SF32LB55x

User Manual

V0.3.4

UM5501-SF32LB55x-EN

SiFli Technologies (Nanjing) Co., Ltd.

http://www.sifli.com

Revision History

Document Status Description

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.

Document Revision History

Date	Version	Release Notes
2025-01-13	0.3.4	Added description of the RTC module
2025-07-13	0.3.3	Added Bluetooth chapt
2025-04-08	0.3.2	Added description of the BUSMON module
2025-03-10	0.3.1	Corrected the description of the data bit width supported by SPI
2025-01-10	0.3	Added description of the package IO and overview of each IP
2024-12-20	0.2	Updated modules such as LCDC and EFUSE
2024-11-28	0.1	Initial Draft

Product Overview

SF32LB55x is a family of highly integrated high-performance System-on-Chip (SoC) MCUs designed for ultra-low-power Artifical Intelligence of Things (AIoT) scenarios. SF32LB55x adopts the big.LITTLE architecture with Arm Cortex-M33 STAR-MC1 processors, combining the best-in-class 2.5D GPU, neural network accelerator, and Bluetooth Low Energy 5.2. SF32LB55x can be used in a wide variety of applications such as wearables, smart display terminals, and smart homes.

The high-performance processor ("big core") of SF32LB55x can operate at up to 240MHz, providing 360DMIPS of computing power, and 965 EEMBC CoreMark; the low-power processor ("LITTLE core") can operate at up to 48MHz, serving as low-power sensor hub and running the Bluetooth protocol stack at the same time. The big.LITTLE architecture brings the balance between compute performance required for Human-Machine Interface (HMI) and power efficiency for sensor operations.

SF32LB55x integrates a world-class BLE5.2 transceiver, supporting 125kbps/500kbps/1Mbps/2Mbps modes. The maximum transmit power is 10dBm, the receiver has a peak power consumption of 2mA@3.3V, and sensitivity of -100dBm (1Mbps), with the link budget reaching 110dB.

SF32LB55x has rich internal and external memories. The high-performance processor has up to 1152KB SRAM, while the low-power processor has 256KB SRAM and 384KB ROM. The fully packaged chip has four QSPI memory interfaces, dedicated OPI-PSRAM interface, and SD/eMMC interfaces, which can be accessed by the big or LITTLE core processor depending on the requirements of task.

SF32LB55x provides a full range of display interfaces, including 8080, SPI/Dual-SPI/Quad-SPI, DPI, JDI interfaces, and MIPI-DSI (with extra QSPI for Always On Display), supporting 1024×1024, aRGB8888, and up to 60fps.

