peripheral request, and follows these steps for transmission:

1. When the peripheral needs to transfer data (e.g., when the receive cache is full or the send cache is empty), it sends a request signal to the corresponding DMAC channel;
2. The DMAC processes the request based on the priority of the relevant channel; when this channel has the highest priority among all requesting channels, it initiates either a unit transfer or a block transfer for that channel;
3. After the unit transfer or block transfer is completed, the DMAC sends an acknowledgment signal to the peripheral;
4. Once the peripheral receives the acknowledgment signal from the DMAC, it will release the request;
5. Upon detecting the release of the peripheral request, the DMAC will then release the acknowledgment signal;
6. After the peripheral detects the release of the response signal, if there is still a transmission requirement, it can continue to send requests and restart the DMAC unit transfer or block transfer.

In memory transfer mode, the transfer is initiated by software and is repeated for unit transfers until completion, provided that the channel priority is sufcient.

6.1.4.5 Transfer Enable

The DMAC features 8 independent channel enables. When CCRx_EN is set to 1 and CNDTRx is not 0, the channel transfer initiates, responding to peripheral requests in peripheral transfer mode and immediately commencing the transfer in memory transfer mode. It is important to note that when the channel is enabled, even if the previous transfer has been completed, rewriting a non-zero CNDTRx will immediately initiate the DMAC transfer. Prior to this, ensure that other parameters have been configured, or confrm that CCRx_EN is set to 0 when writing to CNDTRx.

6.1.4.6 Transfer Unit

The basic unit of transfer for the DMACis a transfer unit, which consists of 3steps:

1. Read data from the source address through the bus in a single operation, configurable for single-byte/double-byte/four-byte;
2. Write the read data to the target address in a single operation, configurable for single-byte/double-byte/four-byte;
3. Update the count, stopping the transfer based on the count result, or calculate the address for the next transfer.

6.1.4.7 Transfer Quantity

The quantity transferred by the DMAC for each channel is controlled by CNDTRx , counted in transfer units, supporting a range of 0~65535 . For example, if configured for fourbyte transfers, the maximum data amount for a single transfer by the channel would be 65535×4=262140 bytes. After configuring CNDTRx and starting the unit transfer (CCRx_EN=1) , the value of the CNDTRx register decreases by 1 after each completed transfer unit. In non-circular mode (CCRx_CIRC=0) , the transfer will stop when CNDTRx decreases to 0 . In circular mode (CCRx_CIRC=1) , when CNDTRx decreases to 0 , it will immediately reload the initial value configured in software and continue the transfer.

6.1.4.8 Circular Mode

In circular mode, the transmission does not stop automatically. After the last data transmission is completed, the CNDTRx register will automatically reload the initial programmed value. The current internal address register will reload the base values from the CPARx and CM0ARx registers. The functionality of ping-pong caching can be achieved in circular mode by monitoring the half transmission and transmission complete flags.

6.1.4.9 Transmission Direction

The transmission direction of the DMAC is determined by the CCRx_DIR .

Table 6-2: **DMAC Transmission Direction**

	Source Starting Address (Read)	Destination Starting Address (Write)
DIR=0	CPARx	CM0ARx
DIR=1	CM0ARx	CPARx

6.1.4.10 Transmission Bit Width

In DMAC transmission, access to the source and destination addresses on the bus can be independently configured as single-byte/double-byte/four-byte types through CCRx_MSIZE and CCRx_PSIZE. DMAC does not provide data packing and unpacking functions; therefore, when the data bit widths of the source and destination addresses are inconsistent, data may be truncated or padded, as illustrated in the example below.

Table 6-3: **DMAC Transmission Bit Width**

Source Size	Destination Size	Source Data	Destination Data
byte	byte	0xB0 @0x0 0xB1 @0x1 0xB2 @0x2 0xB3 @0x3	0xB0 @0x0 0xB1 @0x1 0xB2 @0x2 0xB3 @0x3
byte	half-word(16bit)	0xB0 @0x0 0xB1 @0x1 0xB2 @0x2 0xB3 @0x3	0x00B0 @0x0 0x00B1 @0x2 0x00B2 @0x4 0x00B3 @0x6
byte	word(32bit)	0xB0 @0x0 0xB1 @0x1 0xB2 @0x2 0xB3 @0x3	0x000000B0 @0x0 0x000000B1 @0x4 0x000000B2 @0x8 0x000000B3 @0xC
half-word	byte	0xB1B0 @0x0 0xB3B2 @0x2 0xB4B4 @0x4	0xB0 @0x0 0xB2 @0x1 0xB4 @0x2

Continued on the next page...

Table 6-3: Transmission Bit Width (Continued)

Source Size	Destination Size	Source Data	Destination Data
		0xB7B6 @0x6	0xB6 @0x3
half-word	half-word	0xB1B0 @0x0 0xB3B2 @0x2 0xB4B4 @0x4 0xB7B6 @0x6	0xB1B0 @0x0 0xB3B2 @0x2 0xB4B4 @0x4 0xB7B6 @0x6
half-word	word	0xB1B0 @0x0 0xB3B2 @0x2 0xB4B4 @0x4 0xB7B6 @0x6	0x0000B1B0 @0x0 0x0000B3B2 @0x4 0x0000B5B4 @0x8 0x0000B7B6 @0xC
word	byte	0xB3B2B1B0 @0x0 0xB7B6B5B4 @0x4 0xBBBAB9B8 @0x8 0xBFBEBDBC @0xC	0xB0 @0x0 0xB4 @0x1 0xB8 @0x2 0xBC @0x3
word	half-word	0xB3B2B1B0 @0x0 0xB7B6B5B4 @0x4 0xBBBAB9B8 @0x8 0xBFBEBDBC @0xC	0xB1B0 @0x0 0xB5B4 @0x2 0xB9B8 @0x4 0xBDBC @0x6
word	word	0xB3B2B1B0 @0x0 0xB7B6B5B4 @0x4 0xBBBAB9B8 @0x8 0xBFBEBDBC @0xC	0xB3B2B1B0 @0x0 0xB7B6B5B4 @0x4 0xBBBAB9B8 @0x8 0xBFBEBDBC @0xC

6.1.4.11 Transmission Address

The starting source and destination addresses for transmission are determined by CPARx,CM0ARx, and CCRx_DIR. If increment mode is enabled (with CCRx_MINC and CCRx_PINC configured independently), the address for the next transmission will be the previous transmission's address plus 1, 2, or 4 (based on the transmission bit width).

During the transmission process, these registers will retain their initially programmed values. The software cannot access the address currently being transmitted. Typically, the transmission address configuration for peripherals FIFO is set to non-incrementing, while the transmission address configuration for memory is set to incrementing.

6.1.4.12 Channel Arbitration

The DMAC arbitrator manages the priorities among different channels. When the arbitrator grants priority to a valid channel (either through a hardware request or a software trigger), it initiates a unit transfer or block transfer for that channel. Subsequently, the arbitrator will arbitrate again among valid channels and select the one with the highest priority.

The arbitrator determines priority based on the following criteria:

1. The channel for peripheral transfer mode takes precedence over the channel for memory transfer mode.
2. Among channels that are both in peripheral transfer mode or memory transfer mode, the channel with a lower CCRx_PLhas a higher priority.
3. If the transfer modes and CCRx_PLare the same, the channel with the smaller channel number has a higher priority (for example, channel 1has a higher priority than channel 2).

During the transmission process of a specific channel, if another channel with a higher priority issues a transfer request, the arbiter will switch the transfer to the higher priority channel after the current channel's unit transfer or block transfer is completed.

6.1.4.13 Block Transfer

Block transfer is only effective in peripheral transfer mode. CBSRx defines how many consecutive unit transfers the DMAC must complete for each peripheral request. For example, when CBSRx is set to 4, every time a peripheral initiates a request, the DMAC will complete four consecutive unit transfers before providing a response signal, during which CNDTRx will also decrease by 4. If the remaining CNDTRx to be transferred is less than CBSRx, only the remaining CNDTRx unit transfers will be executed.

6.1.4.14 Notifcation Mechanism

The DMAC can independently generate interrupts and PTC trigger signals for each channel, with each type of interrupt configurable for enablement. An HTIFx interrupt and trigger is generated when at least half of the channel transfer is completed. A TCIFx interrupt and trigger is generated upon full completion of the channel transfer. A TEIFx interrupt and trigger is generated if a bus access error occurs during the channel transfer. When any of these three types of interrupts occur, a GIFx interrupt is generated. Channel interrupts can be cleared individually or collectively using CGIFx.

6.1.4.15 Channel Configuration Process

When configuring the DMAC channel, please follow these steps:

1. Set the peripheral register address in the CPARx register.
   After a peripheral request occurs, or after enabling the channel in memory mode, data will be transferred from this address to memory or from memory to this address.
2. Set the memory address in the CM0ARx Register.
   After a peripheral request occurs, or after enabling the channel in memory mode, data will be written to memory or read from memory.
3. Ensure that CCRx_EN is set to 0, and write the number of units to be transmitted into the CNDTRx Register.
   This value will decrement after each data transfer.
4. Configure the following parameters in the CCRx Register:
   - Channel priority
   - Data transfer direction
   - Circular Mode
   - Peripheral and memory increment mode
   - Peripheral and Memory Data Size
   - Enable interrupts for transfer completion halfway and / or fully completed, and /or when a transfer error occurs.
5. Set CCRx_ENto 1to activate the channel.
   Once enabled, the channel can process requests from the connected peripherals or initiate memory-to-memory transfers.

6.1.4.16 Transfer Completion Handling

In non-circular mode, after a transfer is completed on a specific channel of the DMAC, CNDTRx is automatically reset to zero. The software must write 0 to CCRx_EN for that channel to prevent erroneous triggering during the next configuration. If an interrupt flag appears, the software should clear the interrupt flag after performing the necessary handling.

In circular mode, the DMAC does not automatically stop the transfer. When the software intends to stop the transfer, it should write 0 to the CCRx_EN of that channel and clear the interrupt flag.

6.1.5 DMAC Register

DMAC base address is 0x50081000.

Table 6-4: **DMAC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	
[31]	r	1'h0	TEIF8	channel transfer error flag
[30]	r	1'h0	HTIF8	channel half transfer flag
[29]	r	1'h0	TCIF8	channel transfer complete flag
[28]	r	1'h0	GIF8	channel global interrupt flag
[27]	r	1'h0	TEIF7	channel transfer error flag
[26]	r	1'h0	HTIF7	channel half transfer flag
[25]	r	1'h0	TCIF7	channel transfer complete flag
[24]	r	1'h0	GIF7	channel global interrupt flag
[23]	r	1'h0	TEIF6	channel transfer error flag
[22]	r	1'h0	HTIF6	channel half transfer flag
[21]	r	1'h0	TCIF6	channel transfer complete flag
[20]	r	1'h0	GIF6	channel global interrupt flag
[19]	r	1'h0	TEIF5	channel transfer error flag
[18]	r	1'h0	HTIF5	channel half transfer flag
[17]	r	1'h0	TCIF5	channel transfer complete flag
[16]	r	1'h0	GIF5	channel global interrupt flag
[15]	r	1'h0	TEIF4	channel transfer error flag
[14]	r	1'h0	HTIF4	channel half transfer flag
[13]	r	1'h0	TCIF4	channel transfer complete flag
[12]	r	1'h0	GIF4	channel global interrupt flag
[11]	r	1'h0	TEIF3	channel transfer error flag
[10]	r	1'h0	HTIF3	channel half transfer flag
[9]	r	1'h0	TCIF3	channel transfer complete flag
[8]	r	1'h0	GIF3	channel global interrupt flag
[7]	r	1'h0	TEIF2	channel transfer error flag
[6]	r	1'h0	HTIF2	channel half transfer flag
[5]	r	1'h0	TCIF2	channel transfer complete flag
[4]	r	1'h0	GIF2	channel global interrupt flag
[3]	r	1'h0	TEIF1	channel transfer error flag. Set when bus error detected. Cleared when write 1 to CTEIF or CGIF.
[2]	r	1'h0	HTIF1	channel half transfer flag. Set when half NDT are transferred. Cleared when write 1 to CHTIF or CGIF.
[1]	r	1'h0	TCIF1	channel transfer complete flag. Set when all NDT are transferred. Cleared when write 1 to CTCIF or CGIF.
[0]	r	1'h0	GIF1	channel global interrupt flag. Set when any of TEIF/HTIF/TCIF asserted. Cleared when TEIF/HTIF/TCIF all cleared.
0x04			**IFCR**	
[31]	w	1'h0	CTEIF8	CTEIF, transfer error flag clear
[30]	w	1'h0	CHTIF8	CHTIF, half transfer flag clear
[29]	w	1'h0	CTCIF8	CTCIF, transfer complete flag clear
[28]	w	1'h0	CGIF8	CGIF, global interrupt flag clear
[27]	w	1'h0	CTEIF7	CTEIF, transfer error flag clear
[26]	w	1'h0	CHTIF7	CHTIF, half transfer flag clear

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[25]	w	1'h0	CTCIF7	CTCIF, transfer complete flag clear
[24]	w	1'h0	CGIF7	CGIF, global interrupt flag clear
[23]	w	1'h0	CTEIF6	CTEIF, transfer error flag clear
[22]	w	1'h0	CHTIF6	CHTIF, half transfer flag clear
[21]	w	1'h0	CTCIF6	CTCIF, transfer complete flag clear
[20]	w	1'h0	CGIF6	CGIF, global interrupt flag clear
[19]	w	1'h0	CTEIF5	CTEIF, transfer error flag clear
[18]	w	1'h0	CHTIF5	CHTIF, half transfer flag clear
[17]	w	1'h0	CTCIF5	CTCIF, transfer complete flag clear
[16]	w	1'h0	CGIF5	CGIF, global interrupt flag clear
[15]	w	1'h0	CTEIF4	CTEIF, transfer error flag clear
[14]	w	1'h0	CHTIF4	CHTIF, half transfer flag clear
[13]	w	1'h0	CTCIF4	CTCIF, transfer complete flag clear
[12]	w	1'h0	CGIF4	CGIF, global interrupt flag clear
[11]	w	1'h0	CTEIF3	CTEIF, transfer error flag clear
[10]	w	1'h0	CHTIF3	CHTIF, half transfer flag clear
[9]	w	1'h0	CTCIF3	CTCIF, transfer complete flag clear
[8]	w	1'h0	CGIF3	CGIF, global interrupt flag clear
[7]	w	1'h0	CTEIF2	CTEIF, transfer error flag clear
[6]	w	1'h0	CHTIF2	CHTIF, half transfer flag clear
[5]	w	1'h0	CTCIF2	CTCIF, transfer complete flag clear
[4]	w	1'h0	CGIF2	CGIF, global interrupt flag clear
[3]	w	1'h0	CTEIF1	CTEIF, transfer error flag clear. Write 1 to clear TEIF.
[2]	w	1'h0	CHTIF1	CHTIF, half transfer flag clear. Write 1 to clear HTIF.
[1]	w	1'h0	CTCIF1	CTCIF, transfer complete flag clear. Write 1 to clear TCIF.
[0]	w	1'h0	CGIF1	CGIF, global interrupt flag clear. Write 1 to clear all TEIF/HTIF/TCIF.
0x08			**CCR1**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x0C			**CNDTR1**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x10			**CPAR1**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x14			**CM0AR1**	
[31:0]	rw	32'h0	MA	memory address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x18			**CBSR1**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x1C			**CCR2**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x20			**CNDTR2**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x24			**CPAR2**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x28			**CM0AR2**	

Continued on the next page...

Table 6-4: **DMAC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x2C			**CBSR2**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x30			**CCR3**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x34			**CNDTR3**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x38			**CPAR3**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x3C			**CM0AR3**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x40			**CBSR3**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x44			**CCR4**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x48			**CNDTR4**	
[31:16]			RSVD	

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x4C			**CPAR4**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x50			**CM0AR4**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x54			**CBSR4**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x58			**CCR5**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9:8]	rw	2’h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1’h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1’h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1’h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1’h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1’h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1’h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1’h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x5C			**CNDTR5**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x60			**CPAR5**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x64			**CM0AR5**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x68			**CBSR5**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x6C			**CCR6**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled

Continued on the next page...

Table 6-4: **DMAC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x70			**CNDTR6**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x74			**CPAR6**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x78			**CM0AR6**	

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x7C			**CBSR6**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x80			**CCR7**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[7]	rw	1'h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw	1'h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1'h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1'h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x84			**CNDTR7**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x88			**CPAR7**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x8C			**CM0AR7**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x90			**CBSR7**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non memory-to-memory mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x94			**CCR8**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved
[9:8]	rw	2'h0	PSIZE	peripheral size Defines the data size of each DMA transfer to the identified peripheral. In memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1’h0	MINC	memory increment mode Defines the increment mode for each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 0: disabled 1: enabled
[6]	rw	1’h0	PINC	peripheral increment mode Defines the increment mode for each DMA transfer to the identified peripheral. n memory-to-memory mode, this field identifies the memory destination if DIR = 1 and the memory source if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral destination if DIR = 1 and the peripheral source if DIR = 0. 0: disabled 1: enabled
[5]	rw	1’h0	CIRC	circular mode 0: disabled 1: enabled
[4]	rw	1’h0	DIR	data transfer direction This bit must be set only in memory-to-peripheral and peripheral-to-memory modes. 0: read from peripheral Source attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Destination attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode. 1: read from memory Destination attributes are defined by PSIZE and PINC, plus the CPARx register. This is still valid in a memory-to-memory mode. Source attributes are defined by MSIZE and MINC, plus the CM0ARx register. This is still valid in a peripheral-to-peripheral mode.
[3]	rw	1’h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1’h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1’h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1’h0	EN	channel enable When a channel transfer error occurs, this bit is cleared by hardware. It can not be set again by software (channel x re-activated) until the TEIFx bit of the ISR register is cleared (by setting the CTEIFx bit of the IFCR register). 0: disabled 1: enabled
0x98			**CNDTR8**	
[31:16]			RSVD	

Continued on the next page...

Table 6-4: DMAC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x9C			**CPAR8**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0xA0			**CM0AR8**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0xA4			**CBSR8**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0xA8			**CSELR1**	
[31:30]			RSVD	
[29:24]	rw	6'h0	C4S	DMA channel 4 selection
[23:22]			RSVD	
[21:16]	rw	6'h0	C3S	DMA channel 3 selection
[15:14]			RSVD	
[13:8]	rw	6'h0	C2S	DMA channel 2 selection
[7:6]			RSVD	
[5:0]	rw	6'h0	C1S	DMA channel 1 selection
0xAC			**CSELR2**	
[31:30]			RSVD	
[29:24]	rw	6'h0	C8S	DMA channel 8 selection
[23:22]			RSVD	
[21:16]	rw	6'h0	C7S	DMA channel 7 selection
[15:14]			RSVD	
[13:8]	rw	6'h0	C6S	DMA channel 6 selection
[7:6]			RSVD	
[5:0]	rw	6'h0	C5S	DMA channel 5 selection

6.2 ExtDMA

6.2.1 Introduction

ExtDMA (Extended Direct Memory Access) can effciently transfer data between two different address ranges on the bus. Compared to the DMAC, ExtDMA is more effcient when accessing external memory (such as FLASH and PSRAM), but it has only one channel, supports only 4-byte aligned transfers, and does not respond to peripheral requests.

6.2.2 Main Features

- Single AHBmaster control, capable of accessing SRAM, PSRAM, FLASH, etc., and supporting BURST transfers.
- A single transmission channel with a built-in depth of 16and a bit width of 32bits for FIFO.
- Both source and destination addresses are 4 bytes in size, supporting automatic address increment.
- The maximum number of transmission units for a single configuration is 2^{20}-1, with each unit fxed at 4bytes transfer, resulting in a maximum single transfer of 4M bytes.
- The channel supports transfer completion, half transfer, and transfer error event flags, and can generate interrupt requests and PTC triggers.

6.2.3 Function Description

6.2.3.1 Transfer Effciency

ExtDMA shares the AHB bus with the CPU and other master devices. When the CPU and ExtDMA simultaneously access the same memory, requests from ExtDMA may pause CPU access to the system bus, with the bus arbiter performing round-robin scheduling to ensure that the CPU can at least obtain some bandwidth on the bus. Similarly, when other master devices access the same AHB target as ExtDMA, the access bandwidth for ExtDMA will also decrease. ExtDMA preferentially employs the AHB Burst transfer mode, thereby achieving better transfer effciency than DMAC when accessing memory that is friendly to Burst transfers.

6.2.3.2 Data Address and Bit Width

The source address SRCAR and the destination address DSTAR of ExtDMA must both be aligned to four bytes, and the source data width CCR_SRCSIZE and the destination data width CCR_DSTSIZE must also be set to four bytes.

If address increment is enabled (the source address and destination address are configured by CCR_SRCINC and CCR_DSTINC respectively), each time ExtDMA completes a source data read or destination data write operation, the address for the next read or write operation will be the previous address plus 4 ; otherwise, it will remain the same. The non-incrementing address mode is primarily used for accessing fxed entry addresses of FIFO , such as the data entry for CRC.

6.2.3.3 Transfer Enable

CCR_EN is the channel enable register. When CCR_EN is set to 1 and CNDTR is not 0 , data transmission will commence. It is important to note that when the channel is enabled, even if the previous transmission has completed, rewriting a non-zero CNDTR will immediately initiate the ExtDMA transmission. Therefore, it is advisable to ensure that other parameters are configured beforehand; otherwise, the channel enable should be turned off first.

6.2.3.4 Transfer Quantity

The quantity of data transfer is configured through CNDTR , counted in four-byte units, supporting a range from 0 to 2^{20} -1 . For example, when CNDTR is configured to 1000, the amount of data transferred will be 4000 bytes. After initiating the data transfer, for each completed four-byte read, CNDTR is decremented by 1 . CMPRNDTR represents the amount of data to be written and does not require software configuration. After initiating the data transfer, CMPRNDTR automatically copies the initial CNDTR , and thereafter, for each completed four-byte write, CMPRNDTR is decremented by 1 . When both CNDTR and CMPRNDTR are reduced to 0, all data reading and writing are completed, and ExtDMA stops the transfer.

6.2.3.5 Notifcation Mechanism

ExtDMA can generate interrupts and PTC trigger signals, and can individually configure the enablement of each type of interrupt. When at least half of the transfer is complete, it generates HTIF interrupts and triggers. After the entire transfer is complete, it generates TCIF interrupts and triggers. When a bus access error occurs during the transfer, it generates TEIF interrupts and triggers.

Interrupts are enabled through control bits such as TCIE/HTIE/TEIE in the CCR register. The interrupt status can be accessed through the ISR register and cleared using the IFCR register.

6.2.3.6 Exception Handling

If the ExtDMA transfer cannot be completed due to a configuration error, you can set CCR_RESET to 1 to reset the ExtDMA logic. After the reset, CCR_RESET will automatically clear to 0 . Other configuration registers will not be affected by this operation.

6.2.3.7 Recommended Configuration Process

1. Set the source address SRCARand destination address DSTARto be four-byte aligned.
2. Ensure that CCR_EN is set to 0, and write the amount of data to be transferred in four-byte units to the CNDTRRegister.
3. In Configure the following parameters in the CCR Register:
   - Address increment mode CR_SRCINC and CCR_DSTINC
   - The source data width CCR_SRCSIZE and destination data width CCR_DSTSIZE are both four bytes.
   - Interrupt Enable

4.Set CCR_EN to 1 to initiate data transfer.
5.Wait for the interrupt to occur and handle it, then set CCR_EN to 0.

6.2.4 ExtDMA Register

EXTDMA base address is 0x50001000;

Table 6-5: **ExtDMA Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	interrupt status register
[31:4]			RSVD	
[3]	r	1’h0	TEIF	TEIF, transfer error flag
[2]	r	1’h0	HTIF	HTIF, half transfer flag

Continued on the next page...

Table 6-5: ExtDMA Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	r	1'h0	TCIF	TCIF, transfer complete flag
[0]	r	1'h0	GIF	GIF, global interrupt flag
0x04			**IFCR**	interrupt clear register
[31:4]			RSVD	
[3]	w1s	1'h0	CTEIF	CTEIF, transfer error flag clear
[2]	w1s	1'h0	CHTIF	CHTIF, half transfer flag clear
[1]	w1s	1'h0	CTCIF	CTCIF, transfer complete flag clear
[0]	w1s	1'h0	CGIF	CGIF, global interrupt flag clear
0x08			**CCR**	channel control register
[31]	w1s	1'h0	RESET	Software reset, will clear extdma status. Active high. Will be cleared by HW automatically
[30:20]			RSVD	
[19:18]	rw	2'h3	SRCBURST	source burst transfer configuration 00: single transfer 01: INCR4 (incremental burst of 4 beats) 10: INCR8 (incremental burst of 8 beats) 11: INCR16 (incremental burst of 16 beats)
[17:16]	rw	2'h3	DSTBURST	destination burst transfer configuration 00: single transfer 01: INCR4 (incremental burst of 4 beats) 10: INCR8 (incremental burst of 8 beats) 11: INCR16 (incremental burst of 16 beats)
[15:12]			RSVD	
[11:10]	rw	2'h2	SRCSIZE	source size Defines the data size of each DMA transfer to the source memory. Should be fixed to 10 (32 bits), word access allowed only.
[9:8]	rw	2'h2	DSTSIZE	destination size Defines the data size of each DMA transfer to the destination memory. Should be fixed to 10 (32 bits), word access allowed only.
[7]	rw	1'h1	SRCINC	source increment mode Defines the increment mode for each DMA transfer to the source memory. 0: disabled 1: enabled
[6]	rw	1'h1	DSTINC	destination increment mode Defines the increment mode for each DMA transfer to the destination memory. 0: disabled 1: enabled
[5:4]			RSVD	
[3]	rw	1'h0	TEIE	transfer error interrupt enable 0: disabled 1: enabled
[2]	rw	1'h0	HTIE	half transfer interrupt enable 0: disabled 1: enabled
[1]	rw	1'h0	TCIE	transfer complete interrupt enable 0: disabled 1: enabled
[0]	rw	1'h0	EN	extdma enable. Will be cleared if ccr_reset is written
0x0C			**CNDTR**	number of data register
[31:20]			RSVD	

Continued on the next page...

Table 6-5: **ExtDMA Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[19:0]	rw	20’h0	NDT	number of data to transfer (0 to 220 - 1) This field is updated by hardware when the channel is enabled: It is decremented after each transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached. If this field is zero, no transfer can be served whatever the channel enabled or not
0x10			**SRCAR**	source address register
[31:0]	rw	32’h0	SRCADDR	source address It contains the base address of the source data to be read. Should be word aligned
0x14			**DSTAR**	destination 0 address register
[31:0]	rw	32’h0	DSTADDR	destination address It contains the base address of the destination data to be written. Should be word aligned

7 Connecting Peripheral

7.1 I2C

HPSYS has four I2C modules.

7.1.1 Introduction

The I2C (Inter-Integrated Circuit) interface supports both master and slave roles, allowing it to communicate with I2C peripherals as a master device and respond to external I2C master devices as a slave. TheI2C features an integrated 8 byte FIFO , enabling both single read/write operations and bulk data read/write via DMA. The I2C supports standard mode (standard-mode) , fast mode (fast-mode) , fast-mode plus (fast-mode plus) , and high-speed mode (high-speed-mode) , with a maximum speed of up to 3.4Mbps.

7.1.2 Main Features

Can operate as both a master and a slave device simultaneously

- Supports multi-master bus architecture
- Supports standard mode (up to 100 kbps)
- Supports fast mode (up to 400 kbps)
- Supports enhanced fast mode (up to 1 Mbps)
- Supports high-speed mode (up to 3.4 Mbps)
- As a master device, supports access to 7bit or 10bit addressing
- As a slave device, supports 7bit addressing
- Configurable bus timing
- Supports clock stretching
- 8 byte FIFO, supports DMA
- Configurable digital debounce circuit
- Independent functional clock, supports dynamic adjustment of the system clock

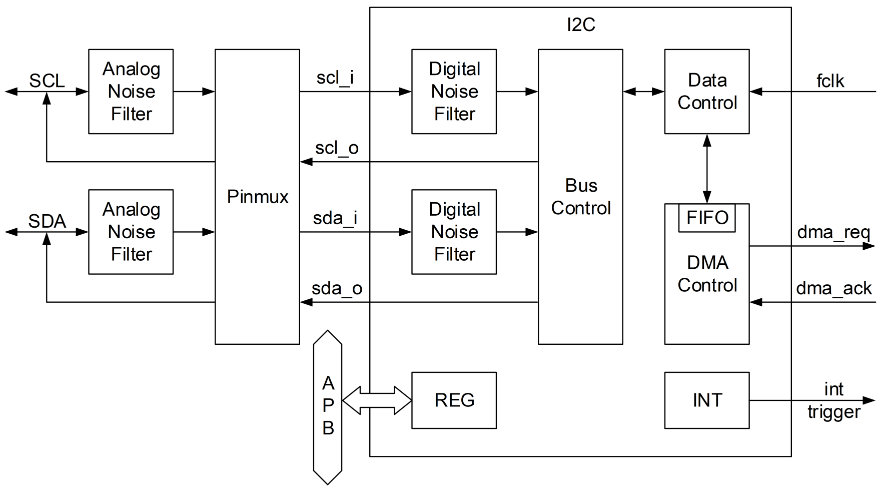


Figure 7-1: **I2C Schematic Diagram**

7.1.3 I2C Function Description

7.1.3.1 Two-Wire Transmission

The I2C bus utilizes SCL and SDA for transmission over two lines, where SCL is commonly referred to as the clock line and SDA as the data line. Both lines support bidirectional transmission, and the outputs are in open-drain mode; therefore, pull-up resistors must be added externally to the chip, with the resistance value determined by the maximum transmission rate. When the I2C bus is idle, both SCL and SDA are pulled high. The current I2C bus signal levels can be queried through BMR_SCL and BMR_SDA.

7.1.3.2 Input Filter

The SCL and SDA input signals can have glitches fltered out by both an analog flter and a digital flter. The analog flter can eliminate glitches shorter than 50ns and is configured within the PINMUX module. The digital flter can be configured via CR_DNF to select the upper limit of the glitch width to be fltered, with a maximum configurable limit of 7 fclk cycles (approximately 146ns).

7.1.3.3 Transmission Rate

The transmission rate of I2C is primarily determined by the master device; however, it can also be influenced by the slave device when it supports clock stretching.

The I2C interface timing is generated based on fclk. This clock originates from the chip's peripheral clock and is independent of the system clock, meaning that changes in the system clock frequency will not affect the transmission rate ofI2C..

According to the I2C protocol, the maximum bit rate for standard mode (standard-mode) is 100 kbps, for fast mode (fast-mode) is 400kbps, for enhanced fast mode (fast-mode plus) is 1 Mbps, and for high-speed mode (high-speed mode) is 3.4 Mbps.

The base bit rate for standard mode can be calculated using the following approximate formula:

$$bit\_rate = Ffclk/(LCR\_SLV + max(LCR\_SLV, (WCR\_CNT \times 2 + 6)) + 7 + CR\_DNF)$$

Where Ffclk is the frequency of fclk (48 MHz). WCR_CNT is used to adjust the offset between the SDA and SCL edges to ensure that the setup and hold times comply with the I2C protocol; when configured properly, it will not affect the bit rate.

Fast mode/The basic bit rate for enhanced fast mode can be calculated using the following approximate formula:

$$bitrate = Ffclk/(LCR\_FLV + max(LCR\_FLV, (WCR\_CNT \times 2 + 6)) + 7 + CR\_DNF)$$

7.1.3.4 Transmission sequence

A complete I2Ctransmission should follow the subsequent transmission sequence:

1. Start bit. The master device issues this to initiate the transmission.
2. 7 bits from the address. The master device issues this to select the slave device.
3. R/nW bit. The master device issues this to indicate the direction of reception or transmission.
4. ACK bit. The slave device issues this in response to the master device's request. ACK=0 indicates a successful response. ACK=1 indicates a transmission failure.
5. 8 bits of data. Issued by the slave device during reception and by the master device during transmission. Depending on the access method of different slave devices, this may represent a register address or data.
6. ACK Bit. Issued by the master device during reception and by the slave device during transmission, it serves as a response to the previous 8bits of data.
7. Repeat steps 5-6until the data is complete or an ACK=1 is received.
8. Repeat the start bit (return to step 1) or the stop bit. Issued by the master device, this either restarts the transmission or halts it.

7.1.3.5 Operating Modes and States

I2C is by default in master mode, capable of initiating active transmissions but not monitoring address transmissions on the bus. When the software initiates a transmission, the I2C enters either master transmit or master receive state.

If CR_SLVEN is set to 1, then I2C enters slave mode, allowing it to initiate active transmissions while monitoring address transmissions on the bus. When the software initiates a transmission, the I2C enters either master transmit or master receive state. Upon detecting a start bit followed by a 7 bit address that matches SAR_ADDR, the I2C enters slave transmit or slave receive state based on the R/nW bit.

7.1.3.6 I2C Initialization Process

1. Configure the I2C speed mode (CR_MODE) and set the relevant timing registers (LCR, WCR, etc.) according to the speed.
2. Configure the IER to enable the required interrupts.
3. Enable the I2C by setting CR_SCLE=1 and CR_IUE=1.

7.1.3.7 Master Transmission Process

1. Left shift the slave device address by 1 bit, append the least signifcant bit (0), and write to the DBR.
2. TCR=TCR_START; TCR|=TCR_TB。
3. Poll the SR_TE until it is 1, or wait for the transmission complete (TE) interrupt.
4. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, send the stop bit and abort the transmission.

5. Write the data to be sent into the DBR.
6. TCR=TCR_TB。
7. Poll SR_TE until it reads 1, or wait for the TEinterrupt.
8. Write 1 to SR_TE to clear the flag. Check SR_NACK ; if it reads 1 , send the stop bit and abort the transmission.
9. Repeat steps 5-8; if this is the last piece of data, thenTCR=TCR_TB|TCR_STOP, and the stop bit will be automatically generated after the transmission is complete

7.1.3.8 Master Receiving Process

1. Left shift the slave device address by 1bit, append the least signifcant bit 1, and write to DBR.
2. TCR=TCR_START; TCR|=TCR_TB。
3. Poll SR_TE until it reads 1, or wait for the TEinterrupt.
4. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, send the stop bit and abort the transmission.
5. TCR=TCR_TB。
6. Poll SR_RFuntil it reads 1, or wait for the reception complete RFinterrupt.
7. SR_RF Write 1 Clear the flag. Retrieve the received data from DBR.
8. Repeat steps 5-7 ; if this is the last piece of data, then TCR=TCR_TB|TCR_NACK.
9. TCR=TCR_MA ; send the stop bit.
10. Poll SR_UB until it is 0 , indicating that the stop bit has been sent, and TCR=0.

7.1.3.9 Slave Transmisson process

1. When the address detection interrupt SAD is triggered, automatically respond with ACK.
2. SR_SAD Write 1 Clear the flag. Read SR_RWM , where 1 indicates sending and 0 indicates receiving. Assuming the current SR_RWM is 1 , enter the sending state.
3. Write the data to be transmitted into DBR.
4. TCR=TCR_TB。
5. Poll SR_TE until it reads 1, or wait for the TEinterrupt.
6. Write 1 to SR_TE to clear the flag. Check SR_NACK ; if it reads 1 , abort the transmission.
7. Repeat steps 3-6until the stop bit interrupt SSDis detected.
8. Write 1 to SR_SSD to clear the flag.

7.1.3.10 Slave receiving process

1. When the address detection interrupt SAD is triggered, automatically respond with ACK.
2. Write 1 to SR_SAD to clear the flag. Read SR_RWM , where 1 indicates transmission and 0 indicates reception. Assuming the current SR_RWM is 0 , enter the receiving state.
3. TCR=TCR_TB。
4. Poll SR_RF until it reads 1 , or wait for the RF interrupt to signal the completion of reception.
5. Write 1 to SR_RF to clear the flag. Retrieve the received data from DBR.
6. Repeat steps 3-5; if the cache is nearing capacity, setTCR=TCR_TB|TCR_NACK, until the stop bit interrupt SSDis detected.
7. Write 1to SR_SSD to clear the flag.

7.1.3.11 DMA Transfer

When I2C is in master transmit or master receive mode, DMA transfer can be enabled. DMA only applies to the data segment and cannot be used to transfer the slave device address and R/nW bit. I2C supports a single DMA transfer of up to 511 bytes of continuous data. If there is a need for additional data transfer, DMA can be restarted.

Once the DMA is initiated, the transfer process does not require CPU involvement, as the I2C module interacts directly with the DMACmodule to complete the transfer and generate an interrupt DMA DONE. The I2C module features a built-in 8 byte FIFO for caching data during the DMA transfer process. If a FIFO overflow occurs, an overflow interrupt OF or an underflow interrupt UF will be triggered. If an ACK=1 is received from the slave device during the master transmission process, the data transfer will be aborted, and an interrupt DMADONE will be triggered.

The number of bytes transferred by the DMA is configured by writing to DNR_NDT . The remaining bytes to be transferred can be retrieved by reading DNR_NDT . When the interrupt DMADONE occurs, reading DNR_NDT and SR_NACK will indicate whether the DMA transfer has been completed successfully.

Set Setting CR_LASTSTOP to 1 enables the automatic sending of a stop bit after the current DMA transfer is completed. Setting CR_LASTNACK to 1 enables an automatic reply of ACK=1 after the current DMA transfer is completed.

The process for using DMAfor master transmission or master reception is as follows:

1. Shift the slave device address left by 1bit, append the least signifcant bit R/nW, and write to DBR.
2. TCR=TCR_START; TCR|=TCR_TB。
3. Poll SR_TE until it reads 1, or wait for the transmission complete TEinterrupt.
4. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, send the stop bit and abort the transmission.
5. Configure the DMAC module. Set the channel peripheral selection to the current I2C , data width to single byte, peripheral address to the current I2C's FIFO register, and start the DMAC channel.
6. Configure the DMA for I2C . Set DNR_NDT to the number of bytes to be transmitted. If a stop bit needs to be automatically sent after the DMA transfer is completed, set CR_LASTSTOP to 1 . If an automatic reply of ACK=1 is required after the DMA transfer is completed, set CR_LASTNACK to 1. Set CR_DMAEN to 1 to enable DMA .
7. Waiting for the DMADONEinterrupt.
8. SR_DMADONE write 1to clear the flag.
9. If DMA needs to be restarted, repeat steps 5-8.
10. Poll SR_UBuntil it reads 0, indicating that the stop bit transmission is complete.

7.1.3.12 Bus Exception Recovery

Due to electrical interference or device anomalies, the I2C bus may occasionally become unresponsive, resulting in continuous failures in sending or receiving I2C data. When a bus hang is suspected, the following methods may be attempted for recovery.

1. Reset the I2C module. Set CR_UR to 1 , wait 100us , and then set it back to 0.
2. If both BMR_SCL and BMR_SDA are 1, set CR_RSTREQ to 1 and monitor CR_RSTREQ until it changes to 0. During this period, I2C will continuously send RCCR_RSTCYC cycles of clock signals, which may enable the device to recover from an abnormal state upon detecting such bus signals.
3. If either BMR_SCLor BMR_SDAremains 0,suspect a failure of the pull-up resistor or that the slave device is unresponsive; the hardware circuit should be inspected or the slave device should be reset.

7.1.4 I2C Registers

I2C1 base address is 0x5009C000.

I2C2 base address is 0x5009D000.

I2C3 base address is 0x5009E000.

I2C4 base address is 0x5009F000.

Table 7-1: **I2C Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control register
[31]	rw	1'h0	UR	Unit Reset. Software need first assert to reset then deassert to release. 0 = No reset. 1 = Reset I2C module.
[30]	rw	1'h0	RSTREQ	I2C will do bus reset upon this bit set. Will be cleared by HW automatically after RSTCYC cycles of SCL generated. 1 = request for i2c bus reset 0 = bus reset finished
[29]	rw	1'h0	BRGRST	Reset bus related state machine and signals. Will be cleared by HW automatically 1 = request for reset 0 = reset finished
[28:15]			RSVD	
[14:12]	rw	3'h0	DNF	Digital noise filter These bits are used to configure the digital noise filter on SDA and SCL input. The digital filter will filter spikes with a length of up to DNF*Tfclk. 0: Digital filter disabled 1: Digital filter enabled and filtering capability up to 1 Tfclk ... 7: digital filter enabled and filtering capability up to 7 Tfclk Digital filter is added to analog filter. Digital filter will introduce delay on SCL and SDA processing, which is essential in hs-mode.
[11]	rw	1'h0	SLVEN	Slave mode Enable for SCL. 0 = Disable slave mode. Will not monitor slave address on I2C bus. 1 = Enable slave mode. Will monitor slave address on I2C bus.
[10]			RSVD	
[9]	rw	1'h0	SCLPP	Push-pull mode Enable for SCL. 0 = open drain output for SCL. 1 = Push-pull output for SCL
[8]	rw	1'h0	MSDE	Master Stop Detected Enable: 0 = Master Stop Detect (MSD) status is not enabled. 1 = Master Stop Detect (MSD) status is enabled.
[7]			RSVD	
[6]	rw	1'h0	LASTSTOP	Generate STOP for last DMA transfer
[5]	rw	1'h0	LASTNACK	Generate NACK for last DMA Read transfer
[4]	rw	1'h0	DMAEN	DMA Enable for both TX and RX 0 = DMA mode is NOT enabled 1 = DMA mode enabled
[3]	rw	1'h0	SCLE	SCL Enable: 0 = Disables the I2C from driving the SCL line. 1 = Enables the I2C clock output for master-mode operation.

Continued on the next page...

Table 7-1: **I2C Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	IUE	I2C Unit Enable: 0 = Disables the unit and does not master any transactions or respond to any slave transactions. 1 = Enables the I2C (defaults to slave-receive mode). Software must guarantee the I2C bus is idle before setting this bit.
[1:0]	rw	2'h0	MODE	Bus Mode (Master operation): 2'b00: standard-mode 2'b01: fast-mode and fast-mode plus 2'b10: HS-mode (standard mode when not doing a high speed transfer) 2'b11: HS-mode (fast mode when not doing a high speed transfer) Bus Mode (Slave operation): 2'b0x: HS-mode is disabled. I2C unit uses Standard/Fast mode timing on the SDA pin. 2'b1x: HS-mode is enabled. I2C unit uses HS-mode timing on the SDA pin when a master code is received.
0x04			**TCR**	Transfer Control register
[31:8]			RSVD	
[7]	w1s	1'h0	ABORTDMA	Abort DMA operation. Will be cleared by HW automatically
[6]	w1s	1'h0	RXREQ	Request DMA RX. Will be cleared by HW automatically
[5]	w1s	1'h0	TXREQ	Request DMA TX. Will be cleared by HW automatically
[4]	rw	1'h0	MA	Master Abort: Used by the I2C in master mode to generate a Stop without transmitting another data byte: 0 = The I2C transmits Stop on if TCR[STOP] is set. 1 = The I2C sends Stop without data transmission. When in master-transmit mode, after transmitting a data byte, the TCR[TB] bit is cleared. When no more data bytes need to be sent, setting master abort bit sends the Stop. The TCR[TB] bit must remain clear. In master-receive mode, when a NAK is sent without a Stop (TCR[STOP] bit was not set) and CPU does not send a repeated Start, setting this bit sends the Stop. Once again, the TCR[TB] bit must remain clear. Master Abort can be done immediately after the address phase (Master Transmit mode only).
[3]	rw	1'h0	NACK	The positive/negative acknowledge control bit, defines the type of acknowledge pulse sent by the I2C when in master receive mode: 0 = Send a positive acknowledge (ACK) pulse after receiving a data byte. 1 = Send a negative acknowledge (NACK) pulse after receiving a data byte. The I2C automatically sends an ACK pulse when responding to its slave address or when responding in slave-receive mode, regardless of the NACK control-bit setting.
[2]	rw	1'h0	STOP	Stop: Used to initiate a Stop condition after transferring the next data byte on the I2C bus when in master mode. In master-receive mode, the NACK control bit must be set in conjunction with the STOP bit. 0 = Do not send a Stop. 1 = Send a Stop.
[1]	rw	1'h0	START	Start: Used to initiate a Start condition to the I2C unit when in master mode. 0 = Do not send a Start pulse. 1 = Send a Start pulse.

Continued on the next page...

Table 7-1: **I2C Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	TB	Transfer Byte: Used to send or receive a byte on the I2C bus: 0 = Cleared by I2C when the byte is sent/received. 1 = Send/receive a byte. CPU can monitor this bit to determine when the byte transfer has completed. In master or slave mode, after each byte transfer including acknowledge pulse, the I2C holds the SCL line low (inserting wait states) until TB is set.
0x08			**IER**	Interrupt Enable register
[31:16]			RSVD	
[15]	rw	1'h0	UFIE	FIFO Underflow Interrupt Enable 0 = FIFO Underflow interrupt is not enabled 1 = FIFO Underflow interrupt is enabled
[14]	rw	1'h0	OFIE	FIFO Overflow Interrupt Enable 0 = FIFO Overflow interrupt is not enabled 1 = FIFO Overflow interrupt is enabled
[13]	rw	1'h0	DMADONEIE	DMA Transaction Done Interrupt Enable 0 = DMA Transaction done interrupt is not enabled. 1 = DMA Transaction done interrupt is enabled.
[12]	rw	1'h0	MSDIE	Master Stop Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C unit to interrupt upon detecting a Master Stop sent by the I2C unit.
[11]			RSVD	
[10]	rw	1'h0	BEDIE	Bus Error Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt for the following I2C bus errors: As a master transmitter, no ACK was detected after a byte was sent. As a slave receiver, the I2C generated a NACK pulse. Software is responsible for guaranteeing that misplaced Start and Stop conditions do not occur.
[9]	rw	1'h0	SADIE	Slave Address Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt upon detecting a slave address match or a general call address.
[8]			RSVD	
[7]	rw	1'h0	RFIE	DBR Receive Full Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt when the DBR has received a data byte from the I2C bus.
[6]	rw	1'h0	TEIE	DBR Transmit Empty Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt after transmitting a byte onto the I2C bus.
[5]	rw	1'h0	ALDIE	Arbitration Loss Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt upon losing arbitration while in master mode.
[4]	rw	1'h0	SSDIE	Slave Stop Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt when it detects a Stop condition while in slave mode.
[3:0]			RSVD	
0x0C			**SR**	Status register
[31:16]			RSVD	

Continued on the next page...

Table 7-1: **I2C Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[15]	rw1c	1’h0	UF	FIFO Underflow Flag. Asserted when FIFO is empty and a POP request generated without a PUSH. Cleared if write 1
[14]	rw1c	1’h0	OF	FIFO Overflow Flag. Asserted when FIFO is full and a PUSH request generated without a POP. Cleared if write 1
[13]	rw1c	1’h0	DMADONE	DMA Transaction Done. Asserted when both APB and I2C bus have finished transfer. Cleared if write 1
[12]	rw1c	1’h0	MSD	Master Stop Detected: 0 = No Master Stop Detected. 1 = This bit is set by the I2C unit when all of the following are true: This bit is enabled (CR[MSDE] = 1); I2C unit is configured as a master; I2C transmits a STOP signal
[11]	r	1’h0	EBB	Early Bus Busy 0 = I2C bus is idle or the I2C is using the bus (that is, unit busy). 1 = Set when the unit detects that the SCL or SDA line is low without a START condition. Bit will remain set until the I2C unit detects the bus is idle by detecting a STOP condition. Bit will also be set whenever the IBB bit is set.
[10]	rw1c	1’h0	BED	Bus Error Detected: 0 = No error detected. 1 = The I2C sets this bit when it detects one of the following error conditions: As a master transmitter, no ACK was detected on the interface after a byte was sent. As a slave receiver, the I2C generates a NACK pulse. When an error occurs, I2C bus transactions continue. Software must guarantee that misplaced Start and Stop conditions do not occur. Cleared if write 1
[9]	rw1c	1’h0	SAD	Slave Address Detected: 0 = No slave address was detected. 1 = The I2C detected a seven-bit address that matches the general call address or SAR. An interrupt is signalled when enabled in the CR. Cleared if write 1
[8]			RSVD	
[7]	rw1c	1’h0	RF	DBR Receive Full: 0 = The DBR has not received a new data byte or the I2C is idle. 1 = The DBR register received a new data byte from the I2C bus. An interrupt is signalled when enabled in the CR. Cleared if write 1
[6]	rw1c	1’h0	TE	DBR Transmit Empty: 0 = The data byte is still being transmitted. 1 = The I2C has finished transmitting a data byte on the I2C bus. An interrupt is signalled when enabled in the CR. Cleared if write 1
[5]	rw1c	1’h0	ALD	Arbitration Loss Detected: Used during multi-master operation: 0 = Cleared when arbitration is won or never took place. 1 = Set when the I2C loses arbitration. Cleared if write 1
[4]	rw1c	1’h0	SSD	Slave Stop Detected: 0 = No Stop detected. 1 = Set when the I2C detects a Stop while in slave-receive or slave-transmit mode. Cleared if write 1

Continued on the next page...

Table 7-1: **I2C Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[3]	r	1'h0	IBB	I2C Bus Busy: 0 = I2C bus is idle or the I2C is using the bus (that is, unit busy). 1 = Set when the I2C bus is busy but local I2C is not involved in the transaction.
[2]	r	1'h0	UB	Unit Busy: 0 = I2C not busy. 1 = Set when local I2C is busy. This is defined as the time between the first Start and Stop.
[1]	r	1'h0	NACK	ACK/NACK Status: 0 = The I2C received or sent an ACK on the bus. 1 = The I2C received or sent a NACK.on the bus. This bit is used in slave-transmit mode to determine when the byte transferred is the last one. This bit is updated after each byte and ACK/NACK information is received.
[0]	r	1'h0	RWM	Read/write Mode: 0 = The I2C is in master-transmit or slave-receive mode. 1 = The I2C is in master-receive or slave-transmit mode. This is the R/nW bit of the slave address. It is cleared automatically by hardware after a Stop state.
0x10			**DBR**	Data Buffer register
[31:8]			RSVD	
[7:0]	rw	8'h0	DATA	use the I2C Data Buffer register to transmit and receive data from the I2C bus. The DBR is accessed by software on one Side and by the I2C Shift register on the other. The DBR receives data coming into the I2C unit after a full byte is received and acknowledged. CPU writes data going out of the I2C to the DBR and sends it to the serial bus. When the I2C is in transmit mode (master or slave), CPU writes data to the DBR over the internal bus. CPU write data to the DBR when a master transaction is initiated or when the DBR transmit-empty interrupt is signalled. Data moves from the DBR to the Shift register when the transfer byte bit is set. The DBR transmit-empty interrupt is signalled (if enabled) when a byte is transferred on the I2C bus and the acknowledge cycle is complete. If the DBR is not written, and a Stop condition is not in place before the I2C bus is ready to transfer the next byte packet, the I2C unit inserts wait states until CPU writes the DBR and sets the transfer byte bit. When the I2C is in receive mode (master or slave), CPU reads DBR data over the internal bus. CPU reads data from the DBR when the DBR receive-full interrupt is signalled. The data moves from the Shift register to the DBR when the acknowledge cycle is complete. The I2C inserts wait states until the DBR is read. After the software reads the DBR, CR[NACK] are written by the software, allowing the next byte transfer to proceed to the I2C bus. In DMA mode, DBR is automatically filled from FIFO in master transmit mode, or fetched and stored in FIFO in master receive mode until DMA done or aborted.
0x14			**SAR**	Slave Address Register
[31:7]			RSVD	
[6:0]	rw	7'h47	ADDR	The seven-bit address to which the I2C responds when in slave-receive mode
0x18			**LCR**	Load Count Register
[31:27]	rw	5'h1	HLVH	Decrementer Load value for High Speed Mode SCL (master mode) for high phase. Thigh=Tfclk*(HLVH+4+DNF)

Continued on the next page...

Table 7-1: I2C Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[26:18]	rw	9’h7	HLVL	Decrementer Load value for High Speed Mode SCL (master mode) for low phase. Tlow=Tfclk*(HLVL+3+DNF). Data rate is generated as 1/(Thigh+Tlow), or Ffclk/(HLVH+HLVL+7+2*DNF). 3.2Mbps data rate is generated by default if fclk is 48MHz. HLVL also controls setup time and hold time for START and STOP condition in High Speed Mode(master mode). Thdsta=Tsusta=Tsusto=Tfclk*(HLVL+1)
[17:9]	rw	9’h39	FLV	Decrementer Load value for Fast Mode (or Fast Mode Plus) SCL (master mode) for both high and low phase. Data rate is generated as Ffclk/(FLV+max(FLV,CNT*2+6)+7+DNF) approximately. 400kbps data rate is generated by default if fclk is 48MHz. FLV also controls setup time and hold time for START and STOP condition in Fast Mode(master mode). Thdsta=Tsusta=Tsusto=Tfclk*FLV
[8:0]	rw	9’hED	SLV	Decrementer Load value for Standard Mode SCL (master mode) for both high & low phase. Data rate is generated as Ffclk/(SLV+max(SLV,CNT*2+6)+7+DNF) approximately. 100kbps data rate is generated by default if fclk is 48MHz. SLV also controls setup time and hold time for START and STOP condition in Standard Mode(master mode). Thdsta=Tsusta=Tsusto=Tfclk*SLV
0x1C			**WCR**	Wait Count Register
[31:8]			RSVD	

[7:0]	rw	8’hA	CNT	Controls the counter values defining the setup and hold times in standard and fast mode
				Tvddat=Thddat=Tfclk*(CNT+2)
				Tsudat=max(Tlow-Thddat,Thddat)
				Lower counter values may violate setup and hold times.

0x20			**RCCR**	Bus Reset Cycle Counter Register
[31:4]			RSVD	
[3:0]	rw	4’h9	RSTCYC	The cycles of SCL during bus reset
0x24			**BMR**	Bus Monitor Register
[31:2]			RSVD	
[1]	r	1’h1	SCL	value of the SCL pin. Software can check bus level when the I2C bus is hung and the I2C unit must be reset.
[0]	r	1’h1	SDA	value of the SDA pin.
0x28			**DNR**	DMA number register
[31:9]			RSVD	
[8:0]	rw	9’h0	NDT	Write as number of data to transfer in byte. Read as left data number to transfer
0x30			**FIFO**	FIFO Register
[31:8]			RSVD	
[7:0]	rw	8’h0	DATA	Write to push send data into FIFO. Read to pop received data from FIFO

7.2 SPI

HPSYS has two SPI modules

7.2.1 Introduction

SPI supports three communication formats: SSP/SPI/Microwire. SSP/SPI is a full-duplex communication protocol, allowing the controller to be configured as either Master or Slave mode. Microwire is a half-duplex communication protocol, with the controller configurable only as Master mode. The SPI controller features a built-in transmit / receive FIFO. The transmit FIFO and receive FIFO share the same address; reading this address accesses the receive FIFO, while writing to this address accesses the transmit FIFO. The FIFO supports both software access mode and DMA access mode.

7.2.2 Main Features

- Supportsthreecommunication formats: SSP/SPI/Microwire
- Supports 8 to 32-bitdata width
- The clock polarity and phase in SPI format can be configured through registers SPO and SPH.
- The polarity of the chip select signal is configurable.
- The FIFO depth is 32-bits×16 entries.
- Both receive and transmit modes support DMA.
- The maximum clock frequency for SPI in HPSYS is 48 MHz.

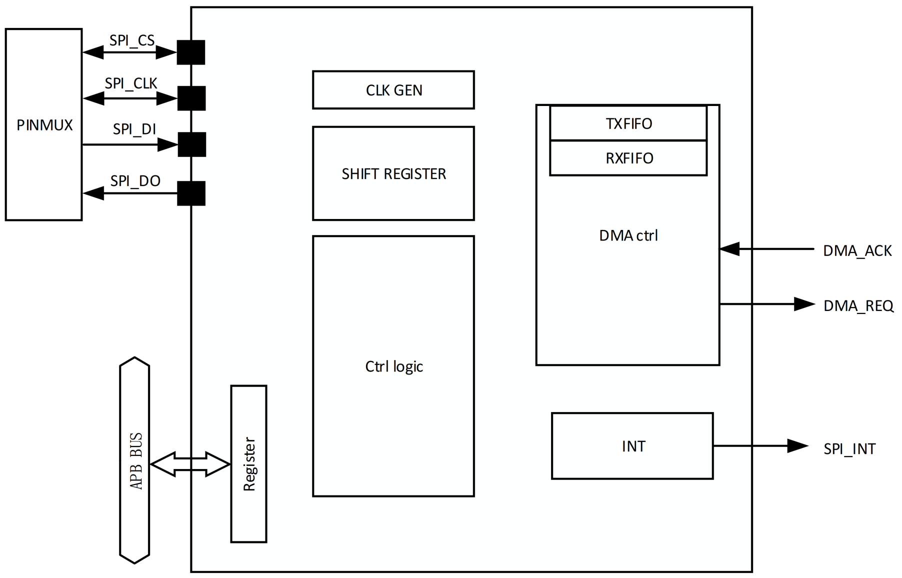


Figure 7-2: **SPI Block Diagram**

7.2.3 Interface Signals

SPI_CS is used as the chip select signal or the data frame start signal.

When configured for communication under the SPI protocol, SPI_CS serves as the chip select signal, where the transition of SPI_CS from 1 to 0 indicates the start of a data transmission, and from 0 to 1 indicates the end of a data transmission. When communicating as a Master, SPI_CS is an output signal driven internally. When communicating as a Slave, SPI_CS is an input signal driven externally.

When communicating under the SSP protocol, the SPI_CS signal indicates the start of a frame data transmission. A high pulse signifes the beginning of a frame transmission, with the pulse width corresponding to the duration required to transmit a single bit of data. Prior to the start of each frame transmission, when operating as a Master, SPI_CS functions as an output signal driven internally. Conversely, when operating as a Slave, SPI_CS functions as an input signal driven externally.

When communicating under the Microwire protocol, the SPI_CS serves as a chip select signal. A transition of SPI_CS from

1 to 0 indicates the start of a data transmission, while a transition from 0 to 1 indicates the end of a data transmission. Under the Microwire protocol, the SPI can only function as a Master, where SPI_CS acts as an output signal driven internally.

SPI_CLK is the clock signal for serial communication. It serves as an output signal when operating as a Master and as an input signal when functioning as a Slave.

SPI_DO is the data signal transmitted outward. The data in TX_FIFO is sent out via SPI_DO in MSB first order. Regardless of whether it is configured as a Master or a Slave,it remains an output signal.

SPI_DI is the data signal received from external sources. The data received from SPI_DI is stored in RX_FIFO and can be accessed by the CPU or DMA. Regardless of whether it is configured as a Master or a Slave,it remains an input signal.

7.2.4 FIFO

The SPI controller includes TXFIFO and RXFIFO, both with a bit width of 32 bits and a depth of 16. Both FIFOs can be accessed by the CPU or DMA From the perspective of address mapping, both FIFO share the same address. Writing data to this address will place the data into TXFIFO, while reading from this address will retrieve the earliest data from RX_FIFO.

A single access to FIFO can only write or read one piece of data (regardless of data width), and the data width for accessing FIFO must be 32 bits. If the configured width is not 32 bits, during transmission, the SPI controller will disregard the portion of the 32 bits in TXFIFO that exceeds the configured width, and during reception, the SPI controller will write the portion exceeding the configured width into RXFIFO after padding it with 0.

FIFO can interact with the CPU through interrupts or by having the CPU poll the FIFO's status register. Based on the interaction results, the CPU writes data to the TXFIFO or reads data from the RXFIFO.

FIFO interacts with DMA controller through DMA interface, notifying DMA to write data to TXFIFO or read data from RXFIFO.

When FIFO interacts with the CPU through interrupts, the conditions for generating an interrupt are as follows:

- When the number of data items in RXFIFO exceeds the value of RFT in the FIFO_CTRL register, the SPI controller will generate an interrupt to notify the CPU to read data from RXFIFO.
- When the number of data items in TXFIFO is less than the value of TFT in the FIFO_CTRL register plus 1, the SPI controller will generate an interrupt to notify the CPU to write data to TXFIFO.

When FIFO interacts with DMA controller through DMA interface,, the conditions for generating DMA_REQare as follows:

- When the number of data items in the RXFIFO exceeds the value of RFT in the FIFO_CTRL register, the SPI controller will generate a DMA_REQ to notify the DMA to read the data from the RXFIFO.
- When the number of data items in the TXFIFO is less than the value of TFT in the FIFO_CTRL register plus one, the SPI controller will generate a DMA_REQ to notify the DMA to write data to the TXFIFO

In both cases, it is essential to set the parameters appropriately to avoid RXFIFO overflow or TXFIFO underrun.

7.2.5 Data Format

- SPI Format SPI is a full-duplex synchronous serial communication protocol, divided into four sub-modes based on the settings of SPO and SPH in the TOP_CTRL register. SPO sets the polarity of SPI_CLK when the SPI controller is either not enabled or is in IDLE state. When SPO is 0, SPI_CLK has a low voltage polarity, with the first edge being a rising edge; when SPO is 1, SPI_CLK has a high voltage polarity, with the first edge being a falling edge.SPH determines the clock edge for driving data and the timing for sampling data. When SPH is 0, the SPI controller begins sampling data on SPI_DI from the first edge of SPI_CLK and drives SPI_DO (from SPI_DO the default is the MSB of the data to be sent) starting from the second edge of SPI_CLK; whenSPH is 1, the SPI controller drives SPI_DO from the first edge of SPI_CLK and samples data on SPI_DI from the second edge of SPI_CLK. For specific communication timing, please refer to Figure7-3 and Figure7-4。

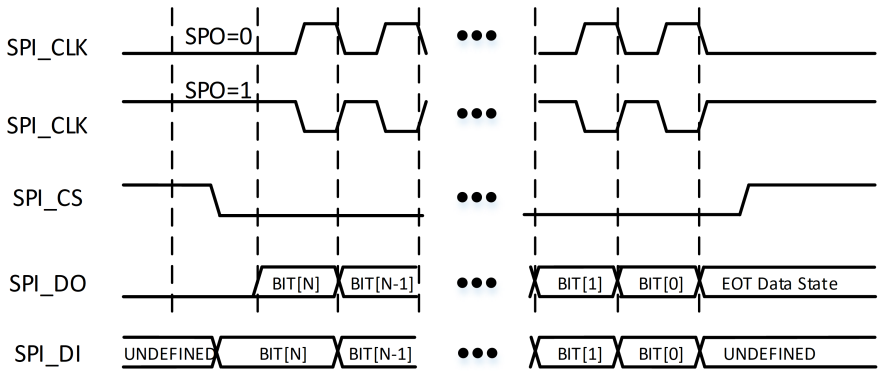


Figure 7-3: SPI communication when SPH is 0

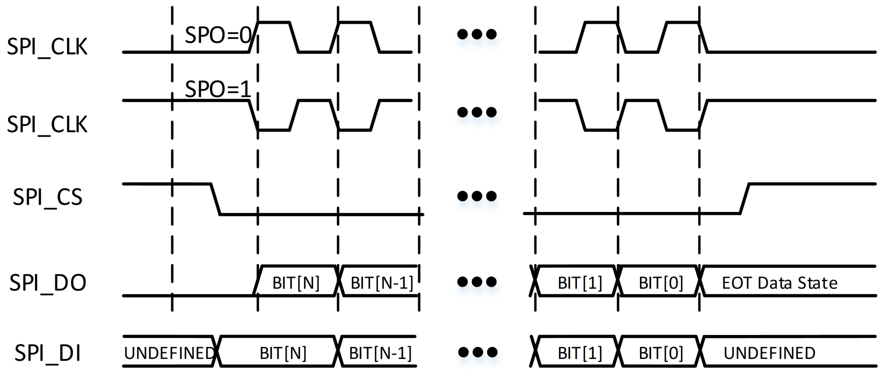


Figure 7-4: SPI communication when SPH is 1

The following section describes the SPI protocol communication process under the conditions of SPH=0 and SPO=0, as illustrated in Figure7-3.When the SPI controller is not enabled or is in an IDLE state after being enabled, SPI_DO is at a low level, SPI_CS is at a high level, and SPI_CLK is at a low level. The SPI_CS signal is pulled low to initiate a data transmission, and it remains low until the data transmission of that frame is complete. Half a SPI_CLK cycle later, the MSB of the transmission data is driven onto SPI_DO, and after another half SPI_CLK cycle, SPI_CLK goes high to start the first rising edge and samples SPI_DI. Before all data in this transmission is completed, SPI_CLK continues to toggle at the set frequency. After the last sample is completed, SPI_CLK returns to a low level, and after another half SPI_CLK cycle, SPI_CS is pulled high to conclude this transmission.
When operating as a MASTER, if there are multiple data entries waiting to be transmitted in the TXFIFO, theSPIcontroller will transmit the data continuously. During this process, the SPI_CSsignal remains low. The transmission timing is illustrated in Figure7-5。

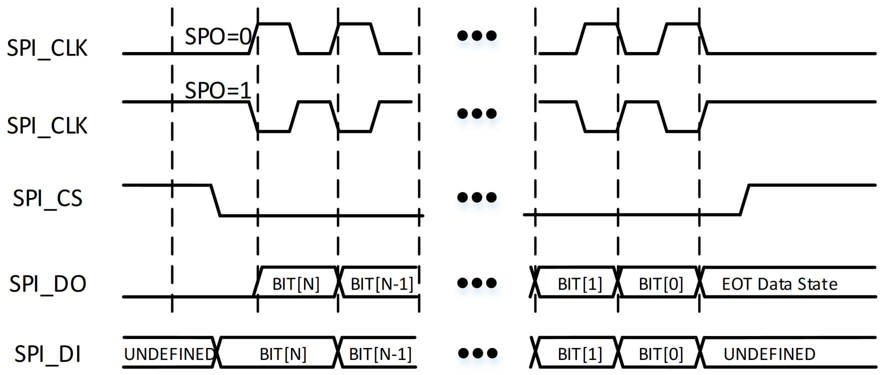


Figure 7-5: SPI Protocol Continuous Transmission Timing

In practical applications, there are scenarios where the SPI_CSneeds to be pulled low once to transmit multiple data entries. In such cases, if the TXFIFO data is not filled promptly, it may cause the SPI_CS to go high during transmission, resulting in communication failure. In these scenarios, special software control is necessary to ensure a stable low state for SPI_CS ; details of the settings can be found in the following chapters.