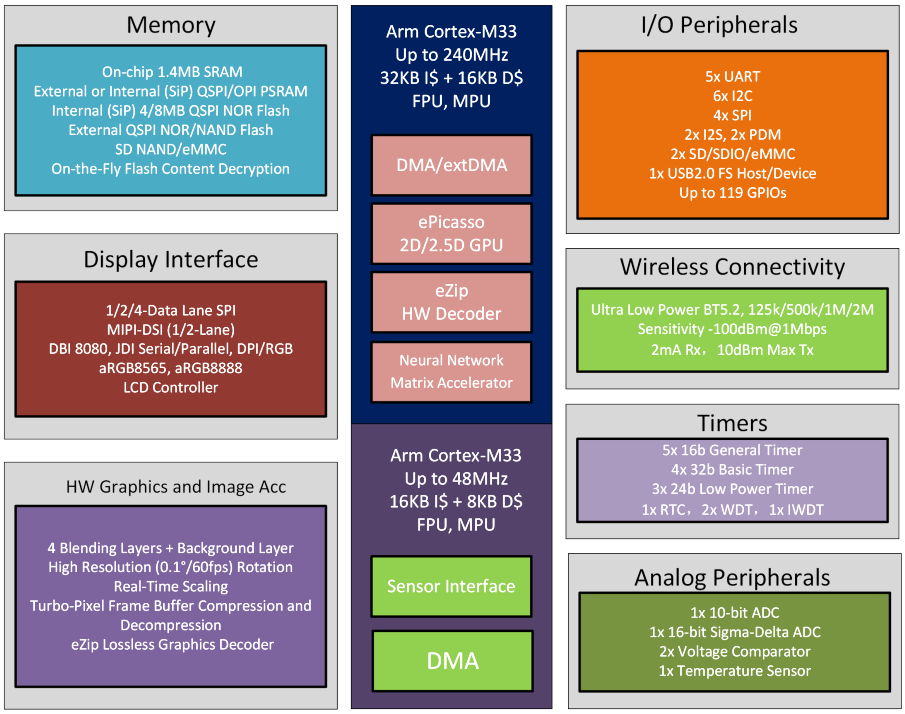


Figure 0-1: Functional Block Diagram

Product Features

CPU and Memory

- High-Performance Processor/ Big Core (HCPU)
  - Processor: Arm Cortex-M33 STAR-MC1
  - CPU Clock Speed: up to 240MHz, adjustable
  - Max. 360DMIPS, 965 EEMBC CoreMark
  - I/D-TCM: 64KB (dedicated)+128KB (shared with SRAM)
  - I/D-Cache: 32KB (2-way)+16KB (4-way)
  - SRAM: 1088KB (64 KB retention SRAM)
  - ROM: 64KB
  - CoreMark Power (@3.3V):
    - 34uA/MHz
    - 8.46uA/CoreMark
    - 118CoreMark/mA
  - Single Precision Floating Point Unit (FPU)
  - Memory Protection Unit (MPU)
  - Neural Network Co-Processor
- Low-Power Processor/ LITTLE Core (LCPU)
  - Processor: Arm Cortex-M33 STAR-MC1
  - CPU Clock Speed: up to 48MHz, adjustable
  - Max. 72DMIPS, 193 EEMBC CoreMark
  - I/D-TCM: 16KB (dedicated)+16KB (dedicated)
  - I/D-Cache: 16KB (2-way)+8KB (4-way)
  - SRAM: 224KB (all retention SRAM)
  - ROM: 384KB
  - CoreMark Power (@3.3V):
    - 11.8uA/MHz
    - 2.93uA/CoreMark
    - 341CoreMark/mA
  - Single Precision Floating Point Unit (FPU)
  - Memory Protection Unit (MPU)
  - Neural Network Co-Processor

Wireless Connectivity

- Bluetooth Low Energy 5.2: 125Kbps, 500Kbps, 1Mbps, 2Mbps
- Sensitivity at 1Mbps mode: -100dBm
- Max. transmit power: 10dBm
- Single antenna output, on-chip Balun, no external matching needed
- Rx peak power consumption: 2.0mA@3.3V
- Tx power consumption @0dBm: 2.6mA@3.3V
- Average power (3.3V, 200ms interval, Tx@0dBm):
  - ADV mode: 43.0uA
  - Connected mode: 30.1uA

2.5D GPU – ePicasso™

- 4-layer alpha blending +1 background layer architecture: 1 transform layer, 3 generic layers, 1 background layer
- Transform layer: Max. resolution 640×640, hardware-accelerated rotation, scaling and mirroring
- Generic layer: Max. resolution 1024×1024
- Support concatenated operation with eZip™, including transform layer operations
- Industry leading blending throughput performance: up to 1.5 cycle/pixel
- Support RGB565, RGB888, aRGB565, aRGB888, and alpha blending

Lossless Decompression Accelerator – eZip™

- Hardware decoding of proprietary lossless graphics compression format, compression ratio comparable to PNG
- Support generic data decompression, compression ratio comparable to gzip
- Independent DMA mode, for memory-to-memory image decompression
- Concatenated operation mode with ePicasso™, no need to buffer decompressed image in memory to complete ePicasso™operations

LCD Controller in High-Performance Domain

- MIPI-DBI: 8080, SPI, Dual-SPI, Quad-SPI
- Support MIPI-DPI
- MIPI-DSI: up to two data lanes, each supporting up to 240MHz/480Mbps
- JDI serial and parallel reflective display interfaces
- Support two-layer alpha blending, and solid color background layer
- TurboPixel™frame buffer decompression: hardware compression by extDMA, and decompression by the LCD controller, to enable high frame rate output with limited bandwidth

- Support RGB565, RGB888, aRGB565, aRGB888, and alpha blending

LCD Controller in Low-Power Domain

- MIPI-DBI: SPI/Dual-SPI/Quad-SPI for Always On Display (AOD)
- Support two-layer alpha blending, and solid color background layer
- Support RGB565, RGB888, aRGB565, aRGB888, and alpha blending
- Works with low-power processor/ LITTLE core, no high-performance processor/big core required

Neural Network Matrix Accelerator

- Highly efficient matrix convolution operation for TinyML scenarios and seamless API interface with Arm CMSIS-NN for easy porting of different neural network framework
- Independent bus interface and fully pipelined computing to maximize data throughput
- Max. processing power: up to 1.92GOPS
- Power consumption efficiency: up to 5.73TOPS/W

DMA

- General purpose DMA: support high-efficiency data transfer between internal memory and peripherals
- ExtDMA: for high-efficiency data transfer between internal memory and external memory, support TurboPixel™frame buffer compression, coupled with the decompression function of the LCD controller

Security

- AES accelerator
  - Symmetric encryption and decryption, support AES128, AES192, AES256 and SM4
  - Encryption and decryption modes including ECB, CTR and CBC
  - AES processes the source data in DMA mode, and writes data back to destination address
  - Support external Key, as well as Root Key
- CRC: Support fully programmable 7/8/16/32-bit generation polynomial, support 8/16/32-bit input data width, and can be driven by CPU or DMA for the CRC calculation
- True Random Number Generator (TRNG)
- Secure Boot
- 1024-bit eFuse to store Root of Trust and UID
- PSA Certified Level 1

Memory Interfaces

- 4×QSPI
  - Support QSPI-NOR Flash, QSPI-NAND Flash, or QSPI-PSRAM, Max. frequency 96MHz
  - QSPI1 is specially used for SiP flash
  - QSPI1/QSPI2 can be extended to dual QSPI-NOR Flash interface, to achieve 8-data lane throughput performance
  - QSPI1/QSPI2/QSPI4 support on-the-fly decryption (AES128-CTR or AES256-CTR) when fetching code from NOR Flash
  - QSPI4 is specially used for resouce extension of low-power processor/ LITTLE core, support NOR Flash or QSPI-PSRAM in specific application scenarios
- 1×OPI-PSRAM interface: support DDR, can be configured to 8-bit PSRAM interface, or be expanded to a set of 16-bit PSRAM interface by two sets of 8-bit PSRAM
- 2×SD/SDIO/eMMC (1×8b and 1×4b), support SD3.0, SDIO3.0, and eMMC4.51

Timers

- 5×16b General Purpose Timer (GPTIM)
- 4×32b Basic Timer (BTIM)
- 3×24b Low-Power Timer (LPTIM)
- 1×Real Time Clock (RTC)
- 2×24b Watch Dog Timer (WDT)
- 1×Independent Watch Dog Timer (IWDT)

Analog Peripherals

- 1×10-bit general purpose SAR ADC, up to 8 channels
- 1×16-bit Sigma-Delta ADC, up to 5 channels
- 1×On-chip temperature sensor
- 2×Low-power voltage comparator

I/O Peripherals

- Up to 119GPIOs
- 5×UART
- 6× I^2C
- 4×SPI
- 2× I^2S

- 2× PDM
- 1×USB2.0 FS Host/Device
- Peripheral Task Controller (PTC)

Power Management

- Input voltage: 1.7V – 3.6V
- Two on-chip high-efficiency bucks and low-power LDO
- Sleep current at RTC wake-up configuration: 600nA
- Sleep current at pin wake-up configuration: 280nA

Others

- Operation temperature: -40°C to 85°C
- Multiple package options available, providing rich GPIOs and interfaces for high-performance (HCPU) domain and low-power (LCPU) domain
  - BGA169, 119 (HCPU71+LCPU48) GPIOs, 7mm × 7mm × 0.94mm, 0.5mm pitch
  - BGA145, 95 (HCPU55+LCPU40) GPIOs, 7mm × 7mm × 0.94mm, 0.5mm pitch
  - BGA125, 84 (HCPU52+LCPU32) GPIOs, 7mm × 7mm × 0.94mm, 0.5mm pitch
  - QFN68L, 49 (HCPU28+LCPU21) GPIOs, 7mm × 7mm × 0.75mm, 0.35mm pitch

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

Home Automation

- Smart lighting
- Smart home appliance
- Smart door lock
- Voice and gesture remote
- Stylus pen for pad, tablet PC, and smartphone