- TI-SSP Format
  TI-SSP is a full-duplex synchronous serial communication protocol. During data transmission, the SPI_CS issues a high pulse with a width of one clock cycle to indicate the start of transmission. Subsequently, data is driven to SPI_DO at a rate of one bit per cycle, in MSB first order. Data is driven onto the data line on the rising edge of SPI_CLK, and sampled by the SPI controller on the falling edge of SPI_CLK. The timing for a single communication is illustrated in Figure7-6。

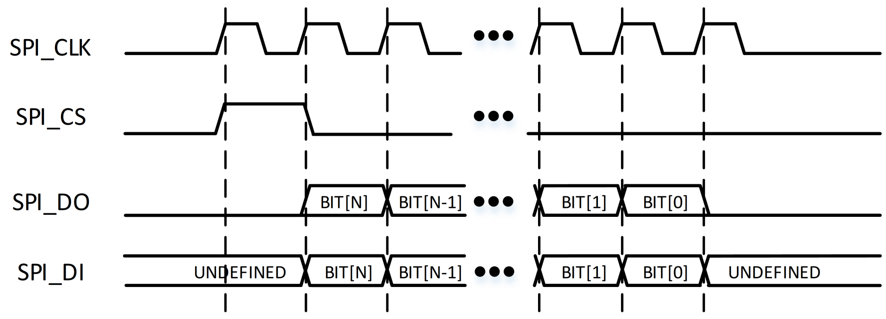


Figure 7-6: **TI-SSP Protocol Communication Timing**

When operating as a MASTER , if there are multiple data entries awaiting transmission in the TXFIFO , theSPI controller will transmit the data continuously. The transmission timing is illustrated in Figure7-7.

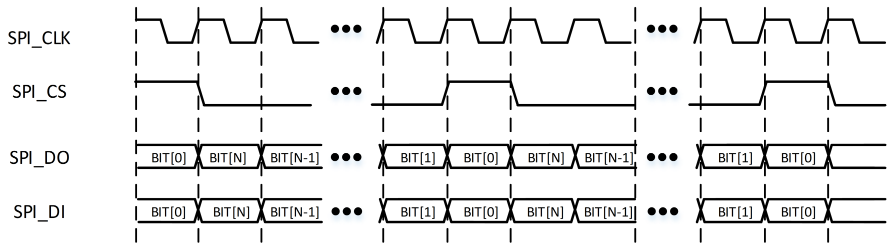


Figure 7-7: **TI-SSP Protocol Continuous Communication Timing**

When multiple data packets are transmitted continuously, and data B immediately follows data A, the SPI_CS is pulled high for one cycle during the transmission of the last bit of data A, and the transmission of the MSB of data B begins on the next rising edge of SPI_CLK.

- Microwire Protocol

  The Microwire Protocol is a half-duplex protocol. The SPI controller only supports communication as a Master. In this communication, the Master first issues an 8 or 16-bit command word on SPI_DO, and after one SPI_CLK cycle, the Slave returns data on SPI_DI. The timing for a single transmission is illustrated in Figure 7-8

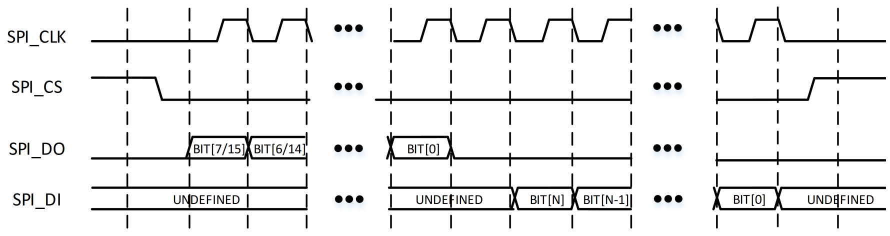


Figure 7-8: **Microwire Protocol Single Communication Timing**

When multiple transmissions occur consecutively, the command word for the next transmission is immediately output to SPI_DO after the last bit returned by the Slave , and SPI_CS remains low until all transmissions are complete. The timing for consecutive transmissions is illustrated in Figure 7-9.

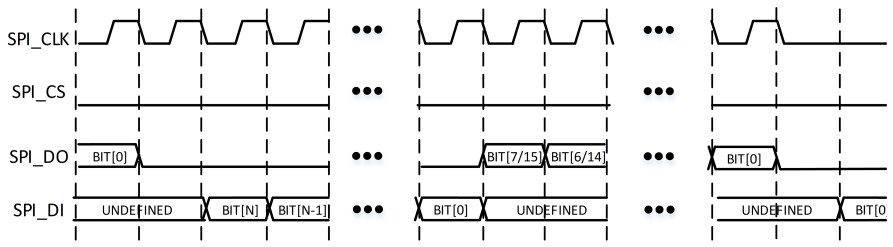


Figure 7-9: **Microwire Protocol Continuous Transmission Timing**

7.2.5.1 Related System Resources

SPI Controller communication requires the correct configuration of PINMUX . The interfaces that need to be configured in PINMUX are SPI_CS/SPI_CLK/SPI_DI/SPI_DO . It is important to note that the function of the PIN used as SPI_DO must be selected as SPI_DIO . In three-wire half-duplex communication, the SPI_DI_SEL in the register CLK_CTRL must be set to 1 .

7.2.5.2 Communication Process

To perform SPIcontroller communication, the following operations are required.

1. Set the PINMUXto configure the corresponding PINfor SPIcommunication functionality.
2. Communication Protocol Settings:
    - Configure the communication protocol using the FRFin the TOP_CTRLRegister.
    - FRF=0： SPI Protocol
    - FRF=1： TI-SSP Protocol
    - FRF=2： Microwire Protocol
3. Master/Slave Mode Settings:
    - SPI_CS/SPI_CLK can be configured for Master/Slave mode. In Master mode, the SPI controller drives the communication, while in Slave mode, the SPI controller receives signals driven externally.
    - To configure the Master/Slave mode of SPI_CS, use the SFRMDIR in the TOP_CTRL register:
    - SFRMDIR=0 : Master mode; SFRMDIR=1 : Slave mode.
    - To configure the Master/Slave mode of SPI_CLK, use the SCLKDIR in the TOP_CTRL register:
    - SCLKDIR=0 : Master mode; SCLKDIR=1 : Slave mode.
    - Typically, SFRMDIR and SCLKDIR are configured to the same mode.
4. Clock Frequency Settings:
    - The procedure for setting the SPI clock is as follows:
        (a) Set the CLK_DIV in the CLK_CTRL register to configure the clock division ratio.
        (b) Set the CLK_SEL in the CLK_CTRL register to select the clock source: CLK_SEL=0 : select divided clock; CLK_SEL=1 : Select the source clock.
        (c) Set the register CLK_CTRL to enable CLK_SSP_EN for the SPI clock, where 1 indicates that the SPI clock is enabled.
    - The SPI clock is derived from either the source clock or the divided source clock. The frequency of the divided clock is the source clock frequency divided by CLK_DIV, with the source clock frequency being 48 MHz in HPSYS
5. Data Width Setting:
    - Supports data widths of 8 to 32 bits. The data bit width can be configured by setting the DSS in the register TOP_CTRL, where the data bit width =DSS + 1.

6. Data Operations:
   - Data is categorized into transmit data and receive data.
     - Transmit data: Write data to be sent into the TXFIFO
     - Receive data: Read already received data from the RXFIFO
   - Data operations can be performed either by CPU software directly accessing FIFOs or via DMA.
   - When using CPU for data operations, the SPI controller typically notifes the CPU via interrupts to write toTXFIFO or read from RXFIFO. The interrupt corresponding toTXFIFO is enabled by setting the TIE in the register to 1. Once enabled, when the number of data inTXFIFO is less than or equal to the value of TFT in the register, the SPI controller issues aTX interrupt notifcation to the CPU. The interrupt corresponding to RXFIFO is enabled by setting the RIE in the register to 1. Once enabled, when the number of data in RXFIFO exceeds the value of RFT in the register, the SPI controller issues a RX interrupt notifcation to the CPU.
   - The CPU can also operate the FIFO by polling its status. FIFO STATUS Register

STATUS Register	Meaning
TNF	0： TXFIFO is full; 1: TXFIFOis not full.
TFL	The number of data items in the TXFIFO. When the value read is 0, TXFIFO is either full or empty, and this should be assessed in conjunction with the value of TNF.
TUR	1: An UNDERRUN occurs in the TXFIFO, indicating that a read operation from the TXFIFO was performed by the SPI controller while the TXFIFO was empty
RNE	0： RXFIFO is empty; 1: RXFIFOis not empty.
RFL	The number of data items in the RXFIFO. When the value read is 0xF, RXFIFO is either empty or full, and this should be assessed in conjunction with the value of RNE.
ROR	1: RXFIFO has encountered an OVERRUN , indicating that RXFIFO is in a full state while the SPI controller has attempted to perform a write operation to RXFIFO.

   - The CPU can read the STATUS Register, and provided that TXFIFO is not full, it can write data to TXFIFO data to TXFIFO; when RXFIFO is not empty, it can read the received data from RXFIFO.
   - When using DMA for data operations, the SPI controller initiates the operation by sending a request to DMA to start DMA. Write RESE to 1 to enable DMA for RXFIFO; when the data in RX FIFO exceeds RFT , the SPI will issue a DMA request. Write TESE to 1 to enable DMA for TXFIFO; when the number of data in TXFIFO is less than or equal to TFT value in the register, the SPI controller will issue a DMA request.
7. Enable SPI Controller
   - Set SSE in Register TOP_CTRL to 1 to enable the SPI Controller.
8. Disable SPI Controller
   - Check the BUSY bit in the STATUS Register. If BUSY is 0, it indicates that the SPI Controller is currently idle. Then write 0 to SSE to disable the SPI Controller.
9. SPI_CS Signal Control
   - In certain communication scenarios, it is necessary for SPI_CS to remain low during the transmission of multiple data packets. Under default settings, if TXFIFO becomes empty during the process, it may cause SPI_CS to temporarily go high, and it will return to low once TXFIFO is not empty. Software intervention can ensure that the SPI_CS signal remains stable at a low level throughout the communication. Before initiating a communication, set HOLD_FRAME_LOW in Register TOP_CTRL to 1 to ensure that SPI_CS stays low during the communication. Please remember to set HOLD_FRAME_LOW to 0 after this communication is completed to prepare for subsequent communications.

- Configure the SPIcontroller according to the following procedure:
  1. Set up pinmux
  2. Configure the communication protocol
  3. Set the clock frequency
  4. Define the data bit width
  5. Establish the master-slave mode
  6. Enable the SPIcontroller
  7. Access the FIFOto initiate transmission and reception

8. Complete the transmission and reception, then disable the SPIcontroller

7.2.5.3 Receive-Only Mode

The SPI controller supports the Receive-Only mode. In this mode, when the data format is SPI or TI-SSP, the SPI controller will toggle the SPI_CLK regardless of whether there are pending data in TXFIFO. Simultaneously, the data received from SPI_DI will be stored in RXFIFO. The configuration for using the Receive-Only mode is as follows:

- Configure the RWOT_CCM Register, which specifes the required SPI_CLKcycle count.
- Set bothSET_RWOT_CYCLE and RWOT_CYCLEin the RWOT_CTRL Register to 1to enable the RWOT counter.
- Set RWOT in the RWOT_CTRL Register to 1 to enable Receive-Only mode.

After completing the above settings, enabling the SPI controller will cause the SPI_CLK to toggle regardless of whether the TXFIFO is empty; the received data will be stored in RXFIFO.

7.2.5.4 Three-Wire Mode

The software-assisted SPIcontroller supports three-wire mode, which allows for half-duplex communication. The required configuration and usage process are as follows:

1. To configure PINMUX, communication must occur through a PINwith SPI_DIOfunctionality, and the function of this PINshould be designated as SPI_DIO.
2. Select the SPIprotocol, configure the clock frequency, set the data bit width, and establish the master-slave mode.
3. Set Set SPI_TRI_WIRE_EN in the SPI controller register TRIWIRE_CTRL to 1 . This enables the SPI controller.
4. When transmitting data, set TXD_OEN in the TRIWIRE_CTRL register to 0 . Write the data to be sent into TXFIFO . In three-wire mode, the SPI controller does not write to RXFIFO when transmitting data, so after sending the data, the software does not need to process RXFIFO .
5. When receiving data, set TXD_OEN in the TRIWIRE_CTRL register to 1 . If SPI_CLK is in master mode, the SPI controller must drive SPI_CLK . There are two methods to drive SPI_CLK :
   - Write the same amount of data as the data to be received into TXFIFO to drive SPI_CLK.
   - UsingReceive-Onlymode, the process is as follows:
     (a) Disable the SPI controller's enable.
     (b) Set the RWOT_CCMRegister, where the value of this register indicates the required number of SPI_CLKcycles.
     (c) Set both SET_RWOT_CYCLE and RWOT_CYCLE in the RWOT_CTRL Register to 1 to enable the RWOT counter.
     (d) Set RWOTin the RWOT_CTRLRegister to 1to enableReceive-Onlymode.
     (e) After completing the above settings, enable the SPI controller.
     (f) Read data from RXFIFO.
6. After completing the transmission and reception, please turn off the SPIcontroller.

Generally, this half-duplex communication mode requires the SPI_CS to maintain a stable low level during the process. To maintain a stable low level, please refer to the SPI_CS signal control settings.

7.2.6 SPI Register

SPI1 base address is 0x50095000。

SPI2 base address is 0x50096000。

Table 7-2: SPI Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**TOP_CTRL**	Top Control Register
[31:19]			RSVD	
[18]	rw	1'b0	TTELP	SPI_DO Three-state Enable On Last Phase (can be set only when TI-SSP) 0: SPI_DO is three-stated 1/2 clock cycle after the beginning of the LSB 1: SPI_DO output signal is three-stated on the clock edge that ends the LSB
[17]	rw	1'b0	TTE	SPI_DO Three-State Enable 0: SPI_DO output signal is not three-stated 1: SPI_DO is three-stated when not transmitting data
[16]			RSVD	
[15]	rw	1'b0	IFS	Invert Frame Signal 0: SPI_CS polarity is as defined in protocol 1: SPI_CS will be inverted from normal-SPI_CS
[14]	rw	1'b0	HOLD_FRAME_LOW	Hold Frame Low Control 0:After this field is set to 1 and the SPI controller is operating in master mode,the output frame signal SPI_CS will be determined by control FSM. 1:After this field is set to 1 and the SPI controller is operating in master mode, the output frame signal SPI_CS will hold low.
[13]	rw	1'b0	TRAIL	Trailing Byte 0: Trailing bytes are handled by CPU 1: Trailing bytes are handled by DMA bursts
[12]			RSVD	
[11]	rw	1'b0	SPH	Motorola SPI SPI_CLK phase setting 0: SPI_CLK is inactive until one cycle after the start of a frame and active until 1/2 cycle before the end of a frame 1: SPI_CLK is inactive until 1/2 cycle after the start of a frame and active until one cycle before the end of a frame
[10]	rw	1'b0	SPO	Motorola SPI SPI_CLK Polarity Setting 0: The inactive or idle state of SPI_CLK is low 1: The inactive or idle state of SPI_CLK is high
[9:5]	rw	5'h0	DSS	SPI controller Work data size, register bits value 7~31 indicated data size 8~32 bits, usually use data size 8bits, 16bits, 24bits, 32bits
[4]	rw	1'b0	SFRMDIR	SPI_CS Direction 0: Master mode, SPI controller drives SPI_CS 1: Slave mode, SPI controller receives SPI_CS
[3]	rw	1'b0	SCLKDIR	SPI_CLK Direction 0: Master mode, SPI controller drives SPI_CLK 1: Slave mode, SPI controller receives SPI_CLK
[2:1]	rw	2'h0	FRF	Frame Format 0x0: Motorola* Serial Peripheral Interface (SPI) 0x1: Texas Instruments* Synchronous Serial Protocol (SSP) 0x2: National Semiconductor Microwire* 0x3: RSVD
[0]	rw	1'b0	SSE	SPI controller Enable 0: SPI controller is disabled 1: SPI controller is enabled
0x04			**FIFO_CTRL**	FIFO Control Register
[31:18]			RSVD	

Continued on the next page...

Table 7-2: SPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[17]	rw	1'h0	RXFIFO_AUTO_FULL_CTRL	Rx FIFO Auto Full Control: After this field is set to 1 and the SPI controller is operating in master mode, the controller FSM returns to IDLE state and stops the SPI_CLK. When Rx FIFO is full, the controller FSM continues transferring data after the RxFIFO is not full. This field is used to avoid an RxFIFO overrun issue. 1: Enable Rx FIFO auto full control
[16]	rw	1'h0	FPCKE	FIFO Packing Enable 0: FIFO packing mode disabled 1: FIFO packing mode enabled
[15:14]	rw	2'h0	TXFIFO_WR_ENDIAN	apb_pwdata Write to TxFIFO Endian 0x0: txfifo_wdata[31:0] = apb_pwdata[31:0] 0x1: fifo_wdata[31:0] = apb_pwdata[15:0], apb_pwdata[31:16] 0x2: txfifo_wdata[31:0] = apb_pwdata[7:0], apb_pwdata[15:8], apb_pwdata[23:16], apb_pwdata[31:24] 0x3: txfifo_wdata[31:0] = apb_pwdata[23:16], apb_pwdata[31:24], apb_pwdata[7:0], apb_pwdata[15:8]
[13:12]	rw	2'h0	RXFIFO_RD_ENDIAN	apb_prdata Read from Rx FIFO Endian 0x0 = apb_prdata[31:0] = rxfifo_wdata[31:0] 0x1 = apb_prdata[31:0] = rxfifo_wdata[15:0], rxfifo_wdata[31:16] 0x2 = apb_prdata[31:0]= rxfifo_wdata[7:0], rxfifo_wdata[15:8], rxfifo_wdata[23:16], rxfifo_wdata[31:24] 0x3 = apb_prdata[31:0]= rxfifo_wdata[23:16], rxfifo_wdata[31:24], rxfifo_wdata[7:0], rxfifo_wdata[15:8]
[11]	rw	1'h0	RSRE	Receive Service Request Enable 0: RxFIFO DMA service request is disabled 1: RxFIFO DMA service request is enabled
[10]	rw	1'h0	TSRE	Transmit Service Request Enable 0: TxFIFO DMA service request is disabled 1: TxFIFO DMA service request is enabled
[9:5]	rw	5'h0	RFT	RXFIFO Trigger Threshold This field sets the threshold level at which RXFIFO asserts interrupt. The level should be set to the preferred threshold value minus 1.
[4:0]	rw	5'h0	TFT	TXFIFO Trigger Threshold This field sets the threshold level at which TXFIFO asserts interrupt. The level should be set to the preferred threshold value minus 1.
0x08			**INTE**	Interrupt Enable Register
[31:6]			RSVD	
[5]	rw	1'h0	TIM	Transmit FIFO Underrun Interrupt Mask 0 : TUR events generate an SPI interrupt 1 : TUR events do NOT generate an SPI interrupt
[4]	rw	1'h0	RIM	Receive FIFO Overrun Interrupt Mask 0: ROR events generate an SPI interrupt 1: ROR events do NOT generate an SPI interrupt
[3]	rw	1'h0	TIE	Transmit FIFO Interrupt Enable 0: TxFIFO threshold-level-reached interrupt is disabled 1: TxFIFO threshold-level-reached interrupt is enabled

Continued on the next page...

Table 7-2: SPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1’h0	RIE	Receive FIFO Interrupt Enable 0: RxFIFO threshold-level-reached interrupt is disabled 1: RxFIFO threshold-level-reached interrupt is enabled
[1]	rw	1’h0	TINTE	Receiver Time-out Interrupt Enable 0: Receiver time-out interrupt is disabled 1: Receiver time-out interrupt is enabled
[0]			RSVD	
0x0C			**TO**	SPI Time Out Register
[31:24]			RSVD	
[23:0]	r	24’h0	TIMEOUT	Timeout Value TIMEOUT value is the value (0 to 2^{24}-1) that defines the time-out interval. The time-out interval is given by the equation shown in the TIMEOUT Interval Equation.
0x10			**DATA**	SPI DATA Register
[31:0]	rw	32’h0	DATA	DATA This field is used for data to be written to the TXFIFO read from the RXFIFO.
0x14			**STATUS**	Status Register
[31:24]			RSVD	
[23]	r	1’h0	OSS	Odd Sample Status 0: RxFIFO entry has two samples 1: RxFIFO entry has one sample Note that this bit needs to be looked at only when FIFO Packing is enabled (FPCKE field in FIFO Control Register is set). Otherwise, this bit is zero. When SPI controller is in Packed mode and the CPU is used instead of DMA to read the RxFIFO, the CPU should make sure that [Receive FIFO Not Empty] = 1 AND this field = 0 before it attempts to read the RxFIFO.
[22]	r	1’h0	TX_OSS	TX FIFO Odd Sample Status When SPI controller is in packed mode, the number of samples in the TX FIFO is: ([Transmit FIFO Level]*2 + this field), when [Transmit FIFO Not Full] = 1 32, when [Transmit FIFO Not Full] = 0. The TX FIFO cannot accept new data when [Transmit FIFO Not Full] = 1 and [Transmit FIFO Level] = 15 and this field = 1. (The TX FIFO has 31 samples). 0: TxFIFO entry has an even number of samples 1: TxFIFO entry has an odd number of samples Note that this bit needs to be read only when FIFO Packing is enabled ([FIFO Packing Enable] in the FIFO Control Register is set). Otherwise, this bit is zero.
[21]			RSVD	
[20]	w1c	1’h0	ROR	Receive FIFO Overrun 0: RXFIFO has not experienced an overrun 1: Attempted data write to full RXFIFO, causes an interrupt request
[19]			RSVD	
[18:15]	r	4’h0	RFL	Receive FIFO Level This field is the number of entries minus one in RXFIFO. When the value 0xF is read, the RXFIFO is either empty or full, and software should read the [Receive FIFO Not Empty] field.
[14]	r	1’h0	RNE	Receive FIFO Not Empty 0: RXFIFO is empty 1: RXFIFO is not empty

Continued on the next page...

Table 7-2: SPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13]	r	1'h0	RFS	Receive FIFO Service Request 0: RXFIFO level is at or below RFT threshold (RFT) or SPI controller is disabled 1: RXFIFO level exceeds RFT threshold (RFT), causes an interrupt request
[12]	w1c	1'h0	TUR	Transmit FIFO Underrun 0: The TXFIFO has not experienced an underrun 1: A read from the TXFIFO was attempted when the TXFIFO was empty, causes an interrupt if it is enabled ([Transmit FIFO Underrun Interrupt Mask] in the INT EN Register is 0)
[11]			RSVD	
[10:7]	r	4'h0	TFL	Transmit FIFO Level This field is the number of entries in TXFIFO.When the value 0x0 is read, the TXFIFO is either empty or full, and software should read the [Transmit FIFO Not Full] field.
[6]	r	1'h0	TNF	Transmit FIFO Not Full 0: TXFIFO is full 1: TXFIFO is not full
[5]	r	1'h0	TFS	Transmit FIFO Service Request 0: TX FIFO level exceeds the TFT threshold (TFT + 1) or SPI controller is disabled 1: TXFIFO level is at or below TFT threshold (TFT + 1), causes an interrupt request
[4]			RSVD	
[3]	w1c	1'h0	TINT	Receiver Time-out Interrupt 0: No receiver time-out is pending 1: Receiver time-out pending, causes an interrupt request
[2]			RSVD	
[1]	r	1'h0	CSS	Clock Synchronization Status 0: SPI controller is ready for slave clock operations 1: SPI controller is currently busy synchronizing slave mode signals
[0]	r	1'h0	BSY	SPI controller Busy 0: SPI controller is idle or disabled 1: SPI controller is currently transmitting or receiving framed data
0x24			**RWOT_CTRL**	RWOT Control Register
[31:5]			RSVD	
[4]	rw	1'h0	MASK_RWOT_LAST_SAMPLE	Mask last_sample_flag in RWOT Mode 1: Mask 0: Unmask
[3]	rw	1'h0	CLR_RWOT_CYCLE	Clear Internal rwot_counter This field clears the rwot_counter to 0. This field is self cleared after SSE = 1. 1: Clear rwot_counter
[2]	rw	1'h0	SET_RWOT_CYCLE	Set RWOT Cycle This field is used to set the value of the RWOT_CCM register to the internal rwot_counter. This field is self-cleared after SSE = 1. 1: Set rwot_counter
[1]	rw	1'h0	CYCLE_RWOT_EN	Enable RWOT Cycle Counter Mode 1: Enable

Continued on the next page...

Table 7-2: SPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	RWOT	Receive Without Transmit 0: Transmit/receive mode 1: Receive without transmit mode
0x28			**RWOT_CCM**	RWOT Counter Cycles Match Register
[31:0]	rw	32'h0	RWOTCCM	It's just total SPI_CLK Cycles. The value of this register defines the total number of SPI_CLK cycles when SPI controller works in master and RWOT mode. When the rwot_counter matches this value, SPI controller returns to IDLE state and does not output SPI_CLK anymore.
0x2C			**RWOT_CVWRn**	RWOT Counter Value Write for Red Request Register
[31:0]	rw	32'h0	RWOTCVWR	RWOTCVWR This register prevents the risk of instability on rwot_counter value reading, it's only valid after SPI controller has been enabled Write 0 = No effect Write 1 = Capture value of rwot_counter Read: Returns the captured value of rwot_counter
0x3C			**CLK_CTRL**	CLK Control Register
[31:10]			RSVD	
[9]	rw	1'h0	SPI_DI_SEL	Select spi_di source. 0: from port SPI_DI. 1: from port SPI_DIO.
[8]	rw	1'h0	CLK_EN	enable clk for internal logic
[7]	rw	1'h0	CLK_SEL	0: select clk_div as clk for SPI controller 1: select clk_sys as clk for SPI controller
[6:0]	rw	7'h0	CLK_DIV	div ratio from clk_sys
0x54			**TRIWIRE_CTRL**	Three Wire Mode Control Register
[31:3]			RSVD	
[2]	rw	1'h0	WORK_WIDTH_DYN_CHANGE	WORK_WIDTH_DYN_CHNAGE 1: SW can dynamicly change TOP_CTRL[9:5] without disabling TOP_CTRL[0] and re-enabling TOP_CTRL[0]
[1]	rw	1'h0	TXD_OEN	TXD_OEN control when TRI-WIRE mode 1: SPI_DIO is input 0: SPI_DIO is output
[0]	rw	1'h0	SPI_TRI_WIRE_EN	SPI_THREE_WIRE_MODE_EN 0: normal mode 1: enable TRI-WIRE mode

7.3 PTC

7.3.1 Introduction

The PTC (Peripheral Task Controller) is an independent peripheral controller that can automatically complete the coordination and control tasks of various peripherals without needing to wake up the CPU. Based on event triggers from selected peripherals, the PTC can automatically rewrite the operating modes or states of these peripherals and can link these tasks into an automatically triggered task sequence, thereby completing complex and rapid response task chains. During the execution of the task chain, the CPU can remain in sleep mode, effectively conserving power.

The PTC features 8 channels, each capable of selecting an independent trigger source and configuring independent tasks. The executable tasks consist of two types: writing specifed data directly to a designated address; and reading the content of a specifed address to perform XOR / AND / OR / addition operations with specifed data before writing it back. Upon completion of a task on each channel, a trigger signal can be generated to activate tasks on other channels. Each channel can be configured for the number of triggers. Some channels support executing tasks after a configurable delay following the trigger.

7.3.2 Main Features

- 8 independently configured channels can operate simultaneously.
- Each channel trigger can be selected from 128 trigger sources, including the PTC's own trigger sources.
- Access to the AHB and APB peripheral address space is available, supporting only word-aligned access.
- Supports direct data writing or reading followed by rewriting.
- Supports 32-bit XOR, AND, OR, and addition operations.
- Configurable trigger count ranging from 1 to 1023, or infnite triggering.
- Configurable trigger delay of 0 to 65535 HCLK cycles.
- Fixed priority arbitration; the smaller the channel number, the higher the priority.
- 4-word register space for data caching.

7.3.3 Function Description

7.3.3.1 Channel Trigger

Each channel can select 1 trigger from 128 trigger sources, with the selection register being TCRx_TRIGSEL. Trigger sources are typically generated by various peripherals to indicate the occurrence of specific events, such as DMA transfer completion, IO toggling, timer updates, etc., and also include the PTC's own channel completion events. The polarity of the trigger source can be selected via TCRx_TRIGPOL. When selecting IO input signals as trigger sources, only 4 can be selected simultaneously from every 32 IOs, with specific selections made through the PTC's GPIO31_0, GPIO63_32, and other registers. IO trigger sources do not support debouncing.

The channel can also be triggered directly by configuring the TCRx_SWTRIG register through the CPU. When TCRx_TRIGSEL is set to 0, the channel can only be triggered via the TCRx_SWTRIG register.

Table 7-3: PTC1 Trigger Source

TRIGSEL	PTC1 trigger source								TRIGSEL
127	PTC1_CH8	PTC1_CH7	PTC1_CH6	PTC1_CH5	PTC1_CH4	PTC1_CH3	PTC1_CH2	PTC1_CH1	120
119	USB_RX	USB_TX	I2S1_OF	I2S1_UF	/	TRNG_RANDGEN	TRNG_SEEDGEN	HCPU_SLEEPDEEP	112
111	EPIC_LINEHIT	EPIC_DONE	/	LCDC1_ERR	LCDC1_LINE	LCDC1_LINEHIT	LCDC1_FMARK	LCDC1_DONE	104
103	LCDC1_BUSY	/	EZIP1_ROW	EZIP1_DONE	USART3_TXBYTE	USART3_RXBYTE	EXTDMA_HT	EXTDMA_TC	96
95	/	/	/	/	USART2_TXBYTE	USART2_RXBYTE	USART1_TXBYTE	USART1_RXBYTE	88
87	/	/	SDMMC1_DATIDLE	SDMMC1_CMDBUSY	SPI2_START	SPI2_DONE	SPI1_START	SPI1_DONE	80
79	I2C3_RF	I2C3_TE	I2C3_DMADONE	TSEN_DONE	I2C2_RF	I2C2_TE	I2C2_DMADONE	AES_DONE	72
71	I2C1_RF	I2C1_TE	I2C1_DMADONE	/	/	/	/	/	64
63	PA63_32_D	PA63_32_C	PA63_32_B	PA63_32_A	PA31_0_D	PA31_0_C	PA31_0_B	PA31_0_A	56
55	MAILBOX1_C4INT7	MAILBOX1_C3INT7	MAILBOX1_C2INT7	MAILBOX1_C1INT7	/	/	/	/	48
47	/	/	/	/	/	/	/	/	40
39	DMAC1_HT8	DMAC1_HT7	DMAC1_HT6	DMAC1_HT5	DMAC1_HT4	DMAC1_HT3	DMAC1_HT2	DMAC1_HT1	32
31	DMAC1_TC8	DMAC1_TC7	DMAC1_TC6	DMAC1_TC5	DMAC1_TC4	DMAC1_TC3	DMAC1_TC2	DMAC1_TC1	24
23	/	ATIM1_COM	ATIM1_CH4	ATIM1_CH3	ATIM1_CH2	ATIM1_CH1	ATIM1_TRIG	ATIM1_UPDATE	16
15	BTIM2_UPDATE	BTIM1_UPDATE	/	/	/	GPTIM2_CH1	GPTIM2_TRIG	GPTIM2_UPDATE	8
7	GPTIM1_CH4	GPTIM1_CH3	GPTIM1_CH2	GPTIM1_CH1	GPTIM1_TRIG	GPTIM1_UPDATE	HCPU_SLEEPING	0	0

7.3.3.2 Channel Tasks

The tasks that a channel can execute fall into two categories, configured through TCRx_OP. When TCRx_OP is set to 0, after the channel is triggered, the data content in the TDRx register will be directly written to the address pointed to by the TARx register. When TCRx_OP is set to 0x4~0x7, after the channel is triggered, the data at the address pointed to by the TARx register will first be read, and then an XOR/AND/OR/addition operation will be performed with the data in the TDRx register before writing it back to the address pointed to by the TARx register. The TARx register typically points to the address of a peripheral register; therefore, the usual purpose of the channel task is to automatically configure the peripheral when a specific event from the trigger source occurs, thereby changing the operating mode or state of the peripheral.

By default,TCRx_REPEN is set to 0 , allowing the channel to be triggered to execute tasks an unlimited number of times. When TCRx_REPEN is set to 1 , the number of times the channel can execute tasks can be specifed through the RCRx_REP Register, with a maximum of 1023 times. When RCRx_REP is greater than 0 , each time a trigger occurs, the task is executed once, and RCRx_REP is decremented by 1 ; this continues until RCRx_REP is decremented to 0 , after which the task will not be executed even if the trigger condition occurs.

After the channel performs a write operation to the address pointed to by the TARx Register, a channel completion event is generated, which can serve as a PTC trigger source for another channel, simultaneously generating the ISR_TCIFx flag, and an interrupt is generated when IER_TCIEx is set to 1 . When TCRx_REPIRQ is set to 1 , an interrupt is generated only after all tasks specifed by RCRx_REP are completed; otherwise, an interrupt is generated after each task completion. When TCRx_REPTRIG is set to 1 , a PTC trigger for channel completion is generated only after all tasks specifed by RCRx_REP are completed; otherwise, a PTC trigger is generated after each task completion.

If the address pointed to by the TARx Register is an inaccessible bus address for PTC, a bus error will occur during access, which can generate the ISR_TEIFx flag and produce an interrupt when IER_TEIE is set to 1.

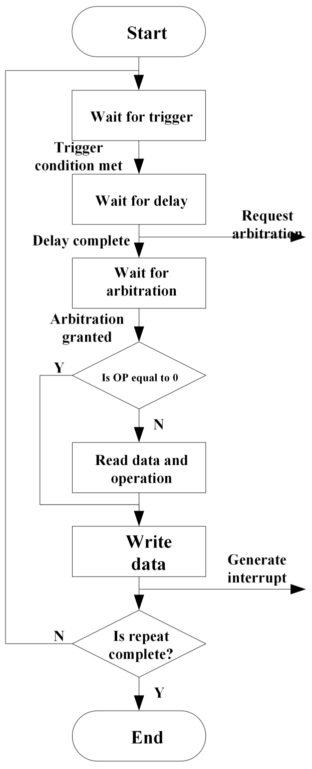


Figure 7-10: PTC Channel Execution Flowchart

7.3.3.3 Channel Arbitration

PTC has a total of 8 channels, which are arbitrated based on the principle that a lower channel number indicates a higher priority. Each channel enters arbitration after being triggered. When multiple channels enter arbitration simultaneously, the channel with the smallest number is granted permission to start task execution first, while the remaining channels remain in a suspended state until the task of the channel with the smallest number is completed, at which point arbitration is conducted again.

7.3.4 PTC Register

PTC1 base address is 0x50080000.

Table 7-4: **PTC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	interrupt status register
[31:24]			RSVD	
[23]	r	1'h0	TEIF8	transfer error flag for task 8
[22]	r	1'h0	TEIF7	transfer error flag for task 7
[21]	r	1'h0	TEIF6	transfer error flag for task 6
[20]	r	1'h0	TEIF5	transfer error flag for task 5
[19]	r	1'h0	TEIF4	transfer error flag for task 4
[18]	r	1'h0	TEIF3	transfer error flag for task 3
[17]	r	1'h0	TEIF2	transfer error flag for task 2
[16]	r	1'h0	TEIF1	transfer error flag for task 1
[15:8]			RSVD	
[7]	r	1'h0	TCIF8	task complete interrupt flag for task 8
[6]	r	1'h0	TCIF7	task complete interrupt flag for task 7
[5]	r	1'h0	TCIF6	task complete interrupt flag for task 6
[4]	r	1'h0	TCIF5	task complete interrupt flag for task 5
[3]	r	1'h0	TCIF4	task complete interrupt flag for task 4
[2]	r	1'h0	TCIF3	task complete interrupt flag for task 3
[1]	r	1'h0	TCIF2	task complete interrupt flag for task 2
[0]	r	1'h0	TCIF1	task complete interrupt flag for task 1
0x04			**ICR**	interrupt clear register
[31:17]			RSVD	
[16]	w1s	1'h0	CTEIF	clear transfer error flag
[15:8]			RSVD	
[7]	w1s	1'h0	CTCIF8	clear task complete interrupt flag for task 8
[6]	w1s	1'h0	CTCIF7	clear task complete interrupt flag for task 7
[5]	w1s	1'h0	CTCIF6	clear task complete interrupt flag for task 6
[4]	w1s	1'h0	CTCIF5	clear task complete interrupt flag for task 5
[3]	w1s	1'h0	CTCIF4	clear task complete interrupt flag for task 4
[2]	w1s	1'h0	CTCIF3	clear task complete interrupt flag for task 3
[1]	w1s	1'h0	CTCIF2	clear task complete interrupt flag for task 2
[0]	w1s	1'h0	CTCIF1	clear task complete interrupt flag for task 1
0x08			**IER**	interrupt enable register
[31:17]			RSVD	
[16]	rw	1'h0	TEIE	enable transfer error flag
[15:8]			RSVD	
[7]	rw	1'h0	TCIE8	enable task complete interrupt for task 8
[6]	rw	1'h0	TCIE7	enable task complete interrupt for task 7
[5]	rw	1'h0	TCIE6	enable task complete interrupt for task 6

Continued on the next page...

Table 7-4: **PTC Register Mapping Table (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h0	TCIE5	enable task complete interrupt for task 5
[3]	rw	1'h0	TCIE4	enable task complete interrupt for task 4
[2]	rw	1'h0	TCIE3	enable task complete interrupt for task 3
[1]	rw	1'h0	TCIE2	enable task complete interrupt for task 2
[0]	rw	1'h0	TCIE1	enable task complete interrupt for task 1
0x10			**TCR1**	task 1 control register
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source 0: task will only be triggered by SWTRIG others: task will be triggered by selected source or SWTRIG
0x14			**TAR1**	task 1 address register
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x18			**TDR1**	task 1 data register
[31:0]	rw	32'h0	DATA	data value for task operation
0x1C			**RCR1**	task 1 repetition and delay counter register
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x20			**TCR2**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x24			**TAR2**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x28			**TDR2**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x2C			**RCR2**	task 2 repetition and delay counter register
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x30			**TCR3**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x34			**TAR3**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x38			**TDR3**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x3C			**RCR3**	task 3 repetition and delay counter register
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x40			**TCR4**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x44			**TAR4**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x48			**TDR4**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x4C			**RCR4**	task 4 repetition and delay counter register

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x50			**TCR5**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x54			**TAR5**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x58			**TDR5**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x5C			**RCR5**	task 5 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x60			**TCR6**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x64			**TAR6**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x68			**TDR6**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x6C			**RCR6**	task 6 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x70			**TCR7**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x74			**TAR7**	

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x78			**TDR7**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x7C			**RCR7**	task 7 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x80			**TCR8**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x84			**TAR8**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x88			**TDR8**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x8C			**RCR8**	task 8 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0xD0			**MEM1**	temporary memory 1
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0xD4			**MEM2**	temporary memory 2
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0xD8			**MEM3**	temporary memory 3
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0xDC			**MEM4**	temporary memory 4
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0xE0			**GPIO31_0**	

Continued on the next page...

Table 7-4: PTC Register Mapping Table (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:29]			RSVD	
[28:24]	rw	5’h0	SELD	select trigger D of GPIO 31~0
[23:21]			RSVD	
[20:16]	rw	5’h0	SELC	select trigger C of GPIO 31~0
[15:13]			RSVD	
[12:8]	rw	5’h0	SELB	select trigger B of GPIO 31~0
[7:5]			RSVD	
[4:0]	rw	5’h0	SELA	select trigger A of GPIO 31~0 0: select GPIO 0 1: select GPIO 1 ...... 31: select GPIO 31
0xE4			**GPIO63_32**	
[31:29]			RSVD	
[28:24]	rw	5’h0	SELD	select trigger D of GPIO 63~32
[23:21]			RSVD	
[20:16]	rw	5’h0	SELC	select trigger C of GPIO 63~32
[15:13]			RSVD	
[12:8]	rw	5’h0	SELB	select trigger B of GPIO 63~32
[7:5]			RSVD	
[4:0]	rw	5’h0	SELA	select trigger A of GPIO 63~32 0: select GPIO 32 1: select GPIO 33 ...... 31: select GPIO 63
0xE8			**GPIO95_64**	
[31:29]			RSVD	
[28:24]	rw	5’h0	SELD	select trigger D of GPIO 95~64
[23:21]			RSVD	
[20:16]	rw	5’h0	SELC	select trigger C of GPIO 95~64
[15:13]			RSVD	
[12:8]	rw	5’h0	SELB	select trigger B of GPIO 95~64
[7:5]			RSVD	
[4:0]	rw	5’h0	SELA	select trigger A of GPIO 95~64 0: select GPIO 64 1: select GPIO 65 ...... 31: select GPIO 95

7.4 USART

HPSYS has three USART modules. The Universal Asynchronous Receiver-Transmitter supports full-duplex mode, providing baud rates of up to 6Mbps and various configurable data formats, offering flexible and effective data interaction methods for communication with external standardized devices. It also supports DMA for multi-packet transmission and reception.

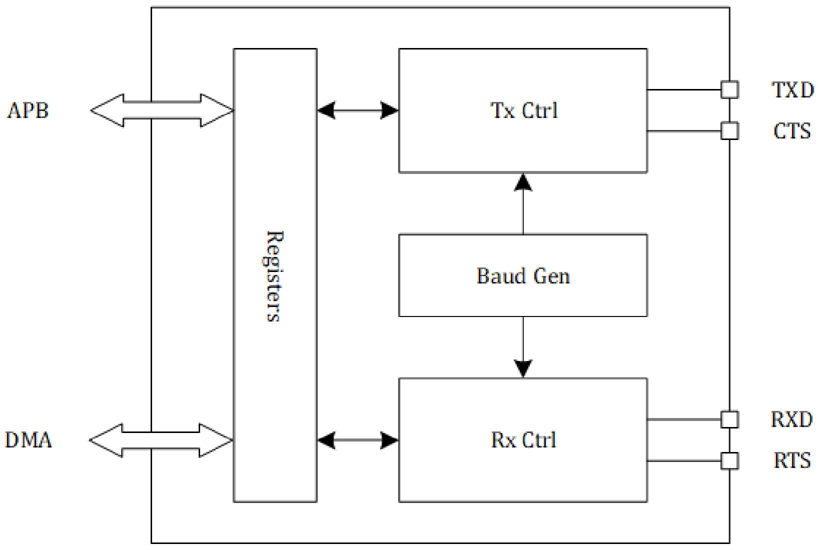


Figure 7-11: Universal Asynchronous Transceiver

Main Features of the Universal Asynchronous Transceiver:

- Full-Duplex Asynchronous Communication
- Configurable 16 Times Oversampling or 8 Times Oversampling, with a Choice of Frequency Priority or Clock Tolerance Priority
- Flexible Baud Rate Configuration; When the Input Clock is 48MHz and the Oversampling Rate is 16, the Baud Rate is 3Mbps
- Configurable Packet Length (7/8/9 Bits)
- Configurable Stop Bits (1/2 Bits)
- Hardware Flow Control (CTS/RTS)
- DMA Multi-Packet Transmission and Reception
- Receive Parity Check and Transmit Parity Generation
- Receive and Transmit Interrupts, as well as Other Error Interrupts

Baud Rate Calculation Instructions

Assuming the input clock is fxed at 48MHz, the baud rate calculation formula is as follows:

$$Baud\ Rate = \frac{48MHz}{(BRR_{INT} + \frac{BRR_{FRAC}}{16})(16\ or\ 8)}$$

7.4.1 USART Register

USART1 base address is 0x50084000。

USART2 base address is 0x50085000。

USART3 base address is 0x50086000

Table 7-5: **USART Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	Control Register 1
[31:29]			RSVD	
[28:27]	rw	2'h2	M	Mode bit indicates the length of the packet, including data bits and parity. Stop bits not included. 0: 6 bits (e.g. 6 data bits + no parity bit) 1: 7 bits (e.g. 6 data bits + 1 parity bit) 2: 8 bits (e.g. 7 data bits + 1 parity bit, or 6 data bits + 2 parity bits) 3: 9 bits (e.g. 8 data bits + 1 parity bit, or 7 data bits + 2 parity bits)
[26]			RSVD	
[25]			RSVD	
[24:20]			RSVD	
[19:15]			RSVD	
[14]	rw	1'h0	OVER8	Oversampling mode 0: Oversampling by 16 1: Oversampling by 8
[13]			RSVD	
[12]			RSVD	
[11]			RSVD	
[10]	rw	1'h0	PCE	Parity check enable. If enabled, parity bit is inserted at the MSB position 0: parity check disabled 1: parity check enabled
[9]	rw	1'h0	PS	Parity select 0: even parity 1: odd parity
[8]	rw	1'h0	PEIE	Parity error interrupt enable 0: interrupt disabled 1: interrupt is generated whenever PE=1 in the ISR register
[7]	rw	1'h0	TXEIE	Tx empty interrupt enable 0: interrupt disabled 1: interrupt is generated whenver TXE=1 in the ISR register
[6]	rw	1'h0	TCIE	Transfer compelete interrupt enable 0: interrupt disabled 1: interrupt is generated whenever TC=1 in the ISR register
[5]	rw	1'h0	RXNEIE	Rx not empty interrupt enable 0: interrupt disabled 1: interrupt is generated whenever RXNE=1 in the ISR register
[4]	rw	1'h0	IDLEIE	Idle line interrupt enable 0: interrupt disabled 1: interrupt is generated whenever IDLE=1 in the ISR register
[3]	rw	1'h0	TE	Transmitter enable 0: transmitter is disabled 1: transmitter is enabled
[2]	rw	1'h0	RE	Receiver enable 0: receiver is disabled 1: receiver is enabled
[1]			RSVD	
[0]	rw	1'h0	UE	USART enable 0: disabled 1: enabled
0x04			**CR2**	Control Register 2
[31:24]			RSVD	
[23]			RSVD	

Continued on the next page...

Table 7-5: Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[22:21]			RSVD	
[20]			RSVD	
[19]			RSVD	
[18]			RSVD	
[17]			RSVD	
[16]			RSVD	
[15]			RSVD	
[14]			RSVD	
[13:12]	rw	2'h0	STOP	Stop bits 0/1: 1 stop bit 2/3: 2 stop bits
[11]			RSVD	
[10]			RSVD	
[9]			RSVD	
[8]			RSVD	
[7]			RSVD	
[6]			RSVD	
[5]			RSVD	
[4]			RSVD	
[3:0]			RSVD	
0x08			**CR3**	Control Register 3
[31:25]			RSVD	
[24]			RSVD	
[23]			RSVD	
[22]			RSVD	
[21:20]			RSVD	
[19:17]			RSVD	
[16]			RSVD	
[15]			RSVD	
[14]			RSVD	
[13]			RSVD	
[12]	rw	1'h0	OVRDIS	Overrun disable 0: overrun error flag (ORE) will be set if new data received but previous data not read. New data will not overwrite the content in RDR register. 1: overrun disabled. If new data is received before previous data is read, the new data will overwrite the content in RDR register and ORE flag remains unset.
[11]	rw	1'h0	ONEBIT	One bit sampling mode 0: 3-bit sampling mode, the sampling value is determined by the voted result out of 3 bits 1: 1-bit sampling mode
[10]	rw	1'h0	CTSIE	CTS interrupt enable 0: interrupt disabled 1: interrupt is generated whenever CTSIF=1 in the ISR register
[9]	rw	1'h0	CTSE	CTS enable 0: CTS hardware flow control disabled 1: CTS hardware flow control enabled, data is transmitted only when CTS input is asserted low
[8]	rw	1'h0	RTSE	RTS enable 0: RTS hardware flow control disabled 1: RTS hardware flow control enabled, RTS output is asserted low when new data can be received

Continued on the next page...

Table 7-5: Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h0	DMAT	Transmitter DMA enable 0: DMA mode disabled for transmission 1: DMA mode enabled for transmission
[6]	rw	1'h0	DMAR	Receiver DMA enable 0: DMA mode disabled for reception 1: DMA mode enabled for reception
[5]			RSVD	
[4]			RSVD	
[3]			RSVD	
[2]			RSVD	
[1]			RSVD	
[0]	rw	1'h0	EIE	Error interrupt enable 0: interrupt disabled 1: interrupt is generated whenever FE=1 or ORE=1 or NF=1 in the ISR register
0x0C			**BRR**	Baud Rate Register
[31:16]			RSVD	
[15:4]	rw	12'h3	INT	Integer part of baud rate prescaler If OVER8 = 0, Baud Rate = 48000000 / (INT + FRAC/16) / 16 If OVER8 = 1, Baud Rate = 48000000 / (INT + FRAC/16) / 8 For example: OVER=0, INT=3, FRAC=0, Baud Rate = 48000000/(3+0)/16 = 1Mbps OVER=0, INT=3, FRAC=4, Baud Rate = 48000000/(3+4/16)/16 = 923077 = 921600 + 1.6‰ OVER=1, INT=52, FRAC=1, Baud Rate = 48000000/(52+1/16)/8 = 115246 = 115200 + 0.4‰
[3:0]	rw	4'h0	FRAC	Fractional part of baud rate prescaler
0x18			**RQR**	Request Register
[31:5]			RSVD	
[4]	w	1'h0	TXFRQ	Tx data flush request Reserved-Do not modify
[3]	w	1'h0	RXFRQ	Rx data flush request. Write 1 to clear the RXNE flag and discard the current data in RDR
[2]			RSVD	
[1]			RSVD	
[0]			RSVD	
0x1C			**ISR**	Interrupt and Status Register
[31:26]			RSVD	
[25]			RSVD	
[24:23]			RSVD	
[22]			RSVD	
[21]			RSVD	
[20]			RSVD	
[19]			RSVD	
[18]			RSVD	
[17]			RSVD	
[16]			RSVD	
[15]			RSVD	
[14]			RSVD	
[13]			RSVD	
[12]			RSVD	
[11]			RSVD	
[10]	r	1'h0	CTS	CTS input. Read this bit to get the raw status of the CTS line.

Continued on the next page...

Table 7-5: **Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	r	1'h0	CTSIF	CTS interrupt flag. This bit is set by hardware whenever CTS input toggles. 0: no change on the CTS line 1: there is a change on the CTS line
[8]			RSVD	
[7]	r	1'h1	TXE	Tx data empty 0: data is ready in TDR 1: data is already transferred to shift register, i.e. transmission is in progress or complete
[6]	r	1'h1	TC	transmission complete. This bit is set by hardware if the transmission is complete 0: transmission is not complete 1: transmission is complete
[5]	r	1'h0	RXNE	Rx data not empty. This bit is set by hardware when the received data is transferred into RDR register. 0: data is not received 1: data is ready in RDR to be read
[4]	r	1'h0	IDLE	Idle line detected 0: no idle line is detected 1: idle line is detected
[3]	r	1'h0	ORE	Overrun error. When new data is received but Rx buffer is not empty (i.e. previous data is not read yet), ORE is asserted and current RDR content is not lost. This feature can be disabled by set CR3_OVRDIS to 1. 0: no overrun error 1: overrun error is detected
[2]	r	1'h0	NF	Noise flag. Noise means the samping values in the 3-bit sampling mode are not the same. 0: no noise is detected 1: noise is detected
[1]	r	1'h0	FE	Framing error. This bit is set by hardware when stop bit is not correctly received 0: no framing error is detected 1: framing error is detected
[0]	r	1'h0	PE	Parity error. This bit is set when a parity error is detected in the received packet. 0: no parity error 1: parity error detected
0x20			**ICR**	Interrupt flag Clear Register
[31:21]			RSVD	
[20]			RSVD	
[19:18]			RSVD	
[17]			RSVD	
[16:13]			RSVD	
[12]			RSVD	
[11]			RSVD	
[10]			RSVD	
[9]	w1c	1'h0	CTSCF	CTS clear flag. Writing 1 to this bit clears the CTSIF flag in the ISR register.
[8]			RSVD	
[7]			RSVD	
[6]	w1c	1'h0	TCCF	Transmission complete clear flag. Writing 1 to this bit clears the TC flag in the ISR register.
[5]			RSVD	
[4]	w1c	1'h0	IDLECF	Idle line detected clear flag. Writing 1 to this bit clears the IDLECF flag in the ISR register.
[3]	w1c	1'h0	ORECF	Overrun error clear flag. Writing 1 to this bit clears the ORE flag in the ISR register.

Continued on the next page...

Table 7-5: Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	w1c	1'h0	NCF	Noise detected clear flag. Writing 1 to this bit clears the NF flag in the ISR register.
[1]	w1c	1'h0	FECF	Framing error clear flag. Writing 1 to this bit clears the FE flag in the ISR register.
[0]	w1c	1'h0	PECF	Parity error clear flag. Wriring 1 to this bit clears the PE flag in the ISR register.
0x24			**RDR**	Receive Data Register
[31:9]			RSVD	
[8:0]	r	9'h0	RDR	Received data
0x28			**TDR**	Transmit Data Register
[31:9]			RSVD	
[8:0]	rw	9'h0	TDR	Transmit data
0x2C			**MISCR**	Miscellaneous Register
[31]	rw	1'h0	AUTOCAL	
[30:8]			RSVD	
[7:4]	rw	4'h2	RTSBIT	assert RTS ahead of the frame completion (in number of bits) Reserved-Do not modify
[3:0]	rw	4'h6	SMPLINI	initial sample count, count down from this value to zero to reach the middle of the start bit in Rx Reserved-Do not modify
0x30			**DRDR**	Debug Receive Data Register
[31:0]	r	32'h0	DATA	
0x34			**DTDR**	Debug Receive Data Register
[31:0]	rw	32'h0	DATA	
0x38			**EXR**	Mutual Exclusive Register
[31:5]			RSVD	
[4]	r	1'b0	ID	
[3:1]			RSVD	
[0]	rw	1'h1	BUSY	

7.5 USB

This chip integrates a full-speed (FS) USB 2.0 Host/Device interface with the following functions.

- Software configurable endpoint settings, support suspend/ resume
- Support dynamic FIFO size
- Support session request protocol and host negotiation protocol
- Support full speed and slow speed modes
- On-chip integrated USB2.0 FS PHY

8 Analog Peripheral

8.1 GPADC

8.1.1 Introduction

The GPADC is a 12-bit precision SARADC that supports 0-3.3V input voltage, providing an output of 12-bit data. The input voltage can be configured as either single-ended or differential, and the output data can be accessed via the APB bus or DMA interface.

8.1.2 Main Features

- Supports 1.8V or 3.3V AVDD(depending on the chip series)
- Input voltage range:0~AVDD, with 12-bit resolution
- Supports both single-ended and differential inputs
- Supports 7channels of single-ended analog input and 1channel for dedicated battery voltage measurement input, or 3pairs of differential analog inputs
- Supports single measurement mode and continuous measurement mode.
- Each measurement can be divided into 8time slots, with each slot configurable for analog input channels.
- Supports software (register write) and hardware (such as timer) triggering methods.
- Supports DMAchannels.
- Sampling frequency is configurable, with a maximum sampling frequency of 4MHz (when 1.8V AVDD is 2MHz).
- An interrupt is generated upon completion of the conversion.

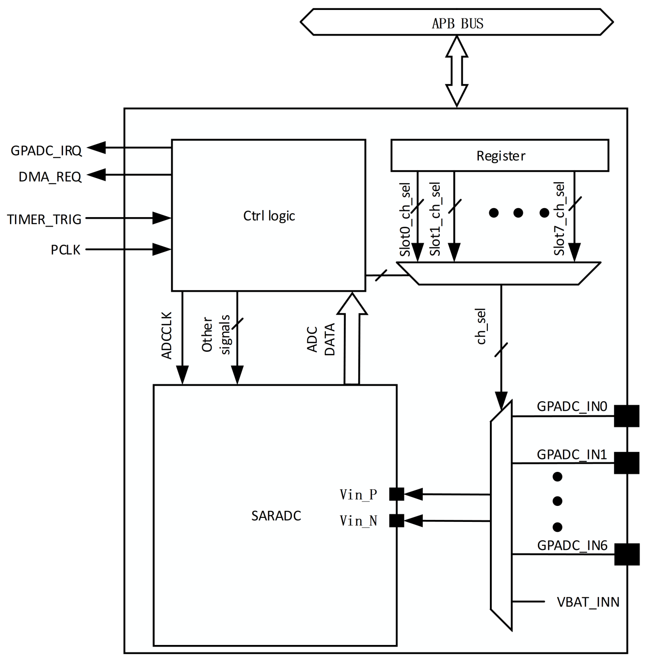


Figure 8-1: Block Diagram

8.1.3 Function Description

8.1.3.1 GPADC Clock Generation

The GPADC clock is generated by dividing the HPSYS PCLK and is configured through the ADC_CTRL_REG with DATA_SAMP_DLY and ADC_CTRL_REG2 with CONV_WIDTH and SAMP_WIDTH to set the ADCCLK frequency, calculated using the formula:

$$f_{ADCCLK} = f_{PCLK}/(DATA\_SAMP\_DLY + CONV\_WIDTH + SAMP\_WIDTH + 2)$$

Since ADCCLK is generated by dividing PCLK,it is essential to confrm the PCLK frequency when setting the ADCCLK frequency.

8.1.3.2 Time Slot Configuration

In each round of GPADC sampling, there are a total of 8 time slots that operate sequentially. Each time slot can be independently enabled or disabled by setting the SLOT_EN in ADC_SLOT*_REG; the input channels for each time slot can be independently configured by setting the PCHNL_SEL and NCHNL_SEL in ADC_SLOT*_REG.

8.1.3.3 Single-Ended/Differential Mode

By setting ANAU_GPADC_SE in the ADC_CFG_REG to 1, the GPADC will operate in single-ended input mode, and it is sufcient to set the PCHNL_SEL in the corresponding configuration register for each time slot to select the input channel.

Setting ANAU_GPADC_SE in register ADC_CFG_REG to 0 configures GPADC to differential input mode. You should set the corresponding configuration registers PCHNL_SEL and NCHNL_SEL for each time slot to select the input channels corresponding to Vin_P and Vin_N.

8.1.3.4 Input Channel Selection

The GPADC features 8 input channels. Among these 8 input channels, channels 0-6 are connected to the external interface via PAD, while channel 7 is connected to an internal node for measuring battery voltage.

By configuring PCHNL_SEL and NCHNL_SEL in ADC_SLOT*_REG, the input channels to be sampled for each time slot can be specifed.

In single-ended mode, only PCHNL_SELneeds to be set to select the input channel.

8.1.3.5 Sampling Mode

If the ADC_CTRL_REG's ADC_OP_MODE is set to 0, then the GPADC is in single sampling mode. In this mode, each time the GPADC is activated, it will complete one round of sampling according to the configured time slots and then return to the waiting trigger state.

If the ADC_CTRL_REG's ADC_OP_MODE is set to 1, then the GPADC is in continuous sampling mode. In this mode, each time the GPADC is activated, it will continuously sample according to the configured time slots. Setting the ADC_CTRL_REG's ADC_STOP to 1 will return the GPADC to the waiting trigger state.

8.1.3.6 Activate the GPADC

- Write Register to Activate
  Setting ADC_STARTinADC_CTRL_REGto 1will initiate GPADC.
  After triggering GPADC , if it is in single-sampling mode, it will return to the waiting trigger state after completing one round of sampling. If it is in continuous sampling mode, you must set ADC_STOP in ADC_CTRL_REG to 1 to return GPADC to the waiting trigger state
- Timer Trigger
  GPADC supports TIMERtriggering. To enable the TIMERtrigger function, setTIMER_TRIG_EN in ADC_CTRL_REGto 1. There are 8 trigger sources available, which can be selected via TIMER_TRIG_SRC_SEL in ADC_CTRL_REG . The corresponding relationships are shown in the table below:

TIMER_TRIG_SRC_SEL	TRIG_SRC
0	GPTIM1 TRGO
1	GPTIM2 TRGO
2	APTIM1 TRGO
3	BTIM1 TRGO
4	BTIM2 TRGO
5	GPTIM1 CH0 Output
6	GPTIM1 CH1 Output
7	GPTIM1 CH2 Output

  After triggering GPADC , if it is in single-sampling mode, it will return to the waiting trigger state after completing one round of sampling. If it is in continuous sampling mode, you must set ADC_STOP in ADC_CTRL_REG to 1 to return GPADC to the waiting trigger state.

8.1.3.7 Data Access

GPADC The converted data can be accessed in the following two ways:

- Register Read
  Software can directly read the conversion results of GPADC by accessing the register. The data for each time slot is stored in the register ADC_RDATA* , and the correspondence between the register and the data for each time slot is shown in the table below:

ADC_RDATA0[31:0]	
SLOT1_RDATA	SLOT0_RDATA
ADC_RDATA1[31:0]	
SLOT3_RDATA	SLOT2_RDATA
ADC_RDATA2[31:0]	
SLOT5_RDATA	SLOT4_RDATA
ADC_RDATA3[31:0]	
SLOT7_RDATA	SLOT6_RDATA

  In the register ADC_RDATA* , the LSB of the GPADC output data corresponding to each time slot is right-aligned to either bit 0 or bit 16 of the register. Taking ADC_RDATA0 as an example, the alignment of time slot data in the register is shown in the table below:

SLOT0_RDATA（ADC_RDATA0[31:16]）																SLOT0_RDATA（ADC_RDATA0[15:0]）															
0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

- DMA Mode
  Read the conversion results of the GPADC from DMAC1 in HPSYS. The process is as follows:
  Set the DMA_EN bit of the ADC_CTRL_REG register to 1.
  The source address for DMA is set to 0x5007034. The alignment of ADC data at this address is:

Bit[15：0]															
0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

  Refer to the DMAC section for additional DMA settings.

8.1.3.8 Notifcation Mechanism

The GPADC generates an interrupt upon completing the sampling conversion, which is reported to the CPU.

This interrupt can be masked by setting the GPADC_IMR bit of the GPADC_IRQ register to 1.

This interrupt can be cleared by setting the GPADC_ICR bit of the GPADC_IRQ register to 1.

8.1.3.9 System Configuration Dependencies

The clock for the GPADCis generated byHPSYS PCLKthrough frequency division, and the frequency ofHPSYS PCLKmust be specifed when setting the frequency.

The GPADC can connect to 7 external channels, which are linked through PAD and the external measured voltage. When in use, the corresponding PINMUX settings must be configured. The correspondence between GPADC channel and PAD is as follows:

GPADC channel	PAD	CH_SEL_value
GPADC_CH0	PAD_PA28	0
GPADC_CH1	PAD_PA29	1
GPADC_CH2	PAD_PA30	2
GPADC_CH3	PAD_PA31	3
GPADC_CH4	PAD_PA32	4
GPADC_CH5	PAD_PA33	5
GPADC_CH6	PAD_PA34	6

In When the CH_SEL value is 7 and the EN_VBAT_MON bit in the ANAU_CR register of the HPSYS_CFG module is set to 1, the internal measurement sampling point for battery voltage is selected. The sampling point voltage is derived from the battery voltage through internal voltage divider resistors, with the division ratio being 0.5 at 3.3V AVDD and 0.3 at 1.8V AVDD, with the division ratio factory calibrated.

When the GPADC is operational, it is necessary to enable the bandgap in ANAU. Set the EN_BG bit in the ANAU_CR register of the HPSYS_CFG module to 1 to activate the bandgap.

8.1.3.10 Configuration Startup Process

The GPADC generally follows the process outlined below:：

- Configure the PINMUX.
- Configure the GPADC clock frequency, input channel selection, and other parameters.

- Set Set the EN_BG bit in the ANAU_CR register of the HPSYS_CFG module to 1 to enable the Bandgap.
- Set the ANAU_GPADC_LDOREF_EN bit in the ADC_CFG_REG1 register to 1 to enable the LDO that provides reference voltage to the GPADC.
- Set the FRC_EN_ADC bit in the ADC_CTRL_REG register to 1 to enable the GPADC module.
- Trigger the GPADC, initiate sampling, and read the data.
- Once sampling is complete, set the FRC_EN_ADC in the register ADC_CTRL_REG to 0 to disable the GPADC module. If operating in continuous sampling mode, you must first set the ADC_STOP in the register ADC_CTRL_REG to 1 and then to 0 to interrupt the sampling process.
- Set the ANAU_GPADC_LDOREF_EN in the register ADC_CFG_REG1 to 0 to disable the LDO that supplies reference voltage to the GPADC.
- The Bandgapis shared with the TSENmodule, and it is advisable not to disable it.

The process must satisfy certain circuit stabilization times.

- After configuring the pinmux, the voltage connected to the input channel viaPADrequires a specific stabilization time.
- After opening the bandgap, only then should the LDO that provides the reference voltage be activated; the LDO requires a 200us stabilization time.
- After enabling the GPADC,at least a 200usstabilization time is necessary.
- Finally, the GPADC can be triggered by writing to the register or by using a Timer.

8.1.4 GPADC Register

GPADC base address is 0x50087000.

Table 8-1: **GPADC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ADC_CFG_REG1**	ADC Analog Config Register 1
[31:30]			RSVD	
[29:25]	rw	5'h16	ANAU_GPADC_CMM	Tune CDAC CM voltage 375mV range (increasing) / 25mV step, 8: for 0.5V Vcm,in
[24:22]	rw	3'h3	ANAU_GPADC_CMPCL	Tune ADC comparator CL= 3: 40f, range: 10fF (0) ~ 80fF (7) / 10fF step
[21:20]	rw	2'h2	ANAU_GPADC_VSP	Set comparator input CM in sampling phase, 0.539V (0) / 0.578V (1) / 0.642V (2) / 0.784V (3)
[19]	rw	1'h0	ANAU_GPADC_LDOREF_EN	Enable LDORF for ADC VREF
[18:15]	rw	4'hA	ANAU_GPADC_LDOVREF_SEL	Set reference voltage for LDOREF, range = 0.35V(0) ~ 0.65V(15), step = 20mV
[14:12]	rw	3'h1	ANAU_GPADC_SEL_PCH	Select P-side input channel for GPADC, 0 for channel 0, 7 for channel 7, effective when force on
[11:9]	rw	3'h0	ANAU_GPADC_SEL_NCH	Select N-side input channel for GPADC, 0 for channel 0, 7 for channel 7, effective when force on
[8]	rw	1'h0	ANAU_GPADC_MUTE	Short GPADC P & N input to CMREF, i.e., VREF/2
[7]	rw	1'h0	ANAU_GPADC_SE	Set GPADC in single-ended mode, signal range at P-input: 0 ~ VREF
[6]	rw	1'h0	ANAU_GPADC_EN_V18	
[5:3]	rw	3'h2	ANAU_GPADC_CL_DLY	
[2]	rw	1'h0	ANAU_GPADC_P_INT_EN	
[1]			RSVD	
[0]	rw	1'h0	ANAU_GPADC_CMREF_FAST_EN	
0x04			**ADC_Slot0_REG**	ADC Slot0 Config Register
[31:14]			RSVD	

Continued on the next page...

Table 8-1: GPADC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x08			**ADC_Slot1_REG**	ADC Slot1 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x0C			**ADC_Slot2_REG**	ADC Slot2 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x10			**ADC_Slot3_REG**	ADC Slot3 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x14			**ADC_Slot4_REG**	ADC Slot4 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x18			**ADC_Slot5_REG**	ADC Slot5 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x1C			**ADC_Slot6_REG**	ADC Slot6 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x20			**ADC_Slot7_REG**	ADC Slot7 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x24			**ADC_RDATA0**	ADC Read Data0
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT1_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT0_RDATA	
0x28			**ADC_RDATA1**	ADC Read Data1

Continued on the next page...

Table 8-1: GPADC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT3_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT2_RDATA	
0x2C			**ADC_RDATA2**	ADC Read Data2
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT5_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT4_RDATA	
0x30			**ADC_RDATA3**	ADC Read Data3
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT7_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT6_RDATA	
0x34			**ADC_DMA_RDATA**	ADC Read Data For DMA
[31:29]			RSVD	
[28:16]	r	13'h0	DMA_RDATA_RAW	
[15:13]			RSVD	
[12:0]	r	13'h0	DMA_RDATA	
0x38			**ADC_CTRL_REG**	ADC Control Register
[31:21]			RSVD	
[20:17]	rw	4'h4	DATA_SAMP_DLY	
[16]	rw	1'b0	DMA_DATA_SEL	0: combined data 1: raw data
[15]	rw	1'b0	TIMER_TRIG_TYP	0: pulse no edge detect needed 1: level,need edge detect
[14:12]	rw	3'h0	TIMER_TRIG_SRC_SEL	Timer trigger source select
[11]	rw	1'b0	FRC_EN_ADC	Enable GPADC core
[10]	rw	1'b0	CHNL_SEL_FRC_EN	Enable input channel setting in ADC_CFG_REG1
[9]	rw	1'b1	TIMER_TRIG_EN	Enable timer trigger function
[8]			RSVD	
[7]	rw	1'b1	DMA_EN	Enable DMA interface
[6:3]	rw	4'h6	INIT_TIME	GPADC will wait INIT_TIME ADCCLK cycles to start sample/conversion after being trigged
[2]	rw	1'b0	ADC_STOP	Write 1 to stop GPADC in continuous mode(need write 0 to clear)
[1]	w1s	1'b0	ADC_START	Write 1 to start GPADC,(don't need clear)
[0]	rw	1'b0	ADC_OP_MODE	0: single conversion mode 1: continuous conversion mode
0x3C			**ADC_CTRL_REG2**	ADC Control Register2
[31:24]	rw	8'h80	CONV_WIDTH	
[23:0]	rw	24'h8000	SAMP_WIDTH	
0x40			**GPADC_STATUS**	GPADC Status Register
[31:12]			RSVD	
[11:9]	r	3'h0	CUR_SLOT	
[8:1]	r	8'h0	SLOT_DONE	
[0]	r	1'h0	ADC_DONE	
0x44			**GPADC_IRQ**	GPADC IRQ Register
[31:4]			RSVD	
[3]	r	1'b0	GPADC_ISR	
[2]	r	1'b0	GPADC_IRSR	
[1]	rw	1'b0	GPADC_IMR	
[0]	w1s	1'b0	GPADC_ICR	

8.2 TSEN

8.2.1 Introduction

TSEN converts temperature into digital encoding by sampling the voltage of the internal temperature-sensitive resistor, assisting the system in real-time monitoring of the chip temperature.

8.2.2 Main Features

- The resolution is 0.2°C
- The supported temperature range is -40°C to 125°C
- Reading can be performed in either polling or interrupt mode.

8.2.3 Function Description

8.2.3.1 Clock Signal

The operating clock of TSEN is derived from the PCLK of LPSYS. The division ratio is set by the ANAU_TSEN_CLK_DIV in the TSEN_CTRL_REG register. The frequency calculation relationship is as follows:

$$ftsen = fpclk/ANAU\_TSEN\_CLK\_DIV$$

8.2.3.2 Reading Process

The conversion result is read from the TSEN_RDATA register, and the read result is converted to temperature using the following formula:

$$Temp = (Dec(TSEN_RDATA) + 3000)/10100 * 749.2916 - 277.5391$$

The process for reading through polling is as follows:

After starting TSEN, poll the TSEN_IRQ register for the TSEN_IRSR. When the TSEN_IRSR value is 1, it indicates that the temperature data conversion is complete. After reading the data, set the TSEN_ICR in the TSEN_IRQ register to 1 and clear the TSEN_IRSR.

The process for reading via interrupts is as follows::

Enable the TSEN interrupt and set the TSEN_IMR in the TSEN_IRQ register to 0. Start the TSEN; after the conversion is complete, it will send a TSEN_IRQ interrupt to the CPU. During interrupt processing, set the TSEN_ICR in the TSEN_IRQ register to 1 to clear the interrupt and read the data.

8.2.3.3 Usage Process

The operation of TSEN must follow the process outlined below.

1. Configure the division ratio according to the PCLK frequency; the clock frequency of TSEN should be either 1MHz or 2MHz.
2. Set Set EN_BG to 1 in the HPSYS_CFG register ANAU_CR to enable the Bandgap.

3. Set ANAU_TSEN_EN in register TSEN_CTRL_REG to 1 to enable the clock for TSEN.
4. SetANAU_TSEN_PUinregisterTSEN_CTRL_REGto 1andANAU_TSEN_RUNto 0.
5. First, setANAU_TSEN_RSTBinregisterTSEN_CTRL_REGto 0and then to 1,maintaining a low level for at least 20us.
6. Set ANAU_TSEN_RUN in register TSEN_CTRL_REG to 1, then read the result using polling or interrupts. Alternatively, you may wait for 3ms absolute time before reading.
7. To disable TSEN: set ANAU_TSEN_RUN/ANAU_TSEN_PU/ANAU_TSEN_EN in register TSEN_CTRL_REG to 0, and set ANAU_TSEN_RSTB to 1.
8. Bandgap shared with GPADC; it is recommended not to disable it. If it must be disabled, care should be taken not to interfere with GPADC's operation

8.2.4 TSEN Register

TSEN base address is 0x50089000.

Table 8-2: **TSEN Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**TSEN_CTRL_REG**	TSEN Analog Control Register
[31:18]			RSVD	
[17:12]	rw	6'h30	ANAU_TSEN_CLK_DIV	gen tsen clk by divide hclk by anau_tsen_clk_div
[11]	rw	1'h0	ANAU_TSEN_EN	Enable tsen digital module
[10]	r	1'h0	ANAU_TSEN_RDY	tsen ready
[9]	rw	1'h0	ANAU_TSEN_SER_PAR_SEL	serial-parallel output selection
[8]	rw	1'b0	ANAU_TSEN_SGN_EN	signature-mode enable
[7:6]	rw	2'h1	ANAU_TSEN_FCK_SEL	select internal clock frequency
[5:3]	rw	3'h1	ANAU_TSEN_IG_VBE	bias current selection to tune vba
[2]	rw	1'h0	ANAU_TSEN_RUN	enable tsen run
[1]	rw	1'h1	ANAU_TSEN_RSTB	resetb for tsen
[0]	rw	1'h0	ANAU_TSEN_PU	power up tsen
0x04			**TSEN_RDATA**	Tsen Read Data
[31:12]			RSVD	
[11:0]	r	12'h0	TSEN_RDATA	
0x08			**TSEN_IRQ**	Tsen IRQ Register
[31:4]			RSVD	
[3]	r	1'b0	TSEN_ISR	
[2]	r	1'b0	TSEN_IRSR	
[1]	rw	1'b0	TSEN_IMR	
[0]	w1s	1'b0	TSEN_ICR	

9 Timer

9.1 ATIM

HPSYS has one ATIM module, The working clock of ATIM is pclk_hpsys, and it should be noted that counting may be affected when the system dynamically adjusts the frequency.

9.1.1 Introduction

ATIM (Advanced Timer) is based on a 32-bit counter, capable of timing, measuring the pulse length of input signals (input capture) or generating output waveforms (output compare and PWM) , among other functions. ATIM supports 6 channels of PWM complementary output with dead time protection, allows for multi-channel PWM simultaneous commutation, and features 2 brake inputs that can quickly switch the output to a safe state. The counter itself can perform incrementing, decrementing, or increment/decrement counting, with the counting clock selectable from system PCLK , IO input signals, or cascading input signals, and can have a pre-scaling factor of 1~65536. ATIM has a total of 6 channels, which can be independently configured for input capture or output mode. The results of counting, input capture, and output compare can generate interrupts, DMA requests, or PTC events. ATIM includes master-slave mode interfaces, enabling multi-level cascading for multi-level counting or synchronized triggering functions.

9.1.2 Main Features

- 32 bit increment, decrement, increment/decrement auto-reload counter
- 16 bit programmable (can be modifed in real-time)prescaler, with a division factor for the counter clock frequency ranging from 1 to 65536 for any value
- 16 bit configurable repeat count
- Supports one pulse mode (OPM), which automatically stops the counter upon completion of the repeat count.
- 6 independent channels
  - Channels 1 to 3 can be configured as input or output modes, with each channel capable of outputting two complementary PWM signals with dead-time protection.
  - Channel 4can be configured as either input or output mode, capable of outputting a single PWM
  - Channels 5~6can be configured for Output Compare Mode
- Input Mode
  - Rising Edge/Falling Edge Capture
  - PWM Pulse Width and Period Capture (requires two channels)
  - Optional one of 4 input ports or 1 external trigger port, supporting debounce fltering and pre-frequency reduction
- Output Mode
  - Force output to high/low level
  - Output high/low/flip level upon reaching the comparison value
  - PWM output, with configurable pulse width and period
  - Multi-channel PWM composite output, capable of generating multiple PWM with interrelated characteristics
  - Single Pulse/Re-triggerable One Pulse Mode Output

- Master-Slave Mode
  - Supports multiple counter interconnections, enabling it to generate control signals as a master device while being controlled by external inputs or other master devices as a slave.
  - Control modes include reset, trigger, gating, and others.
  - Supports synchronous starting and resetting of multiple counters.
- Encoding mode input for controlling counter increment/decrement counting.
- Supports Hall sensor circuits for positioning.
- 2 Brake inputs, featuring debounce fltering, can quickly place the output in a safe state. Brake signal sources include::
  - CPU Exception
  - Comparator
  - External Input
  - Software Trigger
- An interrupt/DMArequest/PTCis generated when the following events occur:
  - Update: Counter increment overflow/decrement overflow, counter initialization (via software or internal/external trigger)
  - Trigger Events (counter start, stop, initialization, or counting triggered by internal/external sources)
  - Input Capture
  - Output Compare
  - Brake
  - Communitation

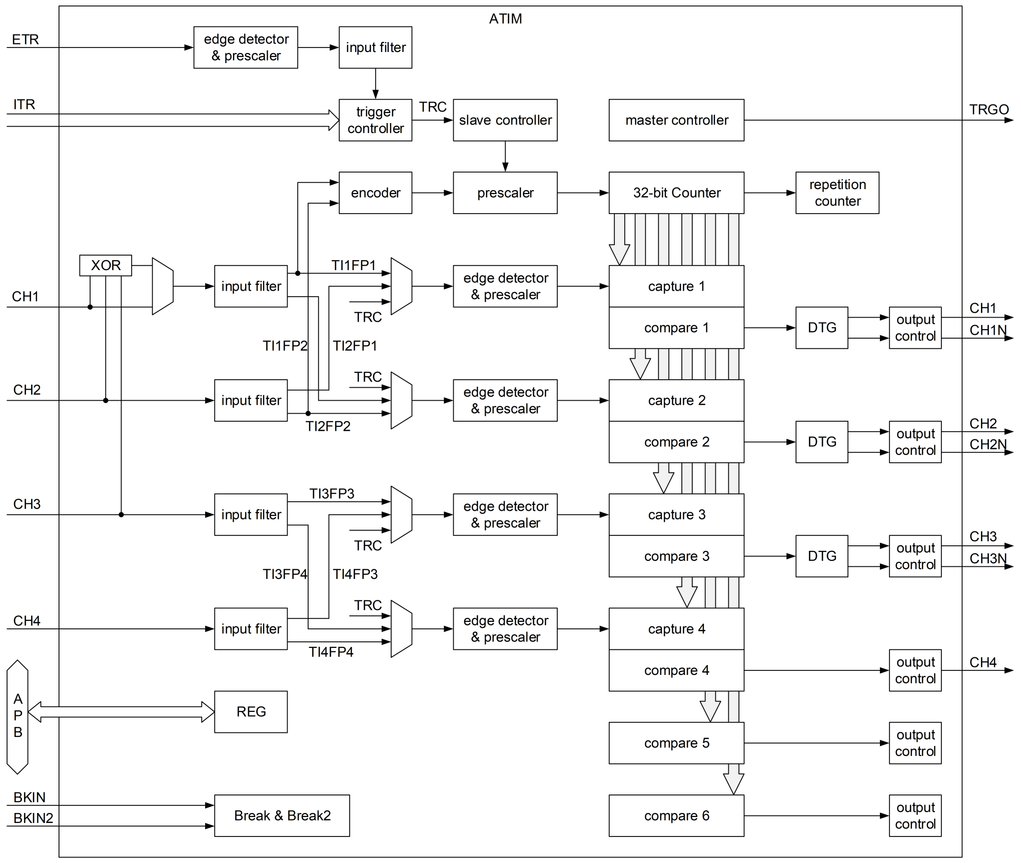


Figure 9-1: **ATIM Structure Diagram**

9.1.3 ATIM Function Description

9.1.3.1 Counter

All functions of ATIM are based on a 32 -bit counter. The counter operates based on events, with the most fundamental event being a PCLK clock edge. Depending on the configuration, other counting events may include toggles from external inputs, output toggles from other timers, and decoding outputs from quadrature encoder interfaces.

Counting events will only enter the counter after being processed by a prescaler. The prescaler count ranges from 1 to 65536 (PSC+1), meaning that the counter's value will only change once after (PSC+1) counting events have occurred.

The counter features three counting modes: incrementing, decrementing, and center-aligned. In the incrementing counting mode (CR1_CMS=0 and CR1_DIR=0) , the counter counts from 0 to the auto-reload value ARR , then restarts counting from 0 and generates a counter overflow event. In the decrementing counting mode (CR1_CMS=0 and CR1_DIR=1) , the counter counts down from ARR to 0 , then restarts counting from ARR and generates a counter underflow event. In the center-aligned mode (CR1_CMS is not 0) , the counter counts from 0 to ARR −1 , generating a counter overflow event, then counts down from ARR to 1 and generates a counter underflow event, after which it restarts counting from 0 again.

The count value can be read through CNT. The counting direction can be read from CR1_DIR.

9.1.3.2 Update Event(UEV)

An update event is used to indicate the end of a counting unit. The most basic update event occurs on every overflow or underflow of the counter (when repeat counting is not enabled) . An update event is also generated when the software sets EGR_UG to 1 . Update events can trigger interrupts, DMA requests, and PTC triggers, making it the most fundamental notifcation function of the timer.

By setting CR1_UDIS to 1 in software, the generation of update events can be disabled. This prevents the shadow register from being updated when writing new values to the preload register. No update events will occur until the UDIS bit is written to 0 .

If CR1_URS (update request selection) is set to 1 , setting EGR_UG to 1 will generate an Update Event, but will not set the UIF flag to 1 (therefore, no interrupts or DMA requests will be sent). Consequently, if the counter is cleared when a capture event occurs, neither an update interrupt nor a capture interrupt will be generated simultaneously.

When an Update Event occurs, the RCR , ARR , and PSC registers will be reloaded, and the update flag SR_UIF will be set to 1 (when CR1_URS=0). This function ensures that modifcations to the basic parameters of these counters do not affect the current counting unit, taking effect only in the next counting cycle.

9.1.3.3 Repeated Counting

If the Repeat Counter (RCR > 0) is configured, it will decrement each time the counter overflows or underflows, and an Update Event will only occur when the Repeat Counter reaches 0. When an Update Event occurs, the Repeat Counter will reload the value of RCR.

The current value of the Repeat Counter cannot be read.

9.1.3.4 Shadow Register

Modifcations to the RCR, ARR, and PSC registers will not be directly reflected in the current counting unit; they will only be updated when an Update Event occurs. Before the Update Event, the counter uses the values from the Shadow Registers. This ensures that dynamically changing these register values during counting does not affect the integrity of the current counting unit, which is signifcant for applications such as PWM output.

If CR1_APRE is 0, theARRregister will take effect in real-time after configuration, without waiting for an update event.

The output compare register CCRx also features a shadow register. When CCMRx_OCxPE is 0 , the configured CCRx will take effect immediately; otherwise, it will only take effect when an update event occurs.

9.1.3.5 Master-Slave Mode

The timer can operate simultaneously in master mode and slave mode. Master mode indicates that the timer can output the TRGO signal to the ITR input of other timers on the chip, which is used to control the counting behavior of those timers. Slave mode indicates that the counting behavior of the timer is influenced by the external input ETR , which is output from other timers to this timer. The device's ITR signal, or the control of the timer channel input CHx.

Multiple timers can achieve Timer Synchronization through a master-slave configuration, enabling functions such as multi-level frequency division, simultaneous start, and gated counting.

The master mode can output the TRGO signal upon various events, such as updates, enabling, input capture, and output comparison, as selected by CR2_MMS.

The slave mode can select behaviors such as counter reset, trigger start, counting enable, and counting events, as determined by SMCR_SMS. The trigger signal TRGI on which the slave mode depends can be flexibly configured, with options to select from ETR, ITR, and channel inputs, as well as to choose signal polarity for pre-scaling, fltering, and other operations

- When the timer is in reset from mode(SMCR_SMS=0100)and the TRGIchanges, both the counter and its prescaler are reinitialized. If CR1_URSis 0,an update event UEVwill be generated, causing all preload registers ARRand CCRxto be updated.
- When the timer is in gated from mode (SMCR_SMS=0101), the counting occurs only when TRGI meets the high or low level requirements; otherwise, the counter remains unchanged.
- When the timer is in triggered from mode (SMCR_SMS=0110), the software does not need to configure CR1_CEN to initiate counting; instead, the counter is automatically started when TRGI meets specific trigger requirements.
- When the timer is in external clock slave mode (SMCR_SMS=0111) , the counting event is modifed to count on the rising edge of TRGI , and counting occurs only when TRGI changes state.
- When the timer is in reset trigger slave mode (SMCR_SMS=1000) , the counter is reset and automatically restarted when TRGI meets specific trigger requirements.

9.1.3.6 Channel Input and Output

Some channels of the timer can be independently configured as input capture mode(CCMRx_CCxS!=0) or output mode(CCMRx_CCxS=0).

In input capture mode, when the corresponding trigger signal is valid, the value of the counter is recorded into CCRx , and an interrupt or other notifcation signal is generated. The trigger signal can be selected from ETR , ITR , and channel input CHx , and the signal polarity can be selected, along with pre-scaling, fltering, and other operations. The notifcation signals generated by the channel include interrupts, DMA requests, and PTC triggers. Input capture mode can record the moments of external signal changes, measure PWM periods, and duty cycles, among other functions.

In output mode, the channel compares the counter value with the size of CCRx , generating a fxed level on the channel output CHx/CHxN , or producing a PWM output signal based on the comparison results of this channel and other channels, along with generating interrupt and other notifcation signals. The parameters of the PWM signal, including the number of pulses, frequency, duty cycle, and phase, are adjustable. Multiple channels can also collaborate to produce specific relationships of PWM combinations, such as six-channel complementary PWM with dead time protection. The notifcation signals generated by the channel include interrupts, DMA requests, and PTC triggers.

In output mode, in the event of an emergency, the output enable can be urgently disabled through the circuit interruption input signals BKIN and BKIN2 , or the output can be set to a preset level to protect the external circuits connected to the timer.

9.1.3.7 Input Capture Mode

In Input Capture Mode, when a rising or falling edge of the corresponding trigger signal on the channel is detected, the value of the counter will be latched using CCRx . When a capture event occurs, the corresponding SR_CCxIF flag will be set to 1 , and an interrupt, a DMA request (if enabled), or a PTC trigger signal may be sent. If the SR_CCxIF flag is already high when the capture event occurs, the repeat capture flag SR_CCxOF will be set to 1 . The SR_CCxIF can be cleared by software by writing 0 to SR_CCxIF or by reading the captured data stored in CCRx . Writing 0 to SR_CCxOF will also clear it.

The following example illustrates how to capture the counter value into CCR1 when a rising edge occurs at the CH1 input. The specific steps are as follows:

1. Select the valid input: Channel 1 is to be connected to the CH1 input, so write 01 to CCMR1_CC1S.
2. Configure the desired input fltering bandwidth based on the signal connected to the timer.
   Assuming that the CH1 signal edge changes with a maximum jitter of 5 PCLK cycles, the fltering bandwidth should be set to greater than 5 PCLK cycles. Set CCMR1_IC1F to 0011(0x3) so that when eight consecutive sampling points (sampled at PCLK frequency) are detected to be at the new level, the transition edge of CH1 can be confrmed.
3. Set Set CCER_CC1Pand CCER_CC1NPto 0to select the valid conversion edge on CH1as the rising edge.
4. Program the input prescaler.
   In this example, we want to perform a capture operation on every valid conversion; therefore, the prescaler is disabled (set CCMR1_IC1PS to 00).
5. SetCCER_CC1Eto 1to enable channel 1and allow the counter value to be captured in CCR1.
6. If necessary, set DIER_CC1IE to 1to enable the corresponding interrupt request, or set DIER_CC1DE to 1to enable DMA requests.

Once configured, the channel will execute the following actions when a rising edge appears on the CH1 input:

1. CCR1 Register records the value of the counter.
2. SR_CCxIF Flag set to 1 (Interrupt flag). If at least two consecutive captures occur without clearing SR_CCxIF, then the SR_CCxOF capture overflow flag will be set to 1.
3. An interrupt is generated based on CCER_CC1IE.
4. A DMA request is generated based on DIER_CC1DE.

To handle repeated captures, it is recommended to read the data before accessing SR_CCxOF. This can prevent the loss of repeated capture information that may occur between reading SR_CCxOF and the data.

Setting EGR_CCxGto 1via software can immediately generate a capture and produce a channel capture interrupt and DMA request.

9.1.3.8 PWM Input Capture

PWM Input Capture is an advanced application of Input Capture, which can be utilized to measure the period and duty cycle of the PWM input signal. To implement this functionality, both channels must be configured in Input Capture Mode, with the trigger signals mapped to the rising and falling edges of the PWM input, and the counter reset mode must be activated.

The following example demonstrates how to measure the period and duty cycle of the PWM input from CH1 using Channel 1 and Channel 2. The specific steps are as follows:：

1. Designate the valid input for Channel 1as the CH1input by writing 01to CCMR1_CC1S.
2. Select channel 1 The effective polarity of the input signal (used for capturing in CCR1 and resetting the counter) is set by writing 0 to CCER_CC1P and CCER_CC1NP, selecting the effective transition edge on CH1 as the rising edge.
3. The effective input for channel 2 is also the CH1 input; write 10(0x2) to CCMR1_CC2S.
4. Select channel 2 The effective polarity of the input signal (for CCR2 capture) , write 1 to CCER_CC2P and 0 to CCER_CC1NP , selecting the effective conversion edge on CH1 as the falling edge.
5. Set the slave mode control signal to CH1 by writing 101(0x5) to SMCR_TS, selecting TI1FP1.
6. Configure the mode controller to reset mode by writing 0100 (0x4)to SMCR_SMS.
7. Enable channel 1 and channel 2 by setting CCER_CC1E and CCER_CC2E to 1.。

After configuration, on each rising edge of CH1 , the counter's value is recorded in CCR1 , while the counter is reset and begins counting again; on each falling edge of CH1 , the counter's value is recorded in CCR2 . Multiplying the value of CCR1 by the period of PCLK calculates the period of PWM. Set Multiplying the value of CCR2 by the period of PCLK calculates the duration of the high level of PWM, thus determining the duty cycle of PWM.

9.1.3.9 Output Compare Mode

In Output Compare Mode, when the count value satisfes a specific relationship with CCRx , particular outputs can be generated on the corresponding CHx and CHxN , which are typically used to control output waveforms or to indicate that a certain time period has elapsed.

Specifically, the channel will execute the following operations when CCRx matches the counter:

1. A programmable value will be assigned for the corresponding CHx and CHxN , as defined by the Compare Mode Register CCMRx_OCxM and the Output Polarity Register CCER_CCxP/CCxNP . Upon matching, the output pin can either maintain its level (CCMRx_OCxM=0000) or be set to active level (CCMRx_OCxM=0001) , inactive level (CCMRx_OCxM=0010) , or toggle (CCMRx_OCxM=0011) .
2. Set the interrupt status register flag SR_CCxIF to 1.
3. An interrupt is generated based on CCER_CC1IE.
4. Generate DMA requests based on DIER_CC1DE and CR2_CCDS.

Configure CCMRx_OCxPE to allow the CCRx register to be set with or without a shadow register. When CCMRx_OCxPE is 0, software modifcations to CCRx take effect in real-time, enabling custom waveform output by modifying the next matching CCRx in each interrupt.

The outputs of CH and CHxN only take effect after BDTR_MOE is set to 1.

9.1.3.10 Basic PWM Output

Using output compare mode, the timer can generate multiple PWM outputs with controllable period, duty cycle, and phase. The period of the PWM output is determined by ARR, while the duty cycle is determined by CCRx. There are various modes for PWM output, independently selected by each channel's CCMRx_OCxM. The most basic single-channel PWM output requires only one channel and can be achieved using the basic PWM mode. More complex PWM signals or PWM combinations require multiple channels and careful allocation of each channel's PWM mode and CCRx.

In the basic PWM mode, the counter value CNT is compared with CCRx , generating a comparison output signal OCxREF that contains either a valid level or an invalid level based on the current counting direction of the counter. The polarity of the valid level can be configured through CCER_CCxP , and the output CHx is enabled according to the registers CCER_CCxE and BDTR_MOE . The PWM output takes effect only after setting BDTR_MOE to 1 .

For instance, in the increment counting mode, if CCMR1_OC1M and CCMR1_OC2M are configured to 0110(0x6) , the PWM output is illustrated in Figure 9-2. When the counter value CNT is less than CCR1/2 , a high level is output; otherwise, a low level is output.

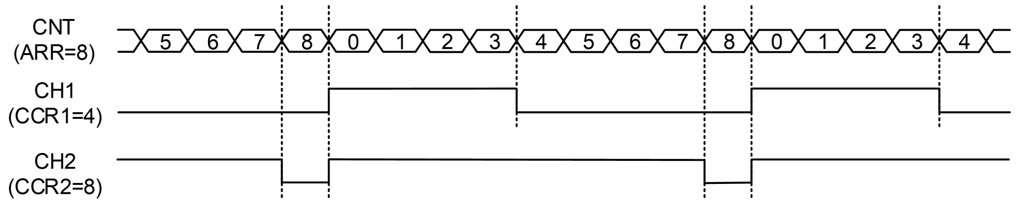


Figure 9-2: PWM output in increment counting mode

In center-aligned counting mode, if CCMR1_OC1M and CCMR1_OC2M are configured to 0110(0x6) , the PWM output is illustrated in Figure9-3 . When the increment stage counting value CNT is less than CCR1/2 , the output is high; otherwise, it is low. When the decrement stage counting value CNT is greater than CCR1/2 , the output is low; otherwise, it is high.

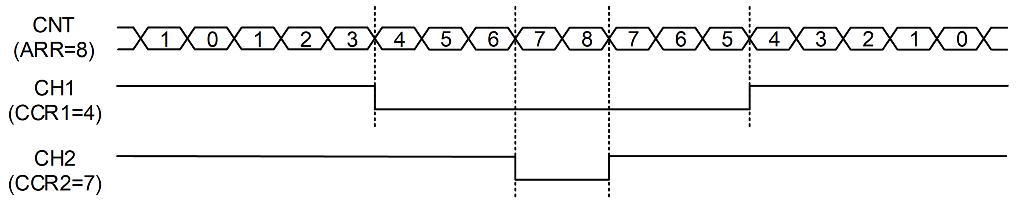


Figure 9-3: PWM output in center-aligned counting mode

9.1.3.11 Asymmetric PWM Output

In asymmetric PWM mode, there is a programmable phase shift between the two PWM signals generated. This mode is restricted to when the counter is in center-aligned mode. The frequencies of the two PWM signals are identical, determined by the value of ARR , while the duty cycle and phase shift are each determined by a pair of CCRx Registers. Each PWM output occupies two CCRx Registers, controlling the behavior during increment and decrement counting periods, allowing the rising and falling edges of the PWM to be configured independently. CCR1 and CCR2 jointly control the output of CH1/2 , while CCR3 and CCR4 jointly control the output of CH3/4 .

CH1/2 and CH3/4can independently select different asymmetric PWMmodes by configuring CCMRx_OCxM to 1110(0xe) or 1111(0xf).

If CCMR1_OC1M and CCMR2_OC3M are configured to 1110(0xe) , the PWM output is illustrated in Figure9-4 . During the increasing phase (0->ARR-1) , when the count value CNT is less than CCR1/3 , a high-level output is generated; otherwise, a low-level output is produced. During the decreasing phase (ARR->1) , when the count value CNT is greater than CCR2/4 , a low-level output is generated; otherwise, a high-level output is produced.

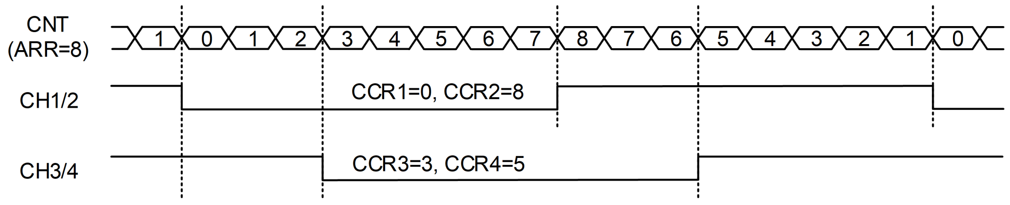


Figure 9-4: Asymmetric PWM Output

9.1.3.12 Combined PWM Output

In combined PWM mode, there is a programmable delay and phase shift between the two PWM signals generated. The counter can operate in incrementing, decrementing, or center-aligned mode, and the frequency of the two PWM signals is the same, determined by the value of ARR. The duty cycle and phase shift are each determined by a pair of CCRx Registers. Each output PWM utilizes two CCRx Registers, formed by the logical AND or OR combination of two basic PWM output waveforms. CCR1 and CCR2 jointly control the output of CH1/2, while CCR3 and CCR4 jointly control the output of CH3/4.

CH1/2 and CH3/4 can independently select different combinations of PWM modes, configuring CCMRx_OCxM to 1100(0xc) or 1101(0xd). When CH1 or CH3 is configured to the combined PWM mode 1100(0xc), CH2 or CH4 must be configured to 0111(0x7) or 1101(0xd) or 1111(0xf). When CH1 or CH3 is configured to the combined PWM mode 1101(0xd), CH2 or CH4 must be configured to 0110(0x6) or 1100(0xc) or 1110(0xe).

For example, if CCMR1_OC1M is configured to 1101(0xd),CCMR1_OC2M to 0110(0x6),CCMR2_OC3M to 1100(0xc),CCMR2_OC4M is 0111(0x7) , the PWM output is illustrated in Figure9-5 . When the count value CNT is less than CCR1/4 , OC1REF/OC4REF is low; otherwise, it is high. When the count value CNT is less than CCR2/3 , OC2REF/OC3REF is high; otherwise, it is low. The CH1 output represents the logical AND of OC1REF and OC2REF . The CH3 output represents the logical OR of OC3REF and OC4REF .

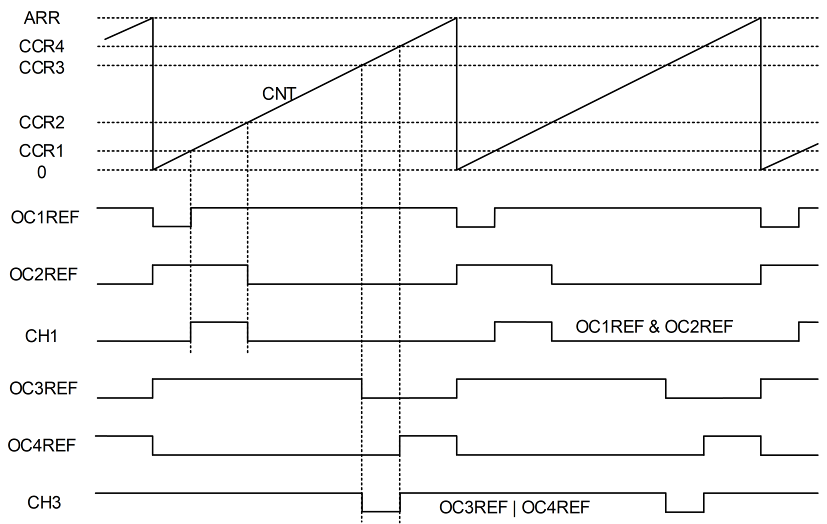


Figure 9-5: Combined PWM Output

9.1.3.13 Complementary PWM output with dead time

ATIM can output two phase-opposite complementary PWM signals CHx and CHxN , and insert a specific delay at the edges of the transitions of the two signals. This delay is commonly referred to as dead time, and users must adjust the dead time according to the characteristics of the devices connected to the output (such as the inherent delay of level shifters, delays caused by switching devices, etc.).

The channels 1/2/3 of ATIM can each output a set of complementary PWM signals, with a maximum of 3 sets outputting a total of 6 complementary signals simultaneously. Each output can independently select its output polarity through CCER_CCxP and CCER_CCxNP. Configuring BDTR_MOE and the corresponding channel's CCER_CCxE and CCER_CCxNE to 1 enables the complementary output.

An example of complementary output is illustrated in Figure9-6. The rising edge of CHx has a dead time delay relative to the rising edge of the reference output OCxREF generated by that channel, while the rising edge of CHxN has a dead time delay relative to the falling edge of OCxREF for that channel.

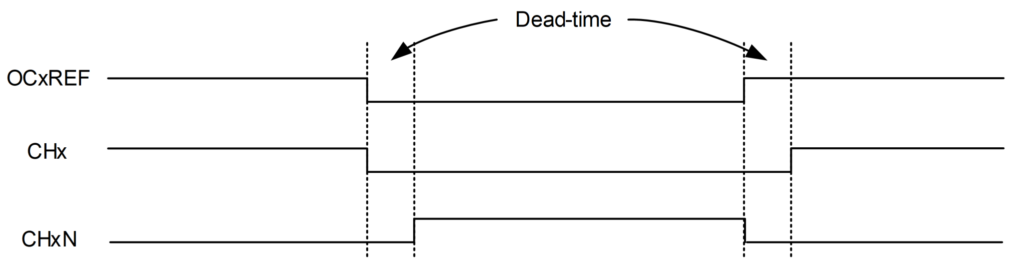


Figure 9-6: Complementary PWM output with dead time

The dead time is adjustable within a certain range. When BDTR_DTPSC is 0, the dead time is calculated as BDTR_DTG multiplied by the PCLK period. When BDTR_DTPSC is 1, the dead time is calculated as BDTR_DTG multiplied by 16 times the PCLK period. If PCLK is 120MHz, the adjustable range of dead time is 0 to 136µs

9.1.3.14 Emergency Cut-off

The purpose of the cut-off function is to protect the power switches driven by the PWM signal generated by ATIM. The two cut-off inputs BKIN and BKIN 2 are typically connected to the fault outputs of the power stage and three-phase inverter. When activated, the cut-off circuit will disable the PWM output and force it into a predefined safe state. It is also possible to select certain internal chip events to trigger the output shutdown. BKIN can force the output to a predefined level (active or inactive) after the dead time duration. BKIN is capable of forcing the output into an inactive state.

The output enable signal and output level during the break period depend on multiple control bits:BDTR_MOE allows the output to be enabled / disabled via software;BDTR_OSSI defines whether the timer will keep the output in an invalid state or release control to the GPIO controller (typically placing it in high-impedance mode);CR2_OISx/OISxN sets the output to a shutdown level (valid or invalid).

After reset, the break function of ATIMis disabled, and BDTR_MOEis at a low level. Setting BDTR_BKE/BKE2to 1enables the break function.

The polarity of the break input can be selected by configuring BDTR_BKP/BKP2. The software can also configure EGR_BG/B2G to generate break events, independent of the values of BDTR_BKE/BKE2.

The internal circuit of the open circuit also implements write protection to ensure application security. Through this feature, users can freeze multiple parameter configurations, such as dead time duration, output polarity, and state when disabled,PWM mode, open circuit enable, and polarity, among others. This feature is achieved by writing to the AF1_LOCK register, allowing selection from 3 levels of protection.

9.1.3.15 6 Step PWM

6 Step PWM requires the simultaneous switching of each channel's PWM mode at a specific moment during the PWM output process, which can be accomplished through the commutation event (COM) of ATIM . When channels utilize complementary output, CCMRx_OCxM , CCER_CCxE , and CCER_CCxNE feature a preload mechanism. Users can pre-program the configuration for the next step, and when a commutation event occurs, the preload register will transfer to the Shadow Register, simultaneously altering the configuration of all channels. The COM can be generated by software by setting EGR_COM to 1 , or it can be generated by hardware on the rising edge of the input trigger signal. When a commutation event occurs, SR_COMIF will be set to 1 . At this point, if DIER_COMIE is 1 , an interrupt will be generated; if DIER_COMDE is 1 , a DMA request will be generated.

9.1.3.16 One Pulse Mode

Set CR1_OPM Write 1 Enables the One Pulse Mode. In this mode, once the counter is started, it will automatically stop counting upon an Update Event. This mode can be utilized for single counting or can be triggered by an excitation signal to generate a pulse with a programmable width after a programmable delay.

For example, to achieve the functionality where a single pulse of a specific width is generated on CH1 after a certain delay when a rising edge is detected on the CH2 input pin, the configuration method is as follows:

1. CCMR1_CC2S=01 to map TI2FP2to channel 2.
2. CCER_CCxP and CCER_CCxNP are set to 0, with TI2FP2 responding to the positive edge change of CH2.
3. SMCR_TS=110(0x6) configures TI2FP2 to trigger from the mode controller's TRGI.
4. SMCR_SMS=110(0x6) configures the mode controller to trigger mode, enabling counting after the trigger.
5. Configure ARRand CCR1according to the required time delay and pulse width, thereby defning the time delay and pulse width.
6. CCMR1_OC1M=0111(0x7) configures for positive pulse PWM.
7. CR1_OPM=1 , a single trigger generates only one pulse.
8. EGR_UG=1 , manually refresh the ARRand CCR1registers.

In trigger mode, it is not necessary to manually enable CR1_CEN; once a trigger signal is detected, the counter will be automatically enabled.

9.1.3.17 Encoder Interface Mode

In Encoder Interface Mode, channels 1 and 2 can be used to connect external quadrature encoders, converting the signals from the external encoder into changes in the timer's count value, thereby determining the operational status of the external encoder.

If the counter counts only on the rising edge of CH1 , configure SMCR_SMS to 0001 ; if the counter counts only on the rising edge of CH2 , configure SMCR_SMS to 0010(0x2) ; if the counter counts on the rising edges of both CH1 and CH2 , configureSMCR_SMS to 0011(0x3) . CCER_CC1P/CC2P is used to select the polarity of CH1 and CH2 . If necessary, the input flter can also be programmed. The signal conversion sequence of the two inputs will generate counting pulses and direction signals; based on this signal conversion sequence, the counter will increment or decrement accordingly, while the hardware will modify CR1_DIR as needed.

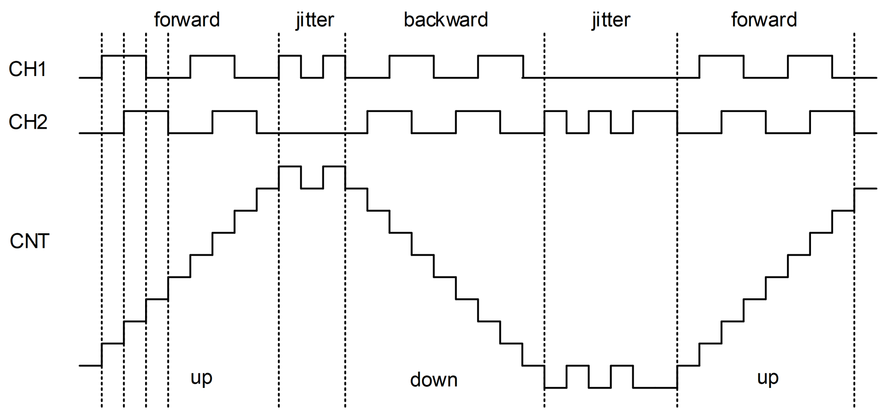
In Encoder Interface Mode, the counting events of the counter are the decoded outputs of the quadrature encoder interface. The counter only performs continuous counting between 0 and ARR (incrementing from 0 to ARR or decrementing from ARR to 0 , depending on the specific counting direction). Therefore, it is necessary to configure ARR before starting. Similarly, the capture, compare, repeat counter, and trigger output functions continue to operate normally. In this mode, the counter is automatically modifed based on the speed and direction of the quadrature encoder, ensuring that its content always represents the position of the encoder. The counting direction corresponds to the rotation direction of the connected sensor. The table below summarizes the possible combinations (assuming CH1 and CH2 do not switch simultaneously).

SMCR_SMS	Condition	CH1 Rising Edge	CH1 Falling Edge	CH2 Rising Edge	CH2 Falling Edge
0001 or 0011	CH2=0	Increment	Decrement	/	/
	CH2=1	Decrement	Increment	/	/
0010 or 0011	CH1=0	/	/	Decrement	Increment
	CH1=1	/	/	Increment	Decrement

The following diagram illustrates how the counter counts based on the signal changes from the quadrature encoder, configured as follows:

CCMR1_CC1S=01 （CH1 mapped to channel 1),CCMR2_CC2S=01(CH2mapped to channel 2),

CCER_CC1P/CC1NP/CC2P/CC2NP=0， SMCR_SMS=0011(0x3)， CR1_CEN=1 .



9.1.3.18 Timer Synchronization

Multiple timers can be interconnected in a master-slave configuration to achieve Timer Synchronization, enabling functionalities such as multi-level frequency division, simultaneous start, and gated counting.

By configuring the master mode timer's TRGO to an Update Event (CR2_MMS=010) and connecting it to another timer set as an external clock slave mode (SMCR_SMS = 0111) , cascading counting of timers can be accomplished. In this scenario, the master mode timer serves as a pre-divider for the slave mode timer, with the total counting bit width equal to the sum of the individual bit widths of both timers. The total counting bit width is equal to the sum of the individual bit widths of both timers.

By configuring the master mode timer's TRGO to Count Enable (CR2_MMS=001) and connecting it to another timer set for Trigger Slave Mode (SMCR_SMS=0110) , Timer Synchronization can be achieved, aligning the start times of multiple timers.

Configure the main mode timer's TRGO to output a comparison signal (CR2_MMS=100) , connected to another timer set to gated slave mode (SMCR_SMS=0101), to achieve gated PWM output. The main mode timer can modulate the PWM carrier output from the slave mode timer.

9.1.3.19 Notifcation Mechanism

The ATIM can generate various notifcation mechanisms, including interrupts, DMA requests, and PTC triggers. The events that can trigger notifcations primarily include update events, trigger events, comparator matches, input captures, interrupt inputs, and commutation events. The DIER register controls whether various events generate interrupts and DMA requests. The status of each event can be checked in the SR register.

9.1.4 ATIM Register

ATIM1 base address is 0x50004000.

Table 9-1: **ATIM Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	TIM control register 1
[31:12]			RSVD	
[11]	rw	1'h0	UIFREMAP	UIF status bit remapping 0: No remapping. UIF status bit is not copied to CNT register bit 31 1: Remapping enabled. UIF status bit is copied to CNT register bit 31.
[10:8]			RSVD	
[7]	rw	1'h0	ARPE	Auto-reload preload enable 0: ARR register is not buffered 1: ARR register is buffered
[6:5]	rw	2'h0	CMS	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set both when the counter is counting up or down.
[4]	rw	1'h0	DIR	Direction 0: Counter used as upcounter 1: Counter used as downcounter
[3]	rw	1'h0	OPM	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	URS	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: Counter overflow/underflow Setting the UG bit Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
[1]	rw	1'h0	UDIS	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: Counter overflow/underflow Setting the UG bit Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
[0]	rw	1'h0	CEN	Counter enable 0: Counter disabled 1: Counter enabled External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware. CEN is cleared automatically in one-pulse mode, when an update event occurs.
0x04			**CR2**	TIM control register 2
[31:19]			RSVD	
[18]	rw	1'h0	OIS6	Output Idle state 6 (OC6 output)
[17]			RSVD	
[16]	rw	1'h0	OIS5	Output Idle state 5 (OC5 output)
[15]			RSVD	
[14]	rw	1'h0	OIS4	Output Idle state 4 (OC4 output)
[13]	rw	1'h0	OIS3N	Output Idle state 3 (OC3N output)
[12]	rw	1'h0	OIS3	Output Idle state 3 (OC3 output)
[11]	rw	1'h0	OIS2N	Output Idle state 2 (OC2N output)
[10]	rw	1'h0	OIS2	Output Idle state 2 (OC2 output)
[9]	rw	1'h0	OIS1N	Output Idle state 1 (OC1N output) 0: OC1N=0 after a dead-time when MOE=0 1: OC1N=1 after a dead-time when MOE=0 This bit, as well as other OISxN, can not be modified as long as LOCK level 1, 2 or 3 has been programmed
[8]	rw	1'h0	OIS1	Output Idle state 1 (OC1 output) 0: OC1=0 (after a dead-time if OC1N is implemented) when MOE=0 1: OC1=1 (after a dead-time if OC1N is implemented) when MOE=0 This bit, as well as other OISx, can not be modified as long as LOCK level 1, 2 or 3 has been programmed
[7]	rw	1'h0	TI1S	TI1 selection 0: The CH1 pin is connected to TI1 input 1: The CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[6:4]	rw	3’h0	MMS	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter enable signal, CNT_EN, is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected. 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO) 100: Compare - OC1REFC signal is used as trigger output (TRGO) 101: Compare - OC2REFC signal is used as trigger output (TRGO) 110: Compare - OC3REFC signal is used as trigger output (TRGO) 111: Compare - OC4REFC signal is used as trigger output (TRGO)
[3]	rw	1’h0	CCDS	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
[2]	rw	1’h0	CCUS	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit or when an edge occurs on TRGI after Trigger selection This bit acts only on channels that have a complementary output.
[1]			RSVD	
[0]	rw	1’h0	CCPC	Capture/compare preloaded control 0: CCxE, CCxNE and OCxM bits are not preloaded 1: CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs (COMG bit set or edge detected on TRGI after Trigger selection, depending on the CCUS bit). This bit acts only on channels that have a complementary output.
0x08			**SMCR**	TIM slave mode control register
[31:20]			RSVD	

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19:16]	rw	4'h0	SMS	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input. 0000: Slave mode disabled. 0001: Encoder mode 1 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 0010: Encoder mode 2 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 0011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 0100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 0101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 0111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. 1000: Combined reset + trigger mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers and starts the counter.
[15]	rw	1'h0	ETP	External trigger polarity This bit selects whether ETR or ETR is used for trigger operations 0: ETR is non-inverted, active at high level or rising edge 1: ETR is inverted, active at low level or falling edge
[14]	rw	1'h0	ECE	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.
[13:12]	rw	2'h0	ETPS	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of CK_INT frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:8]	rw	4'h0	ETF	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fCLK 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[7]	rw	1'h0	MSM	Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
[6:4]	rw	3'h0	TS	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF)
[3:0]			RSVD	
0x0C			**DIER**	TIM DMA/Interrupt enable register
[31:18]			RSVD	
[17]	rw	1'h0	CC6IE	Capture/Compare 6 interrupt enable 0: CC6 interrupt disabled. 1: CC6 interrupt enabled
[16]	rw	1'h0	CC5IE	Capture/Compare 5 interrupt enable 0: CC5 interrupt disabled. 1: CC5 interrupt enabled
[15]			RSVD	
[14]	rw	1'h0	TDE	Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled.
[13]	rw	1'h0	COMDE	COM DMA request enable 0: COM DMA request disabled 1: COM DMA request enabled

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[12]	rw	1'h0	CC4DE	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled. 1: CC4 DMA request enabled
[11]	rw	1'h0	CC3DE	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled. 1: CC3 DMA request enabled.
[10]	rw	1'h0	CC2DE	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
[9]	rw	1'h0	CC1DE	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
[8]	rw	1'h0	UDE	Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled
[7]	rw	1'h0	BIE	Break interrupt enable 0: Break interrupt disabled 1: Break interrupt enabled
[6]	rw	1'h0	TIE	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
[5]	rw	1'h0	COMIE	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
[4]	rw	1'h0	CC4IE	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled. 1: CC4 interrupt enabled
[3]	rw	1'h0	CC3IE	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled. 1: CC3 interrupt enabled
[2]	rw	1'h0	CC2IE	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled. 1: CC2 interrupt enabled.
[1]	rw	1'h0	CC1IE	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled. 1: CC1 interrupt enabled
[0]	rw	1'h0	UIE	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled
0x10			**SR**	TIM status register
[31:18]			RSVD	
[17]	rw0c	1'h0	CC6IF	Compare 6 interrupt flag
[16]	rw0c	1'h0	CC5IF	Compare 5 interrupt flag
[15]			RSVD	
[14]			RSVD	
[13]	rw0c	1'h0	SBIF	System Break interrupt flag This flag is set by hardware as soon as the system break input goes active. It can be cleared by software if the system break input is not active. This flag must be reset to re-start PWM operation. 0: No break event occurred. 1: An active level has been detected on the system break input. An interrupt is generated if BIE=1 in the DIER register.
[12]	rw0c	1'h0	CC4OF	Capture/Compare 4 overcapture flag

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw0c	1'h0	CC3OF	Capture/Compare 3 overcapture flag
[10]	rw0c	1'h0	CC2OF	Capture/Compare 2 overcapture flag
[9]	rw0c	1'h0	CC1OF	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in CCR1 register while CC1IF flag was already set
[8]	rw0c	1'h0	B2IF	Break 2 interrupt flag This flag is set by hardware as soon as the break 2 input goes active. It can be cleared by software if the break 2 input is not active. 0: No break event occurred. 1: An active level has been detected on the break 2 input. An interrupt is generated if BIE=1 in the DIER register.
[7]	rw0c	1'h0	BIF	Break interrupt flag This flag is set by hardware as soon as the break input goes active. It can be cleared by software if the break input is not active. 0: No break event occurred. 1: An active level has been detected on the break input. An interrupt is generated if BIE=1 in the DIER register.
[6]	rw0c	1'h0	TIF	Trigger interrupt flag This flag is set by hardware on trigger event. It is set when the counter starts or stops when gated mode is selected. It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
[5]	rw0c	1'h0	COMIF	COM interrupt flag This flag is set by hardware on COM event (when Capture/compare Control bits - CCxE, CCxNE, OCxM - have been updated). It is cleared by software. 0: No COM event occurred. 1: COM interrupt pending.
[4]	rw0c	1'h0	CC4IF	Capture/Compare 4 interrupt flag
[3]	rw0c	1'h0	CC3IF	Capture/Compare 3 interrupt flag
[2]	rw0c	1'h0	CC2IF	Capture/Compare 2 interrupt flag
[1]	rw0c	1'h0	CC1IF	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value and in retriggerable one pulse mode. It is cleared by software. 0: No match. 1: The content of the counter CNT has matched the content of the CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the CCR1 register. 0: No input capture occurred. 1: The counter value has been captured in CCR1 register.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw0c	1'h0	UIF	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update if repetition counter = 0) and if UDIS=0 in the CR1 register. - When CNT is reinitialized by software using the UG bit in EGR register, if URS=0 and UDIS=0 in the CR1 register. - When CNT is reinitialized by a trigger event, if URS=0 and UDIS=0 in the CR1 register.
0x14			**EGR**	Event generation register
[31:9]			RSVD	
[8]	w	1'h0	B2G	Break 2 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break 2 event is generated. MOE bit is cleared and B2IF flag is set. Related interrupt can occur if enabled.
[7]	w	1'h0	BG	Break generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A break event is generated. MOE bit is cleared and BIF flag is set. Related interrupt or DMA transfer can occur if enabled.
[6]	w	1'h0	TG	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in SR register. Related interrupt or DMA transfer can occur if enabled.
[5]	w	1'h0	COMG	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits This bit acts only on channels having a complementary output.
[4]	w	1'h0	CC4G	Capture/compare 4 generation
[3]	w	1'h0	CC3G	Capture/compare 3 generation
[2]	w	1'h0	CC2G	Capture/compare 2 generation
[1]	w	1'h0	CC1G	Capture/compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	w	1'h0	UG	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. The prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (ARR) if DIR=1 (downcounting).
0x18			**CCMR1**	TIM capture/compare mode register 1
[31:28]	rw	4'h0	OC2M	Output compare 2 mode
[27]	rw	1'h0	OC2PE	Output compare 2 preload enable
[26:25]			RSVD	
[24]	rw	1'h0	OC2CE	Output compare 2 clear enable

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:20]	rw	4'h0	OC1M	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen - The comparison between the output compare register CCR1 and the counter CNT has no effect on the outputs. 0001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter CNT matches the capture/compare register 1 (CCR1). 0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter CNT matches the capture/compare register 1 (CCR1). 0011: Toggle - OC1REF toggles when CNT=CCR1. 0100: Force inactive level - OC1REF is forced low. 0101: Force active level - OC1REF is forced high. 0110: PWM mode 1 - In upcounting, channel 1 is active as long as CNT<CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF=0) as long as CNT>CCR1 else active (OC1REF=1). 0111: PWM mode 2 - In upcounting, channel 1 is inactive as long as CNT<CCR1 else active. In downcounting, channel 1 is active as long as CNT>CCR1 else inactive. 1000: Retriggerable OPM mode 1 - In up-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. In down-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. 1001: Retriggerable OPM mode 2 - In up-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 2 and the channels becomes inactive again at the next update. In down-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes active again at the next update. 1010: Reserved, 1011: Reserved, 1100: Combined PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC is the logical OR between OC1REF and OC2REF. 1101: Combined PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC is the logical AND between OC1REF and OC2REF. 1110: Asymmetric PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down. 1111: Asymmetric PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down. These bits can not be modified as long as LOCK level 3 has been programmed and CC1S=00 (the channel is configured in output). On channels having a complementary output, this bit field is preloaded. If the CCPC bit is set in the CR2 register then the OC1M active bits take the new value from the preloaded bits only when a COM event is generated.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19]	rw	1'h0	OC1PE	Output compare 1 preload enable 0: Preload register on CCR1 disabled. CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on CCR1 enabled. Read/Write operations access the preload register. CCR1 preload value is loaded in the active register at each update event. These bits can not be modified as long as LOCK level 3 has been programmed and CC1S='00' (the channel is configured in output).
[18:17]			RSVD	
[16]	rw	1'h0	OC1CE	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input
[15:12]	rw	4'h0	IC2F	Input capture 2 filter
[11:10]	rw	2'h0	IC2PSC	Input capture 2 prescaler
[9:8]	rw	2'h0	CC2S	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (SMCR register)
[7:4]	rw	4'h0	IC1F	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fCLK 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[3:2]	rw	2'h0	IC1PSC	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=0 (CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[1:0]	rw	2'h0	CC1S	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
0x1C			**CCMR2**	TIM capture/compare mode register 2
[31:28]	rw	4'h0	OC4M	Output compare 4 mode
[27]	rw	1'h0	OC4PE	Output compare 4 preload enable
[26:25]			RSVD	
[24]	rw	1'h0	OC4CE	Output compare 4 clear enable
[23:20]	rw	4'h0	OC3M	Output compare 3 mode
[19]	rw	1'h0	OC3PE	Output compare 3 preload enable
[18:17]			RSVD	
[16]	rw	1'h0	OC3CE	Output compare 3 clear enable
[15:12]	rw	4'h0	IC4F	Input capture 4 filter
[11:10]	rw	2'h0	IC4PSC	Input capture 4 prescaler
[9:8]	rw	2'h0	CC4S	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
[7:4]	rw	4'h0	IC3F	Input capture 3 filter
[3:2]	rw	2'h0	IC3PSC	Input capture 3 prescaler
[1:0]	rw	2'h0	CC3S	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
0x20			**CCER**	Capture/Compare enable register
[31:22]			RSVD	
[21]	rw	1'h0	CC6P	Capture/Compare 6 output Polarity.
[20]	rw	1'h0	CC6E	Capture/Compare 6 output enable.
[19]			RSVD	
[18]			RSVD	
[17]	rw	1'h0	CC5P	Capture/Compare 5 output Polarity.
[16]	rw	1'h0	CC5E	Capture/Compare 5 output enable.
[15]	rw	1'h0	CC4NP	Capture/Compare 4 complementary output polarity
[14]			RSVD	
[13]	rw	1'h0	CC4P	Capture/Compare 4 output Polarity.
[12]	rw	1'h0	CC4E	Capture/Compare 4 output enable.
[11]	rw	1'h0	CC3NP	Capture/Compare 3 complementary output polarity
[10]	rw	1'h0	CC3NE	Capture/Compare 3 complementary output enable
[9]	rw	1'h0	CC3P	Capture/Compare 3 output Polarity.
[8]	rw	1'h0	CC3E	Capture/Compare 3 output enable.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h0	CC2NP	Capture/Compare 2 complementary output polarity
[6]	rw	1'h0	CC2NE	Capture/Compare 2 complementary output enable
[5]	rw	1'h0	CC2P	Capture/Compare 2 output Polarity.
[4]	rw	1'h0	CC2E	Capture/Compare 2 output enable.
[3]	rw	1'h0	CC1NP	Capture/Compare 1 complementary output polarity CC1 channel configured as output: 0: OC1N active high. 1: OC1N active low. CC1 channel configured as input: This bit is used in conjunction with CC1P to define the polarity of TI1FP1 and TI2FP1. Refer to CC1P description. On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the CR2 register then the CC1NP active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit as well as other CCxNP is not writable as soon as LOCK level 2 or 3 has been programmed and CC1S=00 (channel configured as output).
[2]	rw	1'h0	CC1NE	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the CR2 register then the CC1NE active bit takes the new value from the preloaded bit only when a Commutation event is generated.
[1]	rw	1'h0	CC1P	Capture/Compare 1 output Polarity. CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge. Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode). 01: inverted/falling edge. Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode). 10: reserved, do not use this configuration. 11: noninverted/both edges. Circuit is sensitive to both TIxFP1 rising and falling edges (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode). This configuration must not be used for encoder mode. On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the CR2 register then the CC1P active bit takes the new value from the preloaded bit only when a Commutation event is generated. This bit as well as other CCxP is not writable as soon as LOCK level 2 or 3 has been programmed.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	CC1E	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin depending on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (CCR1) or not. 0: Capture disabled. 1: Capture enabled. On channels having a complementary output, this bit is preloaded. If the CCPC bit is set in the CR2 register then the CC1E active bit takes the new value from the preloaded bit only when a Commutation event is generated.
0x24			**CNT**	Counter
[31:0]	rw	32'h0	CNT	bit 30 to 0 is the lower bits of counter value bit 31 depends on IUFREMAP in CR1. If UIFREMAP = 1 this bit is a read-only copy of the UIF bit of the ISR register If UIFREMAP = 0 this bit is counter value bit 31
0x28			**PSC**	Prescaler
[31:16]			RSVD	
[15:0]	rw	16'h0	PSC	Prescaler value The counter clock frequency is fCLK/(PSC+1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of EGR register or through trigger controller when configured in ”reset mode”).
0x2C			**ARR**	Auto-reload register
[31:0]	rw	32'h0	ARR	Auto-reload value ARR is the value to be loaded in the actual auto-reload register.
0x30			**RCR**	Repetition counter register
[31:16]			RSVD	
[15:0]	rw	16'h0	REP	Repetition counter value These bits allow the user to set-up the update rate of the compare registers when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event, any write to the RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to the number of PWM periods in edge-aligned mode or the number of half PWM period in center-aligned mode..
0x34			**CCR1**	Capture/Compare register 1
[31:0]	rw	32'h0	CCR1	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1).
0x38			**CCR2**	Capture/Compare register 2

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32’h0	CCR2	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2).
0x3C			**CCR3**	Capture/Compare register 3
[31:0]	rw	32’h0	CCR3	Capture/Compare value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC3 output. If channel CC3is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3).
0x40			**CCR4**	Capture/Compare register 4
[31:0]	rw	32’h0	CCR4	Capture/Compare value 1. if CC4 channel is configured as output (CC4S bits): CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC4 output. 2. if CC4 channel is configured as input (CC4S bits in CCMR4 register): CCR4 is the counter value transferred by the last input capture 4 event (IC4).
0x44			**BDTR**	TIM break and dead-time register
[31]	rw	1’h0	OSSR	Off-state selection for Run mode This bit is used when MOE=1 on channels having a complementary output which are configured as outputs. OSSR is not implemented if no complementary output is implemented in the timer. 0: When inactive, OC/OCN outputs are disabled (the timer releases the output control, forces a Hi-Z state). 1: When inactive, OC/OCN outputs are enabled with their inactive level as soon as CCxE=1 or CCxNE=1 (the output is still controlled by the timer). This bit can not be modified as soon as the LOCK level 2 has been programmed.
[30]	rw	1’h0	OSSI	Off-state selection for Idle mode This bit is used when MOE=0 due to a break event or by a software write, on channels configured as outputs. 0: When inactive, OC/OCN outputs are disabled (the timer releases the output control, imposes a Hi-Z state). 1: When inactive, OC/OCN outputs are first forced with their inactive level then forced to their idle level after the deadtime. The timer maintains its control over the output. This bit can not be modified as soon as the LOCK level 2 has been programmed.
[29]	rw	1’h0	BK2BID	Break2 bidirectional

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[28]	rw	1’h0	BKBID	Break Bidirectional 0: Break input BRK in input mode 1: Break input BRK in bidirectional mode In the bidirectional mode (BKBID bit set to 1), the break input is configured both in input mode and in open drain output mode. Any active break event asserts a low logic level on the Break input to indicate an internal break event to external devices. This bit cannot be modified as long as LOCK level 1 has been programmed (LOCK bits in BDTR register).
[27]	rw	1’h0	BK2DSRM	Break2 Disarm
[26]	rw	1’h0	BKDSRM	Break Disarm 0: Break input BRK is armed 1: Break input BRK is disarmed This bit is cleared by hardware when no break source is active. The BKDSRM bit must be set by software to release the bidirectional output control (open-drain output in Hi-Z state) and then be polled it until it is reset by hardware, indicating that the fault condition has disappeared.
[25]	rw	1’h0	BK2P	BK2P: Break 2 polarity 0: Break input BRK2 is active low 1: Break input BRK2 is active high This bit cannot be modified as long as LOCK level 1 has been programmed.
[24]	rw	1’h0	BK2E	Break 2 enable This bit enables the complete break 2 protection. 0: Break2 function disabled 1: Break2 function enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
[23:20]	rw	4’h0	BK2F	Break 2 filter This bit-field defines the frequency used to sample BRK2 input and the length of the digital filter applied to BRK2. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK2 acts asynchronously 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8 This bit cannot be modified as long as LOCK level 1 has been programmed.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19:16]	rw	4’h0	BKF	Break filter This bit-field defines the frequency used to sample BRK input and the length of the digital filter applied to BRK. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, BRK acts asynchronously 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8 This bit cannot be modified as long as LOCK level 1 has been programmed.
[15]	rw	1’h0	MOE	Main output enable This bit is cleared asynchronously by hardware as soon as one of the break inputs is active (BRK or BRK2). It is set by software or automatically depending on the AOE bit. It is acting only on the channels which are configured in output. 0: In response to a break 2 event. OC and OCN outputs are disabled In response to a break event or if MOE is written to 0: OC and OCN outputs are disabled or forced to idle state depending on the OSSI bit. 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in CCER register).
[14]	rw	1’h0	AOE	Automatic output enable 0: MOE can be set only by software 1: MOE can be set by software or automatically at the next update event (if none of the break inputs BRK and BRK2 is active) This bit cannot be modified as long as LOCK level 1 has been programmed.
[13]	rw	1’h0	BKP	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high This bit cannot be modified as long as LOCK level 1 has been programmed.
[12]	rw	1’h0	BKE	Break enable This bit enables the complete break protection. 0: Break function disabled 1: Break function enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
[11]	rw	1’h0	DTPSC	Dead-time prescaler This bit-field enables dead-time prescaler. 0: dead-time is tCLK*(DTG+1) if DTG is not zero 1: dead-time is tCLK*(DTG+1)*16 if DTG is not zero This bit cannot be modified as long as LOCK level 1 has been programmed.
[10]			RSVD	

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[9:0]	rw	10'h0	DTG	Dead-time generator setup This bit-field, together with DTPSC, defines the duration of the dead-time inserted between the complementary outputs. If DTG=0, dead-time is disabled. Example if tCLK=8.33ns (120MHz), dead-time possible values are: 16.67ns to 8533.33 ns by 8.33 ns steps if DTPSC=0, 266.67ns to 136.53 us by 133.33 ns steps if DTPSC=1 This bit cannot be modified as long as LOCK level 1 has been programmed.
0x54			**CCMR3**	TIM capture/compare mode register 3
[31:28]	rw	4'h0	OC6M	Output compare 6 mode
[27]	rw	1'h0	OC6PE	Output compare 6 preload enable
[26:25]			RSVD	
[24]	rw	1'h0	OC6CE	Output compare 6 clear enable
[23:20]	rw	4'h0	OC5M	Output compare 5 mode
[19]	rw	1'h0	OC5PE	Output compare 5 preload enable
[18:17]			RSVD	
[16]	rw	1'h0	OC5CE	Output compare 5 clear enable
[15]	rw	1'h0	GC5C3	Group Channel 5 and Channel 3 Distortion on Channel 3 output: 0: No effect of OC5REF on OC3REFC 1: OC3REFC is the logical AND of OC3REFC and OC5REF This bit can either have immediate effect or be preloaded and taken into account after an update event (if preload feature is selected in TIMxCCMR2).
[14]	rw	1'h0	GC5C2	Group Channel 5 and Channel 2 Distortion on Channel 2 output: 0: No effect of OC5REF on OC2REFC 1: OC2REFC is the logical AND of OC2REFC and OC5REF This bit can either have immediate effect or be preloaded and taken into account after an update event (if preload feature is selected in TIMxCCMR1).
[13]	rw	1'h0	GC5C1	Group Channel 5 and Channel 1 Distortion on Channel 1 output: 0: No effect of OC5REF on OC1REFC5 1: OC1REFC is the logical AND of OC1REFC and OC5REF This bit can either have immediate effect or be preloaded and taken into account after an update event (if preload feature is selected in TIMxCCMR1).
[12:0]			RSVD	
0x58			**CCR5**	Capture/Compare register 5
[31:0]	rw	32'h0	CCR5	Capture/Compare 5 value CCR5 is the value to be loaded in the actual capture/compare 5 register (preload value). It is loaded permanently if the preload feature is not selected in the CCMR3 register (bit OC5PE). Else the preload value is copied in the active capture/compare 5 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signaled on OC5 output.
0x5C			**CCR6**	Capture/Compare register 6
[31:0]	rw	32'h0	CCR6	Capture/Compare 6 value CCR6 is the value to be loaded in the actual capture/compare 6 register (preload value). It is loaded permanently if the preload feature is not selected in the CCMR3 register (bit OC6PE). Else the preload value is copied in the active capture/compare 6 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signaled on OC6 output.
0x60			**AF1**	Alternate function option register

Continued on the next page...