Generic Scenario

- Low-power sensor hub
- Bluetooth mesh

Product Series

Table 0-1: SF32LB55x Product Series

Function		SF32LB551	SF32LB553	SF32LB555	SF32LB557
Package	Type	QFN68L	BGA125	BGA145	BGA169
	Dimension (mm×mm×mm)	7×7×0.75	7×7×0.94	7×7×0.94	7×7×0.94
	Pitch (mm)	0.35	0.5	0.5	0.5
High-Performance Processor(HCPU)	Processor	Cortex-M33 STAR-MC1@240MHz			
	Cache	32KB-I$, 16KB-D$			
	TCM	64KB ITCM + 128KB DTCM[1]			
	SRAM	1088KB (Including 64KB Retention SRAM)			
	ROM	64KB			
	SiP PSRAM	4MB/OPI	4MB/OPI	4/8MB/OPI	16MB/OPI + 2MB/QPI
	SiP Flash	4MB QSPI	4MB QSPI	4MB QSPI	1MB QSPI
	Off-chip Flash	QSPI	QSPI	QSPI	QSPI
	SDIO/eMMC	-	-	SD/eMMC ×1	SD/eMMC ×2
Low-Power Processor(LCPU)	Processor	Cortex-M33 STAR-MC1@48MHz			
	Cache	16KB-I$, 8KB-D$			
	TCM	16KB ITCM + 16KB DTCM			
	RAM	224KB （All Retention SRAM）			
	ROM	384KB			
	SiP PSRAM	-	-	-	2MB QPI
	Off-chip Flash	-	-	QSPI (shared)	-
Graphics	Graphics acceleration	ePicasso™high performance 2.5D GPU @240MHz			
	Lossless compression	eZip™decompression @240MHz			
	Display interface	SPI/DSPI/QSPI/8080		SPI/DSPI/QSPI/8080/MIPI-DSI/DPI/JDI	
	Always On Display	-		QSPI (shared)	DSPI (dedicated)
Power Management	Buck1	Y			
	Buck2	-	Y		
Bluetooth Low Energy	Protocol	BT5.2			
	Sensivitity	-100dBm			
	Rx power @3.3V	2.0mA			
	Max. Tx power	10dBm			
Timers (HCPU+LCPU)	16b GPTIM	2+3	2+3	2+3	2+3
	32b BTIM	2+2	2+2	2+2	2+2
	24b LPTIM	1+2	1+2	1+2	1+2
	RTC	1	1	1	1
	24b WDT	1+1	1+1	1+1	1+1
	IWDT	1	1	1	1
Analog Peripherals (HCPU+LCPU)	SAR ADC (8-ch)	0+1	0+1	0+1	0+1
	Sigma-Delta ADC (5-ch)	0+1	0+1	0+1	0+1
	Temperature Sensor	0+1	0+1	0+1	0+1
	LP COMP	0+2	0+2	0+2	0+2
I/O Peripherals (HCPU+LCPU)	GPIO	28+21	52+32	55+40	71+42
	UART	1+1	1+2	2+3	2+3
	I2C	1+2	3+2	3+2	3+3
	SPI	1+1	1+2	2+2	2+2
	I2S	1+0	1+0	1+0	2+0
	PDM	1+0	1+0	1+0	2+0
	SDIO/eMMC	1+0	1+0	1+0	2+0
	USB 2.0 FS	1+0	1+0	1+0	1+0

[1] ITCM: dedicated SRAM; DTCM: shared with SRAM

Contents

List of Figures

List of Tables

1 Introduction

1.1 System Architecture

SF32LB55x is a family of highly integrated and cost-effective System-on-Chip (SoC) MCUs desgined for ultra-low-power Artifical Intelligence of Things (AIoT) scenarios. SF32LB55x adopts the big.LITTLE architecture with two Arm Cortex-M33 STAR-MC1 processors.

- High-Performance Processor/Big Core (HCPU): CPU clock speed up to 240MHz, 32KB instruction cache (I-Cache) and 16KB data cache (D-Cache), 64KB ITCM, 128KB DTCM, 1088KB co-frequency SRAM (of which128KB is shared with D-Cache and 64KB is Retention RAM) and 384KB ROM; as the system master, it can efficiently access internal and external memory and is mainly used for system control, Human-Machine Interaction, and high-performance computing.
- Low-Power Processor/LITTLE Core (LCPU): CPU clock speed up to 48MHz, 16KB instruction cache (I-Cache) and 8KB data cache (D-Cache), 16KB ITCM, 16KB DTCM, 224KB low-power SRAM (all Retention RAM), and 384KB low-power ROM; it is mainly used as the system's ultra-low-power sensor hub and Bluetooth Low Energy connectivity controller which can meet the requirements of data acquisition, processing, transmission and control in ultra-low-power scenarios.

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, and the coprocessor interface is further improved to 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 has access to all address spaces of the HPSYS and can access all address spaces of the LPSYS through HP2LP.

DMAC1 can access all address spaces of HPSYS except HPSYS_ITCM and Retention RAM, and can access all address spaces of LPSYS except LPSYS_DTCM through HP2LP.

Some master devices (LCPU and DMAC2) of the LPSYS can access the address spaces of QSPI1~3, the address spaces of HPSYS_RAM0~5, and all HPSYS_APB peripherals through LP2HP.

The address spaces of HPSYS_ITCM and Retention RAM are only accessible by the HCPU, which are not listed in the figure. 128KB address space is shared between DTCM and HPSYS_RAM0, and can be accessed by the HCPU and other master devices.

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 polling 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, LCDC1 has a fixed priority over AES, and the access order of other master devices will be decided based on the polling arbitration principle.