Table 9-1: **ATIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:30]	rw	2'h0	LOCK	Lock configuration These bits offer a write protection against software errors. 00: LOCK OFF - No bit is write protected. 01: LOCK Level 1 = OISx and OISxN bits in CR2 register, BK2BID, BKBID, BK2DSRM, BKDSRM, BK2P, BK2E, BK2F[3:0], BKF[3:0], AOE, BKP, BKE, OSSI, OSSR, DTPSC and DTG bits in BDTR register, AF1 register and AF2 register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR and OSSI bits can no longer be written. 11: LOCK Level 3 = LOCK Level 2 + CC Control bits (OCxM and OCxPE bits in CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written. The LOCK bits can be written to non-zero only once after reset.
[29:16]			RSVD	
[15:14]	rw	2'h0	ETRSEL	ETR source selection 00: ETR input is connected to I/O 01: LPCOMP output1 (if LPCOMP integrated) 10: LPCOMP output2 (if LPCOMP integrated) 11: ETR input is connected to I/O This bit cannot be modified as long as LOCK level 1 has been programmed.
[13:12]			RSVD	
[11]	rw	1'h0	BKCMP2P	BRK LPCOMP output2 polarity This bit selects the LPCOMP output2 sensitivity (if LPCOMP integrated). It must be programmed together with the BKP polarity bit. 0: LPCOMP output2 is active high 1: LPCOMP output2 is active low This bit cannot be modified as long as LOCK level 1 has been programmed.
[10]	rw	1'h0	BKCMP1P	BRK LPCOMP output1 polarity This bit selects the LPCOMP output1 sensitivity (if LPCOMP integrated). It must be programmed together with the BKP polarity bit. 0: LPCOMP output1 is active high 1: LPCOMP output1 is active low This bit cannot be modified as long as LOCK level 1 has been programmed.
[9]	rw	1'h0	BKINP	BRK BKIN input polarity This bit selects the BKIN input sensitivity. It must be programmed together with the BKP polarity bit. 0: BKIN input is active high 1: BKIN input is active low This bit cannot be modified as long as LOCK level 1 has been programmed.
[8:3]			RSVD	
[2]	rw	1'h0	BKCMP2E	BRK LPCOMP output2 enable This bit enables the LPCOMP output2 (if LPCOMP integrated) for the timer's BRK input. LPCOMP output2 is 'ORed' with the other BRK sources. 0: LPCOMP output2 disabled 1: LPCOMP output2 enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
[1]	rw	1'h0	BKCMP1E	BRK LPCOMP output1 enable This bit enables the LPCOMP output1 (if LPCOMP integrated) for the timer's BRK input. LPCOMP output1 is 'ORed' with the other BRK sources. 0: LPCOMP output1 disabled 1: LPCOMP output1 enabled This bit cannot be modified as long as LOCK level 1 has been programmed.

Continued on the next page...

Table 9-1: ATIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1’h0	BKINE	BRK BKIN input enable This bit enables the BKIN input. BKIN input is ’Ored’ with the other BRK sources. 0: BKIN input disabled 1: BKIN input enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
0x64			**AF2**	Alternate function option register 2
[31:12]			RSVD	
[11]	rw	1’h0	BK2CMP2P	BRK2 LPCOMP output2 polarity This bit selects the LPCOMP output2 sensitivity (if LPCOMP integrated). It must be programmed together with the BK2P polarity bit. 0: LPCOMP output2 is active high 1: LPCOMP output2 is active low This bit cannot be modified as long as LOCK level 1 has been programmed.
[10]	rw	1’h0	BK2CMP1P	BRK2 LPCOMP output1 polarity This bit selects the LPCOMP output1 sensitivity (if LPCOMP integrated). It must be programmed together with the BK2P polarity bit. 0: LPCOMP output1 is active high 1: LPCOMP output1 is active low This bit cannot be modified as long as LOCK level 1 has been programmed.
[9]	rw	1’h0	BK2INP	BRK2 BKIN2 input polarity This bit selects the BKIN2 input sensitivity. It must be programmed together with the BK2P polarity bit. 0: BKIN2 input is active low 1: BKIN2 input is active high This bit cannot be modified as long as LOCK level 1 has been programmed.
[8:3]			RSVD	
[2]	rw	1’h0	BK2CMP2E	BRK2 LPCOMP output2 enable This bit enables the LPCOMP output2 (if LPCOMP integrated) for the timer’s BRK2 input. LPCOMP output2 is ’ORed’ with the other BRK2 sources. 0: LPCOMP output2 disabled 1: LPCOMP output2 enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
[1]	rw	1’h0	BK2CMP1E	BRK2 LPCOMP output1 enable This bit enables the LPCOMP output1 (if LPCOMP integrated) for the timer’s BRK2 input. LPCOMP output1 is ’ORed’ with the other BRK2 sources. 0: LPCOMP output1 disabled 1: LPCOMP output1 enabled This bit cannot be modified as long as LOCK level 1 has been programmed.
[0]	rw	1’h0	BK2INE	BRK2 BKIN input enable This bit enables the BKIN2 input. BKIN2 input is ’Ored’ with the other BRK2 sources. 0: BKIN2 input disabled 1: BKIN2 input enabled This bit cannot be modified as long as LOCK level 1 has been programmed.

9.2 BTIM

HPSYS has two BTIM modules.

The operating clock for BTIM1 is pclk_hpsys , and it should be noted that counting may be affected when the system dynamically adjusts the frequency.

The operating clock for BTIM2 is the divided frequency clk_peri_hpsys (24MHz) , which is independent of the system clock, and counting is not affected when the system dynamically adjusts the frequency.

9.2.1 Introduction

BTIM (Basic Timer) is based on a 32-bit incrementing counter, enabling timing functions. The counting clock can be the system PCLK or a cascading input signal, and it supports a pre-scaling factor of 1~65536 . The timing results can generate interrupts,DMA requests, or PTC triggers. BTIM includes a master-slave mode interface, allowing for multi-level cascading to achieve multi-level counting or synchronized triggering functions.

9.2.2 Main Features

- 32 bit incrementing auto-reload counter
- 16 bit programmable prescaler, with a division factor for the counter clock frequency ranging from 1 to 65536
- Supports one pulse mode (OPM), automatically stopping the counter upon completion
- Master-Slave Mode
  - Supports interconnection with other timers, enabling it to generate control signals as a master device while being controlled by external inputs or other master devices
  - Control modes include reset, trigger, gating, and others.
  - Supports simultaneous start and reset of multiple timers
- Generates interrupts /DMA on counter overflow or initialization

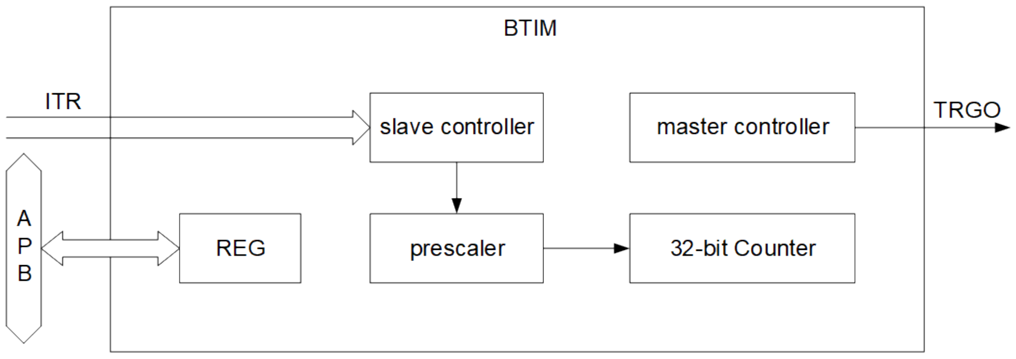


Figure 9-7: **BTIM Block Diagram**

9.2.3 BTIM Function Description

9.2.3.1 Counter

The functions of BTIM are based on a 32-bit counter. The counter operates on an event-counting basis, with the most fundamental event being a PCLKclock edge. Depending on the configuration, other counting events may include edges from external inputs, outputs from other timers, and so forth.

Counting events will only enter the counter after being processed by a prescaler. The prescaler count ranges from 1 to 65536 (PSC+1), meaning that the counter's value will only change once after (PSC+1) counting events have occurred.

The counter is fxed in an incrementing counting mode, counting from 0 to the auto-reload value ARR , then restarting from 0 and generating a counter overflow event. The count value can be read via CNT .

9.2.3.2 Update Event(UEV)

The update event is used to signal the conclusion of a counting unit. The most basic update event occurs with each counter overflow. An update event is also generated when the software sets EGR_UG to 1 . Update events can generate interrupts, DMA requests, and PTC triggers, serving as the most fundamental notifcation function of the timer.

By setting CR1_UDIS to 1 in software, the generation of update events can be disabled. This prevents the shadow register from being updated when writing new values to the preload register. No update events will occur until the UDIS bit is written to 0 .

If CR1_URS (update request selection) is set to 1 , setting EGR_UG to 1 will generate an Update Event, but will not set the UIF flag to 1 (therefore, no interrupts or DMA requests will be sent). Consequently, if the counter is cleared when a capture event occurs, neither an update interrupt nor a capture interrupt will be generated simultaneously.

When an Update Event occurs, the ARR and PSC Registers will be reloaded, and the update flag SR_UIF will be set to 1 (when CR1_URS=0). This feature ensures that modifcations to the basic parameters of these counters do not affect the current counting unit and take effect only in the next counting cycle.

9.2.3.3 Shadow Register

Modifcations to the ARR and PSC Registers will not be directly reflected in the current counting unit; they will only be updated when an Update Event occurs. Before the Update Event, the counter actually uses the values from the Shadow Register. This way, even if the values of these registers are dynamically changed during counting, it will not affect the integrity of the current counting unit.

If CR1_APRE is 0, the ARR register will take effect in real-time after configuration, without waiting for an update event.

9.2.3.4 Master-Slave Mode

The timer can operate simultaneously in both master and slave modes. The master mode allows the timer to output the TRGO signal to the ITR input of other timers on the chip, which is used to control their counting behavior. The slave mode indicates that the counting behavior of this timer is controlled by the ITR signal output from other timers.

Multiple timers can achieve Timer Synchronization through a master-slave configuration, enabling functions such as multi-level frequency division, simultaneous start, and gated counting.

The master mode can output the TRGO signal during various events, such as updates and enabling, as selected by CR2_MMS.

The slave mode can select behaviors such as counter reset, trigger start, and counting events through SMCR_SMS, while also allowing for the selection of gating modes. The triggering signal TRGI and gating signal that the slave mode relies on can be independently selected in ITR, and the polarity of the gating signal can also be chosen.

When the timer is in reset from mode (SMCR_SMS=001) , the counter and its prescaler are reinitialized upon a change in TRGI . If CR1_URS is 0 , an update event UEV is generated, and ARR is updated.

When the timer is in trigger from mode (SMCR_SMS=010) , the software does not need to configure CR1_CEN to enable counting; instead, the counter is automatically started on the rising edge of TRGI. In reset trigger from mode (SMCR_SMS=011) , the counter is reset on the rising edge of TRGI and automatically restarted.

When the timer is in external clock from mode (SMCR_SMS=100) , counting events are modifed to the rising edge of TRGI , and counting occurs only when TRGI changes state.

When the timer gate control is enabled from mode (SMCR_GM=1) , counting will only occur when TRGI meets the high or low level requirements (SMCR_GTP) ; otherwise, the counter remains unchanged.

9.2.3.5 One Pulse Mode

Set Writing 1 to CR1_OPM can enable the one pulse mode. In this mode, once the counter starts, it will automatically stop counting upon the occurrence of an update event. This mode is suitable for single counting.

9.2.3.6 Timer Synchronization

Multiple timers can be interconnected in a master-slave configuration to achieve Timer Synchronization, enabling functionalities such as multi-level frequency division, simultaneous start, and gated counting.

Setting the main mode timer's TRGO to an update event and connecting it to another timer configured as an external clock slave mode allows for cascading timer counting. In this case, the main mode timer functions as a prescaler for the slave mode timer, and the total counting bit width equals the sum of the bit widths of the two timers.

By setting the main mode timer's TRGO to count enable and configuring the slave mode as trigger slave mode, while connecting it to another timer also set as trigger slave mode, multiple timers can be synchronized to trigger simultaneously, thereby aligning their start timings. In this scenario, the main mode timer must also set SMCR_MSM to 1 .

9.2.3.7 Notifcation Mechanism

BTIM can generate various notifcation mechanisms, including interrupts, DMA requests, and PTC triggers. The events that can trigger notifcations are update events. The DIER register controls whether various events generate interrupts and DMA requests. The status of each event can be queried in the SR register.

9.2.4 BTIM Register

BTIM1 base address is 0x50092000.

Table 9-2: **BTIM Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	TIM control register 1
[31:8]			RSVD	
[7]	rw	1'h0	ARPE	Auto-reload preload enable 0: ARR register is not buffered 1: ARR register is buffered
[6:4]			RSVD	
[3]	rw	1'h0	OPM	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Continued on the next page...

Table 9-2: **BTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1’h0	URS	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: Counter overflow Setting the UG bit Update generation through the slave mode controller 1: Only counter overflow generates an update interrupt or DMA request if enabled.
[1]	rw	1’h0	UDIS	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: Counter overflow Setting the UG bit Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
[0]	rw	1’h0	CEN	Counter enable 0: Counter disabled 1: Counter enabled Note: Gated mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware. CEN is cleared automatically in one-pulse mode, when an update event occurs.
0x04			**CR2**	TIM control register 2
[31:6]			RSVD	
[5:4]	rw	2’h0	MMS	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 00: Reset:the UG bit from the EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 01: Enable :the Counter enable signal, CNT_EN, is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in SMCR register). 10: Update:The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 11: Gating:The delayed gating trigger is selected as trigger output (TRGO).
[3:0]			RSVD	
0x08			**SMCR**	TIM slave mode control register
[31:24]			RSVD	

Continued on the next page...

Table 9-2: **BTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[23]	rw	1'h0	GM	Gated Mode. The counter clock is enabled when the selected trigger input (TRGI) is active (according to gating trigger polarity). The counter stops (but is not reset) as soon as the trigger becomes inactive. Both start and stop of the counter are controlled. Gated mode and slave mode can be enabled simutanuously with different trigger selection.
[22]	rw	1'h0	GTP	Gating trigger polarity invert 0: active at high level 1: active at low level
[21:20]	rw	2'h0	GTS	Gating trigger selection in gated mode This bit-field selects the trigger input to be used to enable the counter gating. 00: Internal Trigger 0 (ITR0) 01: Internal Trigger 1 (ITR1) 10: Internal Trigger 2 (ITR2) 11: Internal Trigger 3 (ITR3)
[19]			RSVD	
[18:16]	rw	3'h0	SMS	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input. 000: Slave mode disabled. 001: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 010: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 011: Combined reset + trigger mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers and starts the counter. 100: External Clock Mode - Rising edges of the selected trigger (TRGI) clock the counter.
[15:8]			RSVD	
[7]	rw	1'h0	MSM	Master/Slave mode. This bit should be asserted on master timer if synchronization if needed. 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
[6]			RSVD	
[5:4]	rw	2'h0	TS	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 00: Internal Trigger 0 (ITR0) 01: Internal Trigger 1 (ITR1) 10: Internal Trigger 2 (ITR2) 11: Internal Trigger 3 (ITR3)
[3:0]			RSVD	
0x0C			**DIER**	TIM DMA/Interrupt enable register
[31:9]			RSVD	
[8]	rw	1'h0	UDE	Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled
[7:1]			RSVD	
[0]	rw	1'h0	UIE	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled
0x10			**SR**	TIM status register

Continued on the next page...

Table 9-2: BTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:1]			RSVD	
[0]	rw0c	1'h0	UIF	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred 1: Update interrupt pending. This bit is set by hardware when the registers are updated: At overflow and if UDIS=0 in the CR1 register. When CNT is reinitialized by software using the UG bit in EGR register, if URS=0 and UDIS=0 in the CR1 register. When CNT is reinitialized by a trigger event (refer to the synchro control register description), if URS=0 and UDIS=0 in the CR1 register.
0x14			**EGR**	Event generation register
[31:1]			RSVD	
[0]	w1s	1'h0	UG	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (ARR) if DIR=1 (downcounting).
0x24			**CNT**	Counter
[31:0]	rw	32'h0	CNT	counter value
0x28			**PSC**	Prescaler
[31:16]			RSVD	
[15:0]	rw	16'h0	PSC	Prescaler value The counter clock frequency is equal to fCLK / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of EGR register or through trigger controller when configured in "reset mode").
0x2C			**ARR**	Auto-reload register
[31:0]	rw	32'h0	ARR	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. The counter is blocked while the auto-reload value is null.

9.3 GPTIM

HPSYS has two GPTIM modules.

The operating clock for GPTIM1 is pclk_hpsys, and it should be noted that counting may be affected when the system dynamically adjusts the frequency.

The operating clock for GPTIM2 is clk_peri_hpsys's divided frequency (24MHz) , which is independent of the system clock, and counting is not affected when the system dynamically adjusts the frequency.

9.3.1 Introduction

GPTIM (General Purpose Timer) is based on a 16 -bit counter, which can perform timing, measure the pulse length of input signals (input capture), or generate output waveforms (output compare and PWM) , among other functions. The counter itself can increment, decrement, or perform up/down counting, with the counting clock selectable from the system PCLK , IO input signals, or cascaded input signals, and can be prescaled by 1~65536 times. GPTIM has a total of

4 channels, which can be independently configured for input capture or output mode. The results of counting, input capture, and output compare can generate interrupts, DMA requests, or PTC triggers. GPTIM includes a master-slave mode interface, enabling multi-level cascading to achieve multi-level counting or synchronized triggering functions.

9.3.2 Main Features

- 16 bit increment, decrement, increment/decrement auto-reload counter, with a maximum count of 65535.
- 16 bit programmable (can be modifed in real-time)prescaler, with a division factor for the counter clock frequency ranging from 1 to 65536for any value
- 8 bit configurable repeat count.
- Supports one pulse mode (OPM), which automatically stops the counter upon completion of the repeat count.
- 4 independent channels, which can be configured separately for input or output modes.
- Input Mode
  - Rising Edge/Falling Edge Capture
  - PWM Pulse Width and Period Capture (requires two channels)
  - Optional one of 4 input ports or 1 external trigger port, supporting debounce fltering and pre-frequency reduction
- Output Mode
  - Force output to high/low level
  - Output high/low/flip level upon reaching the comparison value
  - PWM output, with configurable pulse width and period
  - Multi-channel PWM composite output, capable of generating multiple PWM with interrelated characteristics
  - Single Pulse/Re-triggerable One Pulse Mode Output
- Master-Slave Mode
  - Supports interconnection of multiple timers, allowing it to generate control signals as a master device while being controlled by external inputs or other master devices as a slave.
  - Control modes include reset, trigger, gating, and others.
  - Supports simultaneous start and reset of multiple timers
- Encoding mode input for controlling counter increment/decrement counting.
- An interrupt/DMArequest/PTCis generated when the following events occur:
  - Update: Counter increment overflow/decrement overflow, counter initialization (via software or internal/external trigger)
  - Trigger Events (counter start, stop, initialization, or counting triggered by internal/external sources)
  - Input Capture
  - Output Compare

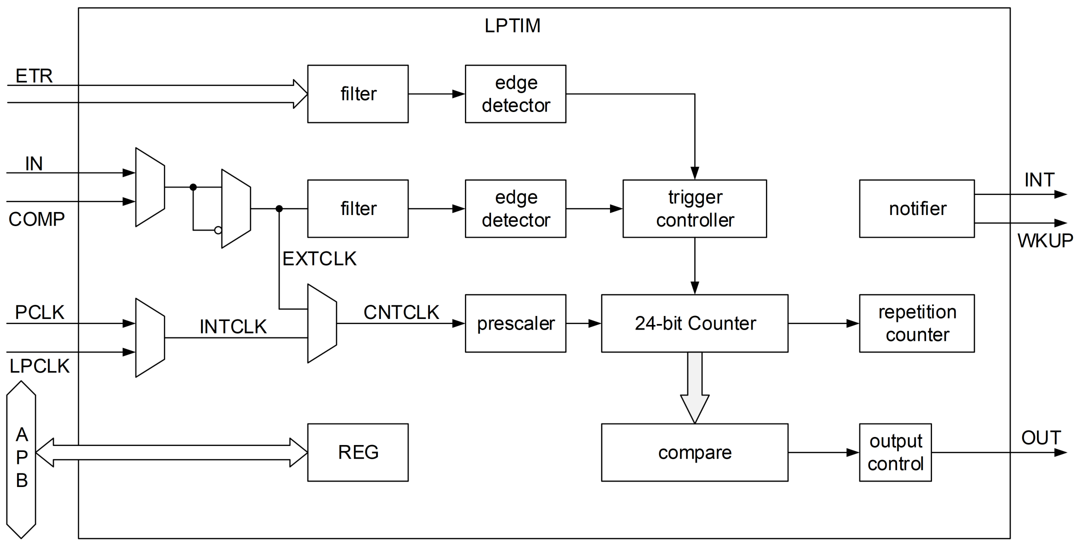


Figure 9-8: **GPTIM Block Diagram**

9.3.3 GPTIM Function Description

9.3.3.1 Counter

The functions of GPTIM are based on a 16 bit counter. The counter operates on an event basis, with the most fundamental event being a PCLK clock edge. Depending on the configuration, other counting events may include external input edges, output edges from other timers, and decoding outputs from the quadrature encoder interface.

Counting events will only enter the counter after being processed by a prescaler. The prescaler count ranges from 1 to 65536 (PSC+1) , meaning that the counter's value will only change once after (PSC+1) counting events have occurred.

The counter features three counting modes: incrementing, decrementing, and center-aligned. In the incrementing counting mode (CR1_CMS=0 and CR1_DIR=0) , the counter counts from 0 to the auto-reload value ARR , then restarts counting from 0 and generates a counter overflow event. In the decrementing counting mode (CR1_CMS=0 and CR1_DIR=1) , the counter counts down from ARR to 0 , then restarts counting from ARR and generates a counter underflow event. In the center-aligned mode (CR1_CMS is not 0) , the counter counts from 0 to ARR −1 , generating a counter overflow event, then counts down from ARR to 1 and generates a counter underflow event, after which it restarts counting from 0 again.

The count value can be read through CNT. The counting direction can be read from CR1_DIR.

9.3.3.2 Update Event(UEV)

An update event is used to indicate the end of a counting unit. The most basic update event occurs on every overflow or underflow of the counter (when repeat counting is not enabled) . An update event is also generated when the software sets EGR_UG to 1 . Update events can trigger interrupts, DMA requests, and PTC triggers, making it the most fundamental notifcation function of the timer.

By setting CR1_UDIS to 1 in software, the generation of update events can be disabled. This prevents the shadow register from being updated when writing new values to the preload register. No update events will occur until the UDIS bit is written to 0 .

If CR1_URS (update request selection) is set to 1 , setting EGR_UG to 1 will generate an Update Event, but will not set the UIF flag to 1 (therefore, no interrupts or DMA requests will be sent). Consequently, if the counter is cleared when a capture event occurs, neither an update interrupt nor a capture interrupt will be generated simultaneously.

When an Update Event occurs, the RCR , ARR , and PSC registers will be reloaded, and the update flag SR_UIF will be set to 1 (when CR1_URS=0). This function ensures that modifcations to the basic parameters of these counters do not affect the current counting unit, taking effect only in the next counting cycle.

9.3.3.3 Repeat Counting

If the Repeat Counter (RCR > 0) is configured, it will decrement each time the counter overflows or underflows, and an Update Event will only occur when the Repeat Counter reaches 0. When an Update Event occurs, the Repeat Counter will reload the value of RCR.

The current value of the Repeat Counter cannot be read.

9.3.3.4 Shadow Register

Modifcations to the RCR, ARR, and PSC registers will not be directly reflected in the current counting unit; they will only be updated when an Update Event occurs. Before the Update Event, the counter uses the values from the Shadow Registers. This ensures that dynamically changing these register values during counting does not affect the integrity of the current counting unit, which is signifcant for applications such as PWM output.

If CR1_APRE is 0, theARRregister will take effect in real-time after configuration, without waiting for an update event.

The output compare register CCRx also features a shadow register. When CCMRx_OCxPE is 0 , the configured CCRx will take effect immediately; otherwise, it will only take effect when an update event occurs.

9.3.3.5 Master-Slave Mode

The timer can operate simultaneously in both master and slave modes. The master mode allows the timer to output a TRGO signal to the ITR input of other timers on the chip, which is used to control their counting behavior. The slave mode indicates that the counting behavior of this timer is influenced by external input ETR, or by the ITR signal output from other timers, or by control from the timer’s channel input CHx.

Multiple timers can achieve Timer Synchronization through a master-slave configuration, enabling functions such as multi-level frequency division, simultaneous start, and gated counting.

The master mode can output the TRGO signal upon various events, such as updates, enabling, input capture, and output comparison, as selected by CR2_MMS.

The slave mode can select behaviors such as counter reset, trigger start, counting enable, and counting events, as determined by SMCR_SMS. The trigger signal TRGI on which the slave mode depends can be flexibly configured, with options to select from ETR, ITR, and channel inputs, as well as to choose signal polarity for pre-scaling, fltering, and other operations

- When the timer is in reset from mode (SMCR_SMS=0100) and the TRGI changes, both the counter and its prescaler are reinitialized. If CR1_URS is 0,an update event UEVwill be generated, causing all preload registers ARRand CCRxto be updated.
- When the timer is in gated from mode (SMCR_SMS=0101), the counting occurs only when TRGI meets the high or

low level requirements; otherwise, the counter remains unchanged.

- When the timer is in triggered from mode (SMCR_SMS=0110), the software does not need to configure CR1_CEN to initiate counting; instead, the counter is automatically started when TRGI meets specific trigger requirements.
- When the timer is in external clock slave mode (SMCR_SMS=0111) , the counting event is modifed to count on the rising edge of TRGI, and counting occurs only when TRGI changes state.
- When the timer is in reset trigger slave mode (SMCR_SMS=1000) , the counter is reset and automatically restarted when TRGI meets specific trigger requirements.

9.3.3.6 Channel Input and Output

Some channels of the timer can be independently configured as input capture mode(CCMRx_CCxS!=0)or output mode(CCMRx_CCxS=0).

In input capture mode, when the corresponding trigger signal is valid, the value of the counter is recorded into CCRx , and an interrupt or other notifcation signal is generated. The trigger signal can be selected from ETR , ITR , and channel input CHx , and the signal polarity can be selected, along with pre-scaling, fltering, and other operations. The notifcation signals generated by the channel include interrupts, DMA requests, and PTC triggers. Input capture mode can record the moments of external signal changes, measure PWM periods, and duty cycles, among other functions.

In output mode, the channel compares the value of the counter with the size of CCRx , generating a fxed level on the channel output CHx/CHxN , or producing a PWM output signal based on the comparison results of this channel and other channels, as well as generating interrupt and other notifcation signals. The parameters of the generated PWM signal, including the number of pulses, frequency, duty cycle, and phase, are adjustable. Multiple channels can also work together to produce specific relationships of PWM combinations. The notifcation signals generated by the channel include interrupts, DMA requests, and PTC triggers, among others.

9.3.3.7 Input Capture Mode

In Input Capture Mode, when a rising or falling edge of the corresponding trigger signal on the channel is detected, the value of the counter will be latched using CCRx . When a capture event occurs, the corresponding SR_CCxIF flag will be set to 1 , and an interrupt, a DMA request (if enabled), or a PTC trigger signal may be sent. If the SR_CCxIF flag is already high when the capture event occurs, the repeat capture flag SR_CCxOF will be set to 1 . The SR_CCxIF can be cleared by software by writing 0 to SR_CCxIF or by reading the captured data stored in CCRx . Writing 0 to SR_CCxOF will also clear it.

The following example illustrates how to capture the counter value into CCR1when a rising edge occurs at the CH1input. The specific steps are as follows:

1. Select valid input: Channel 1 is to be connected to CH1input; therefore, write 01 to CCMR1_CC1S.
2. Configure the required input flter bandwidth based on the signals connected to the timer.
   Assuming that the CH1 signal edge changes with a maximum jitter of 5 PCLK cycles, the fltering bandwidth should be set to greater than 5 PCLK cycles. Set CCMR1_IC1F to 0011(0x3) so that when eight consecutive sampling points (sampled at PCLK frequency) are detected to be at the new level, the transition edge of CH1 can be confrmed.
3. Set Set CCER_CC1P and CCER_CC1NP to 0 to select the valid conversion edge on CH1 as the rising edge.
4. Program the input prescaler.
   In this example, we want to perform a capture operation on every valid conversion; therefore, the prescaler is disabled (set CCMR1_IC1PS to 00).
5. Set CCER_CC1E to 1to enable channel 1and allow the counter value to be captured in CCR1.

6. If necessary, set DIER_CC1IEto 1to enable the corresponding interrupt request, or set DIER_CC1DE to 1 to enable DMArequests.

Once configured, the channel will execute the following actions when a rising edge appears on the CH1 input:

1. CCR1 Register records the value of the counter.
2. SR_CCxIF Flag set to 1(Interrupt flag). If at least two consecutive captures occur, but SR_CCxIF has not been cleared, then SR_CCxOF capture overflow flag will be set to 1.
3. Generate an interrupt based on CCER_CC1IE.
4. Generate a DMA request based on DIER_CC1DE.

To handle repeated captures, it is recommended to read the data before accessing SR_CCxOF. This can prevent the loss of repeated capture information that may occur between reading SR_CCxOF and the data.

Setting EGR_CCxG to 1via software can immediately generate a capture and produce a channel capture interrupt and DMA request.。

9.3.3.8 PWM Input Capture

PWM Input Capture is an advanced application of Input Capture, which can be utilized to measure the period and duty cycle of the PWM input signal. To implement this functionality, both channels must be configured in Input Capture Mode, with the trigger signals mapped to the rising and falling edges of the PWM input, and the counter reset mode must be activated.

The following example demonstrates how to measure the period and duty cycle of the PWM input from CH1 using Channel 1 and Channel 2. The specific steps are as follows:

1. Designate the valid input for Channel 1as the CH1input by writing 01to CCMR1_CC1S.
2. Select channel 1 The effective polarity of the input signal (used for capturing in CCR1 and resetting the counter) is set by writing 0 to CCER_CC1P and CCER_CC1NP, selecting the effective transition edge on CH1 as the rising edge.
3. The effective input for channel 2 is also the CH1 input; write 10(0x2) to CCMR1_CC2S.
4. Select the valid polarity of the input signal for channel 2 (used for CCR2 capture) , set CCER_CC2P to 1 and CCER_CC1NP to 0 , selecting the valid transition edge on CH1 as the falling edge.
5. Set the mode control signal to CH1, write 101(0x5)to SMCR_TS, and select TI1FP1.
6. Configure the mode controller to reset mode by writing 0100 (0x4)to SMCR_SMS.
7. Enable channel 1 and channel 2 by setting CCER_CC1E and CCER_CC2E to 1.

After configuration, on each rising edge of CH1 , the counter's value is recorded in CCR1 , while the counter is reset and begins counting again; on each falling edge of CH1 , the counter's value is recorded in CCR2 . Multiplying the value of CCR1 by the period of PCLK calculates the period of PWM. Set Multiplying the value ofCCR2 by the period of PCLK calculates the duration of the high level of PWM, thus determining the duty cycle of PWM.

9.3.3.9 Output Compare Mode

In Output Compare Mode, when the count value satisfes a specific relationship with CCRx , particular outputs can be generated on the corresponding CHx and CHxN , which are typically used to control output waveforms or to indicate that a certain time period has elapsed.

Specifically, the channel will execute the following operations when CCRxmatches the counter:

1. A programmable value will be assigned for the corresponding CHx and CHxN , as defined by the Compare Mode

Register CCMRx_OCxM and the Output Polarity Register CCER_CCxP/CCxNP . Upon matching, the output pin can either maintain its level (CCMRx_OCxM=0000) or be set to active level (CCMRx_OCxM=0001) , inactive level (CCMRx_OCxM=0010) , or toggle (CCMRx_OCxM=0011) .

2. Set the interrupt status register flag SR_CCxIF to 1.
3. An interrupt is generated based on CCER_CC1IE.
4. Generate DMA requests based on DIER_CC1DE and CR2_CCDS.

Configure CCMRx_OCxPE to allow the CCRx register to be set with or without a shadow register. When CCMRx_OCxPE is 0, software modifcations to CCRx take effect in real-time, enabling custom waveform output by modifying the next matching CCRx in each interrupt.

9.3.3.10 Basic PWM Output

Using output compare mode, the timer can generate multiple PWM outputs with controllable period, duty cycle, and phase. The period of the PWM output is determined by ARR, while the duty cycle is determined by CCRx. There are various modes for PWM output, independently selected by each channel's CCMRx_OCxM. The most basic single-channel PWM output requires only one channel and can be achieved using the basic PWM mode. More complex PWM signals or PWM combinations require multiple channels and careful allocation of each channel's PWM mode and CCRx.

In basic PWM mode, the counter value CNT is compared with CCRx , generating a comparison output signal OCxREF that contains valid or invalid levels based on the current counting direction of the counter. The polarity of the valid level can be configured through CCER_CCxP , and the output CHx is enabled according to the CCER_CCxE and BDTR_MOE registers.

For example, in the increment counting mode, if CCMR1_OC1M and CCMR1_OC2M are configured to 0110(0x6) , the PWM output is as shown in Figure9-9 . When the counter value CNT is less than CCR1/2 , the output is high; otherwise, it is low.

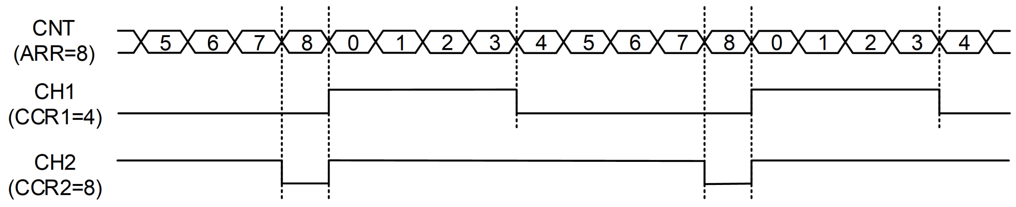


Figure 9-9: PWM output in increment counting mode

In center-aligned counting mode, if you configure CCMR1_OC1M and CCMR1_OC2M to 0110(0x6) , the PWM output is illustrated in Figure9-10 . When the counting value CNT in the increment phase is less than CCR1/2 , the output is high; otherwise, it is low. When the counting value CNT in the decrement phase exceeds CCR1/2 , the output is low; otherwise, it is high.

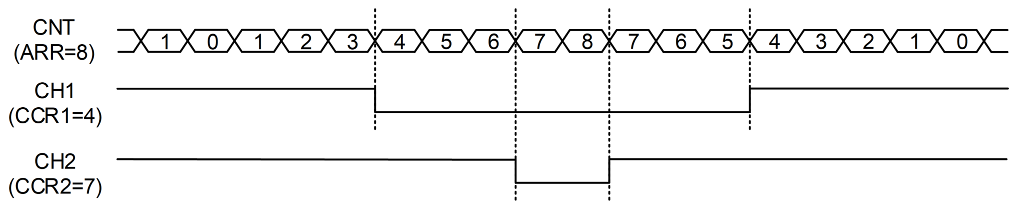


Figure 9-10: PWM output in center-aligned counting mode

9.3.3.11 Asymmetric PWM output

In asymmetric PWM mode, there is a programmable phase shift between the two PWM signals generated. This mode is restricted to when the counter is in center-aligned mode. The frequencies of the two PWM signals are identical, determined by the value of ARR , while the duty cycle and phase shift are each determined by a pair of CCRx Registers. Each PWM output occupies two CCRx Registers, controlling the behavior during increment and decrement counting periods, allowing the rising and falling edges of the PWM to be configured independently. CCR1 and CCR2 jointly control the output of CH1/2 , while CCR3 and CCR4 jointly control the output of CH3/4 .

CH1/2 and CH3/4can independently select different asymmetric PWM modes by configuring CCMRx_OCxM to 1110(0xe)or 1111(0xf).

If you configure CCMR1_OC1M and CCMR2_OC3M to 1110(0xe) , the PWM output is illustrated in Figure9-11. In the increment phase (0->ARR-1) , when the counting value CNT is less than CCR1/3 , the output is high; otherwise, it is low. In the decrement phase (ARR->1) , when the counting value CNT exceeds CCR2/4 , the output is low; otherwise, it is high.

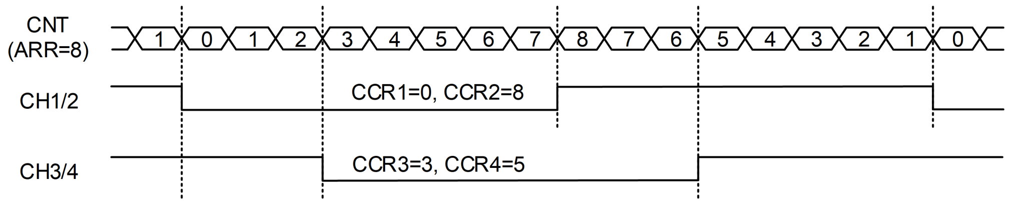


Figure 9-11: **Asymmetric PWMO utput**

9.3.3.12 Combined PWMO utput

In combined PWM mode, there is a programmable delay and phase shift between the two PWM signals generated. The counter can operate in incrementing, decrementing, or center-aligned mode, and the frequency of the two PWM signals is the same, determined by the value of ARR. The duty cycle and phase shift are each determined by a pair of CCRx Registers. Each output PWM utilizes two CCRx Registers, formed by the logical AND or OR combination of two basic PWM output waveforms. CCR1 and CCR2 jointly control the output of CH1/2, while CCR3 and CCR4 jointly control the output of CH3/4.

CH1/2 and CH3/4 can independently select different combinations of PWM modes, configuring CCMRx_OCxM to 1100(0xc) or 1101(0xd). When CH1 or CH3 is configured to the combined PWM mode 1100(0xc), CH2 or CH4 must be configured to 0111(0x7) or 1101(0xd) or 1111(0xf). When CH1 or CH3 is configured to the combined PWM mode 1101(0xd), CH2 or CH4 must be configured to 0110(0x6) or 1100(0xc) or 1110(0xe).

For example, if CCMR1_OC1Mis configured to 1101(0xd),CCMR1_OC2Mto 0110(0x6),CCMR2_OC3Mto 1100(0xc),CCMR2_OC4M is If 0111(0x7), then the PWM output is as shown in Figure9-12. When the count value CNT is less than CCR1/4, OC1REF/OC4REF is low; otherwise, it is high. When the count value CNT is less than CCR2/3, OC2REF/OC3REF is high; otherwise, it is low. The output of CH1 is the logical AND of OC1REF and OC2REF. The output of CH3 is the logical OR of OC3REF and OC4REF.

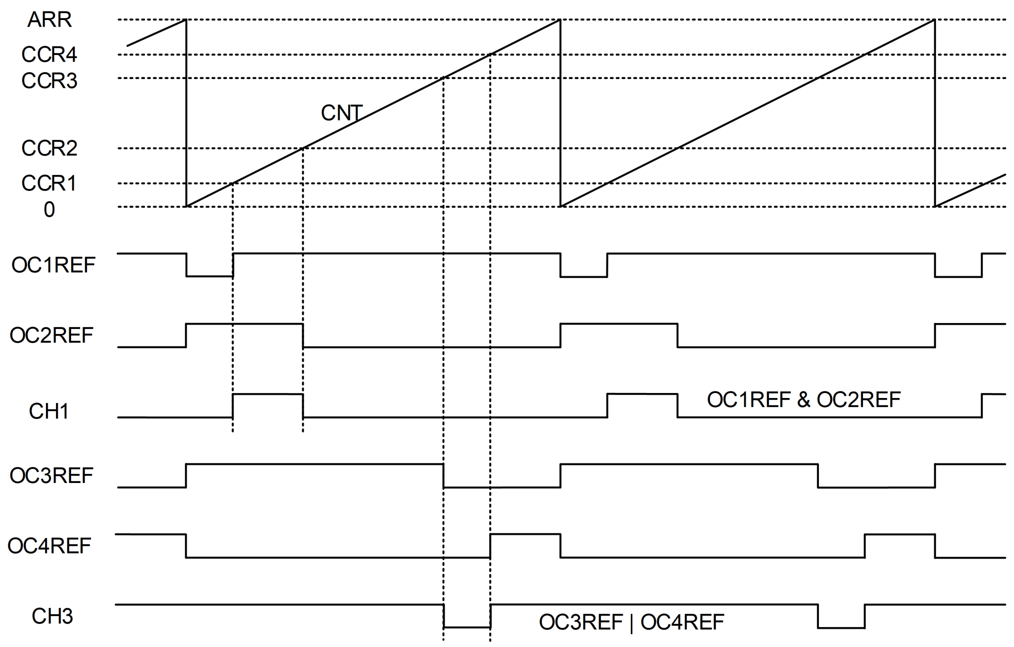


Figure 9-12: **Combined PWM Output**

9.3.3.13 One Pulse Mode

Set CR1_OPM Write 1 Enables the One Pulse Mode. In this mode, once the counter is started, it will automatically stop counting upon an Update Event. This mode can be utilized for single counting or can be triggered by an excitation signal to generate a pulse with a programmable width after a programmable delay.

For example, to achieve the functionality where a single pulse of a specific width is generated on CH1 after a certain delay when a rising edge is detected on the CH2 input pin, the configuration method is as follows:

1. CCMR1_CC2S=01 to map TI2FP2to channel 2.
2. CCER_CCxP and CCER_CCxNP are set to 0, with TI2FP2 responding to the positive edge change of CH2.
3. SMCR_TS=110(0x6) configures TI2FP2 to trigger from the mode controller's TRGI.
4. SMCR_SMS=110(0x6) configures the mode controller to trigger mode, enabling counting after the trigger.
5. Configure ARR and CCR1 according to the required time delay and pulse width, thereby defning the time delay and pulse width.
6. CCMR1_OC1M=0111(0x7) configures for positive pulse PWM.
7. CR1_OPM=1 , a single trigger generates only one pulse.
8. EGR_UG=1 , manually refresh the ARRand CCR1 registers.

In trigger mode, it is not necessary to manually enable CR1_CEN; once a trigger signal is detected, the counter will be automatically enabled.

9.3.3.14 Encoder Interface Mode

In Encoder Interface Mode, channels 1 and 2 can be used to connect external quadrature encoders, converting the signals from the external encoder into changes in the timer's count value, thereby determining the operational status of the external encoder.

If the counter counts only on the rising edge of CH1 , configureSMCR_SMS to 0001 ; if the counter counts only on the rising edge of CH2 , configureSMCR_SMS to 0010(0x2) ; if the counter counts on the rising edges of both CH1 and CH2 , configure SMCR_SMS to 0011(0x3) . CCER_CC1P/CC2P is used to select the polarity of CH1 and CH2 . If necessary, the input flter can also be programmed. The signal conversion sequence of the two inputs will generate counting pulses and direction signals; based on this signal conversion sequence, the counter will increment or decrement accordingly, while the hardware will modify CR1_DIR as needed.

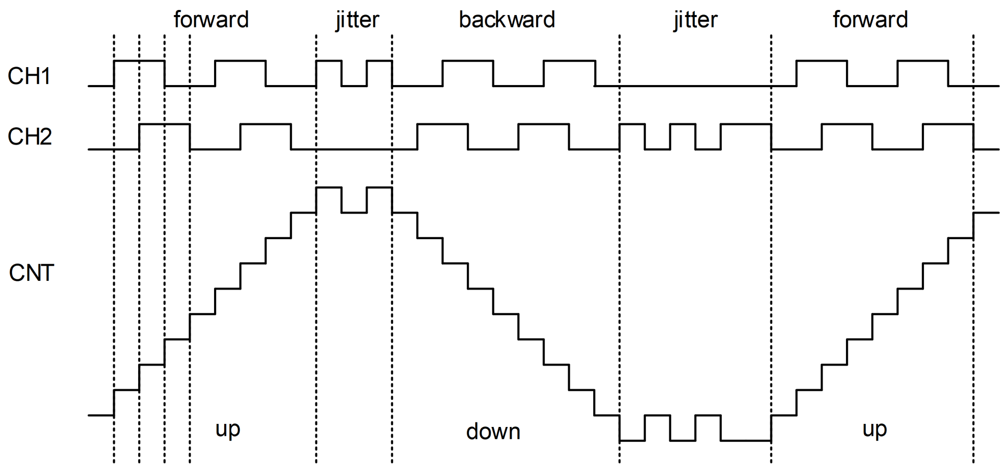
In Encoder Interface Mode, the counting events of the counter are the decoded outputs of the quadrature encoder interface. The counter only performs continuousounting between 0 and ARR (incrementing from 0 to ARR or decrementing from ARR to 0 , depending on the specific counting direction). Therefore, it is necessary to configure ARR before starting. Similarly, the capture, compare, repeat counter, and trigger output functions continue to operate normally. In this mode, the counter is automatically modifed based on the speed and direction of the quadrature encoder, ensuring that its content always represents the position of the encoder. The counting direction corresponds to the rotation direction of the connected sensor. The table below summarizes the possible combinations (assuming CH1 and CH2 do not switch simultaneously).

SMCR_SMS	Condition	CH1 Rising Edge	CH1 Falling Edge	CH2 Rising Edge	CH2 Falling Edge
0001 or 0011	CH2=0	Increment	Decrement	/	/
	CH2=1	Decrement	Increment	/	/
0010 or 0011	CH1=0	/	/	Decrement	Increment
	CH1=1	/	/	Increment	Decrement

The following diagram illustrates how the counter counts based on the signal changes from the quadrature encoder, configured as follows:

CCMR1_CC1S=01 （CH1 mapped to channel 1）, CCMR2_CC2S=01（CH2mapped to channel 2）,

CCER_CC1P/CC1NP/CC2P/CC2NP=0, SMCR_SMS=0011(0x3), CR1_CEN=1。



9.3.3.15 Timer Synchronization

Multiple timers can be interconnected in a master-slave configuration to achieve Timer Synchronization, enabling functionalities such as multi-level frequency division, simultaneous start, and gated counting.

Set the master mode timer's TRGO to update event (CR2_MMS=010) , connected to another timer configured for external clock slave mode (SMCR_SMS = 0111) , to enable timer cascading counting. In this configuration, the master mode timer serves as the prescaler for the slave mode timer, and the total counting bit width is the sum of the bit widths of both timers.

Set the master mode timer's TRGO to count enable (CR2_MMS=001) , and configure the slave mode to trigger slave mode (SMCR_SMS=0110) , while also connecting to another timer set to trigger slave mode (SMCR_SMS=0110) , to achieve synchronized triggering of multiple timers, thereby aligning the start timing of all timers. In this scenario, the master mode timer must also set SMCR_MSM to 1 .

Configure the main mode timer's TRGO to output a comparison signal (CR2_MMS=100) , connected to another timer set to gated slave mode (SMCR_SMS=0101), to achieve gated PWM output. The main mode timer can modulate the PWM carrier output from the slave mode timer.

9.3.3.16 Notifcation Mechanism

GPTIM can generate various notifcation mechanisms, including interrupts, DMA requests, and PTC triggers. The events that can trigger notifcations primarily include update events, trigger events, comparator matches, and input captures. The DIER register controls whether various events generate interrupts and DMA requests. The status of each event can be queried in the SR register.

9.3.4 GPTIM Register

GPTIM1 base address is 0x50090000。

Table 9-3: **GPTIM Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	TIM control register 1
[31:12]			RSVD	
[11]	rw	1'h0	UIFREMAP	UIF status bit remapping 0: No remapping. UIF status bit is not copied to CNT register bit 31 1: Remapping enabled. UIF status bit is copied to CNT register bit 31
[10:8]			RSVD	
[7]	rw	1'h0	ARPE	Auto-reload preload enable 0: ARR register is not buffered 1: ARR register is buffered
[6:5]	rw	2'h0	CMS	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in CCMRx register) are set both when the counter is counting up or down.
[4]	rw	1'h0	DIR	Direction 0: Counter used as upcounter 1: Counter used as downcounter
[3]	rw	1'h0	OPM	One-pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the bit CEN)

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	URS	Update request source This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt or DMA request if enabled. These events can be: Counter overflow/underflow Setting the UG bit Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt or DMA request if enabled.
[1]	rw	1'h0	UDIS	Update disable This bit is set and cleared by software to enable/disable UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: Counter overflow/underflow Setting the UG bit Update generation through the slave mode controller Buffered registers are then loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers keep their value (ARR, PSC, CCRx). However the counter and the prescaler are reinitialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
[0]	rw	1'h0	CEN	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware. CEN is cleared automatically in one-pulse mode, when an update event occurs.
0x04			**CR2**	TIM control register 2
[31:8]			RSVD	
[7]	rw	1'h0	TI1S	TI1 selection 0: The CH1 pin is connected to TI1 input 1: The CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6:4]	rw	3'h0	MMS	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter enable signal is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected. 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO) 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
[3]	rw	1'h0	CCDS	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
[2:0]			RSVD	
0x08			**SMCR**	TIM slave mode control register
[31:20]			RSVD	
[19:16]	rw	4'h0	SMS	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input. 0000: Slave mode disabled. 0001: Encoder mode 1 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 0010: Encoder mode 2 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 0011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 0100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 0101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 0111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. 1000: Combined reset + trigger mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers and starts the counter.
[15]	rw	1'h0	ETP	External trigger polarity 0: ETR is non-inverted, active at high level or rising edge 1: ETR is inverted, active at low level or falling edge

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14]	rw	1'h0	ECE	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.
[13:12]	rw	2'h0	ETPS	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of CK_INT frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
[11:8]	rw	4'h0	ETF	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[7]	rw	1'h0	MSM	Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
[6:4]	rw	3'h0	TS	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF)
[3:0]			RSVD	
0x0C			**DIER**	TIM DMA/Interrupt enable register
[31:15]			RSVD	

Continued on the next page...

Table 9-3: **GPTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[14]	rw	1'h0	TDE	Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled.
[13]			RSVD	
[12]	rw	1'h0	CC4DE	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled. 1: CC4 DMA request enabled
[11]	rw	1'h0	CC3DE	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled. 1: CC3 DMA request enabled.
[10]	rw	1'h0	CC2DE	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
[9]	rw	1'h0	CC1DE	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
[8]	rw	1'h0	UDE	Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled
[7]			RSVD	
[6]	rw	1'h0	TIE	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
[5]			RSVD	
[4]	rw	1'h0	CC4IE	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled. 1: CC4 interrupt enabled
[3]	rw	1'h0	CC3IE	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled. 1: CC3 interrupt enabled
[2]	rw	1'h0	CC2IE	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled. 1: CC2 interrupt enabled.
[1]	rw	1'h0	CC1IE	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled. 1: CC1 interrupt enabled
[0]	rw	1'h0	UIE	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled
0x10			**SR**	TIM status register
[31:13]			RSVD	
[12]	rw0c	1'h0	CC4OF	Capture/Compare 4 overcapture flag
[11]	rw0c	1'h0	CC3OF	Capture/Compare 3 overcapture flag
[10]	rw0c	1'h0	CC2OF	Capture/Compare 2 overcapture flag
[9]	rw0c	1'h0	CC1OF	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in CCR1 register while CC1IF flag was already set
[8:7]			RSVD	

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw0c	1'h0	TIF	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode). It is set when the counter starts or stops when gated mode is selected. It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
[5]			RSVD	
[4]	rw0c	1'h0	CC4IF	Capture/Compare 4 interrupt flag
[3]	rw0c	1'h0	CC3IF	Capture/Compare 3 interrupt flag
[2]	rw0c	1'h0	CC2IF	Capture/Compare 2 interrupt flag
[1]	rw0c	1'h0	CC1IF	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match. 1: The content of the counter CNT has matched the content of the CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the CCR1 register. 0: No input capture occurred. 1: The counter value has been captured in CCR1 register (An edge has been detected on IC1 which matches the selected polarity).
[0]	rw0c	1'h0	UIF	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred 1: Update interrupt pending. This bit is set by hardware when the registers are updated: At overflow or underflow and if UDIS=0 in the CR1 register. When CNT is reinitialized by software using the UG bit in EGR register, if URS=0 and UDIS=0 in the CR1 register. When CNT is reinitialized by a trigger event, if URS=0 and UDIS=0 in the CR1 register.
0x14			**EGR**	Event generation register
[31:7]			RSVD	
[6]	w	1'h0	TG	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in SR register. Related interrupt or DMA transfer can occur if enabled.
[5]			RSVD	
[4]	w	1'h0	CC4G	Capture/compare 4 generation
[3]	w	1'h0	CC3G	Capture/compare 3 generation
[2]	w	1'h0	CC2G	Capture/compare 2 generation

Continued on the next page...

Table 9-3: **GPTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	w	1’h0	CC1G	Capture/compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
[0]	w	1’h0	UG	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the auto-reload value (ARR) if DIR=1 (downcounting).
0x18			**CCMR1**	TIM capture/compare mode register 1
[31:28]	rw	4’h0	OC2M	Output compare 2 mode
[27]	rw	1’h0	OC2PE	Output compare 2 preload enable
[26:25]			RSVD	
[24]	rw	1’h0	OC2CE	Output compare 2 clear enable

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:20]	rw	4’h0	OC1M	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen - The comparison between the output compare register CCR1 and the counter CNT has no effect on the outputs.(this mode is used to generate a timing base). 0001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter CNT matches the capture/compare register 1 (CCR1). 0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter CNT matches the capture/compare register 1 (CCR1). 0011: Toggle - OC1REF toggles when CNT=CCR1. 0100: Force inactive level - OC1REF is forced low. 0101: Force active level - OC1REF is forced high. 0110: PWM mode 1 - In upcounting, channel 1 is active as long as CNT<CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF=’0) as long as CNT>CCR1 else active (OC1REF=1). 0111: PWM mode 2 - In upcounting, channel 1 is inactive as long as CNT<CCR1 else active. In downcounting, channel 1 is active as long as CNT>CCR1 else inactive. 1000: Retriggerable OPM mode 1 - In up-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. In down-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. 1001: Retriggerable OPM mode 2 - In up-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 2 and the channels becomes inactive again at the next update. In down-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes active again at the next update. 1010: Reserved, 1011: Reserved, 1100: Combined PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC is the logical OR between OC1REF and OC2REF. 1101: Combined PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC is the logical AND between OC1REF and OC2REF. 1110: Asymmetric PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down. 1111: Asymmetric PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down.
[19]	rw	1’h0	OC1PE	Output compare 1 preload enable 0: Preload register on CCR1 disabled. CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on CCR1 enabled. Read/Write operations access the preload register. CCR1 preload value is loaded in the active register at each update event.
[18:17]			RSVD	
[16]	rw	1’h0	OC1CE	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:12]	rw	4'h0	IC2F	Input capture 2 filter
[11:10]	rw	2'h0	IC2PSC	Input capture 2 prescaler
[9:8]	rw	2'h0	CC2S	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (SMCR register)
[7:4]	rw	4'h0	IC1F	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fCLK 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[3:2]	rw	2'h0	IC1PSC	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=0. 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
[1:0]	rw	2'h0	CC1S	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
0x1C			**CCMR2**	TIM capture/compare mode register 2
[31:28]	rw	4'h0	OC4M	Output compare 4 mode
[27]	rw	1'h0	OC4PE	Output compare 4 preload enable
[26:25]			RSVD	
[24]	rw	1'h0	OC4CE	Output compare 4 clear enable
[23:20]	rw	4'h0	OC3M	Output compare 3 mode

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19]	rw	1'h0	OC3PE	Output compare 3 preload enable
[18:17]			RSVD	
[16]	rw	1'h0	OC3CE	Output compare 3 clear enable
[15:12]	rw	4'h0	IC4F	Input capture 4 filter
[11:10]	rw	2'h0	IC4PSC	Input capture 4 prescaler
[9:8]	rw	2'h0	CC4S	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
[7:4]	rw	4'h0	IC3F	Input capture 3 filter
[3:2]	rw	2'h0	IC3PSC	Input capture 3 prescaler
[1:0]	rw	2'h0	CC3S	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
0x20			**CCER**	Capture/Compare enable register
[31:16]			RSVD	
[15]	rw	1'h0	CC4NP	Capture/Compare 4 output Polarity.
[14]			RSVD	
[13]	rw	1'h0	CC4P	Capture/Compare 4 output Polarity.
[12]	rw	1'h0	CC4E	Capture/Compare 4 output enable.
[11]	rw	1'h0	CC3NP	Capture/Compare 3 output Polarity.
[10]			RSVD	
[9]	rw	1'h0	CC3P	Capture/Compare 3 output Polarity.
[8]	rw	1'h0	CC3E	Capture/Compare 3 output enable.
[7]	rw	1'h0	CC2NP	Capture/Compare 2 output Polarity.
[6]			RSVD	
[5]	rw	1'h0	CC2P	Capture/Compare 2 output Polarity.
[4]	rw	1'h0	CC2E	Capture/Compare 2 output enable.
[3]	rw	1'h0	CC1NP	Capture/Compare 1 output Polarity. CC1 channel configured as output: CC1NP must be kept cleared in this case. CC1 channel configured as input: This bit is used in conjunction with CC1P to define TI1FP1/TI2FP1 polarity. refer to CC1P description.
[2]			RSVD	

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	CC1P	Capture/Compare 1 output Polarity. CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode). 01: inverted/falling edge Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode). 10: reserved, do not use this configuration. 11: noninverted/both edges Circuit is sensitive to both TIxFP1 rising and falling edges (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode). This configuration must not be used for encoder mode.
[0]	rw	1'h0	CC1E	Capture/Compare 1 output enable. CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (CCR1) or not. 0: Capture disabled 1: Capture enabled
0x24			**CNT**	Counter
[31]	r	1'h0	UIFCPY	Value depends on IUFREMAP in CR1. If UIFREMAP = 1 UIFCPY: UIF Copy This bit is a read-only copy of the UIF bit of the ISR register
[30:16]			RSVD	
[15:0]	rw	16'h0	CNT	counter value
0x28			**PSC**	Prescaler
[31:16]			RSVD	
[15:0]	rw	16'h0	PSC	Prescaler value The counter clock frequency is equal to fCLK / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of EGR register or through trigger controller when configured in 'reset mode').
0x2C			**ARR**	Auto-reload register
[31:16]			RSVD	
[15:0]	rw	16'h0	ARR	Auto-reload value ARR is the value to be loaded in the actual auto-reload register.
0x30			**RCR**	Repetition counter register
[31:8]			RSVD	

Continued on the next page...