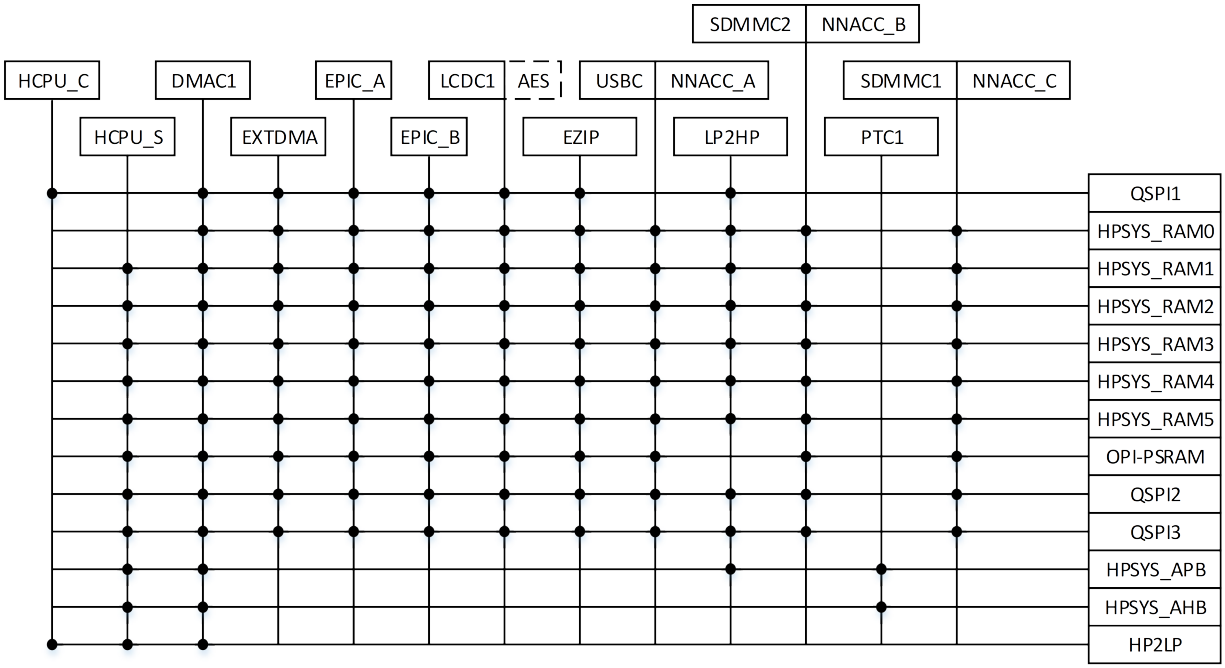


Figure 1-1: Bus Architecture of HPSYS

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 64KB zero-wait-cycle I-TCM with address space 0x0001_0000-0x0001_FFFF. This TCM memory is exclusive to the HCPU and cannot be accessed by other AHB masters. It can be used to place codes and data with high real-time (or delay determinism) requirements.

The HCPU also has 128KB zero-wait-cycle D-TCM with address space 0x2000_0000-0x2001_FFFF. This TCM memory is connected to the bus and can also be accessed by other AHB masters.

1.3.2.3 SRAM

There is a total of 1088KB SRAM on the HPSYS bus, which includes:

- 1024KB zero-wait-cycle SRAM with address space 0x2000_0000 -0x200F_FFFF (the first 128KB shared with DTCM), accessible to all AHB masters. Maximum frequency is 240MHz, which can maximize CPU performance.
- 64KB Retention SRAM with address space 0x0002_0000-0x0002_FFFF, accessible to the HCPU, maximum frequency 120MHz.

1.3.2.4 Off-chip RAM

The HPSYS supports external OPI DDR pSRAM with address space 0x6000_0000 - 0x63FF_FFFF, the actual accessible address is determined by the capacity of the external particles. The maximum interface frequency is DDR 120MHz and the data bit width is 8-bit.

1.3.2.5 Off-chip Flash

The HPSYS supports multiple external NOR/NAND FLASHs, in which

- 0x1000_0000-0x11FF_FFFF address segment for accessing SIP FLASH with a maximum interface frequency of 96MHz
- 0x6400_0000-0x67FF_FFFF address segment can be connected to FLASH2, recommended frequency is 60MHz
- 0x6800_0000-0x6FFF_FFFF address segment can be connected to FLASH3, recommended frequency is 60MHz

1.3.3 Address Mapping

Table 1-1: Address Mapping of HPSYS

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
HPSYS_ITCM	ROM	64KB	0x0000_0000	0x0000_FFFF		
	RAM	64KB	0x0001_0000	0x0001_FFFF		
Retention RAM		64KB	0x0002_0000	0x0002_FFFF		
XIP Memory	QSPI1 Memory	32MB	0x1000_0000	0x11FF_FFFF	0x1000_0000	0x11FF_FFFF
HPSYS_RAM (1024KB) 0x2000_0000 0x200F_FFFF	HPSYS_RAM0	128KB	0x2000_0000	0x2001_FFFF	0x2A00_0000	0x2A01_FFFF
	HPSYS_RAM1	128KB	0x2002_0000	0x2003_FFFF	0x2A02_0000	0x2A03_FFFF
	HPSYS_RAM2	128KB	0x2004_0000	0x2005_FFFF	0x2A04_0000	0x2A05_FFFF
	HPSYS_RAM3	128KB	0x2006_0000	0x2007_FFFF	0x2A06_0000	0x2A07_FFFF
	HPSYS_RAM4	256KB	0x2008_0000	0x200B_FFFF	0x2A08_0000	0x2A0B_FFFF
	HPSYS_RAM5	256KB	0x200C_0000	0x200F_FFFF	0x2A0C_0000	0x2A0F_FFFF

Continued on the next page...

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

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
HPSYS_APB1 (192KB) 0x4000_0000 0x4002_FFFF	RCC1	4KB	0x4000_0000	0x4000_0FFF	0x4000_0000	0x4000_0FFF
	DMAC1	4KB	0x4000_1000	0x4000_1FFF	0x4000_1000	0x4000_1FFF
	MAILBOX1	4KB	0x4000_2000	0x4000_2FFF	0x4000_2000	0x4000_2FFF
	PINMUX1	4KB	0x4000_3000	0x4000_3FFF	0x4000_3000	0x4000_3FFF
	UART1	4KB	0x4000_4000	0x4000_4FFF	0x4000_4000	0x4000_4FFF
	UART2	4KB	0x4000_5000	0x4000_5FFF	0x4000_5000	0x4000_5FFF
	EZIP	4KB	0x4000_6000	0x4000_6FFF	0x4000_6000	0x4000_6FFF
	EPIC	4KB	0x4000_7000	0x4000_7FFF	0x4000_7000	0x4000_7FFF
	LCDC1	4KB	0x4000_8000	0x4000_8FFF	0x4000_8000	0x4000_8FFF
	I2S1	4KB	0x4000_9000	0x4000_9FFF	0x4000_9000	0x4000_9FFF
	I2S2	4KB	0x4000_A000	0x4000_AFFF	0x4000_A000	0x4000_AFFF
	SYSCFG1	4KB	0x4000_B000	0x4000_BFFF	0x4000_B000	0x4000_BFFF
	EFUSEC	4KB	0x4000_C000	0x4000_CFFF	0x4000_C000	0x4000_CFFF
	AES	4KB	0x4000_D000	0x4000_DFFF	0x4000_D000	0x4000_DFFF
	CRC	4KB	0x4000_E000	0x4000_EFFF	0x4000_E000	0x4000_EFFF
	TRNG	4KB	0x4000_F000	0x4000_FFFF	0x4000_F000	0x4000_FFFF
	GPTIM1	4KB	0x4001_0000	0x4001_0FFF	0x4001_0000	0x4001_0FFF
	GPTIM2	4KB	0x4001_1000	0x4001_1FFF	0x4001_1000	0x4001_1FFF
	BTIM1	4KB	0x4001_2000	0x4001_2FFF	0x4001_2000	0x4001_2FFF
	BTIM2	4KB	0x4001_3000	0x4001_3FFF	0x4001_3000	0x4001_3FFF
	WDT1	4KB	0x4001_4000	0x4001_4FFF	0x4001_4000	0x4001_4FFF
	SPI1	4KB	0x4001_5000	0x4001_5FFF	0x4001_5000	0x4001_5FFF
	SPI2	4KB	0x4001_6000	0x4001_6FFF	0x4001_6000	0x4001_6FFF
	EXTDMA	4KB	0x4001_7000	0x4001_7FFF	0x4001_7000	0x4001_7FFF
	PSRAMC	4KB	0x4001_8000	0x4001_8FFF	0x4001_8000	0x4001_8FFF
	NNACC	4KB	0x4001_9000	0x4001_9FFF	0x4001_9000	0x4001_9FFF
	PDM1	4KB	0x4001_A000	0x4001_AFFF	0x4001_A000	0x4001_AFFF
	PDM2	4KB	0x4001_B000	0x4001_BFFF	0x4001_B000	0x4001_BFFF
	I2C1	4KB	0x4001_C000	0x4001_CFFF	0x4001_C000	0x4001_CFFF
	I2C2	4KB	0x4001_D000	0x4001_DFFF	0x4001_D000	0x4001_DFFF
	DSIHOST	4KB	0x4001_E000	0x4001_EFFF	0x4001_E000	0x4001_EFFF
	DSIPHY	4KB	0x4001_F000	0x4001_FFFF	0x4001_F000	0x4001_FFFF
	PTC1	4KB	0x4002_0000	0x4002_0FFF	0x4002_0000	0x4002_0FFF
	BUSMON1	4KB	0x4002_1000	0x4002_1FFF	0x4002_1000	0x4002_1FFF
	I2C3	4KB	0x4002_2000	0x4002_2FFF	0x4002_2000	0x4002_2FFF
	Reserved	54KB	0x4002_3000	0x4002_FFFF	0x4002_3000	0x4002_FFFF
HPSYS_APB2 (64KB) 0x4003_0000 0x4003_FFFF	HPSYS_AON	4KB	0x4003_0000	0x4003_0FFF	0x4003_0000	0x4003_0FFF
	LPTIM1	4KB	0x4003_1000	0x4003_1FFF	0x4003_1000	0x4003_1FFF
	Reserved	4KB	0x4003_2000	0x4003_2FFF	0x4003_2000	0x4003_2FFF
	Reserved	4KB	0x4003_3000	0x4003_3FFF	0x4003_3000	0x4003_3FFF
	Reserved	48KB	0x4003_4000	0x4003_FFFF	0x4003_4000	0x4003_FFFF