Table 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7:0]	rw	8'h0	REP	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event, any write to the RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to the number of PWM periods in edge-aligned mode.
0x34			**CCR1**	Capture/Compare register 1
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR1	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1). The CCR1 register is read-only and cannot be programmed.
0x38			**CCR2**	Capture/Compare register 2
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR2	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2). The CCR2 register is read-only and cannot be programmed.
0x3C			**CCR3**	Capture/Compare register 3
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR3	Capture/Compare value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC3 output. If channel CC3is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3). The CCR3 register is read-only and cannot be programmed.
0x40			**CCR4**	Capture/Compare register 4
[31:16]			RSVD	

Continued on the next page...

Table 9-3: **GPTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[15:0]	rw	16'h0	CCR4	Capture/Compare value 1. if CC4 channel is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC4 output. 2. if CC4 channel is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4). The CCR4 register is read-only and cannot be programmed.

9.3.5 Timer Cascading

Table 9-4: **Timer Cascading**

Lower-Level Timer	Cascading Port	Upper-Level Timer
ATIM1	ITR0	BTIM2
	ITR1	GPTIM2
	ITR2	GPTIM1
	ITR3	BTIM1
GPTIM1	ITR0	GPTIM2
	ITR1	BTIM2
	ITR2	ATIM1
	ITR3	BTIM1
GPTIM2	ITR0	GPTIM1
	ITR1	ATIM1
	ITR2	BTIM1
	ITR3	BTIM2
BTIM1	ITR0	BTIM2
	ITR1	GPTIM1
	ITR2	ATIM1
	ITR3	GPTIM2
BTIM2	ITR0	GPTIM1
	ITR1	BTIM1
	ITR2	GPTIM2
	ITR3	ATIM1

9.4 LPTIM

LPTIM1 and LPTIM2 are located in HPSYS_AON and can remain operational when HPSYS enters Low Power Mode (except for hibernate), with input and output connected to IO(PA) and Low Power IO(PA24~PA27) .

9.4.1 Introduction

LPTIM (Low Power Timer) is based on a 24-bit incrementing counter, capable of timing, generating output waveforms (output compare and PWM) , and waking up the system, among other functions. The counting clock can be the system clock, low power clock, IO input signal, or comparator output, and can support up to 128 times prescaling and up to 256 cycles of counting. Based on the counting results, it can generate PWM outputs, trigger interrupts, or produce wake-up signals to wake the system from Low Power Mode. When using IO input signals as the counting clock, it supports counting and generating wake-up signals independently of the internal clock.

9.4.2 Main Features

- 24-bit up-counting auto-reload counter, maximum count of 16777215(2^{24}-1)
- Counting Clock Selection
  - Internal clock, PCLK2, or low power clock
  - Optional edge-triggered IO input signal or comparator output; can utilize the internal clock for debouncing, or count independently without relying on the internal clock
- 8-level pre-scaler, with a counting clock division factor of 2 raised to the power of 0 to 7
- 1 to 256 loop counts
- Counting Mode
  - Continuous Counting Mode
  - Single Pulse Mode; counting ends after the loop count is completed
- Configurable polarity output mode
  - PWM output, with adjustable pulse width and period
  - Single toggle output
  - Single pulse or a specifed number of pulse outputs
- Trigger mode
  - Software Trigger
  - IO input signal edge-triggered, supporting debounce fltering
- Timeout detection; the counter resets with each external trigger
- An interrupt or wake-up signal is generated when the following events occur:
  - Update
  - Counter overflow
  - Output Compare
  - External trigger

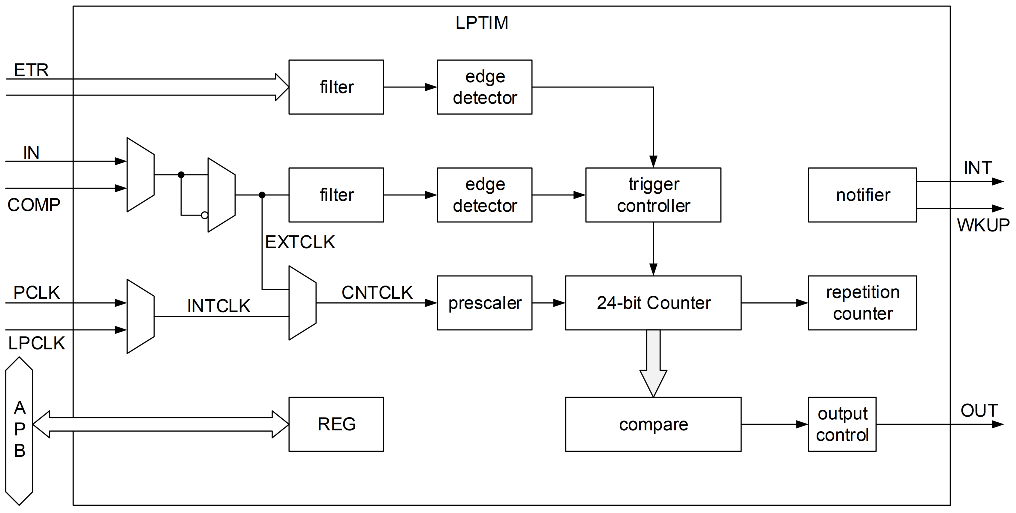


Figure 9-13: **LPTIM structure diagram**

9.4.3 LPTIM function description

9.4.3.1 counter

The functions of LPTIM are based on a 24 bit counter. The counter operates on event counting, including clock edges, external input edges, and comparator result edges.

Counting events will only be registered in the counter after undergoing pre-scaling processing. The number of pre-scales ranges from 1 to 128 ($2^{CFGR\_PRESCPSC}$), meaning that the counter's value will only change once after ($2^{CFGR\_PRESCPSC}$) counting events have occurred.

The counter is fxed in incrementing mode, counting from 0 to the auto-reload value ARR , and then restarting from 0 , which generates a counter overflow event.

The count value can be read through CNT . Since the counting clock is asynchronous with the APB clock, two consecutive readings must yield the same value to be considered a valid data read.

9.4.3.2 Counting Clock

The counting clock of the LPTIM CNTCLK can be selected from multiple sources. The most commonly used default mode is to select the internal low-power clock LPCLK , which allows the LPTIM to continue operating after the chip enters low-power sleep mode. When not in low-power sleep mode, the LPTIM can also select the internal PCLK as the clock. Additionally, the LPTIM can count using an external signal IN or a comparator output signal COMP without relying on the internal clock. The registers related to clock selection include CFGR_EXTCKSEL/INTCKSEL/CKSEL . The polarity of the external clock can be selected using CFGR_CKPOL .

When the internal clock is selected, it is also possible to count the events of the external signal IN or the comparator output signal COMP flipping, and to perform pre-fltering and edge selection processing. In this mode, the flipping frequency of the internal clock must be at least five times that of the external signal IN or the comparator output signal COMP flipping frequency.

9.4.3.3 Update Event(UEV)

The update event is used to signify the end of a counting unit. The most basic update event occurs each time the counter overflows (when repeat counting is disabled) . Update events can generate interrupts and wake-up signals, serving as the most fundamental notifcation function of the timer

9.4.3.4 Repeat Counting

If the Repeat Counter is configured (RCR>0) , it will decrement each time the counter overflows, and an Update Event will only occur when the Repeat Counter reaches 0 .

The current value of the Repeat Counter can be read through RCR . Since the counting clock is asynchronous with the APB clock, the values must match for two consecutive reads to be considered a valid data read.

9.4.3.5 Counter Trigger

The counter can be configured for single-level trigger mode(CFGR_TRIGEN=00)or dual-level trigger mode(CFGR_TRIGEN!=00).

Single-level trigger mode is software-triggered and includes both single trigger and continuous trigger types. Before triggering, the CR_ENABLE must be set to 1 to enable the counter. Then, set CR_SNGSTRT to 1 to initiate a single trigger, causing the counter to start immediately and stop after the Update Event; alternatively, set CR_CNTSTRT to 1 to initiate continuous triggering, causing the counter to start immediately and continue counting until it is disabled or reset.

The two-level trigger mode incorporates a hardware trigger mechanism in addition to software triggering, activating the counter only when the fltered ETRoptional edge arrives.

9.4.3.6 Timeout Monitoring

In the two-level trigger mode, if CFGR_TIMOUT is set to 1 , the counter will reset and restart each time a hardware trigger occurs. This feature can be utilized to monitor the interval between two consecutive trigger signals, generating a comparison or update event when the interval exceeds the expected duration.

9.4.3.7 PWM Output

LPTIM can generate a single-channel PWM output with controllable period and duty cycle to the OUT port. The period of the PWM output is determined by ARR , while the duty cycle is determined by CMP . The counter value CNT is compared with CMP to generate PWM , with polarity configured by CFGR_WAVEPOL . In single trigger mode, a single pulse or multiple pulses can be generated based on the value of RCR . In continuous trigger mode, a sustained PWM can be produced. If CFGR_WAVE is set to 1 , a single pulse waveform can be generated.

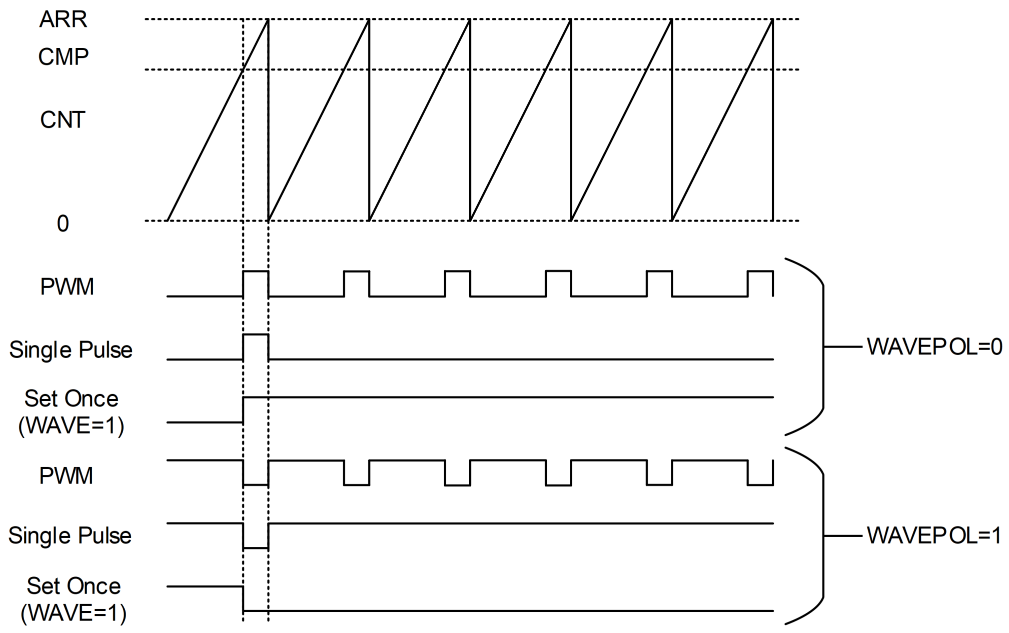


Figure 9-14: **PWM Output**

9.4.3.8 Notifcation Mechanism

LPTIM can generate notifcations such as interrupts and wake-up signals. Interrupts are generated only when the system is in a non-low power sleep state. Wake-up signals can be generated regardless of whether the system is in a low power sleep state and can wake the system. The events that can trigger notifcations primarily include overflow events, update events, trigger events, and comparator matches. The IER register can control whether various events generate interrupts and wake-ups. The status of each event can be queried in the ISR register.

9.4.4 LPTIM Register

LPTIM1 base address is 0x500C1000。

LPTIM2 base address is 0x500C2000。

Table 9-5: **LPTIM Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	LPTIM interrupt and status register
[31:11]			RSVD	
[10]	r	1’h0	OCWKUP	Indicates output compare wakeup occurred The OCWKUP bit is set by hardware when LPTIM_CNT register value reached the LPTIM_CMP register’s value. To clear OCWKUP, first write 0 to the OCWE bit in the LPTIM_IER register to disable, then write 1 to the WKUPCLR bit in the LPTIM_ICR register.
[9]	r	1’h0	OFWKUP	Indicates overflow wakeup occurred OFWKUP is set by hardware when LPTIM_CNT register’s value reached the LPTIM_ARR register’s value and count from zero again. To clear OFWKUP, first write 0 to the OFWE bit in the LPTIM_IER register to disable, then write 1 to the WKUPCLR bit in the LPTIM_ICR register.

Continued on the next page...

Table 9-5: **LPTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	r	1'h0	UEWKUP	Indicates update event wakeup occurred UEWKUP is set by hardware when an update event was generated (overflow occurred while repetition counter reached zero). To clear UEWKUP, first write 0 to the UEWE bit in the LPTIM_IER register to disable, then write 1 to the WKUPCLR bit in the LPTIM_ICR register.
[7:4]			RSVD	
[3]	r	1'h0	ET	External trigger edge event ET is set by hardware to inform application that a valid edge on the selected external trigger input has occurred. If the trigger is ignored because the timer has already started, then this flag is not set. ET flag can be cleared by writing 1 to the ETCLR bit in the LPTIM_ICR register.
[2]	r	1'h0	OC	Output compare match The OC bit is set by hardware to inform application that LPTIM_CNT register value reached the LPTIM_CMP register's value. OC flag can be cleared by writing 1 to the OCCLR bit in the LPTIM_ICR register.
[1]	r	1'h0	OF	Overflow occurred OF is set by hardware to inform application that LPTIM_CNT register's value reached the LPTIM_ARR register's value and count from zero again. OF flag can be cleared by writing 1 to the OFCLR bit in the LPTIM_ICR register.
[0]	r	1'h0	UE	LPTIM update event occurred UE is set by hardware to inform application that an update event was generated when overflow occurred while repetition counter reached zero. UE flag can be cleared by writing 1 to the UECLR bit in the LPTIM_ICR register.
0x04			**ICR**	LPTIM interrupt and status clear register
[31:9]			RSVD	
[8]	w	1'h0	WKUPCLR	wakeup status clear flag Writing 1 to this bit clears all wakeup status flags in the LPTIM_ISR register.
[7:4]			RSVD	
[3]	w	1'h0	ETCLR	External trigger valid edge clear flag Writing 1 to this bit clears the ET flag in the LPTIM_ISR register
[2]	w	1'h0	OCCLR	Output compare clear flag Writing 1 to this bit clears the OC flag in the LPTIM_ISR register
[1]	w	1'h0	OFCLR	Overflow clear flag Writing 1 to this bit clears the OF flag in the LPTIM_ISR register
[0]	w	1'h0	UECLR	Update event clear flag Writing 1 to this bit clear the UE flag in the LPTIM_ISR register.
0x08			**IER**	LPTIM interrupt and wakeup enable register
[31:11]			RSVD	
[10]	rw	1'h0	OCWE	Output compare Wakeup Enable 0: Output compare wakeup disabled 1: Output compare wakeup enabled
[9]	rw	1'h0	OFWE	Overflow Wakeup Enable 0: Overflow Wakeup disabled 1: Overflow Wakeup enabled
[8]	rw	1'h0	UEWE	Update event Wakeup enable 0: Update event Wakeup disabled 1: Update event Wakeup enabled
[7:4]			RSVD	
[3]	rw	1'h0	ETIE	External trigger valid edge Interrupt Enable 0: External trigger interrupt disabled 1: External trigger interrupt enabled

Continued on the next page...

Table 9-5: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	OCIE	Output compare Interrupt Enable 0: Output compare interrupt disabled 1: Output compare interrupt enabled
[1]	rw	1'h0	OFIE	Overflow Interrupt Enable 0: Overflow interrupt disabled 1: Overflow interrupt enabled
[0]	rw	1'h0	UEIE	Update event interrupt enable 0: Update event interrupt disabled 1: Update event interrupt enabled
0x0C			**CFGR**	LPTIM configuration register
[31:24]			RSVD	
[23]	rw	1'h0	COUNTMODE	counter mode in internal clock source mode (CKSEL=0). If CKSEL=1, this bit has no effect. 0: the counter is incremented following each internal clock pulse 1: the counter is incremented following each valid pulse on the external clock
[22]			RSVD	
[21]	rw	1'h0	WAVPOL	Waveform shape polarity The WAVEPOL bit controls the output polarity 0: The LPTIM output reflects the compare results between LPTIM_ARR and LPTIM_CMP registers 1: The LPTIM output reflects the inverse of the compare results between LPTIM_ARR and LPTIM_CMP registers
[20]	rw	1'h0	WAVE	Waveform shape The WAVE bit controls the output shape 0: Deactivate Set-once mode 1: Activate the Set-once mode
[19]	rw	1'h0	TIMOUT	Timeout enable The TIMOUT bit controls the Timeout feature 0: A trigger event arriving when the timer is already started will be ignored 1: A trigger event arriving when the timer is already started will reset and restart the LPTIM counter and the repetition counter
[18:17]	rw	2'h0	TRIGEN	Trigger enable and polarity The TRIGEN bits controls whether the LPTIM counter is started by an external trigger or not. If the external trigger option is selected, three configurations are possible for the trigger active edge: 00: software trigger (counting start is initiated by software) 01: rising edge is the active edge 10: falling edge is the active edge 11: both edges are active edges
[16]			RSVD	
[15:13]	rw	3'h0	TRIGSEL	Trigger selector The TRIGSEL bits select the trigger source that will serve as a trigger event for the LPTIM among the below 8 available sources: 000: lptim_ext0 001: lptim_ext1 010: lptim_ext2 011: lptim_ext3 100: lptim_ext4 101: lptim_ext5 110: lptim_ext6 111: lptim_ext7
[12]			RSVD	

Continued on the next page...

Table 9-5: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:9]	rw	3'h0	PRESC	Clock prescaler The PRESC bits configure the prescaler division factor. It can be one among the following division factors: 000: /1 001: /2 010: /4 011: /8 100: /16 101: /32 110: /64 111: /128
[8]	rw	1'h0	EXTCKSEL	External clock source selector 0: external clock source is from lptim_in 1: external clock source is from LPCOMP (if LPCOMP integrated)
[7:6]	rw	2'h0	TRGFLT	Configurable digital filter for trigger The TRGFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an internal trigger before it is considered as a valid level transition. An internal clock source must be present to use this feature 00: any trigger active level change is considered as a valid trigger 01: trigger active level change must be stable for at least 2 clock periods before it is considered as valid trigger. 10: trigger active level change must be stable for at least 4 clock periods before it is considered as valid trigger. 11: trigger active level change must be stable for at least 8 clock periods before it is considered as valid trigger.
[5]	rw	1'h0	INTCKSEL	Internal clock source selector 0: internal clock source is clk_lp 1: internal clock source is pclk2
[4:3]	rw	2'h0	CKFLT	Configurable digital filter for external clock The CKFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an external clock signal before it is considered as a valid level transition. An internal clock source must be present to use this feature 00: any external clock signal level change is considered as a valid transition 01: external clock signal level change must be stable for at least 2 clock periods before it is considered as valid transition. 10: external clock signal level change must be stable for at least 4 clock periods before it is considered as valid transition. 11: external clock signal level change must be stable for at least 8 clock periods before it is considered as valid transition.
[2:1]	rw	2'h0	CKPOL	Clock Polarity If LPTIM is clocked by an external clock source, CKPOL bits is used to configure the active edge or edges used by the counter: 00: the rising edge is the active edge used for counting 01: the falling edge is the active edge used for counting 10: both edges are active edges. When both external clock signal edges are considered active ones, the LPTIM must also be clocked by an internal clock source with a frequency equal to at least four time the external clock frequency. 11: not allowed

Continued on the next page...

Table 9-5: **LPTIM Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	CKSEL	Clock selector The CKSEL bit selects which clock source the LPTIM will use: 0: LPTIM is clocked by internal clock source, according to INTCKSEL 1: LPTIM is clocked by external clock source, according to EXTCKSEL
0x10			**CR**	LPTIM control register
[31:4]			RSVD	
[3]	rw	1'h0	COUNTRST	Counter reset This bit is set by software and cleared by hardware. When set to 1 this bit will trigger a synchronous reset of the CNT register. Due to the synchronous nature of this reset, it only takes place after a synchronization delay. COUNTRST must never be set to 1 by software before it is already cleared to 0 by hardware. Software should consequently check that COUNTRST bit is already cleared to 0 before attempting to set it to 1.
[2]	w	1'h0	CNTSTRT	Timer start in Continuous mode This bit is set by software and cleared by hardware. In case of software start (TRIGEN[1:0] = 00), setting this bit starts the LPTIM in Continuous mode. If the software start is disabled (TRIGEN[1:0] different than 00), setting this bit starts the timer in Continuous mode as soon as an external trigger is detected. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between ARR and CNT registers and the LPTIM counter keeps counting in Continuous mode.
[1]	w	1'h0	SNGSTRT	LPTIM start in Single mode This bit is set by software and cleared by hardware. In case of software start (TRIGEN[1:0] = 00), setting this bit starts the LPTIM in single pulse mode. If the software start is disabled (TRIGEN[1:0] different than 00), setting this bit starts the LPTIM in single pulse mode as soon as an external trigger is detected. If this bit is set when the LPTIM is in continuous counting mode, then the LPTIM will stop at the following match between ARR and CNT registers. If this bit is set simultaneously with CNTSTRT, then LPTIM will be in continuous counting mode.
[0]	rw	1'h0	ENABLE	LPTIM enable The ENABLE bit is set and cleared by software. 0:LPTIM is disabled 1:LPTIM is enabled
0x14			**CMP**	LPTIM compare register
[31:24]			RSVD	
[23:0]	rw	24'h0	CMP	Compare value CMP is the compare value used by the LPTIM.
0x18			**ARR**	LPTIM autoreload register
[31:24]			RSVD	
[23:0]	rw	24'h0	ARR	Auto reload value ARR is the autoreload value for the LPTIM. This value must be strictly greater than the CMP[15:0] value.
0x1C			**CNT**	LPTIM counter register
[31:24]			RSVD	
[23:0]	r	24'h0	CNT	Counter value When the LPTIM is running with an asynchronous clock, reading the CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical.
0x20			**RCR**	LPTIM repetition register
[31:8]			RSVD	

Continued on the next page...

Table 9-5: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7:0]	rw	8'h0	REP	Repetition register value REP is the repetition value for the LPTIM. Read REP will return left repetition times. It should be noted that for a reliable REP register read access, two consecutive read accesses must be performed and compared. A read access can be considered reliable when the values of the two consecutive read accesses are equal.

9.5 WDT

The chip contains 3 WDTs, with WDT1 located in HPSYS which ceases operation when HPSYS enters deepsleep, standby, or hibernate mode; WDT2 is located in LPSYS and ceases operation when LPSYS enters deepsleep, standby, or hibernate mode; IWDT is located in AON and continues to function after the chip enters low power mode. The specific reset scope for each WDT can be found in the Clock and Reset chapter.

9.5.1 Introduction

The watchdog timer, as a type of counter, is primarily used to reset the system after a predetermined time to prevent software hangs.

Basic functions of the watchdog timer:

- Supports two operating modes:
  - mode0
    - The wdt does not generate an interrupt; it directly resets the system after the set time is reached.
    - Supports up to a 24-bitcounter.
  - mode1
    - Divided into two counting segments; an interrupt is generated after the first segment's set time is reached, and the system is reset after the second segment's set time is reached.
    - Each time segment supports up to a 24-bitcounter.
- Supports write protection to prevent erroneous operations on the wdt.

9.5.2 Operating modes of the WDT

There are two resetgeneration modes based on requirements:

Mode 1: Counts for one round only, directly generating a resetsignal at the end of the count.

Mode 2: Counts for two rounds, generating an interrupt at the end of the first round and a resetsignal at the end of the second round.

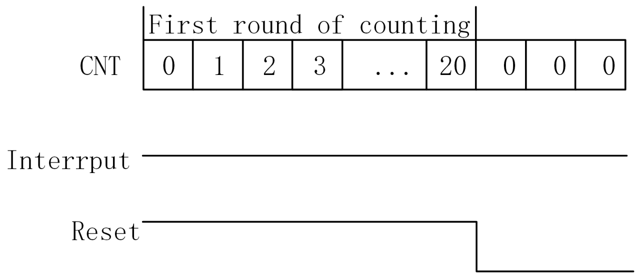


Figure 9-15: **The relationship between the interrupt and resetsignal generation in Mode 1 and the counter (assuming a timeout value of 20, withresetactive low)**

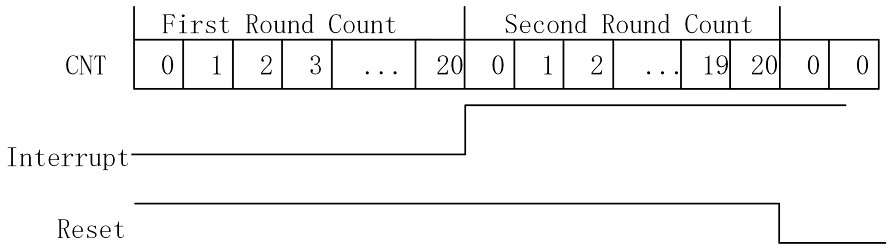


Figure 9-16: **The relationship between the interrupt and resetsignal generation in Mode 2 and the counter (assuming a timeout value of 20, withresetactive low).**

The ’dog feeding’ behavior in both modes:

In Mode 1, if ’feeding the dog’ occurs before the end of the first round of counting, the counter will restart counting from zero.

In mode 2 , if the first round of counting concludes before the ’dog is fed’, the counter will restart from zero; if there is no ’dog feeding’ action before the first round of counting concludes, the wdt will enter the second round of counting. At this point, either clearing the interrupt or performing the ’dog feeding’ action will allow the wdt to restart counting from the beginning of the first round.

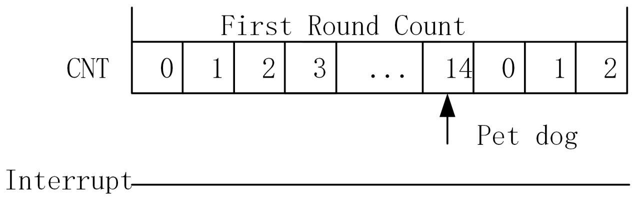


Figure 9-17: **Mode 1 The effect of the ’dog feeding’ action on the counter (assuming the timeout value is 20)**

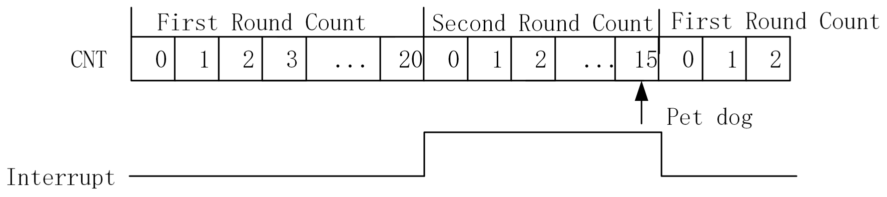


Figure 9-18: **Mode 2 The effect of the ’dog feeding’ operation on the counter during the second round of counting is the same as in the first round (assuming the timeout value is 20)**

Methods to stop wdtin two modes:

In mode 1, configure register0x0C(counter_control)=0x34before the count ends, andwdtwill stop.

In mode 2 , configure register 0x0C(counter_control)=0x34 before the first round count ends, andwdt will stop; if in the second round counting stage, it is necessary to clear the interrupt or perform the 'feed the dog' operation to return wdt to the first round count before configuring register 0x0C(counter_control)=0x34 to stop wdt.

Method for Clearing Interrupts:

In mode 2, wdt will generate an interrupt after the first round of counting is completed, with 0x14 WDT_SR indicating int_assert as 1. At this point, it can be cleared by configuring 0x10's WDT_IDR with int_clr, or by configuring 0xc WDT_CCR with counter_control set to 0x76, which means feeding the watchdog to clear.

9.5.2.1 WDT Register Configuration Process

1. Select the working mode of wdt as needed. Configure 0x08 for the response_mode register, set 0x0 to select mode 1, and set 0x1 to select mode 2.
2. Configure 0x00 for count_value_0 register (the timeout value for the first round of counting in both modes) and 0x04 for count_value_1 (register for the timeout value of the second round counter in mode 2
3. Configure 0x08 for reset lengthr equirements.
4. Configure 0x0c in the counter_control Register (= 0x76) to trigger the wdt to start working.
   The sequence of steps 1 to 3 is not mandatory, provided it is completed before step 4.

9.5.2.2 Note

1. The wdtprovides awrite protectfeature to prevent accidental rewriting of the configurations inwdt, as follows:
   Set 0x18 for wrpt Register to 0x58ab99fc. When wrpt_st in 0x18 Register is 1, it indicates that write protection is enabled. In this state, all registers in wdt cannot be rewritten, but read operations are unaffected. To disable write protection, set 0x18 for wrpt Register to 0x51ff8621.
2. wdt's register address 0x1c, sync_fg register set to 1 indicates that the start , stop , irq clear , and reset flag clear operations have been synchronized from pclk to wdt clk and are now effective.
3. rst_fg set to 1 indicates that this wdt has undergone a reset; this register is only valid for iwdt.
4. If 0xc cannot write to counter_control , first check if the corresponding sys rcc wdt clock enable register is configurd to 1.

9.5.3 WDT Register

WDT1 base address is 0x50094000.

IWDT base address is 0x500CC000.

Table 9-6: **WDT Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**WDT_CVR0**	WatchDog Counter Value 0
[31:24]			RSVD	
[23:0]	rw	24'hffffff	count_value_0	Count Value for 1st TimeOut
0x04			**WDT_CVR1**	WatchDog Counter Value 1
[31:24]			RSVD	
[23:0]	rw	24'hffffff	count_value_1	Count Value for 2nd TimeOut
0x08			**WDT_CR**	WatchDog Control Register

Continued on the next page...

Table 9-6: WDT Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:5]			RSVD	
[4]	rw	1'b1	response_mode	0:reset only, 1:interrupt and reset
[3]			RSVD	
[2:0]	rw	3'b000	reset_length	reset pulse length in number of wdt clock cycles
0x0C			**WDT_CCR**	WatchDog Counter Control Register
[31:8]			RSVD	
[7:0]	rw	8'h0	counter_control	SinglePulse /Write 8'h76 to restart, write8'h34 to stop, else do nothing
0x10			**WDT_ICR**	WatchDog Interrupt Clear Register
[31:1]			RSVD	
[0]	w1c	1'b0	int_clr	SinglePulse /A pulse to clear interrupt
0x14			**WDT_SR**	WatchDog Status Register
[31:2]			RSVD	
[1]	r	1'b0	wdt_active	Watchdog runs when 1, else 0
[0]	r	1'b0	int_assert	Interrupt assert when 1
0x18			**WDT_WP**	WatchDog Write Protect Register
[31]	r	1'b0	wrpt_st	1 indicates write protect is active
[30:0]	w	31'h0	wrpt	write 0x58ab99fc generate write_protect, write 0x51ff8621 to release
0x1C			**WDT_FG**	WatchDog Flag Register
[31:4]			RSVD	
[3]	r	1'b0	sync_fg	1 indicates one transition from system clk to wdt clk has complicated
[2]	w1c	1'b0	sync_fg_clr	SinglePulse/A pulse to clear sync flag
[1]	r	1'b0	rst_fg	1 indicates wdt has already reset system
[0]	w1c	1'b0	rst_fg_clr	SinglePulse/A pulse to clear reset flag

9.6 RTC

9.6.1 RTC Register

RTC base address is 0x500CB000。

Table 9-7: RTC Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**TR**	Time Register
[31]	rw	1'h0	PM	AM/PM notation 0: AM 1: PM
[30:29]	rw	2'h0	HT	Hour tens in BCD format
[28:25]	rw	4'h0	HU	Hour units in BCD format
[24:22]	rw	3'h0	MNT	Minute tens in BCD format
[21:18]	rw	4'h0	MNU	Minute units in BCD format
[17:15]	rw	3'h0	ST	Second tens in BCD format
[14:11]	rw	4'h0	SU	Second units in BCD format
[10]			RSVD	
[9:0]	r	10'h0	SS	Sub-second counter
0x04			**DR**	Date Register
[31]	r	1'h0	ERR	reserved for debug
[30:25]			RSVD	
[24]	rw	1'h0	CB	century bit, 0 - 2000s, 1 - 1900s/2100s
[23:20]	rw	4'h0	YT	Year tens in BCD format

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19:16]	rw	4'h0	YU	Year units in BCD format
[15:13]	rw	3'h1	WD	Week day units 000: forbidden 001: Monday ... 111: Sunday
[12]	rw	1'h0	MT	Month tens in BCD format
[11:8]	rw	4'h1	MU	Month units in BCD format
[7:6]			RSVD	
[5:4]	rw	2'h0	DT	Date tens in BCD format
[3:0]	rw	4'h1	DU	Date units in BCD format
0x08			**CR**	Control Register
[31:22]			RSVD	
[21]	rw	1'h0	COE	
[20:19]	rw	2'h0	OSEL	
[18]	rw	1'h0	POL	
[17]	rw	1'h0	COSEL	
[16]	rw	1'h0	BKP	
[15]	rw	1'h0	SUB1H	
[14]	rw	1'h0	ADD1H	
[13]	rw	1'h0	TSIE	
[12]	rw	1'h0	WUTIE	
[11]	rw	1'h0	ALRMIE	
[10]	rw	1'h0	TSE	
[9]	rw	1'h0	WUTE	
[8]	rw	1'h0	ALRME	
[7]			RSVD	
[6]	rw	1'h0	FMT	
[5]	rw	1'h0	BYPSHAD	
[4]	rw	1'h0	REFCKON	
[3]	rw	1'h0	TSEDGE	
[2]			RSVD	
[1]	rw	1'h0	WUCKSEL	
[0]	rw	1'h0	LPCKSEL	select clk_rtc 0: lcr10 1: lxt32
0x0C			**ISR**	Initialization and Status Register
[31:11]			RSVD	
[10]	rw	1'h0	INIT	
[9]	r	1'h0	INITF	
[8]	r	1'h0	INITS	
[7]	rw0c	1'h0	RSF	
[6]	r	1'h0	SHPF	
[5]	r	1'h0	TSOVF	
[4]	rw0c	1'h0	TSF	
[3]	rw0c	1'h0	WUTF	
[2]	r	1'h1	WUTWF	
[1]	rw0c	1'h0	ALRMF	
[0]	r	1'h1	ALRMWF	
0x10			**PSCLR**	Prescaler Register
[31:24]	rw	8'h80	DIVA_INT	
[23:10]	rw	14'h0	DIVA_FRAC	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9:0]	rw	10’h100	DIVB	
0x14			**WUTR**	Wakeup Timer Register
[31:18]			RSVD	
[17:0]	rw	18’h3ffff	WUT	
0x18			**ALRMTR**	Alarm Time Register
[31]	rw	1’h0	PM	
[30:29]	rw	2’h0	HT	
[28:25]	rw	4’h0	HU	
[24:22]	rw	3’h0	MNT	
[21:18]	rw	4’h0	MNU	
[17:15]	rw	3’h0	ST	
[14:11]	rw	4’h0	SU	
[10]			RSVD	
[9:0]	rw	10’h0	SS	
0x1C			**ALRMDR**	Alarm Date Register
[31:30]			RSVD	
[29]	rw	1’h0	MSKWD	
[28]	rw	1’h0	MSKM	
[27]	rw	1’h0	MSKD	
[26]	rw	1’h0	MSKH	
[25]	rw	1’h0	MSKMN	
[24]	rw	1’h0	MSKS	
[23:20]	rw	4’h0	MSKSS	
[19:16]			RSVD	
[15:13]	rw	3’h1	WD	
[12]	rw	1’h0	MT	
[11:8]	rw	4’h1	MU	
[7:6]			RSVD	
[5:4]	rw	2’h0	DT	
[3:0]	rw	4’h1	DU	
0x20			**SHIFTR**	Shift Control Register
[31]	rw	1’b0	ADD1S	
[30:10]			RSVD	
[9:0]	rw	10’h0	SUBFS	
0x24			**TSTR**	Timestamp Time Register
[31]	r	1’h0	PM	
[30:29]	r	2’h0	HT	
[28:25]	r	4’h0	HU	
[24:22]	r	3’h0	MNT	
[21:18]	r	4’h0	MNU	
[17:15]	r	3’h0	ST	
[14:11]	r	4’h0	SU	
[10]			RSVD	
[9:0]	r	10’h0	SS	
0x28			**TSDR**	Timestamp Date Register
[31:16]			RSVD	
[15:13]	r	3’h1	WD	
[12]	r	1’h0	MT	
[11:8]	r	4’h1	MU	
[7:6]			RSVD	
[5:4]	r	2’h0	DT	
[3:0]	r	4’h1	DU	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x2C			**OR**	Option Register
[31:2]			RSVD	
[1]	rw	1'h0	RTC_OUT_RMP	
[0]	rw	1'h0	RTC_ALARM_TYPE	
0x30			**BKP0R**	Backup 0 Register
[31:0]	rw	32'h0	BKP	
0x34			**BKP1R**	Backup 1 Register
[31:0]	rw	32'h0	BKP	
0x38			**BKP2R**	Backup 2 Register
[31:0]	rw	32'h0	BKP	
0x3C			**BKP3R**	Backup 3 Register
[31:0]	rw	32'h0	BKP	
0x40			**BKP4R**	Backup 4 Register
[31:0]	rw	32'h0	BKP	
0x44			**BKP5R**	Backup 5 Register
[31:0]	rw	32'h0	BKP	
0x48			**BKP6R**	Backup 6 Register
[31:0]	rw	32'h0	BKP	
0x4C			**BKP7R**	Backup 7 Register
[31:0]	rw	32'h0	BKP	
0x50			**BKP8R**	Backup 8 Register
[31:0]	rw	32'h0	BKP	
0x54			**BKP9R**	Backup 9 Register
[31:0]	rw	32'h0	BKP	
0x58			**PBRCR**	PBR Control Register
[31:8]			RSVD	
[7:4]	rw	4'b0	DBG_SEL	reserved for debug
[3:2]			RSVD	
[1]	rw	1'b1	SNS	reserved for debug
[0]	rw	1'b1	RTO	reserved for debug
0x5C			**PBR0R**	PBR0 Register for PA24
[31]	rw	1'b0	FORCE1	1: force output value to 1 when output enabled (PBR0R_OE=1)
[30:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR0R_OUT 1: clk_rtc 2: selected by CR1_PINOUT_SEL0 in HPSYS_AON 3: selected by CR1_PINOUT_SEL1 in HPSYS_AON others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b0	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR0R_SEL is 0
0x60			**PBR1R**	PBR1 Register for PA25
[31:15]			RSVD	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR1R_OUT 1: clk_rtc 2: selected by CR1_PINOUT_SEL0 in HPSYS_AON 3: selected by CR1_PINOUT_SEL1 in HPSYS_AON others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b0	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR1R_SEL is 0
0x64			**PBR2R**	PBR2 Register for PA26
[31:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR2R_OUT 1: clk_rtc 2: selected by CR1_PINOUT_SEL0 in HPSYS_AON 3: selected by CR1_PINOUT_SEL1 in HPSYS_AON others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b0	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR2R_SEL is 0
0x68			**PBR3R**	PBR3 Register for PA27
[31:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR3R_OUT 1: clk_rtc 2: selected by CR1_PINOUT_SEL0 in HPSYS_AON 3: selected by CR1_PINOUT_SEL1 in HPSYS_AON others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b0	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR3R_SEL is 0
0x6C			**PAWK1R**	PA Wakeup Register 1
[31:17]			RSVD	
[16:0]	rw	17'h40	PE	Pull enable of wakeup pin, active high. Keep effective during hibernate mode. Caution: Pull-functions of all pins can be configured in HPSYS_PINMUX registers, but in hibernate mode, such information will be lost, which may lead to wakup pin leakage. Pull-functions set here will keep active in hibernate mode for wakeup pins to avoid leakage. bit 6: pull enable of PA34 bit 7: pull enable of PA35 ...... bit 16: pull enable of PA44
0x70			**PAWK2R**	PA Wakeup Register 2
[31:17]			RSVD	
[16:0]	rw	17'h0	PS	Pull select of wakeup pin. Logic HIGH selects pull-up, logic LOW select pull-down. Keep effective during hibernate mode. bit 6: pull select of PA34 bit 7: pull select of PA35 ...... bit 16: pull select of PA44
0x74			**PAWK3R**	PA Wakeup Register 3
[31:17]			RSVD	
[16:0]	rw	17'h1ffff	IS	Input Select of wakeup pin. Logic LOW selects CMOS input, logic HIGH selects Schmitt input. Keep effective during hibernate mode. bit 6: input select of PA34 bit 7: input select of PA35 ...... bit 16: input select of PA44

10 Graphics

10.1 ePicasso™ High-Performance 2.5D Graphics Engine

In In 2.5D image processing, many common image operations can consume a signifcant amount of CPU computational resources. The ePicasso™ is an acceleration engine specifically designed for 2.5D image operations, capable of providing exponential speed improvements for common functions such as layer overlay, scaling, and rotation in 2.5D image processing. Additionally, the ePicasso™ is compatible with various common RGB image formats, simplifying the conversion of different image formats within the system.

10.1.1 Layer Overlay

ePicasso™ supports up to two foreground layers, one dedicated mask layer, and one monochrome background layer for overlay, with input and output formats including the commonly used RGB565.、RGB888、ARGB8565、ARGB8888、L8、A8、A4、A2、YUV . Each foreground layer features independent overlay modes and overlay areas, while the mask layer is primarily used to extract specific shapes from the image. Additionally, each layer offers separate filter configuration options, enabling the layer to flter out a specific color, a feature that can be utilized for simple image capture.

10.1.2 Graphic Scaling

ePicasso™ includes a layer known as the functional layer, which, in addition to supporting overlay functionality, can also perform graphic scaling. The maximum scaling ratio can reach 1024 times, with a precision of 1/65536 . In the X and Y directions, the scaling ratios can be configured independently to accommodate various requirements.

10.1.3 Graphic Rotation

The functional layer of ePicasso™ not only supports scaling but also enables high-precision image rotation. Users can customize the sine and cosine values of the rotation angle to meet the requirements for any angle of rotation. The rotation and scaling functions can be enabled simultaneously, allowing both operations to be completed at once, thereby enhancing image processing performance.

10.2 LCDC

10.2.1 Introduction

LCDC stands for LCD Controller , whose primary function is to read image data from memory and then send the data to the corresponding screen based on different screen interfaces. LCDC supports a wide variety of screen interfaces, including:

- DBI/8080 Interface: Designed for medium to low resolution displays with GRAM
- SPI/DSPI/QSPI Interface: Intended for small, medium to low resolution screens with GRAM for IoT applications
- JDI Parallel Interface: Designed for JDI vendor-specific reflective displays

In addition to a comprehensive set of interfaces, the LCDC also supports a wide range of image data formats, including RGB565, RGB888, and ARGB8888. Beyond its basic functionality, the LCDC can also facilitate simple layer blending and perform basic image processing acceleration.

10.2.2 Architecture Introduction

The basic architecture diagram of the LCDCis illustrated in the fgure below:

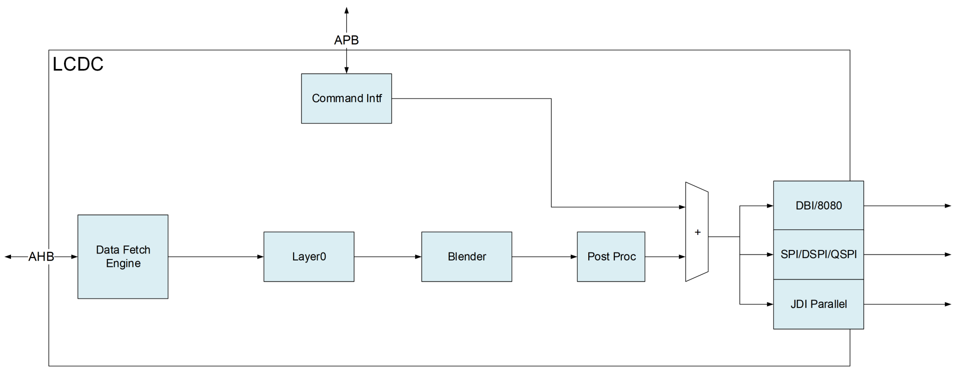


Figure 10-1: LCDC Architecture Diagram

The fgure contains two main data paths.

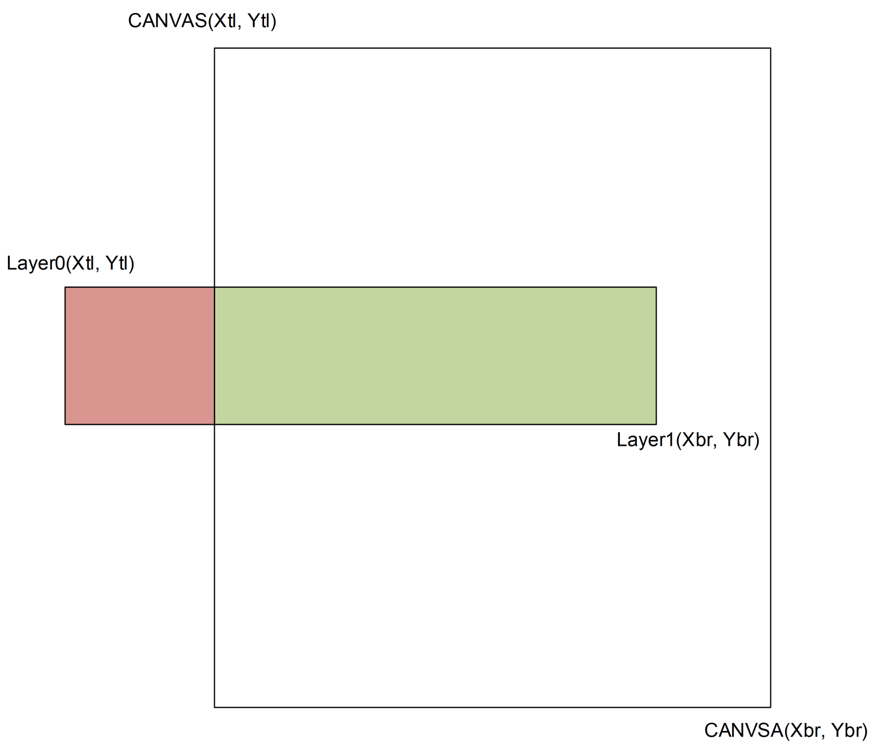
1. Configuration Path: The upper layer transmits screen configuration commands via the APB bus. Upon receiving these commands, the LCDC delivers the corresponding data to the appropriate screen interface through the Command Intf module. This path operates at a lower speed and is primarily utilized for screen initialization configuration and basic functionality verifcation.
2. Image Transmission Path: The Data Fetch Engine of the LCDC retrieves image data from memory via the AHB bus and sends it to Layer0 according to the configuration. When Once the layer0 data preparation is complete, the Blender mixes the image data with the background color and transmits it to the corresponding screen interface. This path serves as the primary route for the LCDC to send image data to the screen. Additionally, it is important to note that when Layer0 is disabled, the Blender can be configured to send monochrome data to the screen.

10.2.3 Configuration Process

The configuration of the LCDC is primarily divided into two parts: the first part involves layer configuration, which pertains to the configuration of two Layers and the Blender ; the second part involves interface configuration, which corresponds to the parameter settings for different interfaces.

10.2.3.1 Layer Configuration

Layer configuration includes the configuration of Layer0 and the Blender . Users can configure the Layer0_CONFIG register to enable Layer0 and set its color format and blending coeffcient. When the layer data is sent to the Blender , it will perform blending operations based on the layer coordinates and canvas coordinates configured by the user. Each layer has its own independent rectangular area, and the canvas also has its own rectangular area; the Blender only calculates the image in the overlapping regions. For specifics, please refer to the fgure below:



In the image CANVAS , both Layer0 and Layer1 have two sets of coordinates representing their respective rectangular areas. (Xtl, Ytl) denotes the coordinates of the top-left corner of the rectangle, while (Xbr, Ybr) denotes the coordinates of the bottom-right corner. It can be observed that Layer0 consists of two parts, with the red part located outside the CANVAS , thus it will not be included in the calculations. The green area overlaps with the CANVAS , so it will undergo blending calculations with the CANVAS . The remaining white area of the CANVAS does not overlap with any layers, so it will directly output the background color of the CANVAS.

In most scenarios, if the LCDC only outputs the image frame buffer (FrameBuffer) to the screen, it is sufcient to configure the layer coordinates to align with those of the CANVAS.

10.2.3.2 Interface Configuration

Due to signifcant differences in transmission protocols across various interfaces, the corresponding configuration parameters also vary. The following lists the different configuration parameters based on the respective interfaces.

DBI/8080 Interface:

DBI/8080 interface is used to drive screens equipped with GRAM . According to the MIPI protocol, the DBI parallel interface is further divided into Type A and Type B categories. Although the signals differ slightly, they can be logically converted between each other. When selecting the LCDC interface, you may also choose the corresponding Type A and Type B .

The main parameter configuration for the DBIinterface is referenced in the table below:

Parameter Name	Description	Configuration
PWH	WRX/RDX Signal Inactive State Cycle Count	The clock corresponding to the cycle is the system clock.
PWL	WRX/RDX Signal Active State Cycle Count	The clock corresponding to the cycle is the system clock.
TAH	WRX/RDX Signal Inactive Arrival CSX Signal Inactive Delay Cycle Count	The clock corresponding to the cycle is the system clock.
TAS	CSX SignalActiveArrival WRX/RDX Signal Active Delay Cycle Count	The clock corresponding to the cycle is the system clock.

The above parameters are the main parameters for configuring the DBI interface, where PWH and PWL determine the rate of the DBI interface, and their sum represents the cycle count for a single data transmission.

Another point to note is that all signals of the DBI interface are active low by default. If it is necessary to adjust the active phase, this can be accomplished by configuring the corresponding signal's POL register to invert the phase.

In addition to the image transmission path, the DBI interface also supports configuration paths. After setting up the DBI interface, the user can first configure the value to be written into the LCD_WR register, and then trigger read/write operations using the WR_TRIG and RD_TRIG registers, obtaining the read value through the LCD_RD register.

SPI/DSPI/QSPI Interface:

SPI, DSPI, and QSPI are all types of SPI interfaces for displays, differing primarily in the number of data lines. SPI typically has a single data line, while DSPI has two, and QSPI has four. In addition to the difference in the number of data lines, the SPI interface can also be categorized based on the D/C (Data/Command selection) signal transmission method into 3-wire SPI and 4-wire SPI. In 3-wire SPI, the D/C is transmitted as a single data bit through SDO, whereas in 4-wire SPI, there is a separate line to indicate the D/C. Consequently, 3-wire SPI has a slightly lower effective bandwidth compared to 4-wire SPI.

The main parameter configuration for the SPIinterface is detailed in the table below:

Parameter Name	Description	Configuration
CLK_DIV	SPI Clock Division Ratio	The source clock is the system clock.
LINE	SPI Mode	Different modes include 3-wire/4-wire as well as corresponding single data line, dual data line, and dead data line modes.
SPI_CS_AUTO_DIS	SPI CS Automatic Stop	During the data transmission phase, if the bus is busy and the LCDC fails to promptly acquire data for the screen, it will automatically control the SPI interface's CS signal, placing it in a non-enabled state.
SPI_CLK_AUTO_DIS	SPI 时 Clock Automatic Stop	During the data transmission phase, if the bus is busy and the LCDC fails to promptly acquire data for the screen, it will automatically stop the clock signal of the SPI interface.

In addition to the main parameters mentioned above, there are additional parameters primarily used for the read operations of the SPI interface, as well as the phase of the SPI signal. For specifics, please refer to the descriptions in the register table.

Similar to the DBI interface, the SPI interface also supports configuration pathways. After selecting the SPI interface, users can trigger read and write operations through the WR_TRIG and RD_TRIG registers, just as they would with DBI. For SPI, users also need to configure WR_LEN and RD_LEN, which determine the number of bytes read or written in a single operation through the SPI interface. A value of 0 indicates 1 byte, with a maximum of 4 bytes.

JDI Parallel Interface:

JDI Parallel interface is a dedicated interface for JDI's reflective screens. As a parallel interface, JDI Parallel can support higher resolution screens with richer colors. It has a variety of configuration parameters, which should be referenced in the JDI Parallel interface protocol for appropriate parameter configuration. The configuration parameters are as follows:

Parameter Name	Description	Configuration
MAX_LINE	Maximum Number of Rows	\
MAX_COL	Maximum Number of Columns	\
ST_LINE	Starting Row Number	Number of Valid Data Rows in the First Row
END_LINE	Ending Row Number	Number of Valid Data Rows in the Last Row
ST_COL	Starting Column Number	Number of Valid Data Columns in the First Column
END_COL	Ending Column Number	Number of Valid Data Columns in the Last Column
HCK_WIDTH	HCK Signal Width	Based on System Clock Cycles
HST_WIDTH	HST Signal Width	Based on System Clock Cycles
VCK_WIDTH	VCK Signal Width	Based on System Clock Cycles
VST_WIDTH	VST Signal Width	Based on System Clock Cycles
VCK_DLY	Delay from VST to VCK Signal	Based on System Clock Cycles
HST_DLY	Delay from VCK to HST Signal	Based on System Clock Cycles
HCK_DLY	Delay from HST to HCK Signal	Based on System Clock Cycles
ENB_ST_COL	ENB Signal starting column number	ENB First valid column number of the signal
ENB_END_COL	ENB Signal ending column number	ENB Last valid column number of the signal
ENB_ST_LINE	ENB Signal starting row number	ENB First valid row number of the signal
ENB_END_LINE	ENB Signal ending column number	ENB Last valid row number of the signal
DP_MODE	DP Mode	Supports dual pixel mode

JDI Parallel The interface configuration parameters are numerous and require reference to the JDI Parallel interface documentation for accurate parameter configuration. The JDI Parallel interface is similar to the DPI interface; once enabled, it continuously reads data from the FrameBuffer address to send to the screen. The JDI Parallel interface will only stop after the current frame data transmission is complete when disabled.

10.2.4 LCDC Register

LCDC base address is 0x50008000.

Table 10-1: **LCDC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**COMMAND**	
[31:2]			RSVD	
[1]	rw	1’h0	reset	1: reset the whole graphics 0: release the reset
[0]	w1t	1’h0	start	write 1 to trigger the lcd interface block
0x04			**STATUS**	
[31:3]			RSVD	
[2]	r	1’h0	JDI_PAR_RUN	JDI parallel interface is running
[1]	r	1’h0	DPI_RUN	DPI interface is running
[0]	r	1’h0	LCD_BUSY	LCS controll busy flag
0x08			**IRQ**	
[31:23]			RSVD	
[22]	rw1c	1’h0	LINE_DONE_RAW_STAT	raw_status of line process done interrupt
[21]	rw1c	1’h0	JDI_PAR_UDR_RAW_STAT	raw_status of jdi parallel interface under run interrupt
[20]	rw1c	1’h0	JDI_PARL_INTR_RAW_STAT	raw_status of jdi parallel interface line interrupt
[19]	rw1c	1’h0	DPI_UDR_RAW_STAT	raw status of dpi under run interrupt
[18]	rw1c	1’h0	DPIL_INTR_RAW_STAT	raw status of dpi line interrupt

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[17]	rw1c	1'h0	ICB_OF_RAW_STAT	raw status of icb overflow interrupt
[16]	rw1c	1'h0	EOF_RAW_STAT	raw status of end of frame interrupt
[15:7]			RSVD	
[6]	rw1c	1'h0	LINE_DONE_STAT	line process done interrupt, masked by mask register
[5]	rw1c	1'h0	JDI_PAR_UDR_STAT	jdi parallel interface under run interrupt, masked by mask register
[4]	rw1c	1'h0	JDI_PARL_INTR_STAT	jdi parallel interface line interrupt, masked by mask register
[3]	rw1c	1'h0	DPI_UDR_STAT	dpi under run interrupt, masked by mask register
[2]	rw1c	1'h0	DPIL_INTR_STAT	dpi line interrupt, masked by mask register
[1]	rw1c	1'h0	ICB_OF_STAT	icb overflow interrupt, masked by mask register
[0]	rw1c	1'h0	EOF_STAT	end of frame interrupt, masked by mask register
0x0C			**SETTING**	
[31:27]			RSVD	
[26:16]	rw	11'h1ff	LINE_DONE_NUM	line number of line process done interrupt
[15:9]			RSVD	
[8]	rw	1'h1	AUTO_GATE_EN	auto clock gating enable
[7]			RSVD	
[6]	rw	1'h0	LINE_DONE_MASK	line process done interrupt, 0: mask the interrupt
[5]	rw	1'h0	JDI_PAR_UDR_MASK	jdi parallel interface under run interrupt mask, 0: mask the interrupt
[4]	rw	1'h0	JDI_PARL_INTR_MASK	jdi parallel interface line interrupt, 0: mask the interrupt
[3]	rw	1'h0	DPI_UDR_MASK	dpi under run interrupt mask, 0: mask the interrupt
[2]	rw	1'h0	DPIL_INTR_MASK	dpi line interrupt, 0: mask the interrupt
[1]	rw	1'h0	ICB_OF_MASK	icb overflow interrupt mask, 0: mask the interrupt
[0]	rw	1'h0	EOF_MASK	end of frame interrupt mask, 0: mask the interrupt
0x10			**CANVAS_TL_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	
0x14			**CANVAS_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	
0x18			**CANVAS_BG**	
[31:28]			RSVD	
[27]	rw	1'h0	H_MIRROR	set 1 to do horizontal mirror for output image
[26]	rw	1'h0	LB_BYPASS	line buffer bypass. Set 1 to bypass line buffer.
[25]	rw	1'h0	ALL_BLENDING_BYPASS	if this bit is set, lcdc is in pure dma mode. No blending operation.
[24]	rw	1'h0	BG_BLENDING_BYPASS	if this bit is set, the layer is not blending with background. The alpha value will be reserved to output.
[23:16]	rw	8'h0	RED	Red color
[15:8]	rw	8'h0	GREEN	green color
[7:0]	rw	8'h0	BLUE	blue color
0x1C			**LAYER0_CONFIG**	
[31]			RSVD	
[30]	rw	1'h0	V_MIRROR	set 1 to do vertical mirror for the layer
[29]	rw	1'h0	ALPHA_BLEND	set 1 to enable alpha blending mode. Use layer alpha as blending factor for image with Alpha. Alpha_out = Layer_alpha * Image_alpha
[28]	rw	1'h0	ACTIVE	layer active flag

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[27]	rw	1'h0	LINE_FETCH_MODE	line fetch mode 0: address skip every single line 1: address skip every two line
[26]	rw	1'h0	PREFETCH_EN	preload 64 bytes extra data when reading pixel from memory
[25:13]	rw	13'h0	WIDTH	source image width(including padding), unit is bytes
[12]	rw	1'h0	FILTER_EN	layer color filter enable
[11:4]	rw	8'h0	ALPHA	layer alpha value
[3]	rw	1'h0	ALPHA_SEL	alpha selection 1'b0: select alpha according to image format 1'b1: select layer alpha
[2:0]	rw	3'h0	FORMAT	overlay layer input format 3'h0: RGB565 3'h1: RGB888 3'h2: ARGB8888 3'h3: ARGB8565 3'h4: RGB332 3'h5: A8 3'h6: L8 others: reserved
0x20			**LAYER0_TL_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	Coordinate X-value
0x24			**LAYER0_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	Coordinate X-value
0x28			**LAYER0_FILTER**	
[31:24]	rw	8'h0	FILTER_MASK	layer color filter mask
[23:16]	rw	8'h0	FILTER_R	filter r color
[15:8]	rw	8'h0	FILTER_G	filter g color
[7:0]	rw	8'h0	FILTER_B	filter b color
0x2C			**LAYER0_SRC**	
[31:0]	rw	32'h0	ADDR	source image RGB data address[31:0]. For RGB565 format, address should be aligned to halfword. For ARGB8888 format, address should be aligned to word.
0x30			**LAYER0_FILL**	
[31:26]			RSVD	
[25]	rw	1'h0	ENDIAN	input 565 data format endian 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]
[24]	rw	1'h0	BG_MODE	not used
[23:16]	rw	8'h0	BG_R	background r color
[15:8]	rw	8'h0	BG_G	background g color
[7:0]	rw	8'h0	BG_B	background b color
0x34			**LAYER0_DECOMP**	
[31:24]			RSVD	
[23:13]	rw	11'h0	col_size	number of colums in a line of original image, max column size is 1024
[12:1]	rw	12'h0	target_words	size of a single channel data before decompression. Unit is half word. Each line has 3 channels. So for each line, the compressed data size is target_words * 3 * 2 bytes.