Continued on the next page...

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

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
HPSYS_AHB (256KB) 0x5000_0000 0x5003_FFFF	GPIO1	4KB	0x5000_0000	0x5000_0FFF		
	QSPI1	4KB	0x5000_1000	0x5000_1FFF		
	QSPI2	4KB	0x5000_2000	0x5000_2FFF		
	QSPI3	4KB	0x5000_3000	0x5000_3FFF		
	SDMMC1	4KB	0x5000_4000	0x5000_4FFF		
	SDMMC2	4KB	0x5000_5000	0x5000_5FFF		
	USBC	4KB	0x5000_6000	0x5000_6FFF		
	Reserved	36KB	0x5000_7000	0x5000_FFFF		
	GFX_RAM	64KB	0x5001_0000	0x5001_FFFF		
	Reserved	128KB	0x5002_0000	0x5003_FFFF		
External Memory (256MB)	OPI-PSRAM	64MB	0x6000_0000	0x63FF_FFFF		
	QSPI2 Memory	64MB	0x6400_0000	0x67FF_FFFF	0x6400_0000	0x67FF_FFFF
	QSPI3 Memory	128MB	0x6800_0000	0x6FFF_FFFF	0x6800_0000	0x6FFF_FFFF

1.3.4 Interrupt List

Table 1-2: Interrupt List of HCPU

IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source
HCPU_NMI	**WDT1**	HCPU_IRQ[24]	**LCPU_IRQ[24]**	HCPU_IRQ[49]	**RTC**	HCPU_IRQ[74]	**UART2**
HCPU_IRQ[0]	**AON1**	HCPU_IRQ[25]	**LCPU_IRQ[25]**	HCPU_IRQ[50]	**DMAC1_CH1**	HCPU_IRQ[75]	**SPI2**
HCPU_IRQ[1]	**LCPU_IRQ[1]**	HCPU_IRQ[26]	**LCPU_IRQ[26]**	HCPU_IRQ[51]	**DMAC1_CH2**	HCPU_IRQ[76]	**I2C2**
HCPU_IRQ[2]	**LCPU_IRQ[2]**	HCPU_IRQ[27]	**LCPU_IRQ[27]**	HCPU_IRQ[52]	**DMAC1_CH3**	HCPU_IRQ[77]	**EXTDMA**
HCPU_IRQ[3]	**LCPU_IRQ[3]**	HCPU_IRQ[28]	**LCPU_IRQ[28]**	HCPU_IRQ[53]	**DMAC1_CH4**	HCPU_IRQ[78]	**PSRAMC**
HCPU_IRQ[4]	**LCPU_IRQ[4]**	HCPU_IRQ[29]	**LCPU_IRQ[29]**	HCPU_IRQ[54]	**DMAC1_CH5**	HCPU_IRQ[79]	**SDMMC1**
HCPU_IRQ[5]	**LCPU_IRQ[5]**	HCPU_IRQ[30]	**LCPU_IRQ[30]**	HCPU_IRQ[55]	**DMAC1_CH6**	HCPU_IRQ[80]	**SDMMC2**
HCPU_IRQ[6]	**LCPU_IRQ[6]**	HCPU_IRQ[31]	**LCPU_IRQ[31]**	HCPU_IRQ[56]	**DMAC1_CH7**	HCPU_IRQ[81]	**NNACC**
HCPU_IRQ[7]	**LCPU_IRQ[7]**	HCPU_IRQ[32]	**LCPU_IRQ[32]**	HCPU_IRQ[57]	**DMAC1_CH8**	HCPU_IRQ[82]	**PDM1**
HCPU_IRQ[8]	**LCPU_IRQ[8]**	HCPU_IRQ[33]	**LCPU_IRQ[33]**	HCPU_IRQ[58]	**LCPU2HCPU**	HCPU_IRQ[83]	**DSI**
HCPU_IRQ[9]	**LCPU_IRQ[9]**	HCPU_IRQ[34]	**LCPU_IRQ[34]**	HCPU_IRQ[59]	**UART1**	HCPU_IRQ[84]	**GPIO1**
HCPU_IRQ[10]	**LCPU_IRQ[10]**	HCPU_IRQ[35]	**LCPU_IRQ[35]**	HCPU_IRQ[60]	**SPI1**	HCPU_IRQ[85]	**QSPI1**
HCPU_IRQ[11]	**LCPU_IRQ[11]**	HCPU_IRQ[36]	**LCPU_IRQ[36]**	HCPU_IRQ[61]	**I2C1**	HCPU_IRQ[86]	**QSPI2**