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1’h0	enable	decompression enable
0x38			**LAYER0_DECOMP_CFG0**	
[31:20]	rw	12’h10	cfg0_reserved	
[19:16]	rw	4’h5	lossless_qidx2	condition to decrease qidx
[15:12]	rw	4’h5	lossless_qidx1	up level for adjusted qidx value for low quality block
[11:8]	rw	4’h9	use_lossless_qidx	condition to increase qidx
[7:4]	rw	4’h8	extra_threshold	the threshold to distinguish high/low quality block
[3:0]	rw	4’h2	extra_high	extra bit for high quality bit
0x3C			**LAYER0_DECOMP_CFG1**	
[31:28]	rw	4’h8	extra_low	extra bit for low quality block
[27:24]	rw	4’h0	block_min_qidx	minimum qidx for block mode
[23:20]	rw	4’h0	line_min_qidx	minimum qidx for line mode
[19:16]	rw	4’h2	failover_bits_b	failover compression mode target bits(Blue)
[15:12]	rw	4’h3	failover_bits_g	failover compression mode target bits(Green)
[11:8]	rw	4’h3	failover_bits_r	failover compression mode target bits(Red)
[7:2]	rw	6’h1	cfg1_reserved	
[1]	rw	1’b1	dither	dithering function 0: off 1: on
[0]	rw	1’b1	block_width	block_size in pixel unit. 0: 16 pixels 1: 32 pixels Small block size will cause more blocks and more bits to store block information.
0x40			**LAYER0_DECOMP_STAT**	
[31:7]			RSVD	
[6:0]	r	7’h0	buf_max_depth	buf max usage
0x60			**LAYER1_CONFIG**	
[31]			RSVD	
[30]	rw	1’h0	V_MIRROR	set 1 to do vertical mirror for the layer
[29]	rw	1’h0	ALPHA_BLEND	set 1 to enable alpha blending mode. Use layer alpha as blending factor for image with Alpha. Alpha_out = Layer_alpha * Image_alpha
[28]	rw	1’h0	ACTIVE	layer active flag
[27]	rw	1’h0	LINE_FETCH_MODE	line fetch mode 0: address skip every single line 1: address skip every two line
[26]	rw	1’h0	PREFETCH_EN	preload 64 bytes extra data when reading pixel from memory
[25:13]	rw	13’h0	WIDTH	source image width(including padding), unit is bytes
[12]	rw	1’h0	FILTER_EN	layer color filter enable
[11:4]	rw	8’h0	ALPHA	layer alpha value
[3]	rw	1’h0	ALPHA_SEL	alpha selection 1’b0: select alpha according to image format 1’b1: select layer alpha
[2:0]	rw	3’h0	FORMAT	overlay layer input format 3’h0: RGB565 3’h1: RGB888 3’h2: ARGB8888 3’h3: ARGB8565 3’h4: RGB332 3’h5: A8 3’h6: L8 others: reserved
0x64			**LAYER1_TL_POS**	

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	Coordinate X-value
0x68			**LAYER1_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	Coordinate X-value
0x6C			**LAYER1_FILTER**	
[31:24]	rw	8'h0	FILTER_MASK	layer color filter mask
[23:16]	rw	8'h0	FILTER_R	filter r color
[15:8]	rw	8'h0	FILTER_G	filter g color
[7:0]	rw	8'h0	FILTER_B	filter b color
0x70			**LAYER1_SRC**	
[31:0]	rw	32'h0	ADDR	source image RGB data address[31:0]. For RGB565 format, address should be aligned to halfword. For ARGB8888 format, address should be aligned to word.
0x74			**LAYER1_FILL**	
[31:26]			RSVD	
[25]	rw	1'h0	ENDIAN	input 565 data format endian 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]
[24]	rw	1'h0	BG_MODE	not used
[23:16]	rw	8'h0	BG_R	background r color
[15:8]	rw	8'h0	BG_G	background g color
[7:0]	rw	8'h0	BG_B	background b color
0x78			**DITHER_CONF**	
[31:13]			RSVD	
[12]	w1t	1'h0	lfsr_load	load lfsr init value
[11:10]	rw	2'h0	lfsr_load_sel	select lfsr 0: none 1: red 2: green 3: blue
[9:7]	rw	3'h0	w_r	red dither width
[6:4]	rw	3'h0	w_g	green dither width
[3:1]	rw	3'h0	w_b	blue dither width
[0]	rw	1'h0	en	dither enable
0x7C			**DITHER_LFSR**	
[31:0]	rw	32'h0	init_val	lfsr init load value
0x80			**LCD_CONF**	
[31:21]			RSVD	
[20:19]	rw	2'h0	SPI_RD_SEL	spi read line select. 0: select line 0 1: select line 1 2: select line 2 3: select line 3
[18]	rw	1'h0	ENDIAN	LCD 565 data format endian, this bit would affect SPI, DPI, DBI and AHB interface 565 format 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[17]	rw	1’h0	DIRECT_INTF_EN	when the target LCD is AHB LCD, this bit enable the direct interface to DSI module. Direct interface has higher bandwidth and speed than AHB interface.
[16:15]	rw	2’h0	JDI_SER_FORMAT	JDI serial format 2’b00: 3-bit mode 2’b01: 4-bit mode 2’b10: 1-bit mode 2’b11: reserved
[14:12]	rw	3’h0	DPI_LCD_FORMAT	DPI LCD format 3’b000: 16-bit conf1 3’b001: 16-bit conf2 3’b010: 16-bit conf3 3’b011: 18-bit conf1 3’b100: 18-bit conf2 3’b101: 24-bit others: Reserved
[11:10]	rw	2’h0	SPI_LCD_FORMAT	SPI LCD format 2’b00: 8-bit RGB 3:3:2 2’b01: 16-bit RGB 5:6:5 2’b10: 24-bit RGB 8:8:8 2’b11: Reserved
[9:8]	rw	2’h0	AHB_FORMAT	AHB LCD/RAM output format: 0: RGB565 1: RGB888 2: ARGB8888 3: RGB332
[7:5]	rw	3’h0	LCD_FORMAT	LCD output format: 3’b000: 8-bit RGB 3:3:2 3’b001: 16-bit RGB 5:6:5 over 8-bit bus, 2 cycles/pixel 3’b010: 12-bit RGB 4:4:4 3’b011: 16-bit RGB 5:6:5 3’b100: 18-bit RGB 6:6:6 3’b101: 24-bit RGB 8:8:8 3’b110: 24-bit RGB 8:8:8 over 16-bit bus, 1.5 cycles/pixel 3’b111: 24-bit RGB 8:8:8 over 8-bit bus, 3cycles/pixel others: Reserved
[4:2]	rw	3’h0	LCD_INTF_SEL	3’b000: 8080 DBI Type B 3’b001: SPI interface 3’b010: DBI to DSI interface 3’b011: DPI interface 3’b100: JDI serial interface 3’b101: JDI parallel interface 3’b110: 8080 DBI Type A 3’b111: DPI to DSI interface
[1:0]	rw	2’h0	TARGET_LCD	The Data can be sent to four destinations: 2’b00: LCD panel 0 2’b01: LCD panel 1 2’b10: AHB LCD 2’b11: AHB RAM
0x84			**LCD_IF_CONF**	
[31:26]			RSVD	
[25]	rw	1’h0	CTRL_DLY_SET	if this bit is set to 1, LCD control output will be delayed for 1 lcdc clock cycle
[24]	rw	1’h0	DO_DLY_SET	if this bit is set to 1, LCD data output will be delayed for 1 lcdc clock cycle
[23]	rw	1’h0	LCD_RSTB	LCD RSTB pin, direct to output

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[22]	rw	1'h0	RD_POL	LCD RD pin polarity. RD is 0 for write operation, 1 for idle if polarity bit is set as 0. RD bit definition is opposite if polarity bit is set as 1.
[21]	rw	1'h0	WR_POL	LCD WR pin polarity. WR is 0 for write operation, 1 for idle if polarity bit is set as 0. WR bit definition is opposite if polarity bit is set as 1.
[20]	rw	1'h0	RS_POL	LCD RS pin polarity. RS is 1 for data access, 0 for command access if polarity bit is set as 0. RS bit definition is opposite if polarity bit is set as 1.
[19]	rw	1'h0	CS1_POL	LCD0 CS pin polarity. CS is 0 for LCD chip select if polarity bit is set as 0. CS bit definition is opposite if polarity bit is set as 1.
[18]	rw	1'h0	CS0_POL	LCD1 CS pin polarity. CS is 0 for LCD chip select if polarity bit is set as 0. CS bit definition is opposite if polarity bit is set as 1.
[17:12]	rw	6'h0	PWH	inactive cycles of LCD_WR/LCD_RD for consecutive write/read operation
[11:6]	rw	6'h0	PWL	active cycles of LCD_WR/LCD_RD
[5:3]	rw	3'h0	TAH	hold cycles, delay from LCD_WR/LCD_RD inactive to LCD_CS inactive
[2:0]	rw	3'h0	TAS	setup cycles, delay from LCD_CS active to LCD_WR/LCD_RD active
0x88			**LCD_MEM**	
[31:0]	rw	32'h0	ADDR	address for AHB LCD/AHB RAM
0x8C			**LCD_O_WIDTH**	
[31:16]			RSVD	
[15:0]	rw	16'h0	OFFSET	AHB RAM address offset for each line
0x90			**LCD_SINGLE**	
[31:4]			RSVD	
[3]	r	1'h0	LCD_BUSY	LCD/SPI LCD interface is busy for single access
[2]	w1t	1'h0	RD_TRIG	Single read operation trigger
[1]	w1t	1'h0	WR_TRIG	Single write operation trigger
[0]	rw	1'h0	TYPE	LCD access type, this bit could affect all LCD interface including SPI, parellel and AHB 1'b0: command 1'b1: data
0x94			**LCD_WR**	
[31:0]	rw	32'h0	DATA	LCD write data
0x98			**LCD_RD**	
[31:0]	r	32'h0	DATA	LCD read data
0x9C			**SPI_IF_CONF**	
[31]			RSVD	
[30]	rw	1'h0	SPI_CLK_INIT	SPI CLK idle state value 1'h0: high 1'h1: low
[29]	rw	1'h0	SPI_CLK_POL	SPI CLK polarity 1'h0: normal 1'h1: inverted
[28]	rw	1'h0	SPI_CS_POL	SPI CS polarity 0: low active 1: high active
[27]	rw	1'h1	SPI_CS_AUTO_DIS	1: SPI CS is automatically disabled after data transaction 0: SPI CS is not disabled after data transaction
[26]	rw	1'h0	SPI_CS_NO_IDLE	1: SPI CS is always active during data transaction 0: SPI CS is IDLE in wait state during data transaction
[25]	rw	1'h0	SPI_CLK_AUTO_DIS	1: SPI clock auto disable in wait state during data transaction 0: SPI clock is always on in wait state during data transaction
[24]	rw	1'h0	SPI_RD_MODE	SPI read mode: 1'b0: normal read. Send write request before read. 1'b1: direct read. Read data without write request.
[23:22]	rw	2'h0	WR_LEN	SPI write data length(single access)

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21:20]	rw	2'h0	RD_LEN	SPI read data length(single access)
[19:17]	rw	3'h0	LINE	SPI line mode 0: 4-line 1: 4-line with 2 data line(support RGB565 and RGB888) 2: 4-line with 4 data line(support RGB565 and RGB888) 3: reserved 4: 3-line 5: 3-line with 2 data line(support RGB565 and RGB888) 6: 3-line with 4 data line(support RGB565 and RGB888) 7: reserved
[16:14]	rw	3'h0	DUMMY_CYCLE	SPI transaction dummy cycle
[13:6]	rw	8'ha	CLK_DIV	SPI clock divider
[5:0]	rw	6'h0	WAIT_CYCLE	SPI line wait cycle, wait cycle is after each line and is according to SPI clock. 0 refers to no wait cycle.
0xA0			**TE_CONF**	
[31:21]			RSVD	
[20]	rw	1'h0	FMARK_SOURCE	TE signal source 1: use TE signal from DSI 0: use TE signal from external pin
[19]	rw	1'h0	FMARK_MODE	TE signal trigger mode 1: edge trigger 0: pulse trigger
[18:3]	rw	16'h0	VSYNC_DET_CNT	vsync signal detect counter, used for mode 1 to detect vsync signal
[2]	rw	1'h0	MODE	0: vsync only TE mode 1: vsync+hsync TE mode
[1]	rw	1'h0	FMARK_POL	TE signal polarity
[0]	rw	1'h0	ENABLE	TE enable
0xA4			**TE_CONF2**	
[31:0]	rw	32'h0	DLY_CNT	TE delay counter
0xA8			**DPI_IF_CONF1**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HSW	dpi hsync width
[15:11]			RSVD	
[10:0]	rw	11'h0	VSH	dpi vsync height
0xAC			**DPI_IF_CONF2**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HBP	horizontal back porch
[15:11]			RSVD	
[10:0]	rw	11'h0	VBP	vertical back porch
0xB0			**DPI_IF_CONF3**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HFP	horizontal front porch
[15:11]			RSVD	
[10:0]	rw	11'h0	VFP	vertical front porch
0xB4			**DPI_IF_CONF4**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HAW	horizontal active width
[15:11]			RSVD	
[10:0]	rw	11'h0	VAH	vertical active height
0xB8			**DPI_IF_CONF5**	
[31:24]			RSVD	

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23]	rw	1'h0	clk_force_on	1: force DPI clock on 0: DPI clock is controlled by hardware
[22:12]	rw	11'h7ff	INT_LINE_NUM	DPI interrupt line number
[11]	rw	1'h0	HSPOL	hsync polarity
[10]	rw	1'h0	VSPOL	vsync polarity
[9]	rw	1'h0	DEPOL	de polarity
[8]	rw	1'h0	PCLKPOL	pixel clock polarity
[7:0]	rw	8'h1	PCLK_DIV	pixel clock divider
0xBC			**DPI_CTRL**	
[31:4]			RSVD	
[3]	rw	1'h0	DPI_UC	dpi update config
[2]	rw	1'h0	DPI_SD	dpi shutdown
[1]	rw	1'h0	DPI_CM	dpi color mode
[0]	rw	1'h0	DPI_EN	dpi interface enable
0xC0			**DPI_STAT**	
[31:16]	r	16'h1	VPOS	dpi vertical position
[15:14]			RSVD	
[13:11]	r	3'h0	HSTAT	horizontal status 0: idle 1: prep 2: hsync 3: hbp 4: hact 5: hfp 6: wait
[10:0]	r	11'h1	HPOS	dpi horizontal position
0xC4			**JDI_SER_CONF1**	
[31:16]			RSVD	
[15:8]	rw	8'h2	CLK_DIV	jdi serial clock divider
[7:5]			RSVD	
[4:0]	rw	5'd1	WR_LEN	jdi single write bit length
0xC8			**JDI_SER_CONF2**	
[31:16]	rw	16'h0	INIT_LINE_CNT	jdi serial init line counter
[15:0]	rw	16'h0	WR_CMD	jdi serial data transfer write command
0xCC			**JDI_SER_CTRL**	
[31:2]			RSVD	
[1]	rw	1'h0	EXTCOMIN	jdi serial interface extcomin control
[0]	rw	1'h0	DISP	jdi serial interface disp control
0xD0			**JDI_PAR_CONF1**	
[31:16]	rw	16'h0	MAX_LINE	jdi parallel interface max line, line number start from 0
[15:0]	rw	16'h0	MAX_COL	jdi parallel interface max column, column number start from 0
0xD4			**JDI_PAR_CONF2**	
[31:16]	rw	16'h0	ST_LINE	jdi parallel interface start line, line number start from 0
[15:0]	rw	16'h0	END_LINE	jdi parallel interface end line, line number start from 0
0xD8			**JDI_PAR_CONF3**	
[31:16]	rw	16'h0	ST_COL	jdi parallel interface start column, column number start from 0
[15:0]	rw	16'h0	END_COL	jdi parallel interface end column, column number start from 0
0xDC			**JDI_PAR_CONF4**	
[31:16]	rw	16'h0	HCK_WIDTH	jdi parallel interface HCK width, HSK width = lcd_ck_cycle * HCK_WIDTH
[15:0]	rw	16'h0	HST_WIDTH	jdi parallel interface HST width, HST width = lcd_ck_cycle * HST_WIDTH
0xE0			**JDI_PAR_CONF5**	
[31:16]	rw	16'h0	VCK_WIDTH	jdi parallel interface VCK width, VCK width = lcd_ck_cycle * VCK_WIDTH

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:0]	rw	16'h0	VST_WIDTH	jdi parallel interface VST width, VST width = lcd_ck_cycle * VST_WIDTH
0xE4			**JDI_PAR_CONF6**	
[31:16]	rw	16'h0	VCK_DLY	jdi parallel interface VST to VCK delay, VST2VCK delay = lcd_ck_cycle * VCK_DLY
[15:0]	rw	16'h0	HST_DLY	jdi parallel interface VCK to HST delay, VCK2HST delay = lcd_ck_cycle * HST_DLY
0xE8			**JDI_PAR_CONF7**	
[31:17]			RSVD	
[16]	rw	1'h0	DP_MODE	double pixel mode. Some jdi parallel screens use large pixel+small pixel structure. Set this bit to 1 to support this structure.
[15:0]	rw	16'h0	HCK_DLY	jdi parallel interface HST to HCK delay
0xEC			**JDI_PAR_CTRL**	
[31:16]	rw	16'hffff	INT_LINE_NUM	jdi parallel interface interrupt line number, line number start from 0.
[15:10]			RSVD	
[9]	rw	1'h0	VSTPOL	jdi parallel vst polarity
[8]	rw	1'h0	VCKPOL	jdi parallel vck polarity
[7]	rw	1'h0	HSTPOL	jdi parallel hst polarity
[6]	rw	1'h0	HCKPOL	jdi parallel hck polarity
[5]	rw	1'h0	ENBPOL	jdi parallel enb polarity
[4]	rw	1'h0	XRST	jdi parallel interface XRST
[3:1]			RSVD	
[0]	rw	1'h0	ENABLE	jdi parallel interface enable
0xF0			**JDI_PAR_STAT**	
[31:16]	r	16'h0	VPOS	jdi parallel vertical position
[15:0]	r	16'h0	HPOS	jdi parallel horizontal position
0xF4			**JDI_PAR_EX_CTRL**	
[31]	r	1'h0	VCOM	VCOM value
[30]	r	1'h0	FRP	FRP value
[29]	r	1'h0	XFRP	XFRP value
[28]	rw	1'h0	CNT_EN	VCOM/FRP/XFRP counter enable
[27:24]			RSVD	
[23:0]	rw	24'h0	MAX_CNT	VCOM/FRP/XFRP max counter
0xF8			**JDI_PAR_CONF8**	
[31:16]	rw	16'h0	ENB_ST_COL	jdi parallel interface enb start column, column number start from 0
[15:0]	rw	16'h0	ENB_END_COL	jdi parallel interface enb end column, column number start from 0
0xFC			**JDI_PAR_CONF9**	
[31:16]	rw	16'h0	ENB_ST_LINE	jdi parallel interface enb start line, line number start from 0
[15:0]	rw	16'h0	ENB_END_LINE	jdi parallel interface enb end line, line number start from 0
0x100			**JDI_PAR_CONF10**	
[31:16]	rw	16'h0	HC_ST_LINE	jdi parallel interface horizontal control start line, line number start from 0
[15:0]	rw	16'h0	HC_END_LINE	jdi parallel interface horizontal control end line, line number start from 0
0x110			**CANVAS_STAT0**	
[31:27]			RSVD	
[26:16]	r	11'h0	y_cor	canvas y cordinate
[15:11]			RSVD	
[10:0]	r	11'h0	x_cor	canvas x cordinate
0x114			**CANVAS_STAT1**	
[31:12]			RSVD	
[11:9]	r	3'h0	fetch_stat	fetch status
[8:6]	r	3'h0	prec_stat	prec status
[5:3]	r	3'h0	postc_stat	postc_status
[2:0]	r	3'h0	fifo_cnt	pre calc fifo count
0x118			**OL0_STAT**	
[31:24]			RSVD	

Continued on the next page...

Table 10-1: LCDC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:22]	r	2'h0	sc_lb0	
[21:20]	r	2'h0	sc_lb1	
[19:16]	r	4'h0	sc_fe	
[15:13]	r	3'h0	sc_be	
[12:11]	r	2'h0	sc_out	
[10:8]	r	3'h0	pf_pr	
[7:6]	r	2'h0	pf_df	
[5:4]	r	2'h0	data_conv	
[3:2]	r	2'h0	prefetch_read	
[1]	r	1'h0	prefetch_out	
[0]	r	1'h0	done_req	
0x11C			**OL1_STAT**	
[31:11]			RSVD	
[10:8]	r	3'h0	pf_pr	
[7:6]	r	2'h0	pf_df	
[5:4]	r	2'h0	data_conv	
[3:2]	r	2'h0	prefetch_read	
[1]	r	1'h0	prefetch_out	
[0]	r	1'h0	done_req	
0x120			**MEM_IF_STAT**	
[31:10]			RSVD	
[9:7]	r	3'h0	arb_main	
[6:4]	r	3'h0	arb_read_port	
[3:0]	r	4'h0	ahb	
0x124			**PERF_CNT**	
[31:0]	rw	32'h0	VAL	lcdc performance counter

10.3 eZip™ Lossless Compression Decoder

The eZip™ decoder is a real-time lossless decompression module based on proprietary algorithms, achieving a compression ratio comparable to the Zip format. It can be utilized to decode general data for storage, thereby enhancing real-time data loading capabilities. When data is transmitted from outside the chip, compressed transmission aids in reducing transfer time and power consumption.

Additionally, the eZip™ also supports proprietary image compression formats, achieving a compression ratio comparable to the PNG format, and supports independent DMA operation or coordinated reading with ePicasso™. When operating independently, eZip™ can flexibly decompress compressed images stored in Flash or RAM and transfer them to the target cache via the DMA mechanism. In coordinated mode, ePicasso™ reads images from storage in real-time through the eZip™ module, decompresses them in real-time, and then performs the necessary 2.5D calculations according to the general graphics process, thus eliminating the need for temporary storage of decompressed images.

Through the aforementioned mechanism, eZip™ can effectively reduce the storage capacity requirements for image assets, maximizing the richness of materials within limited storage, and minimizing the bandwidth requirements for external storage, thereby signifcantly enhancing the overall operational effciency of the system.

The eZip™ module is designed to decode and output compressed images. This module reads compressed data via the AHB bus, and the decoded image data can be configured for output through the AHB bus or directly sent to the epic module for subsequent processing.

This module possesses the following features:

- The data address for input and output via the AHB bus is configurable.
- Output image data can be directly transmitted to the epic module.
- Can output image data from a specifed area.

11 Audio

11.1 PDM

HPSYS has one PDM module. It supports up to two microphone inputs.

11.1.1 Introduction

The PDM (Pulse Density Modulation) interface is primarily used to convert PDM audio signals collected from PDM microphones into PCM (Pulse Code Modulation) audio signals for subsequent audio processing.

Main functions:

- Simultaneously supports left and right stereo signals and can also collect mono signals individually.
- Available PDM microphone clock rate:3.072MHz. 、1.536MHz、0.768MHz、1.024MHz、2.4MHz、1.6MHz、0.8MHz Etc.
- Supported PCM data rate: 48kHz. 、 32kHz、 24kHz、 16kHz、 12kHz、 8kHz Equivalent
- Supports 32 -bit 、 24bit、 16bit、 8-bit PCM Signal
- Supports a resolution of 0.5 dB and adjustable gain from -15 dB to 45 dB.

11.1.2 Usage Instruction

The PDM (Pulse Density Modulation) Module is designed to support digital microphones.

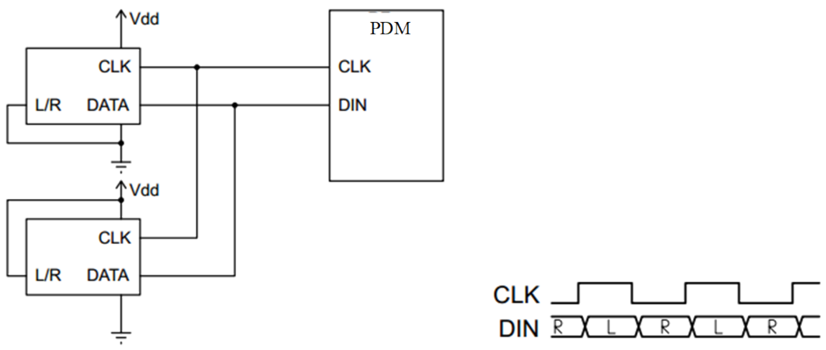


Figure 11-1: **Typical connection of a digital microphone through the PDM Module**

The above fgure illustrates a typical connection of a pair of digital microphones through the PDM Module, where both microphones share the same bit stream clock and data line. Using the microphone configuration pins (L/R) , one microphone can provide valid data on the rising edge of CLK while the other microphone provides valid data on the falling edge of CLK.

11.1.2.1 Overall Structure of the PDM Module

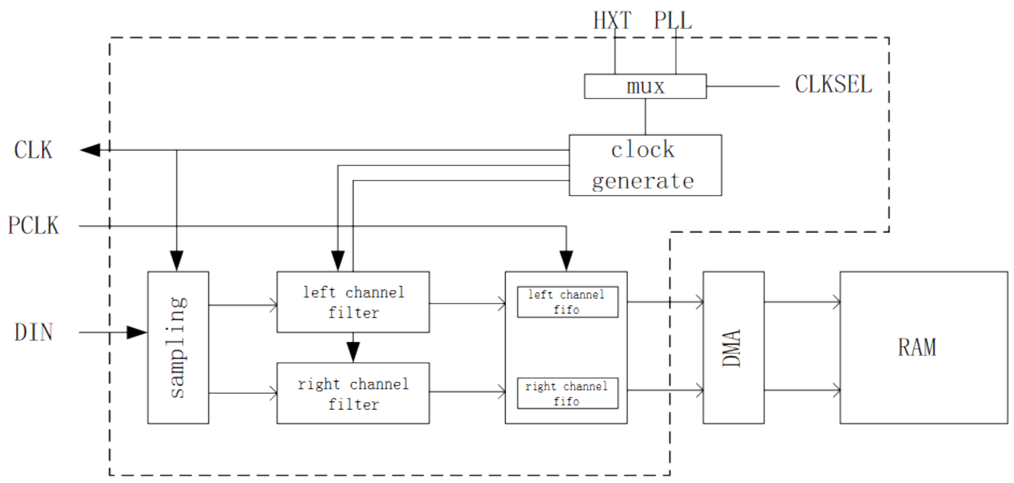


Figure 11-2: Overall Structure of the PDM Module

11.1.2.2 Clock Structure of the PDM Module

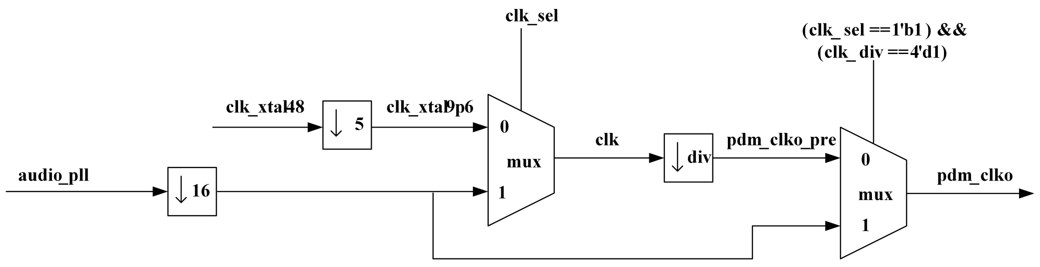


Figure 11-3: Clock Structure of the PDM Module

The clock source of the PDM module consists of two components: one is the system's 48mHz crystal oscillator, and the other is the system's audio PLL after a 16 fold frequency division. The 48mHz crystal oscillator first passes through a 5 fold divider within the PDM module to generate a 9.6mHz clock. This clock is selected between the audio PLL clock and then goes through a configurable divider to produce the fnal clock sent to the digital microphone, referred to as pdm_clko . The final output data rate of the PDM module is pdm_clko/(sinc_rate×lpf_downsample) . The parameters sinc_rate and lpf_downsample are configured according to the following table for the registers sinc_rate and lpf_ds to obtain the final data.

Table 11-1: PDM Microphone Clock Source and Corresponding Output Data Rate Configuration Table

PDM_CLK(MHz)	Fs PCM (Output Rate, KHz)	OSR (Over Sampling Rate)	SINC RATE (CIC Down Sampling Rate)	LPF Post Down Sampling Rate	SINC ORDER
3.072	48	64	32	2	3
3.072	32	96	48	2	3
3.072	24	128	64	2	3
3.072	16	192	96	2	3
3.072	12	256	64	4	3
3.072	8	384	96	4	3
1.536	48	32	16	2	4
1.536	32	48	24	2	4

Continued on the next page...

Table 11-1: Microphone Clock Source and Corresponding Output Data Rate Configuration Table (Continued)

PDM_CLK(MHz)	Fs PCM (Output Rate, KHz)	OSR (Over Sampling Rate)	SINC RATE (CIC Down Sampling Rate)	LPF Post Down Sampling Rate	SINC ORDER
1.536	24	64	32	2	3
1.536	16	96	48	2	3
1.536	12	128	64	2	3
1.536	8	192	96	2	3
0.768	24	32	16	2	4
0.768	16	48	24	2	4
0.768	12	64	32	2	3
0.768	8	96	24	4	4
1.024	32	32	16	2	4
1.024	16	64	32	2	3
1.024	8	128	64	2	3
2.4	48	50	25	2	4
2.4	24	100	50	2	3
2.4	16	150	75	2	3
2.4	12	200	100	2	3
2.4	8	300	75	4	3
1.6	32	50	25	2	4
1.6	16	100	50	2	3
1.6	8	200	100	2	3
0.8	16	50	25	2	4
0.8	8	100	50	2	3
2.4	32	75	75	LPF bypass	3
1.2	48	25	25	LPF bypass	4
1.2	24	50	25	2	4
1.2	16	75	75	LPF bypass	3
1.2	12	100	50	2	3
1.2	8	150	75	2	
2.8224	44.1	64	32	2	3
2.8224	22.05	128	64	2	3
2.8224	11.025	256	64	4	3
1.4112	44.14	32	16	2	4
1.4112	22.05	64	32	2	3
1.4112	11.025	128	64	2	3
0.7056	22.05	32	16	2	4
0.7056	11.025	64	32	2	3

Using the configuration in the first row of the table as an example, the register configuration for the PDM module is explained. The output clock is 3.072mHz , and the output data rate is 48kHz . The output data bit width is 16 bits, with stereo enabled.

Configuration process for the PDM registers:

1. Select the output rate according to the table. Choose the clock source based on the PDM_CLK in the table, and configure 0x00 in the clk_sel register. 0x0 indicates that the input clock for the PDM module is 9.6mHz , while 0x1 indicates that the input clock for the PDM module is the one-sixteenth frequency clock of the audio PLL. In this case, the configuration of clk_sel is 1 .(The configuration of the audio clock is not discussed here.)
2. According to the table, configure the SINC RATE and SINC ORDER in 0x08 for sinc_rate and sinc_order_sel registers, where sinc_order_sel is 1 corresponds to the table where SINC ORDER is 4 , and 0 corresponds to 3 . Based on the LPF post-downsampling rate, configure 0x34 for lpf_ds register and lpf_bypass register. Configuring lpf_ds register to 1 indicates a downsampling of 4 in excel for lpf , while configuring lpf_ds register to 0 indicates a downsampling

of 2 in excel for lpf . Configuring lpf_bypass to 1 corresponds to the LPF BYPASS in the table. In this example, 0x08 sinc_rate=64 , sinc_order_sel=0 , 0x34 lpf_ds=0

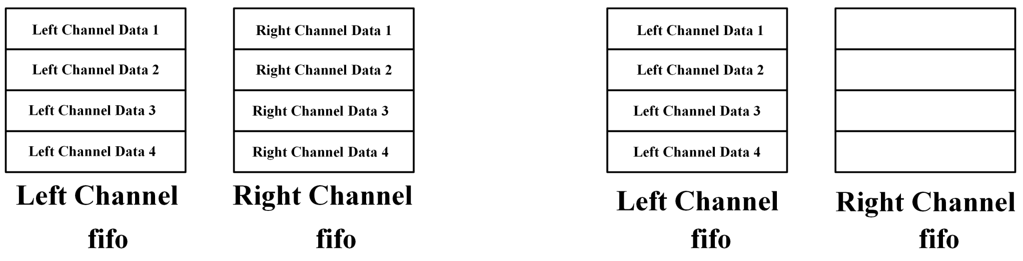
3. Enable the corresponding signal based on the channel configuration:

		Dual Channel		Left Channel	Right Channel
Register 0x0 CFG0	left_en	1	0	1	0
	right_en	1	0	0	1
	stereo_en	0	1	0	0

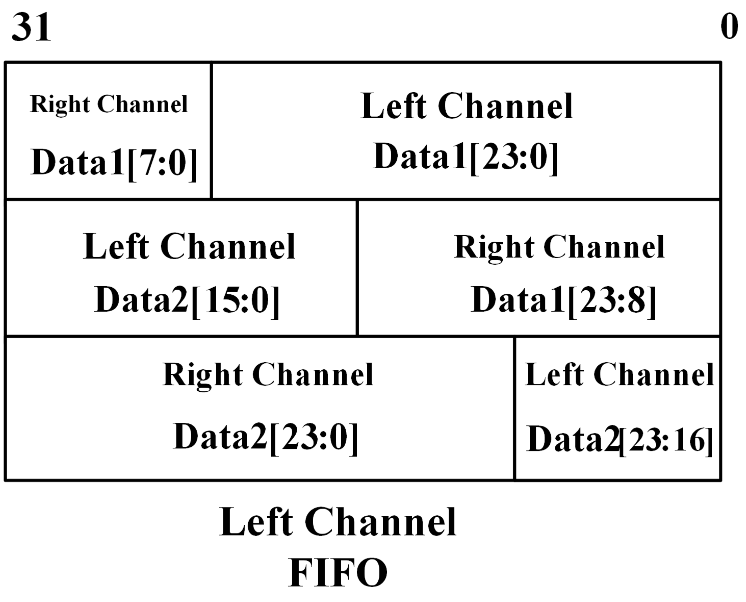
When the swap_en register at 0x0 is set to 0, the PDM module interprets the right channel as collecting data from the rising edge output of the PDM mic, while the left channel collects data from the falling edge output of the PDM mic. Conversely, when the swap_en register at 0x0 is set to 1, the right channel is interpreted as collecting data from the falling edge output of the PDM mic, and the left channel collects data from the rising edge output of the PDM mic. Configure according to requirements. In this example, 0x0 stereo_en=1 left_en=0 right_en=0 or stereo_en=0 left_en=1 right_en=1 .

4. Configure according to the output data format of the pdm module. The different bit widths of the output data configure the 0x38 byte_trunc register. Set 0 for a 24-bit output, set 1 for a 16-bit output, set 2 for an 8-bit output, and set 3 for a 32-bit output. In this example, the 0x38 byte_trunc is 1 .

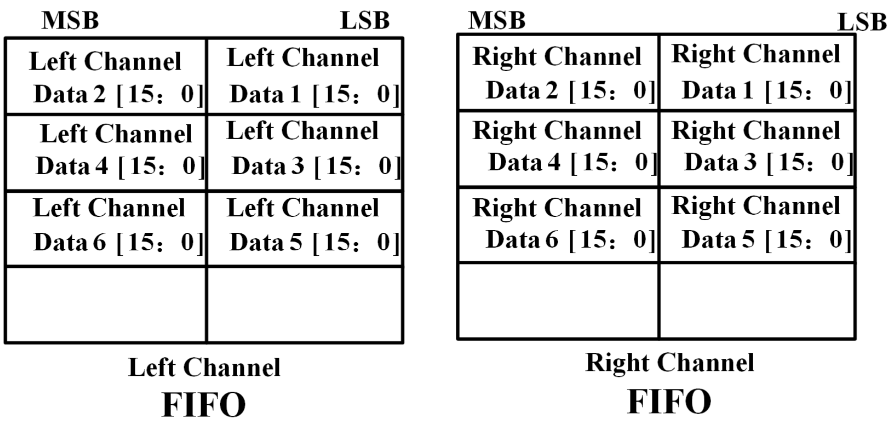
5. Configure the 0x38 byte_con register based on whether the left and right channel data needs to be mixed together. Set 0 to store the data of the left and right channels separately in their respective ffo , while set 1 will store the data of both channels in the left channel's fifo . A schematic diagram illustrates the differences in data storage corresponding to different byte_con register configurations.



If the data is less than 32 bits, the lower bits of the next data will automatically fill the higher bits of the previous data to form 32 bits of data. For 24 bit stereo data, when byte_con is set to 1, it is illustrated in the fgure below.



For 16 bit stereo data, when byte_con is set to 0, it is illustrated in the fgure below.


MSB
LSB
Left Channel Data 2 [15：0]
Left Channel Data 1 [15：0]
Left Channel Data 4 [15：0]
Left Channel Data 3 [15：0]
Left Channel Data 6 [15：0]
Left Channel Data 5 [15：0]
Left Channel FIFO
MSB
LSB
Right Channel Data 2 [15：0]
Right Channel Data 1 [15：0]
Right Channel Data 4 [15：0]
Right Channel Data 3 [15：0]
Right Channel Data 6 [15：0]
Right Channel Data 5 [15：0]
Right Channel FIFO

In this example, there are no specific requirements for the data storage format; users may select the configuration based on their needs.

6. Configure the DMA registers according to the DMA usage instructions to transfer data from the PDM FIFO, one 32-bit data at a time, to the specifed RAM address.
7. Set the PDMCOREEN register at 0x0 to enable the PDM module.

The order of steps 1 to 6 is not fixed, provided that they are completed before step 7.

11.1.2.3 Precaution

All interrupts generated by the PDM module are triggered by errors caused by ffo overflow.

11.1.3 PDM Register

PDM base address is 0x5009A000。

Table 11-2: PDM Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CFG0**	
[31:10]			RSVD	
[9]	rw	1’b0	swap_en	1: Swap right channel and left channel pdm data; 0: Not swap right channel and left channel pdm data
[8]	rw	1’b0	stereo_en	1:Enable double channels pdm data sampling; 0: Disable double channels pdm data sampling
[7]	rw	1’b0	right_en	1: Enable right channel pdm data sampling; 0: Disable right channel pdm data sampling
[6]	rw	1’b0	left_en	1: Enable left channel pdm data sampling; 0: Disable left channel pdm data sampling
[5:2]	rw	4’h4	clk_div	Clock frequency division ratio of 3.072MHz or 9.6MHz according to register clk_sel
[1]	rw	1’b0	clk_sel	1:Clk select dll 3.072MHz; 0: Clk selct xtal 9.6MHz
[0]	rw	1’b0	pdmcoreen	1:Enable pdm module; 0: Disable pdm module
0x04			**CFG1**	
[31:11]			RSVD	
[10:8]	rw	3’h0	sample_dly_r	The number of delay dff before the right data stream in processing
[7:5]	rw	3’h0	sample_dly_l	The number of delay dff before the left data stream in processing
[4:0]			RSVD	
0x08			**SINC_CFG**	
[31:9]			RSVD	
[8]	rw	1’b1	sinc_order_sel	1:select four differentiators in sinc filter; 0:select three differentiators in sinc filter
[7:0]	rw	8’d32	sinc_rate	dowmsampling rate of sinc filter
0x14			**HPF_CFG**	
[31:6]			RSVD	
[5]	rw	1’b1	hpf_rst	1:high-pass filter normal operation ; 0:reset high-pass filter
[4]	rw	1’b0	hpf_bypass	1:bypass-high pass filter ; 0: enable high-pass filter
[3:0]	rw	4’hd	hpf_coeff	coefficient of high-pass filter
0x18			**PGA_CFG**	
[31:14]			RSVD	
[13:7]	rw	7’d0	pga_gain_r	right channel gain control , the range is -15dB~45dB. Resolution is 0.5dB/LSB
[6:0]	rw	7’d0	pga_gain_l	left channel gain control , the range is -15dB~45dB. Resolution is 0.5dB/LSB
0x34			**LPF_CFG6**	
[31:14]			RSVD	
[13]	rw	1’b0	lpf_bypass	1:bypass low-pass filter ; 0: enable low-pass filter
[12]	rw	1’b0	lpf_ds	1:downsampling rate of low pass filter is two;0:No downsampling of low pass filter
[11:0]			RSVD	
0x38			**FIFO_CFG**	
[31:9]			RSVD	
[8]	rw	1’b0	lr_chg	1:exchange storage location of left and right channel; 0: don’t exchange storage location of left and right channel
[7]	rw	1’b0	rx_dma_msk_l	1:disable left channel dma request; 0: enable left channel dma request
[6]	rw	1’b0	rx_dma_msk_r	1:disable right channel dma request; 0: enable right channel dma request
[5:3]	rw	3’h0	pdm_shift	the number of data left shift for higher data accuracy
[2:1]	rw	2’b0	byte_trunc	1: 16bits output ; 0: 24bits output ;2: 8bits output ; 3: 32bits output
[0]	rw	1’b0	byte_con	1: combine left channel and right channel; 0: not combine left channel and right channel
0x44			**FIFO_ST**	
[31:8]			RSVD	
[7]	r	1’h0	full_l	1 indicates left channel fifo is full

Continued on the next page...

Table 11-2: PDM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	r	1'b0	empty_l	1 indicates left channel fifo is empty
[5]	r	1'b0	almost_full_l	1 indicates left channel fifo is less than two full
[4]	r	1'b0	almost_empty_l	1 indicates left channel fifo is less than two datas left
[3]	r	1'h0	full_r	1 indicates right channel fifo is full
[2]	r	1'b0	empty_r	1 indicates right channel fifo is empty
[1]	r	1'b0	almost_full_r	1 indicates right channel fifo is less than two full
[0]	r	1'b0	almost_empty_r	1 indicates right channel fifo is less than two datas left
0x48			**INT_ST**	
[31:2]			RSVD	
[1]	r	1'b0	overflow_l	1 indicates left channel fifo has already overflowed and as irq at same time
[0]	r	1'b0	overflow_r	1 indicates right channel fifo has already overflowed and as irq at same time
0x4c			**INT_MSK**	
[31:2]			RSVD	
[1]	rw	1'b0	int_mask_l	1:disable left channel irq to system; 0: enable left channel irq to system
[0]	rw	1'b0	int_mask_r	1:disable right channel irq to system; 0: enable right channel irq to system
0x50			**INT_CLR**	
[31:2]			RSVD	
[1]	w1c	1'b0	int_clr_l	clear left channel irq
[0]	w1c	1'b0	int_clr_r	clear right channel irq

11.2 I2S

HPSYS has one I2S module. It supports master mode and slave mode.

11.2.1 Introduction

The I2S bus (also known as IIS, which stands for Inter-IC Sound) is a standard established by Philips for audio data transmission between digital audio devices. This bus operates in a master/slave mode and is dedicated to data transmission between audio devices, making it widely used in various multimedia systems.

Currently, I2S supports MSB alignment (left-aligned), LSB alignment (right-aligned), and the I2S standard mode.

The standard I2S mode is illustrated in the fgure below:

Data is transmitted on the second rising edge of BCLK following LRCLK, with the MSB transmitted first, and the subsequent bits following in order until the LSB. The transmission is dependent on the word length, BCLK frequency, and sampling rate (BCLK = Fs x number of channels x number of sampling bits). There should be unused BCLK cycles between the LSB of each sample and the MSB of the next sample. LRCLK is 0 for transmitting left channel data, and LRCLK is 1 for transmitting right channel data.

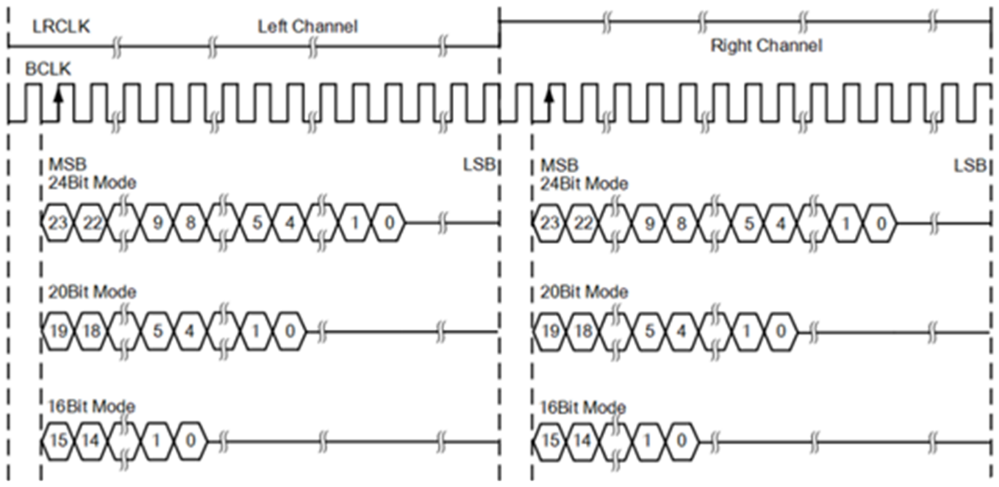


Figure 11-4: **Standard I2S Mode**

The left-aligned I2S mode is illustrated in the fgure below:

In the standard left-aligned format, the MSB of the data has no clock delay relative to BCLK. The MSB of the left and right channel data in the left-aligned format is valid on the first rising edge of BCLK after the LRCLK edge changes. LRCLK is 1 for transmitting left channel data and 0 for transmitting right channel data.

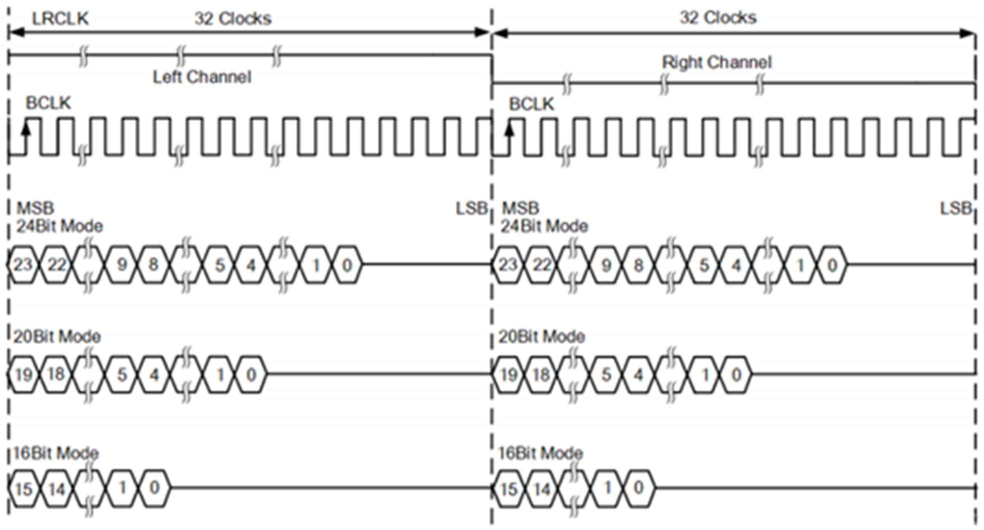


Figure 11-5: **I2S Left Alignment**

I2S Right alignment is illustrated in the following fgure:

Simultaneously with the completion of the sound data LSB transmission, LRCLK completes its second transition, with LRCLK being 1 for transmitting left channel data and 0 for transmitting right channel data.

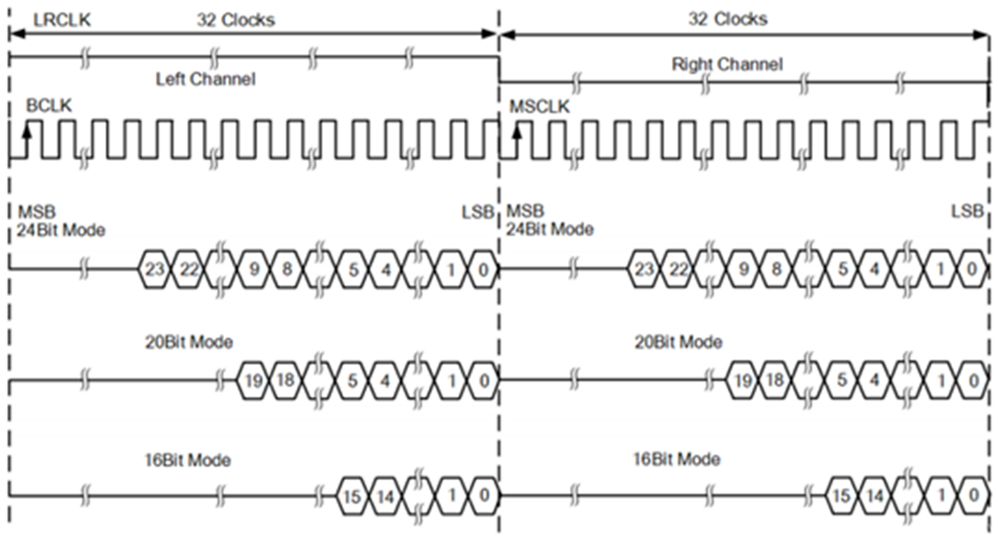


Figure 11-6: **I2S Right Alignment**

11.2.2 I2S Function Description

The I2S module supports both master and slave modes. In master mode, theMCU provides the sampling clock LRCK and the bit clockBCLK. In slave mode, theBCLK and LRCK are supplied externally, with the MCU responsible for the data transmission and reception of I2S.

The I2S module supports three data formats: left-aligned and right-aligned. In master mode, users can define the ratio between BCLK and LRCK according to their requirements.

The current version of the I2S module additionally provides an output for MCLK , which is a higher frequency clock synchronized with BCLK and LRCK , allowing for higher operating frequencies for I2S peripherals. Users can also define the frequency of MCLK according to their requirements. Furthermore, the I2S clock source has added the option of PLL , enhancing the flexibility of the I2S clock.

11.2.3 I2S Register

I2S base address is 0x50009000.

Table 11-3: **I2S Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x10			**TX_PCM_FORMAT**	
[31:6]			RSVD	
[5]	rw	1’h0	track_flag	0: stereo 1: mono

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4:0]	rw	5’h10	dw	tx source pcm data width N(N>=8) common value is 8,13,14,16,18,20,22,24 This data width indicate the tx fifo output data width. When writing to tx fifo, please refer to following format: Mono 8 bit: fifo_data[31:0] = L3,L2,L1,L0, each word contains 4 samples, so four samples need read one word Stereo 8 bit: fifo_data[31:0] = R1,L1,R0,L0 , each word contains 2 samples, so two samples need read one word Mono 13/14/16 bit: fifo_data[31:0] = L1,L0, each word contains 2 samples, so two samples need read one word Stereo 13/14/16 bit: fifo_data[31:0] = R0,L0, each word contains 1 samples, so each sample need read one word Mono 18/20/22/24 bit: fifo_data[31:0] = L0, each word contains 1 samples, so each sample need read one word Stereo 18/20/22/24 bit: fifo_data[31:0][0] = L0, fifo_data[31:0][1]=R0, each 2 words contain 1 samples, so each sample need read two word
0x20			**TX_PCM_SAMPLE_CLK**	
[31:13]			RSVD	
[12:0]	rw	13’d250	fs_duty	source PCM sample clock duty cycle(with GCLK=12MHz): 250 for 48K FS 272 for 44.1K FS 375 for 32K FS 500 for 24K FS 544 for 22.05K FS 750 for 16K FS 1000 for 12K FS 1088 for 11.025K FS 1500 for 8K FS
0x30			**TX_RS_SMOOTH**	
[31:1]			RSVD	
[0]	rw	1’h0	en	0: Disable TX re-sample smooth filter 1: Enable TX re-sample smooth filter This function is not implemented.
0x40			**TX_PCM_CH_SEL**	
[31:4]			RSVD	
[3:2]	rw	2’h0	left_channel_sel	TX re-sampling module setting: 00: TX left = source left 01: TX left = source right 10,11: TX left = (source left + source right)/2
[1:0]	rw	2’h0	right_channel_sel	TX re-sampling module setting: 00: TX right = source right 01: TX right = source left 10,11: TX right = (source left + source right)/2
0x50			**TX_VOL_CTRL**	
[31:4]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4'hf	vol	volume control: 0000: +6dB, 0001: +4.5dB, 0010: +3dB, 0011: +1.5dB, 0100: 0dB, 0101: -1.5dB, 0110: -3.0dB, 0111: -4.5dB, 1000: -6.0dB, 1001: -7.5dB, 1010: -9dB, 1011: -10.5dB, 1100: -12dB, 1101: -13.5dB, 1110: -15dB, 1111: mute Note: 1) +1.5db = 20log(1+1/4-1/16+1/1024) 2) -1.5dB = 20log(1-1/8-1/32-1/512-1/2048)
0x60			**TX_LR_BAL_CTRL**	
[31:6]			RSVD	
[5:4]	rw	2'h0	en	LR balance enable: 00: both left and right in full volume 10: left channel balance volume adjustment enable 01: right channel balance volume adjustment enable 11: reserved, still kepp left and right in full volume
[3:0]	rw	4'h0	bal_vol	Balance volume control: 0000: Reserved, 0001: -1.5dB, 0010: -3.0dB, 0011: -4.5dB, 0100: -6.0dB, 0101: -7.5dB, 0110: -9.0dB, 0111: -10.5dB, 1000: -12dB, 1001: -13.5dB, 1010: -15dB, 1011: -16.5dB, 1100: -18dB, 1101: -19.5dB, 1110: -21dB, 1111: mute Note: 1) bit[5:0] = 101111 for left mute 2) bit[5:0] = 011111 for right mute 3) bit[5:4] = 00 or 11, bit[3:0] is don't care 4) +1.5db = 20log(1+1/4-1/16+1/1024) 5) -1.5dB = 20log(1-1/8-1/32-1/512-1/2048)
0x70			**AUDIO_TX_LRCK_DIV**	
[31:28]			RSVD	
[27:16]	rw	12'd125	duty_high	TX LRCK duty cycle high: 125 for 48K FS 136 for 44.1K FS 185 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS
[15:12]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:0]	rw	12’d125	duty_low	TX LRCK duty cycle low: 125 for 48K FS 136 for 44.1K FS 190 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS Note: 1)duty_cycle = 12M/FS
0x80			**AUDIO_TX_BCLK_DIV**	
[31:6]			RSVD	
[5:0]	rw	6’h5	duty	TX serial bit clock duty cycle 5 for 48K FS 4 for 44.1K FS 5 for 32KFS 10 for 24K FS 8 for 22.05K FS 15 for 16K FS 20 for 12K FS 16 for 11.025K FS 30 for 8KFs
0x90			**AUDIO_TX_FORMAT**	
[31:5]			RSVD	
[4:0]	rw	5’h10	pcm_data_width	I2S out pcm data width M >= 16, common value: 16, 18, 20, 22, 24
0xa0			**AUDIO_SERIAL_TIMING**	
[31:4]			RSVD	
[3]	rw	1’h0	lrck_pol	TX LRCK polarity control. 0: disable TX_LRCK inventor 1: enable TX_LRCK inventor for standard I2S, set tx_lrck_pol to low for Left/Right Justified, set tx_lrck_pol to hgih
[2]	rw	1’h0	slave_en	audio code transmit mode select. 0: master mode, 1: slave mode
[1:0]	rw	2’h0	timing	00: I2S mode 01: Left justified 10: right justified 11: reserved
0xb0			**AUDIO_TX_FUNC_EN**	
[31:2]			RSVD	
[1]	rw	1’h0	tx_intf_sel	1: select external tx interface 0: select internal apb tx interface
[0]	rw	1’h0	tx_en	1: enable 0:disable
0xc0			**AUDIO_TX_PAUSE**	
[31:1]			RSVD	
[0]	rw	1’h0	tx_pause	TX pause control when tx_enable = 1. 1: pause 0: TX work
0xc8			**AUDIO_I2S_SL_MERGE**	
[31:1]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	slave_timing_merge	when work as an I2S slave, and external I2S master TX/RX share an only BCLK/LRCK, we need set this bit high. 0: I2S slave use separated timing control port. TX_BCLK_IN/TX_LRCK_IN and RX_BCLK/RX_LRCK_IN are separated. 1: I2S slave use the same BCLK/LRCK, the TX_BCLK_IN/TX_LRCK also is used for RX controller.
0x100			**AUDIO_RX_FUNC_EN**	
[31:2]			RSVD	
[1]	rw	1'h0	rx_intf_sel	1: select external rx interface 0: select internal apb rx interface
[0]	rw	1'h0	rx_en	1: enable 0: disable
0x110			**AUDIO_RX_PAUSE**	
[31:1]			RSVD	
[0]	rw	1'h0	rx_pause	RX pause control when rx_enable = 1. 1: pause 0: RX work
0x120			**AUDIO_RX_SERIAL_TIMING**	
[31:4]			RSVD	
[3]	rw	1'h0	lrck_pol	RX LRCK polarity control. 0: disable RX_LRCK inventor 1: enable RX_LRCK inventor for standard I2S, set tx_lrck_pol to low for Left/Right Justified, set tx_lrck_pol to hgih
[2]	rw	1'h0	slave_en	audio code receiver mode select. 0: master mode, 1: slave mode
[1:0]	rw	2'h0	timing	00: I2S 01: Left justified 10: right justified 11: reserved
0x130			**AUDIO_RX_PCM_DW**	
[31:5]			RSVD	
[4:0]	rw	5'h10	pcm_data_width	For I2S and left justified mode, M can be 8,13,14,16 For right justified mode, M can be 8, 13, 14, 16, 18, 20, 22, 24
0x140			**AUDIO_RX_LRCK_DIV**	
[31:28]			RSVD	
[27:16]	rw	12'd125	duty_high	RX LRCK duty cycle high: 125 for 48K FS 136 for 44.1K FS 185 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS
[15:12]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:0]	rw	12'd125	duty_low	RX LRCK duty cycle low: 125 for 48K FS 136 for 44.1K FS 190 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS Note: 1)duty_cycle = 12M/FS
0x150			**AUDIO_RX_BCLK_DIV**	
[31:10]			RSVD	
[9:0]	rw	10'h5	duty	RX serial bit clock duty cycle 5 for 48K FS 4 for 44.1K FS 5 for 32KFS 10 for 24K FS 8 for 22.05K FS 15 for 16K FS 20 for 12K FS 16 for 11.025K FS 30 for 8KFs
0x160			**RECORD_DATA_SEL**	
[31:1]			RSVD	
[0]	rw	1'h0	rs_data_sel	0: I2S audio recording 1: BT recording
0x170			**RX_RE_SAMPLE_CLK_DIV**	
[31:13]			RSVD	
[12:0]	rw	13'd250	rs_duty	source PCM sample clock duty cycle: 250 for 48K FS 272 for 44.1K FS 375 for 32K FS 500 for 24K FS 544 for 22.05K FS 750 for 16K FS 1000 for 12K FS 1088 for 11.025K FS 1500 for 8K FS Note: 1)duty_cycle = 12M/FS
0x180			**RX_RE_SAMPLE**	
[31:1]			RSVD	
[0]	rw	1'h0	smooth_en	0: Disable RX re-sample smooth filter 1: Enable RX re-sample smooth filter
0x190			**RECORD_FORMAT**	
[31:2]			RSVD	
[1]	rw	1'h0	track	1: mono recording, 0: stereo recording

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	dw	0: 8bit 1: 16bit RX fifo data format: Mono 8 bit (unsigned): RX FIFO_DIN[31:0] = L3,L2,L1,L0, each four samples need one FIFO write operation Stereo 8 bit (unsigned): RX_FIFO_DIN[31:0] = R1,L1,R0,L0, each tow samples need one FIFO write operation Mono 16 bit (Signed 2's complement): RX_FIFO_DIN[31:0] = L1,L0, each two samples need one FIFO write operation Stereo 16 bit (Signed 2's complement): RX_FIFO_DIN[31:0] = R0,L0, each sample need one FIFO write operation
0x1a0			**RX_CH_SEL**	
[31:4]			RSVD	
[3:2]	rw	2'h0	left_channel_sel	RX re-sampling module setting: 00: RD left = RX left 01: RD left = RX right 10,11: RD left = (RX left + RX right)/2
[1:0]	rw	2'h0	right_channel_sel	RX re-sampling module setting: 00: RD right = RX right 01: RD right = RX left 10,11: RD right = (RX left + RX right)/2
0x200			**BT_PHONE_CTRL**	
[31:6]			RSVD	
[5]	rw	1'h0	bb_i2s_bps_to_cdc	bypass baseband I2S interface to audio codec i2s interface 0: no bypass, 1: bypass
[4]	rw	1'h0	bt_pcm_if_bps	bypass baseband PCM signals to BT VCI master: 0: no bypass, 1: bypass
[3]	rw	1'h0	bt_path_sel	BT path select 0: digital path, 1: analog path
[2]	rw	1'h0	bt_mix_smooth_filter_en	0: disable the smooth filter for background mixer 1: enable the smooth filer for background mixer
[1]	rw	1'h0	bt_back_mix_en	background mixer enable 0: disable, 1: enable
[0]	rw	1'h0	bt_ph_en	BT phone enable 0: disable, 1: enable
0x210			**BB_PCM_FORMAT**	
[31:11]			RSVD	
[10]	rw	1'h0	pcm_clk_pol	input BB pcm clock polarity: 0: rising edge for data transmitting, falling edge for data receiving 1: rising edge for data receiving, falling edge for data transmitting

[9]	rw	1'h0	i2s_lrck_pol	0: no bb_i2s_lrck input inventor
				1: enable bb_i2s_lrck input inventor
				for standard I2S, set tx_lrck_pol to low
				for Left/Right Justified, set tx_lrck_pol to high

[8]	rw	1'h0	pcm_lsb_flag	Serial PCM data bit sequence. 0: MSB first, 1: LSB first
[7]	rw	1'h0	pcm_sync_flag	0: short sync, 1: long sync
[6:5]	rw	2'h0	pcm_tim_sel	00: I2S timing, 01: Left Justified 10: Right Justified, 11: PCM timing
[4:0]	rw	5'h8	pcm_dw	Baseband Master PCM data width (>=8) Common value: 8, 13,14, 16, 18, 20, 22, 24. for I2S/Left Justified/Right Kistified timing, bb_pcm_dw >=16 For PCM timing, only 8, 13, 14, 16 configure value is available.
0x220			**BT_PCM_DW**	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:5]			RSVD	
[4:0]	rw	5'h10	dw	BT PCM master data width (>= 8), common value: 8, 13,14, 16
0x230			**BT_PCM_TIMING**	
[31:3]			RSVD	
[2]	rw	1'h0	clk_pol	BT PCM master output pcm clock polarity: 0: rising edge for data transmitting, falling edge for data receiving 1: rising edge for data receiving, falling edge for data transmitting
[1]	rw	1'h0	sync_flag	0: short sync, 1: long sync
[0]	rw	1'h0	lsb_flag	Serial PCM data bit sequence. 0: MSB first, 1: LSB first
0x240			**BT_PCM_CLK_DUTY**	
[31:10]			RSVD	
[9:0]	rw	10'h0	clk_duty	BT_PCM_CLK duty cycle <= (GCLK/(bt_pcm_sync*bt_pcm_dw))
0x250			**BT_PCM_SYNC_DUTY**	
[31:6]			RSVD	
[5:0]	rw	6'h0	sync_duty	PCM_SYNC duty cycle (bt_pcm_sync frequency = bt_pclk_clk/bt_pcm_sync_duty)
0x260			**BT_VOL_CTRL**	
[31:4]			RSVD	
[3]	rw	1'h0	vol_adj_en	BT volume adjust enable
[2:0]	rw	3'h0	vol	BT master volume
0x300			**INT_MASK**	
[31:2]			RSVD	
[1]	rw	1'h1	tx_fifo_int_mask	Interrupt mask for TX FIFO pop underflow, high active
[0]	rw	1'h1	rx_fifo_int_mask	Interrupt mask for RX FIFO push overflow, high active
0x310			**INT_STATUS**	
[31:2]			RSVD	
[1]	rw	1'h0	tx_fifo_underflow	TX FIFO pop underflow
[0]	rw	1'h0	rx_fifo_overflow	RX FIFO push overflow
0x400			**TX_DMA_ENTRY**	
[31:0]	w	32'h0	tx_dma_entry	TX DMA entry
0x440			**RX_DMA_ENTRY**	
[31:0]	r	32'h0	rx_dma_entry	RX DMA entry
0x480			**DMA_MASK**	
[31:2]			RSVD	
[1]	rw	1'h1	tx_dma_mask	TX DMA mask enable:1: mask0: do not mask
[0]	rw	1'h1	rx_dma_mask	RX DMA mask enable:1: mask0: do not mask
0x500			**DEBUG_LOOP**	
[31:24]			RSVD	
[23:16]	rw	8'h2	sp_clk_div	sp clock divider value
[15:9]			RSVD	
[8]	w1c	1'h0	sp_clk_div_update	update sp clock divider
[7:3]			RSVD	
[2]	rw	1'h0	sp_clk_sel	clock select 0: xtal clock 1: pll clock
[1]	rw	1'h0	ad2da_loop_back	RX–>TX Loop debug control: 0: disable 1: enable, internally connect RX Resampled PCM to TX Resample PCM input

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1’h0	da2ad_loop_back	TX–>RX Loop debug control: 0: disable 1: enable, internally connect TX SDTO to RX SDTI
0x600			**FIFO_STATUS**	
[31:8]			RSVD	
[7:0]	rw	8’h0	fifo_status_out	FIFO Status output: Bit [7:0] = tx_full,tx_empty,tx_almost_full,tx_almost_empty,rx_full,rx_empty,rx_almost_fu
0x700			**TX_EQUALIZER_EN**	
[31:1]			RSVD	
[0]	rw	1’h0	tx_equalizer_en	0: Disable TX equalizer 1: Enable TX equalizer equalizer is not implemented
0x710			**TX_EQUALIZER_GAIN1**	
[31:30]			RSVD	
[29:25]	rw	5’h0	band6_gain	
[24:20]	rw	5’h0	band5_gain	
[19:15]	rw	5’h0	band4_gain	
[14:10]	rw	5’h0	band3_gain	
[9:5]	rw	5’h0	band2_gain	
[4:0]	rw	5’h0	band1_gain	
0x720			**TX_EQUALIZER_GAIN2**	
[31:20]			RSVD	
[19:15]	rw	5’h0	band10_gain	
[14:10]	rw	5’h0	band9_gain	
[9:5]	rw	5’h0	band8_gain	
[4:0]	rw	5’h0	band7_gain	

11.3 Audprc

11.3.1 Introduction

Audprc Module Full Name Audio Process Controller , whose primary function is to process collected and played audio data and transmit the processed data to the designated module. The Audprc module includes audio processing functions such as gain adjustment, mixing, equalization, and sample rate conversion, which users can configure individually as needed.

11.3.2 System Architecture

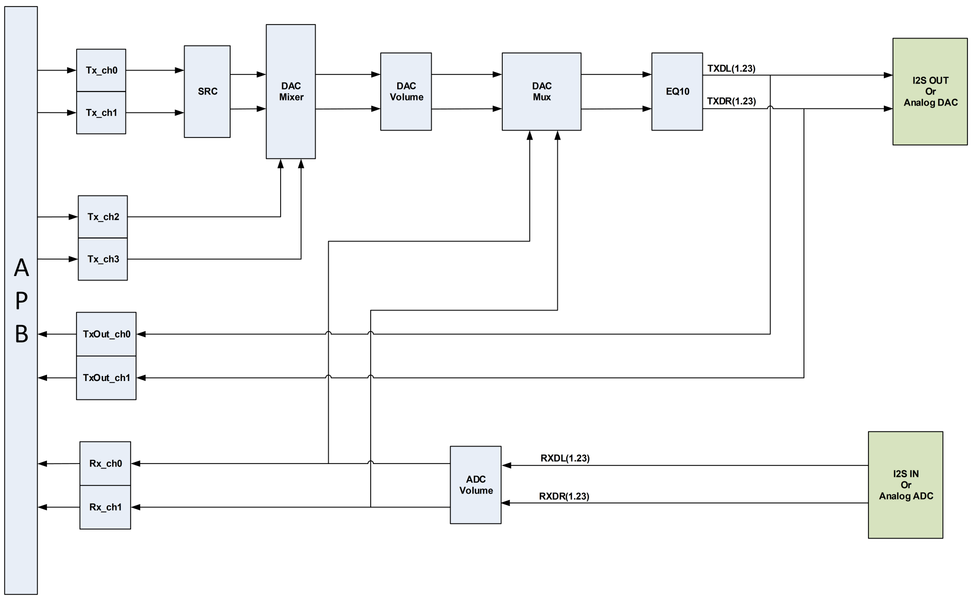


Figure 11-7: **Audprc Block Diagram**

The Audprc consists primarily of two pathways: the output TX pathway and the receiving RX pathway. The TX pathway transmits data from memory space to the I2S output or the analog DAC . The RX pathway receives data from the I2S input or the analog ADC and stores it in memory space.

11.3.3 Function Description

11.3.3.1 Sample Rate Conversion Module

The Sample Rate Conversion Module (SRC) can adjust the sample rate of audio data as required, with a conversion range of 1/16 to 16. This module comprises multiple stages of Half-band flters and Sinc flters. Users must configure the parameters of the Sample Rate Conversion Module based on the input and output sample rates to achieve the desired conversion ratio.

The Sample Rate Conversion Module includes a total of three stages of Half-band flters and one Sinc flter. Each stage of the Half-band flter can perform 2-fold upsampling or downsampling, while the Sinc flter can achieve high-precision sample rate conversion within the range of 0.5 to 2. The overall sample rate conversion ratio is:

Ratio = Ratio(hb1) * Ratio(hb2) * Ratio(hb3) * Ratio(sinc)

The structure of the third-level Half-band flter is identical, and the configuration method is also the same. If a 2 fold upsampling is required, set Hbf enable to 1 and mode to 0 . Conversely, if a 2 fold downsampling is needed, set Hbf enable to 1 and mode to 0 . By default, the Hbf enable is 0 , indicating that this level of the Half-band flter is inactive and will not alter the sampling rate. The configuration of the third-level Half-band filter should be as close as possible to the fnal target sampling rate conversion ratio.

The fnal level Sinc flter has two configuration parameters: enable and the sampling rate conversion ratio Sinc_ratio . The Sinc_ratio is a 31-bit fxed -point unsigned number, comprising 1 integer bit and 30 fractional bits, corresponding to a precision of $1/2^{30}$.

The following example will demonstrate how to configure SRC from 16KHz to 44.1KHz.

There are two relatively simple methods for converting16KHzto 44.1KHz:

- Option One:16KHz->32KHz->44.1KHz
- Option Two:16KHz->32KHz->64KHz->44.1KHz

Both methods are theoretically valid. In Option One, a first-order Half-band flter is employed for 2 fold upsampling, followed by a Sinc flter with a Ratio of 1.378 fold. In Option Two, two stages of Half-band flters are utilized for a total of 4 fold upsampling, followed by a Sinc flter with a Ratio of 0.689 fold. When comparing the two options, the Ratio of the Sinc flter in Option Two is closer to 1, resulting in better flter performance; therefore, Option Two is preferred.

11.3.3.2 Mixing Module

The Mixing Module(DAC Mixer, DAC Mux)is utilized to combine audio sources from different origins along the DAC path.

The DAC Mixer receives four TX data streams as input, which are recombined into two audio outputs. Each output is generated by summing two audio data streams, which may originate from the four TX data streams or be set to mute. For detailed information, please refer to registers mixlsrc0 , mixlsrc1 , mixrsrc0 , and mixrsrc1 .

The DAC Mux receives two data streams from the DAC and two data streams from the ADC. The mixing method employed by the Mux is identical to that of the Mixer, merging four inputs into two outputs, with each audio source being configurable. For detailed information, please refer to registers muxlsrc0 , muxlsrc1 , muxrsrc0 , and muxrsrc1.

11.3.3.3 Gain Adjustment Module

The DACand ADCpaths are managed by an independent gain adjustment module Volume, allowing for independent adjustment of the left and right audio channels. The range is 18~13dB, with a step size of 0.5dB.

11.3.3.4 Equalizer Module

The equalizer module EQ supports up to 10 levels of independent adjustment. Users must configure eq_stage to enable the desired number of levels. Each level consists of 5 parameters, resulting in a total of 50 configurable parameters. Users can utilize tools to generate the desired equalizer effect and subsequently configure the parameters into Audprc . Prior to use, the equalizer must first enable eq_clr , and once eq_clr_done is 1 , the eq_clr can be cleared. This step is essential to eliminate any residual internal temporary data. If this clearing is not performed, the equalizer module may experience a longer convergence time and could potentially produce transient noise upon activation.

11.3.4 Configuration Process

11.3.4.1 Configuration of Tx and Rx Channels

The Tx channel of Audprc supports mono 16bit/24bit and stereo 16bit audio data formats, with a maximum capacity of 4 audio channels (stereo occupies two channels). Each Tx channel can support a single channel 16bit/24bit audio by configuring the format register. For stereo 16bit audio sources (each 32bit data contains 16bit data for both the left and right channels), only Tx channel 0 and channel 2 are supported, which can be configured through the Mode register. When Tx channel 0 is configured to stereo 16bit mode, it will occupy Tx channel 1 and it will no longer be available for use. Similarly, when Tx channel 2 is configured to stereo 16bit mode, it will occupy Tx channel 3 and it will also no longer be available for use.

The Rx channel of the Audprc module supports mono 16bit/24bit audio data formats, accommodating a maximum of 2 audio streams. Each Rx channel can support a single 16bit/24bit audio stream by configuring the format register.

After configuring the audio data structures for Tx and Rx, it is also necessary to configure the corresponding DMA for the data transmission and reception processing of the Tx and Rx channels.

11.3.4.2 Configuring the DAC Path

The DAC path configuration includes various modules mentioned earlier, such as SRC, EQ, Mixer, Mux, and Volume. It is important to note that both SRC and EQ support enabling the two input channels independently; configuration only requires enabling the input channel that contains audio data. Additionally, when configuring the Mux, if the data from the Rx channel is mixed with the data from the Tx channel, it is essential to ensure that both have the same sampling rate; otherwise, synchronization issues may arise during mixing.

After configuring the DAC internal module, you can set the output target module of the DAC via the register dst_sel. If dst_sel is 0, then the DAC data will be directed to the Codec module, which supports up to 24-bit stereo. Following processing by the Codec module, the audio data will be converted into an analog audio signal for output. If dst_sel is 1, then the DAC data will be routed to the I2S module, which supports up to 16-bit stereo, and the data will ultimately be output through the digital I2S interface. If dst_sel is 2, then the DAC data will be sent to the Audprc Tx_out channel, which supports up to 24-bit stereo. Users can configure the DMA to read data from the Tx_out channel and store it in memory.

Additionally, the DAC path must also configure the sampling clock source and the sampling division ratio. The clock source can be selected from the chip's built-in audio PLL or a 48MHz crystal oscillator. It is essential to ensure that the clock source used by the DAC path is consistent with the output target module, and that the divided clock matches the sampling rate of the output module. If the output module is the Tx_out channel, this requirement does not apply.

11.3.4.3 Configuring the ADC Path

The ADC path configures a Volume module. After configuring the Volume, the audio source for the ADC can be set through the register src_sel. If src_sel is 0 , the ADC data source is the Codec module, which supports a maximum of 24-bit stereo. The Codec data is derived from the external audio analog signal collected by the ADC. If src_sel is 1, the ADC data source is the I2S module, which supports a maximum of 16-bit stereo, with audio data sourced from the external I2S digital interface.

Similar to the DAC path, the ADC path requires the configuration of the sampling clock source and the division ratio. The clock source must be consistent with the audio source, and the divided clock must match the audio sampling rate.

When ADC data is sent to the DAC path via the Mux, the ADC data to the Rx channel will remain unaffected, allowing the user to simultaneously collect ADC data from the enabled Rx channel.

11.3.5 Audprc Register

Audprc base address is 0x50005000.

Table 11-4: **Audprc Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**id**	
[31:0]	rw	32'hA0000	rev	revision id

Continued on the next page...

Table 11-4: **Audprc Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0x04			**cfg**	
[31:21]			RSVD	
[20]	w1c	1'h0	audclk_div_update	audprc clock divider update, write 1 to update
[19:16]	rw	4'h2	audclk_div	audprc clock divider, 0 and 1 means divide by 1
[15:10]			RSVD	
[9]	rw	1'h0	stb_clk_sel	audio strobe clock select 0: use xtal clock to generate strobe 1: use pll clock to generate strobe
[8]	rw	1'h0	auto_gate_en	auto clock gating enable, high active
[7]	rw	1'h0	adc_path_en	adc path enable
[6]	rw	1'h0	dac_path_en	dac path enable
[5]	rw	1'h0	adc_path_sreset	adc path software reset, high active
[4]	rw	1'h0	dac_path_sreset	dac path software reset, high active
[3]	rw	1'h0	adc_path_flush	adc path fifo flush, high active
[2]	rw	1'h0	dac_path_flush	dac path fifo flush, high active
[1]	rw	1'h0	sreset	audprc software reset, high active
[0]	rw	1'h0	enable	audprc enable
0x08			**stb**	
[31:16]	rw	16'h1	adc_div	adc strobe divider
[15:0]	rw	16'h1	dac_div	dac strobe divider
0x0C			**irq**	
[31:26]			RSVD	
[25]	rw	1'h0	tx_out1_fifo_uf_mask	tx_out channel 1 fifo underflow mask, 0: mask the interrupt
[24]	rw	1'h0	tx_out0_fifo_uf_mask	tx_out channel 0 fifo underflow mask, 0: mask the interrupt
[23]	rw	1'h0	rx_in_fifo_of_mask	rx input fifo overflow mask, 0: mask the interrupt
[22]	rw	1'h0	tx_out_fifo_uf_mask	tx output fifo underflow mask, 0: mask the interrupt
[21]	rw	1'h0	rx1_fifo_uf_mask	rx channel 1 fifo underflow mask, 0: mask the interrupt
[20]	rw	1'h0	rx0_fifo_uf_mask	rx channel 0 fifo underflow mask, 0: mask the interrupt
[19]	rw	1'h0	tx3_fifo_of_mask	tx channel 3 fifo overflow mask, 0: mask the interrupt
[18]	rw	1'h0	tx2_fifo_of_mask	tx channel 2 fifo overflow mask, 0: mask the interrupt
[17]	rw	1'h0	tx1_fifo_of_mask	tx channel 1 fifo overflow mask, 0: mask the interrupt
[16]	rw	1'h0	tx0_fifo_of_mask	tx channel 0 fifo overflow mask, 0: mask the interrupt
[15:10]			RSVD	
[9]	rw1c	1'h0	tx_out1_fifo_uf	tx_out channel 1 fifo underflow, write 1 to clear
[8]	rw1c	1'h0	tx_out0_fifo_uf	tx_out channel 0 fifo underflow, write 1 to clear
[7]	rw1c	1'h0	rx_in_fifo_of	rx input fifo overflow, write 1 to clear
[6]	rw1c	1'h0	tx_out_fifo_uf	tx output fifo underflow, write 1 to clear
[5]	rw1c	1'h0	rx1_fifo_uf	rx channel 1 fifo underflow, write 1 to clear
[4]	rw1c	1'h0	rx0_fifo_uf	rx channel 0 fifo underflow, write 1 to clear
[3]	rw1c	1'h0	tx3_fifo_of	tx channel 3 fifo overflow, write 1 to clear
[2]	rw1c	1'h0	tx2_fifo_of	tx channel 2 fifo overflow, write 1 to clear
[1]	rw1c	1'h0	tx1_fifo_of	tx channel 1 fifo overflow, write 1 to clear
[0]	rw1c	1'h0	tx0_fifo_of	tx channel 0 fifo overflow, write 1 to clear
0x10			**tx_ch0_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch0

Continued on the next page...

Table 11-4: Audprc Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	mode	tx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, tx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x14			**tx_ch0_entry**	
[31:0]	rw	32'h0	data	tx channel 0 data entry
0x18			**tx_ch1_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch1
[2]			RSVD	
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x1C			**tx_ch1_entry**	
[31:0]	rw	32'h0	data	tx channel 1 data entry
0x20			**tx_ch2_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch2
[2]	rw	1'h0	mode	tx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, tx channel 3 is not working, both left and right audio data comes from channel 2.
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x24			**tx_ch2_entry**	
[31:0]	rw	32'h0	data	tx channel 2 data entry
0x28			**tx_ch3_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch3
[2]			RSVD	
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x2C			**tx_ch3_entry**	
[31:0]	rw	32'h0	data	tx channel 3 data entry
0x30			**rx_ch0_cfg**	

Continued on the next page...

Table 11-4: **Audprc Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	rx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for rx ch0
[2]	rw	1'h0	mode	rx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, rx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	rx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	rx channel 0 enable
0x34			**rx_ch0_entry**	
[31:0]	r	32'h0	data	rx channel 0 data entry
0x38			**rx_ch1_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	rx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for rx ch1
[2]			RSVD	
[1]	rw	1'h0	format	rx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	rx channel 1 enable
0x3C			**rx_ch1_entry**	
[31:0]	r	32'h0	data	rx channel 1 data entry
0x40			**tx_out_ch0_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx out fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx out ch0
[2]	rw	1'h0	mode	tx out mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, rx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	tx out format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx out channel 0 enable
0x44			**tx_out_ch0_entry**	
[31:0]	r	32'h0	data	tx out channel 0 data entry
0x48			**tx_out_ch1_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx out fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx out ch1
[2]			RSVD	
[1]	rw	1'h0	format	tx out format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx out channel 1 enable

Continued on the next page...

Table 11-4: **Audprc Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0x4C			**tx_out_ch1_entry**	
[31:0]	r	32’h0	data	tx out channel 1 data entry
0x50			**dac_path_cfg0**	
[31:30]			RSVD	
[29:28]	rw	2’h0	dst_sel	dac path destination select 2’h0: select audio codec 2’h1: select external interface 2’h2: select apb interface 2’h3: reserved
[27:25]	rw	3’h3	mixrsrc1	dac mixer right channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:tx ch2 3’h3:tx ch3 3’h4:mute other: mute
[24:22]	rw	3’h2	mixrsrc0	dac mixer right channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:tx ch2 3’h3:tx ch3 3’h4:mute other: mute
[21:19]	rw	3’h1	mixlsrc1	dac mixer left channel input source1 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:tx ch2 3’h3:tx ch3 3’h4:mute other: mute
[18:16]	rw	3’h0	mixlsrc0	dac mixer left channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:tx ch2 3’h3:tx ch3 3’h4:mute other: mute
[15:12]	rw	4’h0	fine_vol_r	dac mixer right channel fine volume control range from 0dB to 6dB step is 0.5dB 4’h0: 0dB 4’h1: 0.5dB ...... 4’hb: 5.5dB 4’hc, 4’hd, 4’he, 4’hf: mute

Continued on the next page...

Table 11-4: Audprc Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:8]	rw	4’h6	rough_vol_r	dac mixer right channel rough volume control range from -36dB to 54dB step is 6dB 4’h0: -36dB 4’h1: -30dB ...... 4’h6: 0dB ...... 4’he: 48dB 4’hf: 54dB
[7:4]	rw	4’h0	fine_vol_l	dac mixer left channel fine volume control range from 0dB to 6dB step is 0.5dB 4’h0: 0dB 4’h1: 0.5dB ...... 4’hb: 5.5dB 4’hc, 4’hd, 4’he, 4’hf: mute
[3:0]	rw	4’h6	rough_vol_l	dac mixer left channel rough volume control range from -36dB to 54dB step is 6dB 4’h0: -36dB 4’h1: -30dB ...... 4’h6: 0dB ...... 4’he: 48dB 4’hf: 54dB
0x54			**dac_path_cfg1**	
[31:30]	rw	2’h0	src_ch_clr	clear src channal internal data
[29:28]	r	2’h0	src_ch_clr_done	src channel internal data clear done
[27]	rw	1’h0	src_hbf3_mode	3rd stage hbf mode: 0: upsampling 1: downsampling
[26]	rw	1’h0	src_hbf3_en	3rd stage hbf enable
[25]	rw	1’h0	src_hbf2_mode	2nd stage hbf mode: 0: upsampling 1: downsampling
[24]	rw	1’h0	src_hbf2_en	2nd stage hbf enable
[23]	rw	1’h0	src_hbf1_mode	1st stage hbf mode: 0: upsampling 1: downsampling
[22]	rw	1’h0	src_hbf1_en	1st stage hbf enable
[21:20]	rw	2’h0	src_ch_en	source rate converter channel enable
[19]	rw	1’h0	eq_clr	equalizer clear request
[18]	r	1’h0	eq_clr_done	equalizer clear done flag
[17:14]	rw	4’ha	eq_stage	set equalizer stage, max is 10.
[13:12]	rw	2’h0	eq_ch_en	equalizer channel enable 2’b11: enable both channel 2’b10: enable right chanel only 2’b01: enable left channel only 2’b00: bypass equalizer

Continued on the next page...

Table 11-4: Audprc Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:9]	rw	3’h3	muxrsrc1	dac mux right channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:rx ch0 3’h3:rx ch1 3’h4:mute other: mute
[8:6]	rw	3’h2	muxrsrc0	dac mux right channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:rx ch0 3’h3:rx ch1 3’h4:mute other: mute
[5:3]	rw	3’h1	muxlsrc1	dac mux left channel input source1 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:rx ch0 3’h3:rx ch1 3’h4:mute other: mute
[2:0]	rw	3’h0	muxlsrc0	dac mux left channel input source0 select 3’h0:tx ch0 3’h1:tx ch1 3’h2:rx ch0 3’h3:rx ch1 3’h4:mute other: mute
0x58			**dac_path_cfg2**	
[31]	rw	1’h0	src_sinc_en	sinc filter enable
[30:0]	rw	31’h0	sinc_ratio	sinc filter ratio, s31.30 format. Range from 0~2
0x5C			**dac_path_cfg3**	
[31:18]			RSVD	
[17:16]	r	2’h0	ramp_stat_r	dac mixer right channel ramp module status
[15:12]	rw	4’h0	ramp_interval_r	dac mixer right channel volume ramp interval.
[11]	rw	1’h0	zero_adjust_en_r	dac mixer right channel volume adjustment during 0 volume cross enable
[10]	rw	1’h0	ramp_mode_r	dac mixer right channel volume ramp mode: 1: slowly ramp to target volume. Step is 0.5db 0: directly ramp to target volume.
[9]	rw	1’h0	ramp_en_r	dac mixer right channel volume ramp enable
[8:7]	r	2’h0	ramp_stat_l	dac mixer left channel ramp module status
[6:3]	rw	4’h0	ramp_interval_l	dac mixer left channel volume ramp interval.
[2]	rw	1’h0	zero_adjust_en_l	dac mixer left channel volume adjustment during 0 volume cross enable
[1]	rw	1’h0	ramp_mode_l	dac mixer left channel volume ramp mode: 1: slowly ramp to target volume. Step is 0.5db 0: directly ramp to target volume.
[0]	rw	1’h0	ramp_en_l	dac mixer left channel volume ramp enable
0x60			**adc_path_cfg0**	
[31:19]			RSVD	
[18]	rw	1’h0	rx2tx_loopback	rx to tx loopback enable
[17]	rw	1’h0	data_swap	swap adc path left and right channel data

Continued on the next page...

Table 11-4: **Audprc Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[16]	rw	1’h0	src_sel	adc path source select 1’h0: select audio codec 1’h1: select external interface
[15:12]	rw	4’h0	fine_vol_r	adc right channel fine volume control range from 0dB to 6dB step is 0.5dB 4’h0: 0dB 4’h1: 0.5dB ...... 4’hb: 5.5dB 4’hc, 4’hd, 4’he, 4’hf: mute
[11:8]	rw	4’h6	rough_vol_r	adc right channel rough volume control range from -36dB to 54dB step is 6dB 4’h0: -36dB 4’h1: -30dB ...... 4’h6: 0dB ...... 4’he: 48dB 4’hf: 54dB
[7:4]	rw	4’h0	fine_vol_l	adc left channel fine volume control range from 0dB to 6dB step is 0.5dB 4’h0: 0dB 4’h1: 0.5dB ...... 4’hb: 5.5dB 4’hc, 4’hd, 4’he, 4’hf: mute
[3:0]	rw	4’h6	rough_vol_l	adc left channel rough volume control range from -36dB to 54dB step is 6dB 4’h0: -36dB 4’h1: -30dB ...... 4’h6: 0dB ...... 4’he: 48dB 4’hf: 54dB
0x70			**dac_eq_cfg0**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x74			**dac_eq_cfg1**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x78			**dac_eq_cfg2**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x7C			**dac_eq_cfg3**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x80			**dac_eq_cfg4**	
[31:24]			RSVD	

Continued on the next page...

Table 11-4: **Audprc Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[23:0]	rw	24’h0	coef	
0x84			**dac_eq_cfg5**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x88			**dac_eq_cfg6**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x8C			**dac_eq_cfg7**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x90			**dac_eq_cfg8**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x94			**dac_eq_cfg9**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x98			**dac_eq_cfg10**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0x9C			**dac_eq_cfg11**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xA0			**dac_eq_cfg12**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xA4			**dac_eq_cfg13**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xA8			**dac_eq_cfg14**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xAC			**dac_eq_cfg15**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xB0			**dac_eq_cfg16**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xB4			**dac_eq_cfg17**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xB8			**dac_eq_cfg18**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xBC			**dac_eq_cfg19**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xC0			**dac_eq_cfg20**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	
0xC4			**dac_eq_cfg21**	
[31:24]			RSVD	
[23:0]	rw	24’h0	coef	

Continued on the next page...

Table 11-4: Audprc Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xC8			**dac_eq_cfg22**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xCC			**dac_eq_cfg23**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD0			**dac_eq_cfg24**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD4			**dac_eq_cfg25**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD8			**dac_eq_cfg26**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xDC			**dac_eq_cfg27**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE0			**dac_eq_cfg28**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE4			**dac_eq_cfg29**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE8			**dac_eq_cfg30**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xEC			**dac_eq_cfg31**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xF0			**dac_eq_cfg32**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xF4			**dac_eq_cfg33**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xF8			**dac_eq_cfg34**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xFC			**dac_eq_cfg35**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x100			**dac_eq_cfg36**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x104			**dac_eq_cfg37**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x108			**dac_eq_cfg38**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x10C			**dac_eq_cfg39**	

Continued on the next page...

Table 11-4: Audprc Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x110			**dac_eq_cfg40**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x114			**dac_eq_cfg41**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x118			**dac_eq_cfg42**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x11C			**dac_eq_cfg43**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x120			**dac_eq_cfg44**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x124			**dac_eq_cfg45**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x128			**dac_eq_cfg46**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x12C			**dac_eq_cfg47**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x130			**dac_eq_cfg48**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x134			**dac_eq_cfg49**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x138			**RESERVED_IN**	
[31:24]			RSVD	
[23:16]	rw	8'hf	CTRL_2	reserved control 2
[15:8]	rw	8'hf	CTRL_1	reserved control 1
[7:0]	rw	8'hf	CTRL_0	reserved control 0
0x13C			**RESERVED_OUT**	
[31:8]			RSVD	
[7:0]	r	8'h0	STAT	reserved status

12 Accelerator

12.1 Digital Signal Processing Accelerator

12.1.1 Cordic Co-Processor

The Cordic co-processor is utilized for computing trigonometric and hyperbolic functions, as well as certain arithmetic operations derived from them. A Cordic co-processor is integrated solely within HPSYS.

The features of the Cordicco-processor are as follows:

- SupportsARM coprocessorinstructions.
- Supports ARM Custom Datapath Extension instructions (available only in HPSYS).
- Supports trigonometric operations: cos.、sin、ang、mod、atan、rot
- Supports hyperbolic operations: cosh.、sinh、atanh、angh、modh、mul、div、ln、exp、sqrt
- Supports 32-bit fxed-point input and output.

12.2 CRC

12.2.1 Introduction

CRC (Cyclic Redundancy Check) can perform CRC calculations with specific bit widths, arbitrary generating polynomials, and arbitrary initial values. Data can be input through CPU or DMA, with the minimum input unit being a single byte and no maximum byte limit. A single HCLK cycle can complete the calculation for a single byte input. The checksum result is obtained immediately after all data input is completed. It supports highlow bit reversal for input data and high-low bit reversal for output data. It also accommodates input data with varying valid bit widths.

12.2.2 Main Features

- 7/8/16/32 Bit CRCCalculation
- Arbitrary Custom Polynomial
- Arbitrary Initial Value
- Input data supports single-byte/double-byte/triple-byte/four-byte valid bit widths
- Input data supports byte/double-byte/four-byte high-low bit reversal
- Output data supports high-low bit reversal
- Calculation speed is 1 byte per HCLK cycle

12.2.3 CRC Configuration Method

Before starting the CRC calculation, it is necessary to pre-configure the corresponding Register, including polynomial width, valid data bit width, input-output reversal mode, polynomial, and initial value, etc. The mainstream CRC format configuration methods are shown in the table below.

CRC1 base address is 0x50048000.

Table 12-1: **CRC Configuration Method**

CRC Algorithm	Polynomial Formula	POLYSIZE	POL	INIT	REV_IN	REV_OUT	Result XOR Value
CRC-7/MMC	x^7+x^3+1	3	0x09	0x00	0	0	0x00
CRC-8	x^8+x^2+x+1	2	0x07	0x00	0	0	0x00
CRC-8/ITU	x^8+x^2+x+1	2	0x07	0x00	0	0	0x55
CRC-8/ROHC	x^8+x^2+x+1	2	0x07	0xFF	1	1	0x00
CRC-8/MAXIM	$x^8+x^5+x^4+1$	2	0x31	0x00	1	1	0x00
CRC-16/IBM	$x^{16}+x^5+x^2+1$	1	0x8005	0x0000	1	1	0x0000
CRC-16/MAXIM	$x^{16}+x^5+x^2+1$	1	0x8005	0x0000	1	1	0xFFFF
CRC-16/USB	$x^{16}+x^5+x^2+1$	1	0x8005	0xFFFF	1	1	0xFFFF
CRC-16/ MODBUS	$x^{16}+x^5+x^2+1$	1	0x8005	0xFFFF	1	1	0x0000
CRC-16/CCITT	$x^{16}+x^{12}+x^5+1$	1	0x1021	0x0000	1	1	0x0000
CRC-16/ CCITT-FALSE	$x^{16}+x^{12}+x^5+1$	1	0x1021	0xFFFF	0	0	0x0000
CRC-16/x5	$x^{16}+x^{12}+x^5+1$	1	0x1021	0xFFFF	1	1	0xFFFF
CRC-16/ XMODEM	$x^{16}+x^{12}+x^5+1$	1	0x1021	0x0000	0	0	0x0000
CRC-16/DNP	$x^{16}+x^{13}+x^{12}+x^{11}+x^{10}+x^8+x^6+x^5+x^2+1$	1	0x3D65	0x0000	1	1	0xFFFF
CRC-32	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$	0	0x04C11DB7	0xFFFFFFFF	1	1	0xFFFFFFFF
CRC-32/ MPEG-2	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$	0	0x04C11DB7	0xFFFFFFFF	0	0	0x00000000

The CRCmodule does not perform the XOR operation on the result; this must be handled by software after reading the result.

12.2.4 Data Format

The basic data unit for CRC calculation is a byte. The data written to the DR register is fxed at 4 bytes, and the bytes that participate in the calculation are specifed by CR_DATASIZE . The gray cells in the table below indicate which bytes are included in the calculation.

Table 12-2: **Data Participating in the Calculation**

CR_DATASIZE	DR			
0	BYTE3	BYTE2	BYTE1	BYTE0
1	BYTE3	BYTE2	BYTE1	BYTE0
2	BYTE3	BYTE2	BYTE1	BYTE0
3	BYTE3	BYTE2	BYTE1	BYTE0

Each byte of data participating in the calculation can be specifed individually; however, it is important to note that changing CR_DATASIZE must occur when SR_DONE is 1 ; otherwise, it may affect the calculation result of the current data.

The operation sequence is executed in order by BYTE0, BYTE1, BYTE2, BYTE3. When calculating each byte, it defaults to the order from the most signifcant bit to the least signifcant bit. If the input inversion mode is configured, the order will follow the inverted sequence from the most signifcant bit to the least signifcant bit. The table below provides configuration examples, which can be flexibly adjusted according to the data format in memory.

Table 12-3: **Operation Sequence**

DATASIZE	REV_IN	DR	Inverted Input	First Beat Calculate Byte	Second Beat Calculate Byte	Third Beat Calculate Byte	Fourth Beat Calculate Byte
0	0	0x12345678	/	0x78	/	/	/
1	0	0x12345678	/	0x78	0x56	/	/
2	0	0x12345678	/	0x78	0x56	0x34	/
3	0	0x12345678	/	0x78	0x56	0x34	0x12
3	1	0x12345678	0x482C6A1E	0x1E	0x6A	0x2C	0x48
3	2	0x12345678	0x2C481E6A	0x6A	0x1E	0x48	0x2C
3	3	0x12345678	0x1E6A2C48	0x48	0x2C	0x6A	0x1E

12.2.5 Calculation Rate

CRC Completes the calculation of one byte per HCLKcycle. When data is continuously input, the time required for calculation is approximately the number of bytes×HCLK cycles.

12.2.6 CRC Configuration Process

1. Configure the CRC format and set POL according to requirements., INIT, CR_POLYSIZE, CR_REV_IN, CR_REV_OUT, CR_DATASIZE。
2. Set CR_RESET to 1to initialize the CRC.
3. Continuously transfer the required data to the DR Register using the CPU or DMA.
4. If the remaining number of data bytes does not match CR_DATASIZE, first check SR_DONE; if it is 1, change CR_DATASIZE and write the remaining data into the DR Register.
5. Read the DRRegister to obtain the calculation result, and perform a software XOR operation as needed to derive the fnal CRCvalue.

12.2.7 CRC Register

Table 12-4: **CRC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**DR**	Data register
[31:0]	rw	32’hffffffff	DR	Data register bits. This register is used to write new data to the CRC calculator. It holds the previous CRC calculation result when it is read. If the data size is less than 32 bits, the least significant bits are used to write/read the correct value.
0x04			**SR**	Status register
[31:2]			RSVD	
[1]	r	1’h0	overflow	Overflow when new data arrive while last calculation not done yet
[0]	r	1’h0	done	Done flag. When DR written, done flag will be cleared automatically. The flag will assert after CRC operation of current DR finished.

Continued on the next page...

Table 12-4: CRC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x08			**CR**	Control register
[31:8]			RSVD	
[7]	rw	1'h0	REV_OUT	Reverse output data This bit controls the reversal of the bit order of the output data. 0: Bit order not affected 1: Bit-reversed output format
[6:5]	rw	2'h0	REV_IN	Reverse input data These bits control the reversal of the bit order of the input data 00: Bit order not affected 01: Bit reversal done by byte 10: Bit reversal done by half-word 11: Bit reversal done by word
[4:3]	rw	2'h0	POLYSIZE	Polynomial size These bits control the size of the polynomial. 00: 32 bit polynomial 01: 16 bit polynomial 10: 8 bit polynomial 11: 7 bit polynomial
[2:1]	rw	2'h3	DATASIZE	Valid input data size These bits control the valid size of the input data. 00: lower 8-bit 01: lower 16-bit 10: lower 24-bit 11: all 32-bit
[0]	w	1'h0	RESET	This bit is set by software to reset the CRC calculation unit and set the data register to the value stored in the CRC_INIT register. This bit can only be set, it is automatically cleared by hardware
0x10			**INIT**	Initial CRC value
[31:0]	rw	32'hffffffff	INIT	Programmable initial CRC value
0x14			**POL**	CRC polynomial
[31:0]	rw	32'h04c11db7	POL	Programmable polynomial This register is used to write the coefficients of the polynomial to be used for CRC calculation. If the polynomial size is less than 32 bits, the least significant bits have to be used to program the correct value.

13 Security

13.1 AES

13.1.1 Introduction

The AES_ACC module of the SF32LB52xis primarily designed to accelerate encryption and decryption operations of specialized algorithms in the security domain. Symmetric encryption algorithms include AES128 , AES192 , AES256 , and SM4 . Modes include ECB , CTR , and CBC . Hash algorithms include SHA1 , SHA224 , SHA256 , and SM3 . Upon activation, the AES_ACC module invokes the internal DMA to read raw data, and based on the algorithm, writes the corresponding results to the target address via the internal DMA or stores them in the module's internal registers.

13.1.2 AES Function Description

13.1.2.1 Symmetric Encryption Algorithm

The symmetric encryption algorithms primarily include AES and SM4 , where AES is further categorized into AES128 , AES192 , and AES256 based on the length of the Key . The SM4 algorithm is a national secret symmetric encryption algorithm. In terms of strength, a longer key length results in higher security; however, the time required for encryption and decryption also increases. The SM4 algorithm requires the most time.

13.1.2.2 Symmetric Encryption Mode

Symmetric encryption modes primarily include ECB, CTR, and CBC.

ECB Mode: The ECB mode encrypts and decrypts plaintext data directly using the KEY , processing data in groups of 16 bytes . Each encryption and decryption operation is performed on the entire 16 bytes. The advantage of this mode is that each group of data is independent, allowing for parallel computation and supporting random read and write operations by groups. The disadvantage is that if the plaintext of different groups is identical, the ciphertext will also be identical, making it susceptible to attacks.

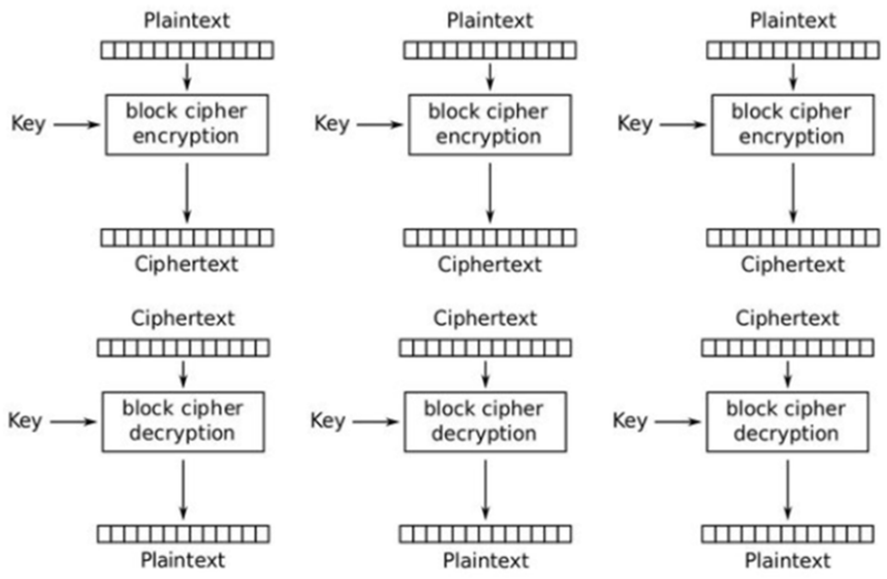


Figure 13-1: **ECB Mode Decryption**

CTR Mode: The CTR mode encrypts a vector composed of a NONCE and a COUNTER using a KEY , and then XORs the encrypted result with the plaintext data to produce the ciphertext, thereby achieving encryption. During decryption, the encrypted result of the same vector is XORed with the ciphertext to retrieve the plaintext data, thus completing the decryption process. In practical applications, the NONCE is typically represented by a constant, while the COUNTER uses the address of the data, ensuring that each data group has a different vector, which in turn makes the XOR data used in the encryption and decryption processes distinct. Understanding the composition of the vector used in CTR , the encryption and decryption of each data group are independent and can be computed in parallel, similar to the ECB mode. Additionally, this mode only involves encryption and XOR operations, making its structure relatively simple. The operations on the vector during encryption and decryption are independent of the data, which is why this mode is commonly used for encrypting and decrypting externally stored data.

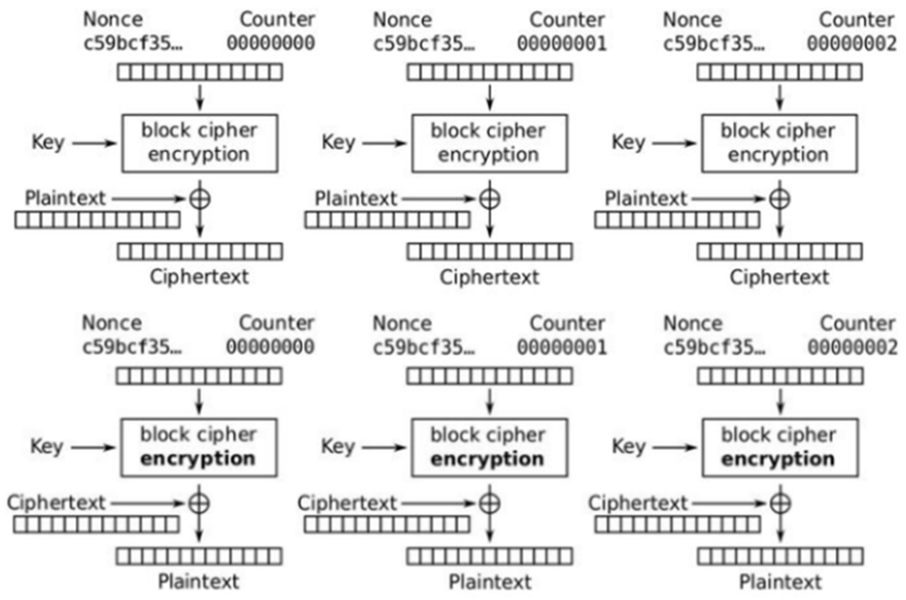


Figure 13-2: **CTR Mode Decryption**

CBC Mode: The CBC mode utilizes the ciphertext of the previous block as the initialization vector, which is then XORed with the current block before being encrypted with KEY to produce the ciphertext. During decryption, the ciphertext is decrypted using KEY and then XORed with the previous block's ciphertext to generate the plaintext. This mode can perform both encryption and decryption, and the last block of ciphertext can also serve as the MAC value for integrity verifcation. Due to the interrelation of each block of data during the encryption and decryption process, it provides enhanced security; however, it cannot execute parallel computations and does not allow for random read or write operations on individual blocks of data.

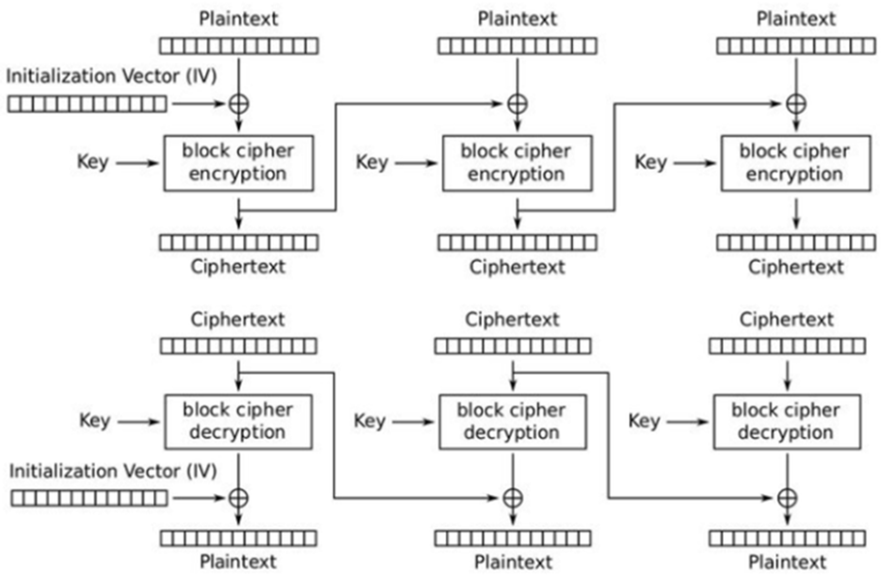


Figure 13-3: **CBC Mode Decryption**

13.1.2.3 Multiple Calls for Symmetric Encryption and Decryption

At times, it is necessary to perform encryption and decryption operations on large volumes of data. However, due to limitations in storage capacity or transmission rates, the data may need to be processed in multiple batches. When repeatedly invoking the symmetric encryption and decryption module for continuous processing of a large segment of data, it is crucial to consider the preservation and maintenance of the data context between calls.

ECB Mode:

The ECB mode conducts separate encryption and decryption operations for each group of 16 bytes of data. For the data to be encrypted or decrypted in a single operation, it must be ensured that the data volume is a multiple of 16 bytes. Any excess data can be merged and processed once the next batch is ready.

CTR Mode:

The CTR mode assigns a corresponding NONCE andCOUNTER value for each group of 16 bytes of data. Typically, theNONCE is a constant, while theCOUNTER serves as the identifer for the current data group, incrementing by 1 for each operation. For the data to be encrypted or decrypted in a single operation, the upper-level software must ensure that the data volume is an integer multiple of 16 bytes , while recording the COUNTER value based on the data volume. In the next call, the accumulated COUNTER value will be used as the initial vector input to the corresponding IV Register. Any excess data can be processed once the next batch is ready.

CBC Mode:

In CBC mode, the initial vector for each group of data is derived from the ciphertext of the previous group. For single encryption and decryption operations, the upper-level software must ensure that the data volume is an integer multiple of 16 bytes , and it is necessary to record the ciphertext of the last group of data from the current operation. This will serve as the initial vector input to the corresponding IV Register in the next operation. Any excess data will be processed once the next batch is ready.

13.1.2.4 Hash Algorithm

Hash algorithms include SHA1 , SHA224 , SHA256 , and SM3 , each with different digest lengths. The digest length for SHA1 is 160 bits , for SHA224 is 224 bits , and both SHA256 and SM3 are 256 bits . In terms of collision resistance, a longer digest results in a lower probability of collision.

13.1.2.5 Multiple Calls for Hash Value Calculation

At times, it is necessary to compute hash values for large datasets. However, due to limitations in storage capacity or transmission rates, the data may need to be divided into multiple batches for processing. When multiple calls to the hash value calculation module are required to handle a large segment of data, the following points should be considered.

First, except for the last processing instance, the amount of data processed each time must be a multiple of 4 bytes . Any excess data from a single processing instance must be saved and merged with the next batch of data for processing.

Second, for all hashing algorithms, before each calculation, the intermediate results of the previous hash calculation H0~H7 must be written into the corresponding registers. The hash value calculation module should then be configured to use the external initial values of H0 H7 and loaded into the module.

Third, each time HASH is called, the HASH LEN RESULT from the previous call must be updated to the current HASH LEN register and loaded into the module.

Fourth, except for the last call, the hash value calculation module should be configured not to use padding.

13.1.3 AES Register

AES base address is 0x5000D000.

Table 13-1: **AES Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**COMMAND**	
[31:5]			RSVD	
[4]	rw	1’h0	AUTO_GATE	auto clock gating
[3]	rw	1’h0	HASH_RESET	HASH_ACC soft reset, 1’h1: reset the HASH_ACC block
[2]	w1t	1’h0	HASH_START	write 1 to trigger the HASH_ACC block
[1]	rw	1’h0	AES_ACC_RESET	AES_ACC soft reset, 1’h1: reset the AES_ACC block
[0]	w1t	1’h0	START	write 1 to trigger the AES_ACC block
0x04			**STATUS**	
[31:3]			RSVD	
[2]	r	1’h0	HASH_BUSY	HASH_ACC block is busy
[1]	r	1’h0	FLASH_KEY_VALID	flash key valid indicator
[0]	r	1’h0	BUSY	AES_ACC block is busy
0x08			**IRQ**	
[31:22]			RSVD	

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21]	rw1c	1'h0	HASH_PAD_ERR_RAW_STAT	HASH_ACC padding error raw status
[20]	rw1c	1'h0	HASH_BUS_ERR_RAW_STAT	HASH_ACC bus error raw status
[19]	rw1c	1'h0	HASH_DONE_RAW_STAT	HASH_ACC done raw status
[18]	rw1c	1'h0	SETUP_ERR_RAW_STAT	AES_ACC setup error raw status
[17]	rw1c	1'h0	BUS_ERR_RAW_STAT	AES_ACC bus error raw status
[16]	rw1c	1'h0	DONE_RAW_STAT	AES_ACC done raw status
[15:6]			RSVD	
[5]	rw1c	1'h0	HASH_PAD_ERR_STAT	HASH_ACC padding error status
[4]	rw1c	1'h0	HASH_BUS_ERR_STAT	HASH_ACC bus error status
[3]	rw1c	1'h0	HASH_DONE_STAT	HASH_ACC done status
[2]	rw1c	1'h0	SETUP_ERR_STAT	AES_ACC setup error status
[1]	rw1c	1'h0	BUS_ERR_STAT	AES_ACC bus error status
[0]	rw1c	1'h0	DONE_STAT	AES_ACC done status
0x0C			**SETTING**	
[31:6]			RSVD	
[5]	rw	1'h0	HASH_PAD_ERR_MASK	HASH_ACC padding error interrupt mask, 0: mask the interrupt
[4]	rw	1'h0	HASH_BUS_ERR_MASK	HASH_ACC bus error interrpt mask, 0: mask the interrupt
[3]	rw	1'h0	HASH_DONE_MASK	HASH_ACC done interrupt mask, 0: mask the interrupt
[2]	rw	1'h0	SETUP_ERR_IRQ_MASK	AES_ACC setup error interrupt mask, 0: mask the interrupt
[1]	rw	1'h0	BUS_ERR_IRQ_MASK	AES_ACC bus error interrupt mask, 0: mask the interrupt
[0]	rw	1'h0	DONE_IRQ_MASK	AES_ACC done interrupt mask, 0: mask the interrupt
0x10			**AES_SETTING**	
[31:9]			RSVD	
[8]	rw	1'h0	AES_BYPASS	1'h0: normal operation 1'h1: bypass
[7]	rw	1'h0	AES_OP_MODE	1'h0: decryption 1'h1: encryption
[6]	rw	1'h0	ALGO_STANDARD	1'h0: AES 1'h1: SM4
[5]	rw	1'h0	KEY_SEL	1'h0: select key from AES_ACC key registers 1'h1: use internal root key
[4:3]	rw	2'h0	AES_LENGTH	AES Length: 2'h0: 128-bit 2'h1: 192-bit 2'h2: 256-bit 2'h3: Reserved
[2:0]	rw	3'h0	AES_MODE	AES Mode: 3'h0: ECB 3'h1: CTR 3'h2: CBC Others: Reserved
0x14			**DMA_IN**	
[31:0]	rw	32'h0	ADDR	AES_ACC input data address
0x18			**DMA_OUT**	
[31:0]	rw	32'h0	ADDR	AES_ACC output data address
0x1C			**DMA_DATA**	
[31:28]			RSVD	
[27:0]	rw	28'h0	SIZE	AES_ACC data block size, AES_ACC only support block aligned transaction. Each block contains 16 bytes.
0x20			**IV_W0**	
[31:0]	rw	32'h0	DATA	Initial Vector Word0
0x24			**IV_W1**	

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	DATA	Initial Vector Word1
0x28			**IV_W2**	
[31:0]	rw	32'h0	DATA	Initial Vector Word2
0x2C			**IV_W3**	
[31:0]	rw	32'h0	DATA	Initial Vector Word3
0x30			**EXT_KEY_W0**	
[31:0]	rw	32'h0	DATA	External Key Word0
0x34			**EXT_KEY_W1**	
[31:0]	rw	32'h0	DATA	External Key Word1
0x38			**EXT_KEY_W2**	
[31:0]	rw	32'h0	DATA	External Key Word2
0x3c			**EXT_KEY_W3**	
[31:0]	rw	32'h0	DATA	External Key Word3
0x40			**EXT_KEY_W4**	
[31:0]	rw	32'h0	DATA	External Key Word4
0x44			**EXT_KEY_W5**	
[31:0]	rw	32'h0	DATA	External Key Word5
0x48			**EXT_KEY_W6**	
[31:0]	rw	32'h0	DATA	External Key Word6
0x4C			**EXT_KEY_W7**	
[31:0]	rw	32'h0	DATA	External Key Word7
0x50			**HASH_SETTING**	
[31:9]			RSVD	
[8]	w1t	1'h0	HASH_LEN_LOAD	write 1 to load hash length
[7]	w1t	1'h0	HASH_IV_LOAD	write 1 to load hash iv
[6]	rw	1'h0	RESULT_ENDIAN	hash result endian setting: 1'h0: little endian 1'h1: big endian
[5]	rw	1'h0	DFT_IV_SEL	HASH default iv select. 1'h0: default iv according to hash mode 1'h1: default iv from HASH_IV_H* registers
[4]	rw	1'h0	BYTE_SWAP	HASH byte swap option. Set 1 to swap byte order when read data from memory.
[3]	rw	1'h0	DO_PADDING	HASH padding enable. Set 1 to do padding after data transfer.
[2:0]	rw	3'h0	HASH_MODE	HASH Mode: 3'h0: SHA-1 3'h1: SHA-224 3'h2: SHA-256 3'h3: SM3 Others: Reserved
0x54			**HASH_DMA_IN**	
[31:0]	rw	32'h0	ADDR	input data address
0x58			**HASH_DMA_DATA**	
[31:0]	rw	32'h0	SIZE	HASH input data byte size.
0x5C			**HASH_IV_H0**	
[31:0]	rw	32'h0	DATA	HASH IV H0
0x60			**HASH_IV_H1**	
[31:0]	rw	32'h0	DATA	HASH IV H1
0x64			**HASH_IV_H2**	
[31:0]	rw	32'h0	DATA	HASH IV H2
0x68			**HASH_IV_H3**	

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	DATA	HASH IV H3
0x6C			**HASH_IV_H4**	
[31:0]	rw	32'h0	DATA	HASH IV H4
0x70			**HASH_IV_H5**	
[31:0]	rw	32'h0	DATA	HASH IV H5
0x74			**HASH_IV_H6**	
[31:0]	rw	32'h0	DATA	HASH IV H6
0x78			**HASH_IV_H7**	
[31:0]	rw	32'h0	DATA	HASH IV H7
0x7C			**HASH_RESULT_H0**	
[31:0]	r	32'h0	DATA	HASH result H0
0x80			**HASH_RESULT_H1**	
[31:0]	r	32'h0	DATA	HASH result H1
0x84			**HASH_RESULT_H2**	
[31:0]	r	32'h0	DATA	HASH result H2
0x88			**HASH_RESULT_H3**	
[31:0]	r	32'h0	DATA	HASH result H3
0x8C			**HASH_RESULT_H4**	
[31:0]	r	32'h0	DATA	HASH result H4
0x90			**HASH_RESULT_H5**	
[31:0]	r	32'h0	DATA	HASH result H5
0x94			**HASH_RESULT_H6**	
[31:0]	r	32'h0	DATA	HASH result H6
0x98			**HASH_RESULT_H7**	
[31:0]	r	32'h0	DATA	HASH result H7
0x9C			**HASH_LEN_L**	
[31:0]	rw	32'h0	DATA	HASH load length l
0xA0			**HASH_LEN_H**	
[31:29]			RSVD	
[28:0]	rw	29'h0	DATA	HASH load length h
0xA4			**HASH_RESULT_LEN_L**	
[31:0]	r	32'h0	DATA	HASH result length l
0xA8			**HASH_RESULT_LEN_H**	
[31:29]			RSVD	
[28:0]	r	29'h0	DATA	HASH result length h

13.2 TRNG

13.2.1 Introduction

TRNG stands for True Random Number Generator . This module utilizes an oscillation circuit composed of digital logic to generate random entropy sources, which undergo a series of verifcations to produce a random number seed. This seed is then used by a pseudo-random number generator to generate random numbers for system use.

13.2.2 Module Architecture

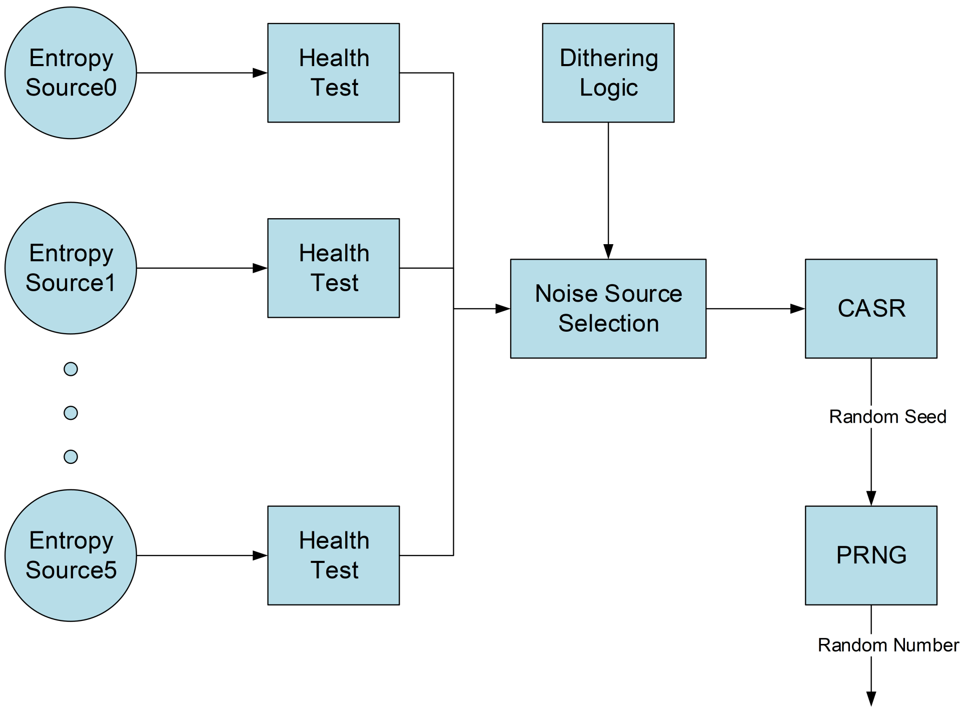


Figure 13-4: **TRNG Schematic Diagram**

The True Random Number Generator primarily consists of 3 components: entropy source, random seed generator, and random number generator.

13.2.3 Function Description

13.2.3.1 Entropy Source

The entropy source consists of six inverter chains. Each time a new random seed is generated, all inverter chains are activated, and a specific verifcation algorithm is employed to generate the random number seed. The generation time for the random number seed is relatively long, and it also consumes a signifcant amount of power. It is generally recommended to regenerate a new seed in two scenarios: first, when the original seed's lifecycle has expired, necessitating the periodic generation of new seeds to overwrite the old ones; and second, when the software requires a manual generation of a new seed under special circumstances.

The entropy source is activated each time a seed is generated. The VN verifer threshold is used to flter the entropy source; a higher threshold allows for a more lenient selection of the entropy source, resulting in faster seed generation, although the randomness may decrease.

Users can directly trigger gen_seed_startto activate the entropy source module for generating new random seeds.

13.2.3.2 Random Seed Generator

The random seed generator collects data from the entropy source and generates random seeds through the CASR (Cellular Automata Shift Register) module, which are provided for use by the next level PRNG. Users can also obtain random seeds directly through the register. When the user configures use_ext_seed to 1, the next level PRNG module will not utilize the seeds generated by the random seed generator, but will instead use external seeds to generate random numbers.

13.2.3.3 Random Number Generator

The random number generator is a pseudo-random number generator based on existing random seeds, with a relatively short and fxed duration. It is important to note that the pseudo-random number generator has a probability of entering a self-locking state due to certain internal data sequences that can cause the linear feedback register to deadlock. When the prng_lockup state is detected, it is advisable to regenerate a new set of random seeds to produce new random numbers.

Users can trigger gen_rand_num_start to initiate the random number generator. If the current random seed has not been generated, the random number generator will first create a random seed and then generate random numbers.

13.2.3.4 Other Functional Modules

The TRNG module additionally provides measurement of the oscillation period of the inverter chain, which can be used to infer the process corner of the chip. The process begins by determining the number of the inverter chain used, as different inverter chains have varying lengths and corresponding oscillation periods. Next, the number of oscillation periods must be set, which corresponds to the frequency of the system pclk . Once the measurement is enabled, two counters will track the clock cycles of pclk and the clock cycles of the inverter chain, stopping when the number of clock cycles of pclk reaches the set threshold. At this point, the frequency of the inverter chain can be calculated using the formula Finv=Fpclk*Npclk/Ninv , based on the clock frequency Fpclk , the number of periods Npclk , and the number of periods of the inverter chain Ninv . This allows for inferring the process corner corresponding to the chip.

13.2.4 TRNG Register

TRNG base address is 0x5000F000.

Table 13-2: **TRNG Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CTRL**	
[31:5]			RSVD	
[4]	rw	1’h0	gen_rand_num_suspend	Set 1 to suspend random number generation and update. Set 0 to recover the process.
[3]	rw	1’h0	gen_rand_num_stop	Set 1 to stop random number generation and update. This will reset the random number generation engine. After release the stop bit, user should write 1 to gen_rand_num_start to trigger the random number engine.
[2]	rw	1’h0	gen_seed_stop	Set 1 to stop random seed generation. This will reset the random seed generation engine. After release the stop bit, user should write 1 to gen_seed_start to trigger the random seed engine.
[1]	w1t	1’h0	gen_rand_num_start	write 1 to trigger the random number generation engine
[0]	w1t	1’h0	gen_seed_start	write 1 to trigger the random seed generation engine
0x04			**STAT**	
[31:4]			RSVD	
[3]	r	1’h0	rand_num_valid	random number valid flag
[2]	r	1’h0	rand_num_gen_busy	random number engine busy flag
[1]	r	1’h0	seed_valid	random seed valid flag
[0]	r	1’h0	seed_gen_busy	random seed engine busy flag
0x08			**CFG**	
[31:16]			RSVD	

Continued on the next page...

Table 13-2: TRNG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:8]	rw	8'ha	reject_threshold	random seed internal VN corrector check threshold
[7:2]			RSVD	
[1]	rw	1'h0	use_ext_seed	set 1 to use external seed to generate random number
[0]	rw	1'h0	auto_clock_enable	auto clock gating enable
0x0c			**IRQ**	
[31:19]			RSVD	
[18]	rw	1'h1	prng_lockup_msk	prng lockup interrupt mask
[17]	rw	1'h1	rand_num_avail_msk	random number available interrupt mask
[16]	rw	1'h1	seed_gen_done_msk	random seed generation done interrupt mask
[15:3]			RSVD	
[2]	rw1c	1'h0	prng_lockup	prng lockup raw interrupt
[1]	rw1c	1'h0	rand_num_avail	random number available raw interrupt
[0]	rw1c	1'h0	seed_gen_done	random seed generation done raw interrupt
0x10			**rand_seed0**	
[31:0]	rw	32'h0	val	random seed value0. If using external random seed, write value to this register will update the random seed in use.
0x14			**rand_seed1**	
[31:0]	rw	32'h0	val	random seed value1. If using external random seed, write value to this register will update the random seed in use.
0x18			**rand_seed2**	
[31:0]	rw	32'h0	val	random seed value2. If using external random seed, write value to this register will update the random seed in use.
0x1c			**rand_seed3**	
[31:0]	rw	32'h0	val	random seed value3. If using external random seed, write value to this register will update the random seed in use.
0x20			**rand_seed4**	
[31:0]	rw	32'h0	val	random seed value4. If using external random seed, write value to this register will update the random seed in use.
0x24			**rand_seed5**	
[31:0]	rw	32'h0	val	random seed value5. If using external random seed, write value to this register will update the random seed in use.
0x28			**rand_seed6**	
[31:0]	rw	32'h0	val	random seed value6. If using external random seed, write value to this register will update the random seed in use.
0x2c			**rand_seed7**	
[31:0]	rw	32'h0	val	random seed value7. If using external random seed, write value to this register will update the random seed in use.
0x30			**rand_num0**	
[31:0]	r	32'h0	val	random number value0
0x34			**rand_num1**	
[31:0]	r	32'h0	val	random number value1
0x38			**rand_num2**	
[31:0]	r	32'h0	val	random number value2
0x3c			**rand_num3**	
[31:0]	r	32'h0	val	random number value3
0x40			**rand_num4**	
[31:0]	r	32'h0	val	random number value4
0x44			**rand_num5**	
[31:0]	r	32'h0	val	random number value5
0x48			**rand_num6**	
[31:0]	r	32'h0	val	random number value6
0x4c			**rand_num7**	

Continued on the next page...

Table 13-2: TRNG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	r	32'h0	val	random number value7
0x50			**cal_cfg**	
[31:16]	rw	16'hff	length	calibration length
[15:6]			RSVD	
[5]	r	1'h0	done	calibration done
[4]	rw	1'h0	enable	calibration enable
[3:1]	rw	3'h0	osc_clk_sel	osc clock select
[0]	rw	1'h0	osc_clk_force_on	osc force enable
0x54			**cal_result**	
[31:16]	r	16'h0	osc_cnt	osc clock calibration counter result
[15:0]	r	16'h0	pclk_cnt	pclk calibration counter result

13.3 efusec

13.3.1 ntroduction

efusec refers to efuse control, which can control the corresponding bits of the efuse through write operations. The information of the blown bit is read back as 1, and the stored information value can be retrieved through read operations.

13.3.2 Main Features

- Can control 4 blocks of 256-bit efuse units
- Can read/write 1024-bit information
- Can control the read/write operation of one efuseunit at a time
- Read/write operations are single bitserial
- Completion of reading/writing can generate an interrupt signal
- The functionality for read and write masking can be implemented
- Certain stored information is directly output through the module interface
- The default value of efuse is all 0

13.3.3 Write Operation

The write operation of efusec is a programming operation for efuse , where the data written is a 1 bit. This bit is set to an open circuit by means of fusing, resulting in a read value of 1 . The information to be written is configured through registers PGM_DATA0~7 , with four units sharing 8 PGM_DATA* registers. The control flow of the write operation is as follows:

- Configure the BANKSEL register to select the efuse unit to be written
- Configure the MODE register to select the write operation
- Configure the IEregister to enable interrupt status.
- Configure the PGM_DATA* register to write programming data.
- Configure the TIMER register to set the critical duration.
- Configure the EN register to initiate the write operation.
- Wait for the interrupt to arrive or query the SR register to obtain the write completion status.

The blown bit cannot be restored to a 0 state, while the unblown bit can be blown again through a write operation.

13.3.4 Read Operation

The read operation of eFusec involves reading the stored information of efuse , and the retrieved information can be queried through the BANK*_DATA* registers, with each unit having an independent register for data querying. The control flow for the read operation is as follows:

- Configure the BANKSEL register to select the efuse unit for reading
- Configure the MODEregister to select the read operation
- Configure the IEregister to enable interrupt status.
- Configure the TIMERregister to set the critical duration.
- Configure the ENregister to initiate the write operation.
- Wait for an interrupt to arrive or query the SRregister to obtain the read completion status
- Read theBANK*_DATA*register to obtain the read value
- A portion of the read value is directly transmitted through the interface signal

13.3.4.1 Read and Write Timing Control

The read and write timing of efuse has specific requirements, particularly regarding the fuse time. When the working clock of the efusec module changes, the clock count value must be adjusted to meet the read and write timing requirements of efuse . The timing control points that need to be addressed are as follows:

- TCKHP : Single bitfuse time, 10us
- THPCK : Hold time for enabling write, >20 ns
- THRCK : Hold time for enabling read, >500 ns

These three times can be configured based on the working clock count value through the TIMEregister. The reset value of the register corresponds to the count value at a 48 MHzworking clock.

13.3.4.2 Read/Write Masking Function

To prevent malicious read and write operations, after configuring the data and ceasing data updates, the read and write functions of the efuse modules in banks 1 to 3 can be masked by controlling the fxed bit of bank0. Once masked, writing to the corresponding efuse is not possible, and the read values cannot be updated.

- bank0[255:254]=2'b11, Mask bank3write function;
- bank0[253:252]=2'b11, Mask bank3read function;
- bank0[251:250]=2'b11, Mask bank2write function;
- bank0[249:248]=2'b11, Mask bank2read function;
- bank0[247:246]=2'b11, Mask bank1write function;
- bank0[245:244]=2'b11, Mask bank1read function;

13.3.4.3 Module Interface Output Signals

Some values of efusewill be output through the module interface signals, which are:

Table 13-3: efuse interface signals

Serial Numbe	Signal Name
1	idsel
2	swddis
3	pkgid[1:0]
4	uid[127:0]
5	rootkey[255:0]

13.3.5 efusec Register

efusec base address is 0x5000C000.

Table 13-4: efusec Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31:5]			RSVD	
[4]	rw	1’h0	IE	Interrupt enable
[3:2]	rw	2’h0	BANKSEL	Bank select
[1]	rw	1’h0	MODE	0 - READ, 1 - PGM
[0]	w1s	1’h0	EN	Write 1 to enable PGM/READ. Self clear
0x04			**TIMR**	Timer Register
[31:21]			RSVD	
[20:10]	rw	11’h78	TCKHP	SCLK high period for PGM. Recommended value ~10us
[9:7]	rw	3’h0	THPCK	SCLK to CSB hold time into PGM mode. Recommended value > 20ns
[6:0]	rw	7’h6	THRCK	SCLK to CSB hold time into READ mode. Recmmended value > 500ns
0x08			**SR**	Status Register
[31:1]			RSVD	
[0]	rw1c	1’h0	DONE	Indicates PGM/READ done. Write 1 to clear
0x0C			**RSVDR**	Reserved Register
0x0C			RSVDR	
[31:0]			RSVD	
0x10			**PGM_DATA0**	Program Data0
[31:0]	rw	32’h0	DATA	
0x14			**PGM_DATA1**	Program Data1
[31:0]	rw	32’h0	DATA	
0x18			**PGM_DATA2**	Program Data2
[31:0]	rw	32’h0	DATA	
0x1C			**PGM_DATA3**	Program Data3
[31:0]	rw	32’h0	DATA	
0x20			**PGM_DATA4**	Program Data4
[31:0]	rw	32’h0	DATA	
0x24			**PGM_DATA5**	Program Data5
[31:0]	rw	32’h0	DATA	
0x28			**PGM_DATA6**	Program Data6
[31:0]	rw	32’h0	DATA	
0x2C			**PGM_DATA7**	Program Data7
[31:0]	rw	32’h0	DATA	
0x30			**BANK0_DATA0**	Bank0 Data0
[31:0]	r	32’h0	DATA	
0x34			**BANK0_DATA1**	Bank0 Data1
[31:0]	r	32’h0	DATA	
0x38			**BANK0_DATA2**	Bank0 Data2

Continued on the next page...

Table 13-4: efusec Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	r	32'h0	DATA	
0x3C			**BANK0_DATA3**	Bank0 Data3
[31:0]	r	32'h0	DATA	
0x40			**BANK0_DATA4**	Bank0 Data4
[31:0]	r	32'h0	DATA	
0x44			**BANK0_DATA5**	Bank0 Data5
[31:0]	r	32'h0	DATA	
0x48			**BANK0_DATA6**	Bank0 Data6
[31:0]	r	32'h0	DATA	
0x4C			**BANK0_DATA7**	Bank0 Data7
[31:0]	r	32'h0	DATA	
0x50			**BANK1_DATA0**	Bank1 Data0
[31:0]	r	32'h0	DATA	
0x54			**BANK1_DATA1**	Bank1 Data1
[31:0]	r	32'h0	DATA	
0x58			**BANK1_DATA2**	Bank1 Data2
[31:0]	r	32'h0	DATA	
0x5C			**BANK1_DATA3**	Bank1 Data3
[31:0]	r	32'h0	DATA	
0x60			**BANK1_DATA4**	Bank1 Data4
[31:0]	r	32'h0	DATA	
0x64			**BANK1_DATA5**	Bank1 Data5
[31:0]	r	32'h0	DATA	
0x68			**BANK1_DATA6**	Bank1 Data6
[31:0]	r	32'h0	DATA	
0x6C			**BANK1_DATA7**	Bank1 Data7
[31:0]	r	32'h0	DATA	
0x70			**BANK2_DATA0**	Bank2 Data0
[31:0]	r	32'h0	DATA	
0x74			**BANK2_DATA1**	Bank2 Data1
[31:0]	r	32'h0	DATA	
0x78			**BANK2_DATA2**	Bank2 Data2
[31:0]	r	32'h0	DATA	
0x7C			**BANK2_DATA3**	Bank2 Data3
[31:0]	r	32'h0	DATA	
0x80			**BANK2_DATA4**	Bank2 Data4
[31:0]	r	32'h0	DATA	
0x84			**BANK2_DATA5**	Bank2 Data5
[31:0]	r	32'h0	DATA	
0x88			**BANK2_DATA6**	Bank2 Data6
[31:0]	r	32'h0	DATA	
0x8C			**BANK2_DATA7**	Bank2 Data7
[31:0]	r	32'h0	DATA	
0x90			**BANK3_DATA0**	Bank3 Data0
[31:0]	r	32'h0	DATA	
0x94			**BANK3_DATA1**	Bank3 Data1
[31:0]	r	32'h0	DATA	
0x98			**BANK3_DATA2**	Bank3 Data2
[31:0]	r	32'h0	DATA	
0x9C			**BANK3_DATA3**	Bank3 Data3
[31:0]	r	32'h0	DATA	
0xA0			**BANK3_DATA4**	Bank3 Data4

Continued on the next page...

Table 13-4: **efusec Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	r	32'h0	DATA	
0xA4			**BANK3_DATA5**	Bank3 Data5
[31:0]	r	32'h0	DATA	
0xA8			**BANK3_DATA6**	Bank3 Data6
[31:0]	r	32'h0	DATA	
0xAC			**BANK3_DATA7**	Bank3 Data7
[31:0]	r	32'h0	DATA	
0xB0			**ANACR**	Bank3 Data7
[31:24]	r	8'h0	RESERVE1	
[23:16]	rw	8'h0	RESERVE0	
[15:11]			RSVD	
[10:8]	rw	3'h0	LDO_DC_TR	
[7:5]			RSVD	
[4]	rw	1'h0	LDO_MODE	
[3:1]	rw	3'h4	LDO_VREF_SEL	
[0]	rw	1'h0	LDO_EN	
0xB4			**DB_SEL**	debug signal select
[31:0]	rw	32'h0	db_sel	debug signal select

14 Storage Interface

14.1 SD/SDIO/eMMC

HPSYS has one SDMMC module. It uses the same io as MPI2(PA12~PA17), so the external Flash cannot be accessed through MPI2 when using the SDMMC function.

14.1.1 Introduction

SDMMC supports SD protocol 3.0 and eMMC standard 4.51 and can function as a HOST controller to interact with SD/SDIO/eMMC devices, working in conjunction with DMAC for data read and write operations. SDMMC supports SDR single line and 4 line modes, but does not support DDR or SPI mode.

14.1.2 Main Features

- Compatible with SD Host Controller Standard Specification Version 3.0
- Compatible with SD 3.0 Physical Layer Specification Version 3.01
- Compatible with SDIO Specification Version 3.0
- Compatible with JEDEC JESD84-B451 eMMC 4.51 Specification
- SupportsSDSC/SDHC/SDXC/SDHScards
- SupportsSDR12/SDR25/SDR50
- Supports SDRsingle-line and4line modes
- Built-in 2K bytes FIFO, with a maximum support for a single block of 512 bytes
- Configurable clock
- Operates with DMACfor data transfer

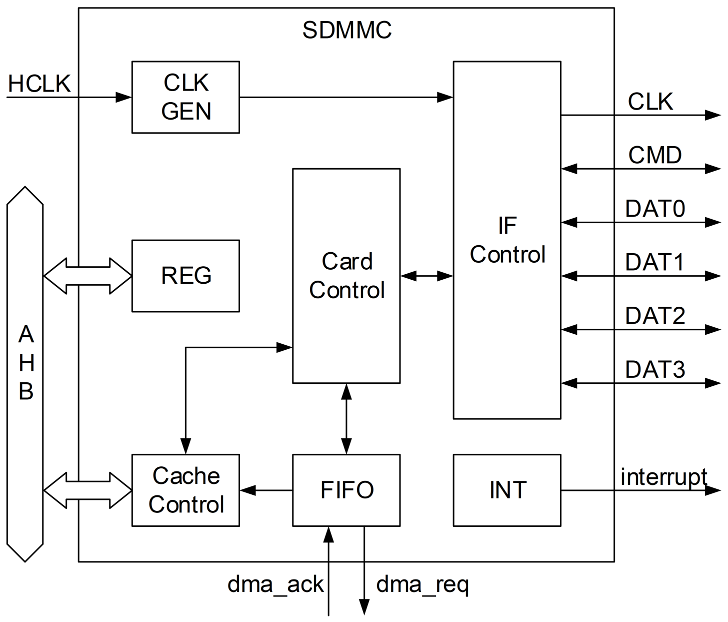


Figure 14-1: **SDMMC Block Diagram**

14.1.3 Function Description

14.1.3.1 SD/eMMC Interface

The controller supports a SD/eMMC interface that includes one CLK line, one CMD line, and 4 DAT lines (DAT0-DAT3). In single-line mode, only one DAT line is utilized(DAT0). The CLK line is used for clock output. The CMD line transmits CMD commands and responses RSP according to the protocol. The DAT lines transmit data streams in accordance with the protocol. This module only supports single-edge (SDR) transmission, with CMD and DAT being valid only on the rising edge of CLK.

14.1.3.2 Clock Settings

The controller operates at the system clock frequency, referred to as HCLK. The output to the SD/eMMC interface's CLK is generated by dividing the HCLK frequency, with the division ratio specifed by CLKCR_DIV. The division formula is CLK frequency = HCLK frequency / (CLKCR_DIV + 1). For example, when HCLK is 48M, to output a CLK of 400KHz, it is necessary to configure CLKCR_DIV to 119. The minimum supported setting for CLKCR_DIV is 1, which means the division ratio is at least 2.

The CLK output is controlled by CLKCR_STOP_CLK. When set to 0, the clock will continue to output; when set to 1, the output will be disabled.

14.1.3.3 Sending Commands

Commands are transmitted by the controller via the CMD line in a fxed-format packet received by SD/eMMC/SDIO devices, which is used to configure the status of SD/eMMC/SDIO devices and control data transmission. The length of the command packet is fxed at 48 bits.

Command Packet Format						
Bit position	47	46	[45:0]	[39:8]	[7:1]	0
Width(bits)	1	1	6	32	7	1
Value	0	1	x	x	x	1
Description	Start bit	Transimission bit	Command index	Argument	CRC7	End bit

Each command is distinguished by a 6-bit command number, accompanied by a 32-bit parameter. The device uses a 7-bit CRC to verify the correctness of the received command and sends a response when necessary.

Based on the 6-bit command number, commands are typically designated as CMDn, where nrepresents the command number configured by the CCR_CMD_INDEX Register.

The 32-bitparameter of the command is configured by the CARRegister.

Certain commands, such as CMD0 and CMD4 , do not require a response; others, such as CMD8 and CMD11 , require a 48-bit response; while commands like CMD2 require a 136-bit response. Before sending a command, it is essential to configure the CCR_CMD_HAS_RSP and CCR_CMD_LONG_RSP registers to select the expected response for the command.

Response Configuration				
Response	CMD Example	RSP Example	CCR_CMD_HAS_RSP	CCR_CMD_LONG_RSP
None	CMD0， CMD4	None	0	0
48bit	CMD8， CMD11	R1,R7	1	0
136bit	CMD2， CMD9	R2	1	1

Before the controller sends a command, it must complete the configuration of the command and set the CCR_CMD_TX_EN bit. Subsequently, it should set the CMD_START bit, and the controller will initiate the command sending process.

There are two ways to conclude the command sending process:：

1.Command completed; SR_CMD_DONEis asserted high, and an interrupt can be generated whenIER_CMD_DONE_MASKis set to 1.When the command is completed and does not require to receive a response, or when a response is received within the timeout period, regardless of whether the response’s CRC is correct, the command will be completed. If a CRC error occurs during response reception, SR_CMD_RSP_CRC will be asserted high, and an interrupt can be generated when IER_CMD_RSP_CRC_MASK is set to 1.

2.Command timeout; SR_CMD_TIMEOUT is asserted high, and an interrupt can be generated when IER_CMD_TIMEOUT_MASK is set to 1. When the command is completed and requires a response, but no response is received within the timeout period configured in the TOR register (no response received for the start bit) , a timeout will occur. Once a command timeout occurs, the controller must be reset (via the RCC module) before sending the next command, and the configuration must be restored (only the SDMMC controller needs to be reconfigured; the state of interaction with the device does not need to change).

The command sending process can be checked via SR_CMD_BUSY to determine if it has completed. When the current command sending process is not fnished, the controller will not respond to new command sending requests. During debugging, if SR_CMD_BUSY remains low for an extended period, please verify whether the TOR settings are appropriate and check if a reset and reconfiguration of the controller were performed after the previous command timed out.

For certain commands, when expecting a response (such as R1b) , it is also necessary to check the busy status feedback from the device through the DAT line, which can be determined by querying the DSR register to see if busy has been cleared.

The command index of the received response is stored in the RIR register, while the argument is stored in RAR1 and RAR4 , with the 48-bit response argument stored in RAR1.

Suggested process for command transmission:

1. Configure index, argument, and expected response type, then initiate the command transmission
2. Wait for either the command completion interrupt or the command timeout interrupt.
3. If the command completes, check the returned response and retrieve the argument.
4. If the command times out, reset the controller and reconfigure (primarily including clock settings and timeout duration, etc.).

14.1.3.4 Data Transmission

Data transmission encompasses reading data from the device and writing data to the device, initiated by the controller sending specific commands (such as CMD17, CMD24, etc.), concluding after a preset length of data transmission, or terminated by the controller sending a specific command (such as CMD12).

Data transmission defined by the SD and SDIO protocols can utilize either single-line or 4 line modes. The data transmission defined by the eMMC protocol can employ single-line, 4 line, or 8 line modes. This controller exclusively supports single-line or 4 line transmission, which is configured through the DCR_WIRE_MODE Register. In single-line mode, data is transmitted via DAT0, which also indicates the busy status of the data transmission. In 4 line mode, data is transmitted through DAT0 DAT3, with DAT0 also indicating the busy status of the data transmission.

Data is transmitted in blocks, with the amount of data per block varying according to the device type. For SD and eMMC devices, the typical block size is 512 bytes , but there are exceptions. Some devices allow the command (such as CMD16) to change the amount of data per block. Before initiating data transmission, the controller should configure the DCR_BLOCK_SIZE Register to the appropriate value (the number of bytes per block equals DCR_BLOCK_SIZE plus 1) . The controller must also configure the DLR_DATA_LEN Register to determine the total amount of data transmitted in a single operation (the total number of bytes equals DLR_DATA_LEN plus 1) . For single block transmission, DLR_DATA_LEN should equal DCR_BLOCK_SIZE . For multi-block transmission, (DLR_DATA_LEN+1) should be an integer multiple of (DCR_BLOCK_SIZE+1) .

The direction of data transmission is configured by DCR_R_WN. The DCR_TRAN_DATA_EN register enables data transmission and should remain high during the transmission process. DCR_DATA_START is used to initiate data transmission.

Data transmission must pass through the internal FIFO cache of the controller. The data to be sent is written to the FIFO by the CPU or DMA, after which the controller transfers the data from the FIFO to the device. Data read from the device is also temporarily stored in the FIFO and is then retrieved by the CPU or DMA. The FIFO register address space occupies a total of 512 bytes, and accessing any aligned address within it yields the same effect. If the FIFO overflows or underflows, it will generate SR_FIFO_OVERRUN or SR_FIFO_UNDERRUN records, which can trigger an interrupt

Data transmission may either complete normally or time out if data or CRC is not received within the specifed time. In either case, SR_DATA_DONE will be asserted high and can generate an interrupt. If a CRC error is detected while reading the device, or if a CRC error is returned by the device during a write operation, an SR_DATA_CRC record will be generated. If the data start bit is not detected within the timeout period configured by TOR while reading the device, or if the CRC response from the device is not received within the timeout period configured by TOR while writing, an SR_DATA_TIMEOUT record will be generated.

The status of data transmission completion can be queried through SR_DATA_BUSY. When the current data transmission is not complete, the controller will not respond to new data transmission requests.

Recommended process for data reading:

1.Configure the single block data amount and total data amount, then initiate data reading:
2. Configure DMACfor reading FIFO, and then start DMAC.
3. Send the read command (e.g. CMD17).
4. Wait for the command interrupt, then process the device response (e.g. R1).
5. Wait for the data transfer completion interrupt.
6. If it is a multi-block read, send the transfer end command (e.g. CMD12).
7. Wait for the DMAcompletion interrupt.

Recommended process for data writing:

1. Send the write command (e.g. CMD24).
2. Wait for the command interrupt, then process the device response (e.g. R1).
3. Configure the single block data amount and total data amount, then initiate data writing.
4. Configure DMACfor writing FIFO, and then start DMAC.
5. Waiting for the data transfer complete interrupt
6. If it is a multi-block write, send the transfer end command (such as CMD12)

14.1.3.5 Interrupt Generation

SDMMC can generate interrupts to notify the CPU of specific events. These events include command completion, command timeout, data completion, data timeout, and various errors. Events that can generate interrupts are configured through the IER register, and occurring events can be queried through the SR register.

14.1.3.6 FIFO Management

SDMMC features a built-in 2KB FIFO for caching read and write data. The FIFO can generate DMA requests based on the fullness of its contents, working with DMAC to complete data transfers. During multi-block data transfers, the bandwidth of DMAC typically meets the device's data transfer bandwidth, allowing for continuous data transfer. However, if the storage space at the other end of DMAC becomes blocked and does not respond to FIFO requests in a timely manner, it may cause FIFO to experience overflow or underflow, resulting in data transfer errors. To prevent this situation, the CLKCR_VOID_ FIFO_ERROR register can be configured so that when the contents of FIFO cannot meet the data transfer requirements, the interface CLK is automatically halted, and data transfer is stopped until the contents of FIFO meet the requirements. At this point, a pause in the CLK can be observed on the interface.

14.1.3.7 eMMC Open Drain Mode

When the eMMC device is in certain states (such as inactive , idle , or identification state; refer to the eMMC protocol for details), the CMD line must be configured as open drain mode by setting CDR_CMD_ODto 1.

When accessing SD/SDIO devices, and when the eMMC enters other states, the CMD line must be configured as push-pull mode, meaning CDR_CMD_OD should be set to 0 .

14.1.3.8 SDIO Interrupt

SDIO devices can generate SDIO interrupts to notify the controller by pulling down DAT1.

If the controller needs to respond to SDIO interrupts, in single-line mode, the CEATA_ENABLE_SDIO_IRQ Register must be set to 1 . In four-line mode, it is additionally required to set CEATA_SDIO_4WIRES_IRQ to 1 .

In four-line mode, interrupt requests during multi-block data transfer intervals for CMD53 are not supported (Block Gap Interrupt) . However, interrupt requests before a single CMD data transfer or after the transfer is completed can be responded to normally.

14.1.3.9 Card Detection

SD cards support hot swapping. The detection of card insertion and ejection is typically implemented in the following 3methods.

1. Through a dedicated CD (card detect) pin for detection. SD card slots that support insertion detection usually provide a dedicated card detection pin (CD) , which has a pull-up resistor connected to the power supply. The chip needs to allocate an independent IO to detect the level of this pin. When no SD card is inserted, the pin level is high. When the SD card is inserted into the slot, the pin connects to ground, and the level is low. By determining the GPIO input level of this IO , the insertion and ejection of the SD card can be detected.

2. Detection is performed through the DAT3 pin. When the SD card is powered on, there is a 50K ohm pull-up resistor from the internal DAT3 of the card to the power supply. The external circuit of the chip must include a weak pull-down resistor to ground on the DAT3 pin (typically greater than 470K ohm). The chip determines whether the card is inserted by detecting the IO level connected to DAT3. When the SD card is not inserted, the pin level is low. When the SD card is inserted into the slot, the pin is pulled up to a high level. By checking the GPIO input level of this IO, the insertion and ejection of the SD card can be detected. When the SD card begins data transmission, this detection should be halted, and DAT3 should be utilized as a normal data line.

3.Detection through command polling. The chip initiates command interactions with the SDcard at regular intervals, determining the card’s presence based on whether a response is received.

SDMMC only supports card detection via DAT3. SR_CARD_EXIST reports the card’s presence based on the level of DAT3, while card insertion and removal events are reported by SR_CARD_INSERT and SR_CARD_REMOVE, which can generate interrupts.

14.1.3.10 Bus Direct Read Mode

SDMMC allows the data space of the SD card to be mapped onto the AHB bus, enabling other master control modules to directly access the address to read data from the SD card via the AHB bus. Before utilizing this feature, the SDMMC must first configure the SD card into data transfer mode. Subsequently, when there is a read access to the mapped space on the bus, the SDMMC automatically issues commands to the SD card and reads multiple blocks of data, storing the data in the built-in cache (shared storage space with FIFO) and returning the data via the bus. If the data to be read from the bus is located in the cache, the SDMMC will not initiate a new read command but will instead return the results directly from the cache.

14.1.3.11 Sampling Clock Adjustment

When communicating with the device via SDMMC , it is essential to sample the CMD and DAT signals from the device on the rising edge of the CLK . Due to signal transmission delays, it may be necessary to adjust the sampling clock edge to ensure accurate sampling. The CLKCR_CLK_TUNE_SEL register can be utilized to adjust the sampling clock edge backward, offering a total of 4 levels for adjustment based on the actual conditions of the product. When the CLK frequency exceeds 50M , it is often necessary to adjust the sampling clock to ensure the proper functionality of the interface.

14.1.4 SDMMC Register

SDMMC1 base address is 0x50045000。

Table 14-1: **SDMMC Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**SR**	command and data status register
[31:18]			RSVD	
[17]	rw	1’h0	cache_err	Detect cache error Read 1: cache error occur Read 0: no cache error Write 1: clear the bit Write 0: no any influence to the bit
[16]	rw	1’h0	sdio	Detect SDIO Card Interrupt Read 1: detect sdio card generating interrupt Read 0: no interrupt Write 1: clear the bit Write 0: no any influence to the bit
[15]	r	1’h0	card_exist	Card exist status Read 1: card exist Read 0: no card exist This bit will be valid after enable detect card.
[14]	rw	1’h0	card_remove	Detect card removed Read 1: detect card removed. When detect card inserted bit is set, the bit will also be back to 0 Read 0: no meaning Write 1: clear the bit Write 0: no any influence to the bit
[13]	rw	1’h0	card_insert	Detect card inserted Read 1: detect card inserted. When detect card removed bit is set, the bit will also be back to 0 Read 0: no meaning Write 1: clear the bit Write 0: no any influence to the bit
[12]	rw	1’h0	cmd_sent	Command sent (perhaps no response back yet) Read 1: command sent. When command start bit is set, the bit will also be back to 0 Read 0: command transferring or others Write 1: clear the bit Write 0: no any influence to the bit
[11]			RSVD	

Continued on the next page...

Table 14-1: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	rw	1'h0	fifo_overrun	FIFO overrun Read 1: FIFO overrun error Read 0: no FIFO overrun error Write 1: clear the bit Write 0: no any influence to the bit
[9]	rw	1'h0	fifo_underrun	FIFO underrun Read 1: FIFO underrun error Read 0: no FIFO underrun error Write 1: clear the bit Write 0: no any influence to the bit
[8]	rw	1'h0	startbit_error	Wide bus start bits error Didn't detect all start bits in data bus Read 1: start bits error Read 0: no start bits error Write 1: clear the bit Write 0: no any influence to the bit
[7]	rw	1'h0	data_timeout	Data timeout Read 1: timeout Read 0: no timeout Write 1: clear the bit Write 0: no any influence to the bit
[6]	rw	1'h0	data_crc	Data CRC error Read 1: data CRC error Read 0: data CRC right Write 1: clear the bit Write 0: no any influence to the bit
[5]	rw	1'h0	data_done	Data transfer done Read 1: transfer data done, and start a new transfer will take the bit into 0 Read 0: data transferring or idle Write 1: clear the bit Write 0: no any influence to the bit
[4]	r	1'h0	data_busy	Transfer Data busy 1: busy, and when busy, start transfer data bit is no usage and you should not modify the relative register. If want to do this, first disable transfer data enable bit, then the busy bit will be back to 0, and this transfer will also be cancelled. 0: data idle
[3]	rw	1'h0	cmd_timeout	Command timeout (response timeout) Read 1: timeout Read 0: no timeout Write 1: clear the bit Write 0: no any influence to the bit
[2]	rw	1'h0	cmd_rsp_crc	Command response CRC error status Read 1: response CRC error Read 0: response CRC right Write 1: clear the bit Write 0: no any influence to the bit
[1]	rw	1'h0	cmd_done	Command done Read 1: transfer command done, and start a new transfer will take the bit into 0 Read 0: command transferring or idle Write 1: clear the bit Write 0: no any influence to the bit

Continued on the next page...

Table 14-1: **SDMMC Register Mapping Table (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	r	1'h0	cmd_busy	Command busy 1: busy, and when busy, start TX command bit is no usage and should not modify the relative register 0: command idle
0x04			**CCR**	command control register
[31:24]			RSVD	
[23:18]	rw	6'h0	cmd_index	Command index
[17]	rw	1'h0	cmd_long_rsp	1: Response will be 136-bit, long response 0: Response will be 48-bit, normal response
[16]	rw	1'h1	cmd_has_rsp	1: Response expected after command 0: No response expected after command
[15:10]			RSVD	
[9]	rw	1'h0	cmd_pend	Command pending enable When prepare to send stop command, this bit should be set. Controller will calculate a proper time point to send out the command to guarantee all the data have been transferred. And this is mainly used in stream mode. Recommend using set_block_count (SD/MMC basis command) to control transferring data for block mode. If send stop command for canceling this transfer (such as CRC error in multi-block), no need to set the bit.
[8]	rw	1'h0	cmd_tx_en	TX command enable 1: enable TX command 0: disable TX command
[7:1]			RSVD	
[0]	rw	1'h0	cmd_start	Command start write 1 to start command TX, and when begin to TX command, the bit will return into 0.
0x08			**CAR**	command argument register
[31:0]	rw	32'h0	cmd_arg	Command argument
0x0C			**RIR**	response command index register
[31:6]			RSVD	
[5:0]	r	6'h0	rsp_index	Response command index
0x10			**RAR1**	response command argument1 register
[31:0]	rw	32'h0	rsp_arg1	Response command content If long response, it is rsp_arg[39:8]
0x14			**RAR2**	response command argument2 register
[31:0]	rw	32'h0	rsp_arg2	Long response, it is rsp_arg[71:40]
0x18			**RAR3**	response command argument3 register
[31:0]	rw	32'h0	rsp_arg3	Long response, it is rsp_arg[103:72]
0x1C			**RAR4**	response command argument4 register
[31:24]			RSVD	
[23:0]	rw	24'h0	rsp_arg4	Long response, it is rsp_arg[127:104]
0x20			**TOR**	timeout count register
[31:0]	rw	32'h1000	timeout_cnt	Used to determine how much time waiting response or data bus busy is timeout, and decreased under card clock. Set to 400000 for 1s timeout if interface clock is 400KHz.
0x24			**DCR**	data control register
[31:27]			RSVD	
[26:16]	rw	11'h1ff	block_size	Data block size is block_size+1 (max 2048 bytes) 0: 1 byte 0x1ff: 512 bytes
[15:13]			RSVD	

Continued on the next page...

Table 14-1: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[12:11]	rw	2'h0	wire_mode	Wide data bus mode 00: 1 wire bus 01: 4 wires wide bus 1X: reserved
[10]	rw	1'h0	stream_mode	Data transfer mode 0: block 1: stream
[9]	rw	1'h0	r_wn	Write or read 0: write data into card 1: read data from card
[8]	rw	1'h0	tran_data_en	Transfer data enable 0: disable transfer data. After disable data transfer, stop command should be sent to card 1: enable data transfer
[7:1]			RSVD	
[0]	rw	1'h0	data_start	Start transfer data set 1 to let the controller begin to transfer data (in fact, go into wait write or wait read state). After begin to transfer, this bit will be back to 0.
0x28			**DLR**	data length register
[31:16]	r	16'h0	block_tran_num	The number of blocks which have been transferred successfully 1 = 1 block transferred It is cleared when start transfer data bit is set.
[15:0]	rw	16'h1ff	data_len	Data length value. The number of data bytes is data_len+1. The number of data bytes should be a multiple of data block size. 0 is 1 byte. 0x1ff is 512 bytes. Max is 63.5KB.
0x2C			**IER**	command and data interrupt mask register
[31:18]			RSVD	
[17]	rw	1'h1	cache_err_mask	cache error mask for interrupt
[16]	rw	1'h0	sdio_mask	Detect SDIO interrupt(data[1]) mask for interrupt
[15]			RSVD	
[14]	rw	1'h0	card_remove_mask	Detect card remove mask for interrupt
[13]	rw	1'h0	card_insert_mask	Detect card insert mask for interrupt
[12]	rw	1'h0	cmd_sent_mask	Command sent mask for interrupt
[11]			RSVD	
[10]	rw	1'h1	fifo_overrun_mask	FIFO overrun bit mask for interrupt
[9]	rw	1'h1	fifo_underrun_mask	FIFO underrun bit mask for interrupt
[8]	rw	1'h0	startbit_error_mask	Wide bus start bits error bit mask for interrupt
[7]	rw	1'h1	data_timeout_mask	Data timeout bit mask for interrupt
[6]	rw	1'h1	data_crc_mask	Data CRC error bit mask for interrupt
[5]	rw	1'h1	data_done_mask	Data transfer done bit mask for interrupt
[4]			RSVD	
[3]	rw	1'h1	cmd_timeout_mask	Command timeout bit mask for interrupt
[2]	rw	1'h1	cmd_rsp_crc_mask	Command CRC error bit mask for interrupt
[1]	rw	1'h1	cmd_done_mask	Command done bit mask for interrupt
[0]			RSVD	
0x30			**CLKCR**	clock control register
[31:21]			RSVD	
[20:8]	rw	13'h80	div	Divide card clock counter. 0 is illegal. sd_clock = hclk/(div + 1) If hclk is 240M and div is 599, 400KHz SD clock will be generated.
[7:4]			RSVD	

Continued on the next page...

Table 14-1: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:2]	rw	2'h0	clk_tune_sel	select clock delay for rx sample 0: no delay 1: delay level 1 (~1.5ns typical) 2: delay level 2 (~3ns typical) 3: delay level 3 (~5ns typical)
[1]	rw	1'h0	void_fifo_error	Void FIFO error 0: close the function 1: open the function If open it, when FIFO will be overrun or underrun soon, the SD_CLK and the clock enable of this module will be closed, and wait to host to read or write FIFO. Note: this function needs to be supported by card.
[0]	rw	1'h1	stop_clk	Disable SD card clock 1: stop SD card clock 0: SD card clock generated
0x3C			**CDR**	card interface control and card detect register
[31:19]	rw	13'h0	otiming	define output timing
[18:6]	rw	13'h0	itiming	define input timing
[5]	rw	1'h0	cmd_od	Open Drain mode for cmd line (for eMMC identification mode) 0: cmd line is push-pull 1: cmd line is open-drain
[4]	rw	1'h1	cd_hvalid	Card detect high level valid 0: detect low level means card exist 1: detect high level means card exist (default)
[3]	rw	1'h1	en_cd	Enable card detect Only when the bit is valid, controller does card detect. If use sd_data[3] to do card detect, the bit should be cleared when transfer valid data.
[2]	rw	1'h0	otiming_sel	select output timing (according to otiming config)
[1]	rw	1'h0	itiming_sel	select input sample timing (according to itiming config)
[0]	rw	1'h1	sd_data3_cd	Use sd_data[3] to do card detect 0: use special pin to do card detect / write protect. (Currently not supported) 1: use sd_data[3] to do card detect (default)
0x40			**DBGR1**	card debug port1 register
[31]			RSVD	
[30:16]	r	15'h1	data_st	data state for debug only
[15:0]	r	16'h1	cmd_st	command state for debug only
0x44			**DBGR2**	card debug port2 register
[31:30]	rw	2'h0	dbg_sel	for debug only
[29:26]			RSVD	
[25:16]	r	10'h0	valid_data_cou	for debug only
[15:14]			RSVD	
[13:0]	r	14'h0	host_word_counter	for debug only
0x48			**CEATA**	CE-ATA/SDIO mode register
[31:4]			RSVD	
[3]	rw	1'h0	sdio_4wires_multi_irq	Select the sdio host 4 wires interrupt on multi-block support 0: host not support 4 wires interrupt on multi-block data transfers 1: host support 4 wires interrupt on multi-block data transfers
[2]	rw	1'h0	sdio_4wires_irq	Select the sdio host 4 wires interrupt support 0: host not support 4 wires interrupt on single-block data transfers 1: host support 4 wires interrupt on single-block data transfers
[1]	rw	1'h0	enable_sdio_irq	Select the sdio card mode, default is sd card 0: sd card mode , no sdio card interrupt 1: sdio card mode , enable sdio card interrupt

Continued on the next page...

Table 14-1: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	ata_mode	Select the card type, default is sd card 0: sd card mode 1: CE-ATA device mode
0x54			**DSR**	data status register
[31:8]			RSVD	
[7:0]	r	8'h0	sd_data_i_ll	The status of each sd data pad status
0x58			**CDCR**	clock duty cycle register
[31:1]			RSVD	
[0]	rw	1'h1	clk_config	1: the sd clock is 50% duty cycle 0: the high level of the sd clock is 1 hclk cycle
0x5C			**CASR**	cache status register
[31:4]			RSVD	
[3]	rw1s	1'h0	cache_flush	Set 1 to flush cache. Should set when cache not busy.
[2]	r	1'h0	cache_busy	Indicates cache is working
[1]	rw1c	1'h0	sd_busy	Read 1 indicates sd is ready for normal access. Ahb access will be hold during sd_busy asserted. After sd normal access done, write 1 to clear, and ahb access will continue
[0]	rw1s	1'h0	sd_req	Set 1 to request sd normal access. sd_req will be cleared automatically after sd_busy asserted
0x60			**CACR**	cache control register
[31]	rw	1'h1	cache_en	enable cache 1: ahb read will return cached data 0: ahb read always return dummy data with no error response
[30]	rw	1'h1	cache_to_en	enable ahb read timeout recover
[29]	rw	1'h0	cache_force_read	force cache read done 1: start new fetch for miss access only after cache read done 0: start new fetch for miss access even when cache is still filling (read will be breaked by cmd12)
[28]	rw	1'h1	cache_sdsc	select card version 1: card size <=2GB, address of cmd18 is in byte 0: card size >2GB, address of cmd18 is in block
[27]	rw	1'h0	cache_nocrc	1: return ahb data without crc check 0: return ahb data after block crc pass
[26]	rw	1'h0	cache_hresp	1: generate ahb error response when error occur 0: no ahb error response generated. Could check cache_err interrupt
[25:24]			RSVD	
[23:20]	rw	4'h8	cache_pref_block	cache prefetch depth is cache_pref_block blocks. Should be no less than cache_block
[19]			RSVD	
[18:16]	rw	3'h4	cache_block	cache depth is cache_block blocks
[15]	rw	1'h0	stop_long_rsp	Stop response is 136-bit, long response
[14]	rw	1'h1	stop_has_rsp	Stop command have a response
[13:8]	rw	6'h0c	stop_index	Command index for stop. CMD12 by default
[7]	rw	1'h0	read_long_rsp	Read response is 136-bit, long response
[6]	rw	1'h1	read_has_rsp	Read command have a response
[5:0]	rw	6'h12	read_index	Command index for cache read. CMD18 by default
0x64			**CACNT**	cache counter register
[31:16]	rw	16'hffff	cache_tor	timeout count register for ahb read
[15:8]	rw	8'h0	cache_ndc	data-cmd interval counter in hclk cycles
[7:0]	rw	8'h20	cache_ncc	cmd-cmd interval counter in hclk cycles
0x68			**CAOFF**	cache offset register
[31:0]	rw	32'h0	cache_offset	offset to map ahb address to sd address for ahb access

Continued on the next page...

Table 14-1: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x200			**FIFO**	FIFO entry
[31:0]	rw	32'h0	data	Entry to access internal FIFO. Access should be word-aligned, ranging from 0x200 to 0x3fc. Inside the range, write to any address will push the data into the FIFO, and read any address will pop a word from the FIFO.

14.2 MPI

The chip features 2 MPIs. MPI1 is connected to IO(SA) for accessing the 8-line pSRAM or NOR Flash within the chip's integrated package (SiP). MPI2 is connected to IO(PA) for accessing the external NOR/NAND Flash.

The MPI (Memory Peripheral Interface)controller is a dedicated memorycommunication interface that supports various external memory components, including:

- SPI NOR Flash , supporting 1line/2lines/4lines, and DTRmode.
- SPI NAND Flash , supporting 1line/2lines/4lines.
- pSRAM , supports x8 and x16 data bit widths, complies with Xccelastandard interface, and is compatible with Legacyinterface
- HyperRAM , supports x8 and x16 data bit widths, and complies with HyperBusstandard interface

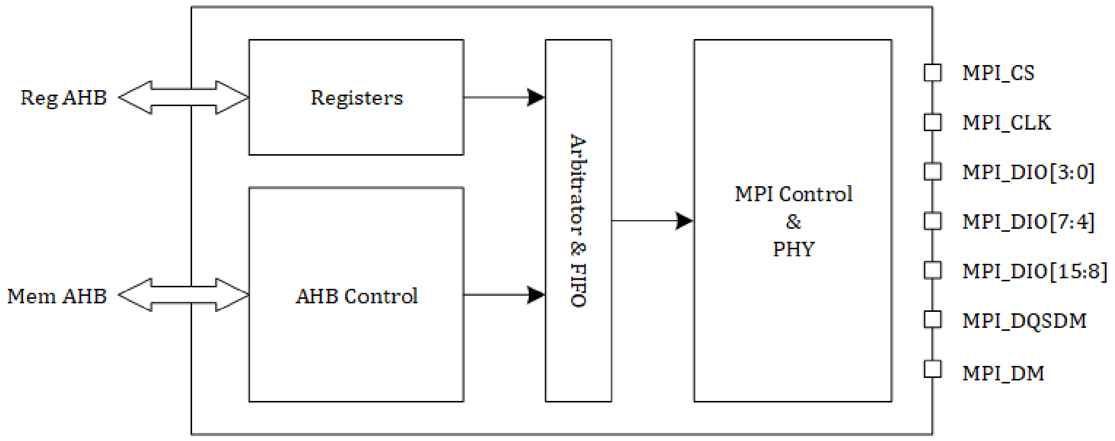


Figure 14-2: **MPI Controller Block Diagram**

The MPI controller supports two operating modes: (1) Register mode and (2) Address-mapped mode. The transition between these modes is automatically managed by hardware and can be dynamically interleaved. In both modes, highly customizable interface timing is supported to ensure compatibility with various memory chips.

Register mode

- In register operation, a command timing sequence is transmitted. This command can also be configured as a status query command that is repeatedly sent until the read-back data is complete achieve a specific preset state.
- Supports sending a sequence that includes two command timings, where the second command can be configured as a status query command that is repeatedly sent until the returned data satisfes a specific preset state
- Supports DMAchannels, facilitating data transfer through the Register FIFOinterface

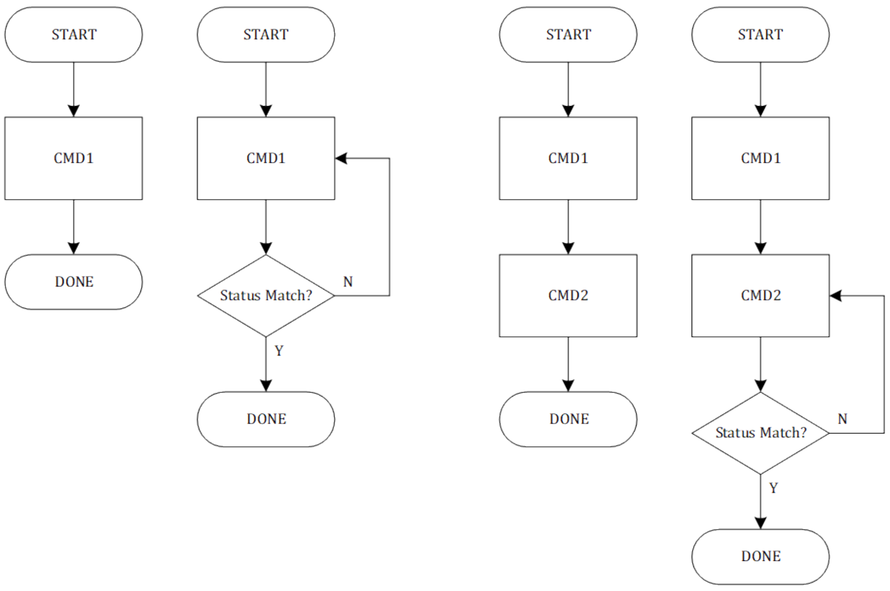


Figure 14-3: **Register mode for single and multiple command timing sequences**

Address mapping mode

- External memory is mapped to the AHB address space, automatically converting AHB bus read and write operations into preset Memory interface timings to enable XIP functionality
- SupportsByte (8-bit), Half-word (16-bit), andWord (32-bit) AHBaccess
- Effcient conversionAHB Wrapoperation, independent of whether the component supports Wrap.
- Supports XIPreal -time (On-The-Fly) decryption, with modes of AES128-CTRor AES256-CTR.
- Supports continuous read and write functionality; if the current AHB read/write address is continuous with the previous one, data transmission begins directly, omitting the command and address portions. This feature can signifcantly enhance effective bandwidth during large data transfers.
- Automatically manages the internal dynamic refresh characteristics for pSRAM and HyperRAM, handling the longest CS low time, the most recent CS access interval, and the maximum burst data length limitations without requiring software intervention.

14.2.1 MPI Register

MPI1 base address is 0x50041000。

MPI2 base address is 0x50042000。

Table 14-2: **MPI Register Mapping Table**

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31]	w1t	1’h0	ABORT	Write 1 to abort internal state machine. For debug purpose only
[30:26]			RSVD	
[25]	rw	1’h0	AHBDIS	Hold hreadyout low if AHB access
[24]	rw	1’h0	DFM	Dual Flash Mode Reserved-Do not modify
[23]	rw	1’b0	MX16	Mode X16 Reserved-Do not modify

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[22]	rw	1'b0	PREFE	Prefetch enable. If enabled, MPI will prefetch at consequtive address following a read transaction. Recommend to use when reading large data in a burst manner. 0: prefetch disabled 1: prefetch enabled
[21]	rw	1'b0	OPIE	OPI interface enable 0: x8 mode disabled 1: x8 mode enabled
[20]	rw	1'b0	HWIFE	Hardware interface enable Reserved-Do not modify
[19]	rw	1'b0	SMM	Status match mode 0: AND mode 1: OR mode
[18]	rw	1'b0	SME2	Status match enable. If enabled, CMD2 will be issued repeatedly until the data match the value in SMR and SMKR 0: disabled 1: enabled
[17]	rw	1'b0	SME1	Status match enable. If enabled, CMD1 will be issued repeatedly until the data match the value in SMR and SMKR 0: disabled 1: enabled (either SME1 or SME2 can be enabled, and SME1 has high priority)
[16]	rw	1'b0	CMD2E	Enable CMD2 0: disabled 1: CMD2 is enabled and will be issued after CMD1 with an interval of TI2
[15:14]			RSVD	
[13]	rw	1'h0	RBXIE	Row boundary crossing interrupt enable
[12]	rw	1'h0	CSVIE	CS max violation interrupt enable
[11]	rw	1'h0	SMIE	Status match interrupt enable
[10]			RSVD	
[9]			RSVD	
[8]	rw	1'h0	TCIE	Transfer complete interrupt enable
[7]	rw	1'h0	CTRM	AES-CTR mode 0: AES-128 1: AES-256
[6]	rw	1'h0	CTRE	AES-CTR on-the-fly decryption enable 0: disabled 1: enabled, data read from memory will be decrypted on the fly by MPI controller
[5]	rw	1'h0	DMAE	DMA enable 0: disabled 1: enable DMA to read or write DR register
[4]	rw	1'h0	HOLD	The value of HOLD when HOLDE is set
[3]	rw	1'h0	HOLDE	Enable HOLD function on IO3. Use this only in SPI or Dual SPI mode
[2]	rw	1'h0	WP	The value of WP when WPE is set
[1]	rw	1'h0	WPE	Enable WP function on IO2. Use this only in SPI or Dual SPI mode
[0]	rw	1'h0	EN	Enable MPI
0x04			**DR**	Data Register
[31:0]	rw	32'h0	DATA	The entry of internal data FIFO
0x08			**DCR**	Device Control Register
[31]	rw	1'h0	FIXLAT	Indicate PSRAM is fixed latency or variable latency. It must be compatible to the configuration in PSRAM registers. Recommend always set to 1. 0: variable latency 1: fixed latency

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[30:26]	rw	5'h0	TRCMIN	Write/Read cycle minimum time in internal MCLK cycles. Please see MCLK frequency in PSCLR description. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and TRCMIN = n, then tRC time = (n+1) * 1000/240 ns which must meet minimum tRC requirement for PSRAM
[25:22]	rw	4'h2	CSHMIN	Minimum CS high deselect time in MCLK cycles. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSHMIN = n, then CS High time = (n+1) * 1000/240 ns which must meet minimum tCPH requirement for PSRAM
[21:18]	rw	4'h0	CSLMIN	Minimum CS low active time in MCLK cycles. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSLMIN = n, then CS Low time = (n+1) * 1000/240 ns which must meet the minimum tCEM requirement for PSRAM
[17:6]	rw	12'h0	CSLMAX	Maximum CS low active time in MCLK cycles For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSLMAX = n, then CS Low time = (n+1) * 1000/240 ns which must meet the maximum tCEM requirement for PSRAM
[5]	rw	1'h0	XLEGACY	Xccela legacy protocol. Set to 1 for AP 32Mb PSRAM only, othersize always set to 0.
[4]	rw	1'h0	HYPER	HyperBus protocol. Set to 1 for HyperRAM.
[3]	rw	1'h0	DQSE	DQS enable. Setting to 1 indicates device provides DQS signal for Rx data latching
[2:0]	rw	3'h0	RBSIZE	Row boundary size. 0: no row boundary 1: $2^{(1+3)}$ = 16 bytes 2: $2^{(2+3)}$ = 32 bytes ... n: $2^{(n+3)}$ bytes
0x0C			**PSCLR**	Prescaler Register
[31:8]			RSVD	
[7:0]	rw	8'h4	DIV	Prescaler divider. 0: MCLK = FCLK/1 1: MCLK = FCLK/1 2: MCLK = FCLK/2 n: MCLK = FCLK/n Note: FLASH clock = MCLK. E.g. FCLK=192M and DIV=2, then FLASH clock = MCLK = 192/2 = 96MHz PSRAM clock = MCLK/2. E.g. FCLK=240M and DIV=1, then PSRAM clock = MCLK/2 = 240/2 = 120MHz
0x10			**SR**	Status Register
[31]	r	1'h0	BUSY	For debug purpose only
[30:6]			RSVD	
[5]	r	1'h0	RBXF	Row boundary crossing flag
[4]	r	1'h0	CSVF	CS max violation flag
[3]	r	1'h0	SMF	Status match flag in Polling Mode
[2]			RSVD	
[1]			RSVD	
[0]	r	1'h0	TCF	Transfer complete flag
0x14			**SCR**	Status Clear Register
[31:6]			RSVD	

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	w1c	1'h0	RBXFC	Write 1 to clear RBXF
[4]	w1c	1'h0	CSVFC	Write 1 to clear CSVF
[3]	w1c	1'h0	SMFC	Write 1 to clear SMF
[2]			RSVD	
[1]			RSVD	
[0]	w1c	1'h0	TCFC	Write 1 to clear TCF
0x18			**CMDR1**	Command Register
[31:8]			RSVD	
[7:0]	rw	8'h0	CMD	Command. Write to this register will trigger the sequence specified in CCR1
0x1C			**AR1**	Address Register
[31:0]	rw	32'h0	ADDR	Address
0x20			**ABR1**	Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	Alternate byte
0x24			**DLR1**	Data Length Register
[31:20]			RSVD	
[19:0]	rw	20'h0	DLEN	Data length 0: one byte 1: two bytes ... n: (n+1) bytes
0x28			**CCR1**	Communication Configuration Register
[31:22]			RSVD	
[21]	rw	1'b0	FMODE	Function Mode 0: read mode 1: write mode
[20:18]	rw	3'h0	DMODE	Data Mode 0: no data phase 1: single line 2: dual lines 3: quad lines 4/5/6: reserved 7: quad lines DDR
[17:13]	rw	5'h0	DCYC	Number of dummy cycles 0: no dummy cycle 1: one dummy cycle 2: two dummy cycles
[12:11]	rw	2'h0	ABSIZE	Alternate byte size 0: one byte 1: two bytes 2: three bytes 3: four bytes
[10:8]	rw	3'h0	ABMODE	Alternate byte mode 0: no alternate byte 1: single line 2: dual lines 3: quad lines 4/5/6: reserved 7: quad lines DDR
[7:6]	rw	2'h0	ADSIZE	Address size 0: one byte 1: two bytes 2: three bytes 3: four bytes

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:3]	rw	3'h0	ADMODE	Address mode 0: no address phase 1: single line 2: dual line 3: quad line 4/5/6: reserved 7: quad line DDR
[2:0]	rw	3'h0	IMODE	Instruction mode 0: no instruction phase 1: single line 2: dual lines 3: quad lines 4/5/6 - reserved 7 - quad lines DDR
0x2C			**CMDR2**	Command Register
[31:8]			RSVD	
[7:0]	rw	8'h0	CMD	Command 2. If CMD2E is enabled, the CMD2 sequence will be issued after CMD1 as specified in CCR2 Note: CMD2 sequence cannot be issue individually
0x30			**AR2**	Address Register
[31:0]	rw	32'h0	ADDR	Address byte in CMD2 sequence
0x34			**ABR2**	Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	Alternate byte in CMD2 sequence
0x38			**DLR2**	Data Length Register
[31:20]			RSVD	
[19:0]	rw	20'h0	DLEN	Data length in CMD2 sequence
0x3C			**CCR2**	Communication Configuration Register
[31:22]			RSVD	
[21]	rw	1'b0	FMODE	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	
[2:0]	rw	3'h0	IMODE	This register specifies the format of CMD2 sequence. Refer to CCR1 description
0x40			**HCMDR**	AHB Command Register
[31:16]			RSVD	
[15:8]	rw	8'h02	WCMD	AHB write command. During XIP, the AHB write transaction will be translated into this Write Command on memory interface
[7:0]	rw	8'h0b	RCMD	AHB read command. During XIP, the AHB read transaction will be translated into this Read Command on memory interface
0x44			**HRABR**	AHB Read Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	
0x48			**HRCCR**	AHB Read Communication Configuration Register
[31:21]			RSVD	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2:0]	rw	3'h0	IMODE	This register specifies the format of AHB read command sequence. Refer to CCR1 description
0x4C			**HWABR**	AHB Write Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	
0x50			**HWCCR**	AHB Write Communication Configuration Register
[31:21]			RSVD	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	
[2:0]	rw	3'h0	IMODE	This register specifies the format of AHB write command sequence. Refer to CCR1 description
0x54			**FIFOCR**	FIFO Control Register
[31:15]			RSVD	
[14:10]	rw	5'h8	TXSLOTS	When DMA enabled, asserts DMA reqeust if TXFIFO vacant slots is greater than or equal to TXSLOTS. Note: this field should be set in accordance to the burst length in DMA. For example, if DMA employs BURST8 transction, then this filed is set to 8
[9]	r	1'h0	TXF	Tx FIFO full flag
[8]	w1c	1'h0	TXCLR	write 1 to clear Tx FIFO
[7:2]			RSVD	
[1]	r	1'h1	RXE	Rx FIFO empty
[0]	w1c	1'h0	RXCLR	write 1 to clear Rx FIFO
0x58			**MISCR**	Miscelaneous Register
[31:28]	rw	4'h0	DBGSEL	
[27]			RSVD	
[26]	rw	1'h0	DTRPRE	Enable pre-sampling for DTR Reserved-Do not modify
[25]	rw	1'h1	SCKINV	Invert output clock. This bit is used to align (coarse tune) the output clock to the center of output data.
[24]	rw	1'h0	RXCLKINV	Invert internal Rx clock to add half-cycle delay (coarse tune) when sampling data. It is usually used for FLASH device w/ higher frequency.
[23:16]	rw	8'h0	DQSDLY	Delay the input DQS signal to the appropriate sampling position. For device w/ DQS signal only. Note: effective 7-bit
[15:8]	rw	8'h0	SCKDLY	Add delay on output clock to fine tune the clock position. Note: effective 7-bit
[7:0]	rw	8'h0	RXCLKDLY	Add delay on internal Rx clock to fine tune the sampling position. Note: effective 5-bit
0x5C			**CTRSAR**	CTR Starting Address Register
[31:10]	rw	22'h0	SA	Starting address of the AES decryption area. Since the lowest 10 bits are zero, the address is always 1KB aligned. Together with CTREAR, the total area is [CTRSAR, CTREAR) For example, CTRSAR = 32'h0, CTREAR = 32'h200000, then the on-the-fly decryption area is 0x0 - 0x1FFFFF
[9:0]			RSVD	
0x60			**CTREAR**	CTR Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the AES decryption area
[9:0]			RSVD	
0x64			**NONCEA**	Nonce A Register
[31:0]	rw	32'h0	NONCEA	Used for on-the-fly decryption
0x68			**NONCEB**	Nonce B Register
[31:0]	rw	32'h0	NONCEB	Used for on-the-fly decryption

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x6C			**AASAR**	Address Aliasing Start Address Register
[31:10]	rw	22'h0	SA	Starting address of the address aliasing area. Always 1KB aligned.Together with AAEAR, the aliasing area is [AASAR, AAEAR). If the address falls into this area, an offset AAOAR is added and the aliased address will be used to access external memory
[9:0]			RSVD	
0x70			**AAEAR**	Address Aliasing Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the address aliasing area
[9:0]			RSVD	
0x74			**AAOAR**	Address Aliasing Offset Address Register
[31:10]	rw	22'h0	OA	The offset to be added to the original address
[9:0]			RSVD	
0x78			**CIR**	Command Interval Register
[31:16]	rw	16'h0	INTERVAL2	The interval between CMD1 and CMD2 (or between CMD2 itself) if CMD2E is enabled. The unit is in MCLK cycles
[15:0]	rw	16'h0	INTERVAL1	The interval between CMD1 itself. The unit is in MCLK cycles
0x7C			**SMR**	Status Match Register
[31:0]	rw	32'h0	STATUS	If status match is enabled, this register is compared with the data read from external memory. Together with SMKR, only the bits with mask=1 will be considered to compare in AND or OR mode as configured in SMM field.
0x80			**SMKR**	Status Mask Register
[31:0]	rw	32'h0	MASK	Status mask 0: the corresponding bit is not considered to compare 1: the corresponding bit is considered to compare
0x84			**TIMR**	Timer Register
[31:16]			RSVD	
[15:0]	rw	16'h0	TIMEOUT	After the transaction is complete, CS remains low for multiple cycles of MCLK as specified by this register. For example if TIMEOUT=n, CS remains active for n cycles, during which if a new transaction occurs and the address is consecutive, the memory access can be resumed w/o sending the command and address again.
0x88			**WDTR**	WDT Register
[31]	r	1'h0	TOF	Timeout flag. Self cleared when HREADYOUT becomes ready
[30:17]			RSVD	
[16]	rw	1'b0	EN	WDT enable. This watchdog is on AHB side such that bus access will not hang in exceptional cases
[15:0]	rw	16'hffff	TIMEOUT	Set timeout value in number of clk_wdt cycles
0x8C			**PRSAR**	Prefetch Starting Address Register
[31:10]	rw	22'h0	SA	Starting address of the prefetch area If prefetch is enabled and the read address falls into [PRSAR, PREAR), controller will prefetch the following data
[9:0]			RSVD	
0x90			**PREAR**	Prefetch Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the prefetch area
[9:0]			RSVD	
0x94			**CALCR**	Calibration Clock Register
[31]	rw	1'h0	EN	calibration enable
[30:9]			RSVD	
[8]	r	1'h0	DONE	calibration done flag
[7:0]	r	8'h0	DELAY	calibration delay result
0x9C			**APM32CR**	APM32 Control Register

Continued on the next page...

Table 14-2: MPI Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:8]			RSVD	
[7:4]	rw	4’h4	TCPHW	For special use by AP 32Mb PSRAM. Reserved-Do not modify
[3:0]	rw	4’h2	TCPHR	For special use by AP 32Mb PSRAM. Reserved-Do not modify
0xA0			**CR2**	Control Register 2
[31:8]			RSVD	
[7:0]	rw	8’h0	LOOP	Repeat CMD1->CMD2 sequence for n times. This filed is only valid when CMD2E=1 and SME2=0. For example if LOOP=0, then the sequence is CMD1 -> CMD2. If LOOP=2, then the sequence is (CMD1->CMD2) -> (CMD1->CMD2) -> (CMD1->CMD2)

15 Debug Interface

15.1 Introduction

The chip integrates a UART debug interface within the USART1 module, providing users with a convenient and effcient debugging method. Users can perform CPU debugging, including pausing, obtaining status, and stepping through operations, simply by using UART communication; they can also access bus addresses to read from and write to memory and registers. While supporting UART debugging functionality, USART1 can still perform standard UART transmission and reception (requires host computer and driver support). Users can permanently disable the UART debug interface by programming the EFUSE, without affecting the normal functionality of USART1.

15.2 Main Features

- Standard UARTinterface , requiring only connections to TX, RX,and GND.
- Custom Debug Frame.
- The baud rate can be configured after connection (default baud rate 1M).
- Compatible with various debugging functions of the CPU.
- Supports access to various memory and registers via bus address.
- Supports block data transfer to enhance access effciency.
- CPU exceptions do not affect bus address access.
- Can coexist with standard UARTfunctionality.
- Can be permanently disabled via EFUSE.

15.3 Debugging Method

The host computer sends a request frame to the receiving port of USART1 (PA18) and receives a response frame from the transmitting port of USART1 (PA19). Both the request frame and the response frame are transmitted according to the UART protocol and conform to the custom debug frame format.

There are a total of 4formats for the request frame, representing entering debug mode, exiting debug mode, reading data, and writing data.

Each request frame corresponds to a response frame, indicating that debug mode has been entered, debug mode has been exited, data reading is complete, and data writing is complete.

After the host computer sends the request frame, it should wait for the response frame to return before sending the next request frame. If the response frame has not been received within a specifed time period (such as 100ms) , it can be considered a request timeout, and the request frame may be resent

The host computer must first send the request frame to enter debug mode and can only begin data reading and writing after receiving the response frame indicating that debug mode has been entered.

When reading data, a read request frame containing the starting address and data length must be issued, and the chip will return a read completion response frame. The data content is included within the read completion response frame.

When writing data, a write request frame containing the starting address, data length, and data content must be issued, and the chip will return a write completion response frame.

After debugging is complete, a request frame to exit debug mode can be issued, and the response frame indicating that debug mode has been exited should be awaited.

15.4 Custom Debug Frame

The custom debug frame is used to differentiate debug data from normal transmission data.

The format of the custom debug frame is as follows, primarily consisting of a frame header Header and frame data Payload , transmitted in order from left to right, from low byte to high byte. The length Length is a total of 2 bytes, indicating the number of bytes in the frame data Payload . The length of the frame data Payload is variable, with transmission commencing from the lowest byte.

Table 15-1: Custom Debug Frame Format

Header(6 bytes)						Payload(Length bytes)			
0x7E	0x79	Length (Lower byte)	Length (higher byte)	0x10	0x00	Byte0	Byte1	···	Byte Length-1

Frame Data Payloadcan further differentiate between various types of frames. During transmission, adhere to the order from left to right, from low byte to high byte.

1. Enter Debug Mode (Request Frame)

Payload (8 bytes)							
0x41	0x54	0x53	0x46	0x33	0x32	0x05	0x21

2. Successfully Entered Debug Mode (Response Frame)

Payload (2 bytes)	
0xD1	0x06

3. Exit Debug Mode (Request Frame)

Payload (8 bytes)							
0x41	0x54	0x53	0x46	0x33	0x32	0x18	0x21

4. Successfully Exited Debug Mode (Response Frame)

Payload (2 bytes)	
0xD0	0x06

5. Read Data (Request Frame)

   Addr indicates the starting address for reading data, which must be 4bytes aligned.

WordCnt indicates the number ofwordsto be read. The total number of bytes read is 4*WordCnt.

Payload (8 bytes)			
0x40	0x72	Addr(4 bytes)	WordCnt(2 bytes)

6. Read Data Complete (Response Frame)

Payload (2+4*WordCnt bytes)		
0xD2	Data(4*WordCnt bytes)	0x06

7. Write Data (Request Frame)
   Addr indicates the starting address for writing data, which must be 4bytes aligned.
   WordCnt indicates the number ofwordsto be written. The total number of bytes written is 4*WordCnt.

Payload (8+4*WordCnt bytes)				
0x40	0x77	Addr(4 bytes)	WordCnt(2 bytes)	Data(4*WordCnt bytes)

8. Data writing completed (Response frame)

Payload (2 bytes)	
0xD3	0x06

15.5 Debugging Example

The following is an example of a complete data read and write debugging process, detailing the specific data transmitted sequentially via UART during the debugging process. Downstream represents the request frame sent from the host computer to the chip, while upstream represents the response frame returned from the chip to the host computer. Assume that data 0x12345678 is already present at address 0x20000000

Downstream: 7E 79 08 00 10 00 41 54 53 46 33 32 05 21 (Entering debug mode)

Upstream: 7E 79 02 00 10 00 D1 06 (Successfully entered debug mode)

Downstream: 7E 79 0C 00 10 00 40 77 04 00 00 20 01 00 DD CC BB AA (Write data, 0xAABBCCDDto0x20000004)

Upstream: 7E 79 02 00 10 00 D3 06 (Write data complete)

Downstream: 7E 79 08 00 10 00 40 72 00 00 00 20 02 00 (Read data , starting from 0x20000000 for 2 words)

Upstream: 7E 79 0A 00 10 00 D2 78 56 34 12 DD CC BB AA 06 (Read data complete)

Downstream: 7E 79 08 00 10 00 41 54 53 46 33 32 18 21 (Exit debug mode)

Upstream: 7E 79 02 00 10 00 D0 06 (Exited debug mode)

15.6 Address Mapping

The UART Debug Interface is based on bus address read and write operations, but the address mapping differs slightly from that of HCPU. The UART Debug Interface also includes several dedicated registers. Please refer to the table below.

Table 15-2: UART Debug Interface Address Mapping

	Debug Interface Address Space	HCPU Address Space
HPSYS_ROM	Starting from 0xA0000000	Starting from 0x0
HCPU System Registers	Starting from 0xF0000000	Starting from 0xE0000000
MPI	Starting from 0x60000000	0x10000000 or Starting from 0x60000000
Other Memory	Identical	
Other Module Registers	Identical	
(Dedicated) Boot Mode	0xFFA57200	/
(Dedicated) HCPU System Reset	0xFFA57201	/
(Dedicated)Chip Reset	0xFFA57203	/

15.7 HCPU Debugging

Debugging the HCPU is accomplished through read and write access to the HCPU system registers. For specific methods of register operation, please refer to the relevant documentation from ARM Corporation.

The UART Debug Interface can access the HCPU system registers through the address space 0xF0000000 0xFFFFFFFF to implement debugging functions for the HCPU, including but not limited to pausing, obtaining status, single stepping, and system reset operations.

15.8 USART1 Behavior

The USART1 module is enabled by default after the chip starts and continuously monitors the custom debug frame sequence at the receiving end (PA18). When a valid custom debug frame is detected, it automatically executes the corresponding operation and returns a response through the transmitting end (PA19). The USART1 module defaults to a 1M baud rate, 8 data bits, 1 stop bit, and no parity.

During debugging, USART1 automatically invokes DMAC1 ’s dedicated channel for bus data read and write, which does not affect the functionality of DMAC1’s other channels but will consume a certain amount of bus bandwidth.

15.9 Debug Interface Failure

The debug interface may fail to connect or read/write successfully under the following circumstances:：

1. Incorrect baud rate. The default baud rate after the chip is powered on is 1M ; however, if the frmware modifes the baud rate of USART1 , it is necessary to connect the debug interface according to the modifed baud rate.
2. The port of USART1has been mapped to other IO.
3. The clock for DMAC1 has been disabled or reset.
4. DMAC1 Bus hang. If MPI is uninitialized, accessing the memory space of this MPI by DMAC1 or UART Debug Interface (e.g., Nor Flash or PSRAM)may potentially cause a bus hang.
5. HPSYS Enters low power mode. When HPSYS is in deepsleep or standby mode, or the chip is in hibernate mode, the UART Debug Interface will not function.
6. The Debug Interface is disabled by the EFUSE control bit.

15.10 Coexistence of debug data and normal data

While using the Debug Interface,USART1 can still transmit and receive general UARTdata, but the following rules must be adhered to in order to avoid conflicts.

1. CPU Before sending data using USART1, it is essential to read the data lock register EXR_BUSY. If the read value is 1, it indicates that it is locked (Debug Interface is in operation), and you must wait for a while before attempting again. If the read value is 0, the data lock will automatically be set to 1, at which point you can begin normal data transmission. After the data transmission is complete, you need to write 1 to EXR_BUSY to unlock.
2. The UART Debug Interface will also automatically check EXR_BUSY before sending debug data. If EXR_BUSY is 1, it will wait until the CPU unlocks before starting the transmission; otherwise, it will send the data directly and automatically lock until the transmission is complete, at which point it will automatically unlock.
3. USART1 All data received can be read by the CPU, including request frames and general data. The frmware must differentiate between different data types using the format of custom debug frames.