HCPU_IRQ[12]	**LCPU_IRQ[12]**	HCPU_IRQ[37]	**LCPU_IRQ[37]**	HCPU_IRQ[62]	**EPIC**	HCPU_IRQ[87]	**QSPI3**
HCPU_IRQ[13]	**LCPU_IRQ[13]**	HCPU_IRQ[38]	**LCPU_IRQ[38]**	HCPU_IRQ[63]	**LCDC1**	HCPU_IRQ[88]	**EZIP**
HCPU_IRQ[14]	**LCPU_IRQ[14]**	HCPU_IRQ[39]	**LCPU_IRQ[39]**	HCPU_IRQ[64]	**I2S1**	HCPU_IRQ[89]	**PDM2**
HCPU_IRQ[15]	**LCPU_IRQ[15]**	HCPU_IRQ[40]	**LCPU_IRQ[40]**	HCPU_IRQ[65]	**I2S2**	HCPU_IRQ[90]	**USBC**
HCPU_IRQ[16]	**LCPU_IRQ[16]**	HCPU_IRQ[41]	**LCPU_IRQ[41]**	HCPU_IRQ[66]	**EFUSEC**	HCPU_IRQ[91]	**I2C3**
HCPU_IRQ[17]	**LCPU_IRQ[17]**	HCPU_IRQ[42]	**LCPU_IRQ[42]**	HCPU_IRQ[67]	**AES**	HCPU_IRQ[92]	**Reserved**
HCPU_IRQ[18]	**LCPU_IRQ[18]**	HCPU_IRQ[43]	**LCPU_IRQ[43]**	HCPU_IRQ[68]	**PTC1**	HCPU_IRQ[93]	**Reserved**
HCPU_IRQ[19]	**LCPU_IRQ[19]**	HCPU_IRQ[44]	**LCPU_IRQ[44]**	HCPU_IRQ[69]	**TRNG**	HCPU_IRQ[94]	**Reserved**
HCPU_IRQ[20]	**LCPU_IRQ[20]**	HCPU_IRQ[45]	**LCPU_IRQ[45]**	HCPU_IRQ[70]	**GPTIM1**	HCPU_IRQ[95]	**Reserved**
HCPU_IRQ[21]	**LCPU_IRQ[21]**	HCPU_IRQ[46]	**LPTIM1**	HCPU_IRQ[71]	**GPTIM2**	\	\
HCPU_IRQ[22]	**LCPU_IRQ[22]**	HCPU_IRQ[47]	**Reserved**	HCPU_IRQ[72]	**BTIM1**	\	\
HCPU_IRQ[23]	**LCPU_IRQ[23]**	HCPU_IRQ[48]	**Reserved**	HCPU_IRQ[73]	**BTIM2**	\	\

1.4 Low-Power Processor (LITTLE Core) System (LPSYS)

1.4.1 Bus Architecture

The LPSYS 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 1-2, 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.

LCPU has access to all address spaces of LPSYS and can access the address spaces of QSPI1~3, HPSYS_RAM0~5, and all HPSYS_APB peripherals through LP2HP.

DMAC2 can access all address spaces of LPSYS except LPSYS_ROM, and can access the address spaces of QSPI1~3, HPSYS_RAM0~5, and all HPSYS_APB peripherals through LP2HP.

Some master devices (HCPU and DMAC1) of the HPSYS are able to access all address spaces of the LPSYS through HP2LP.

LPSYS_ITCM and LPSYS_DTCM can be accessed by LCPU and DMAC2, and also by some master devices (HCPU and DMAC1) of the HPSYS.

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 polling arbitration principle. When multiple master devices access the address spaces of different slave devices at the same time, they will not be affected by each other.

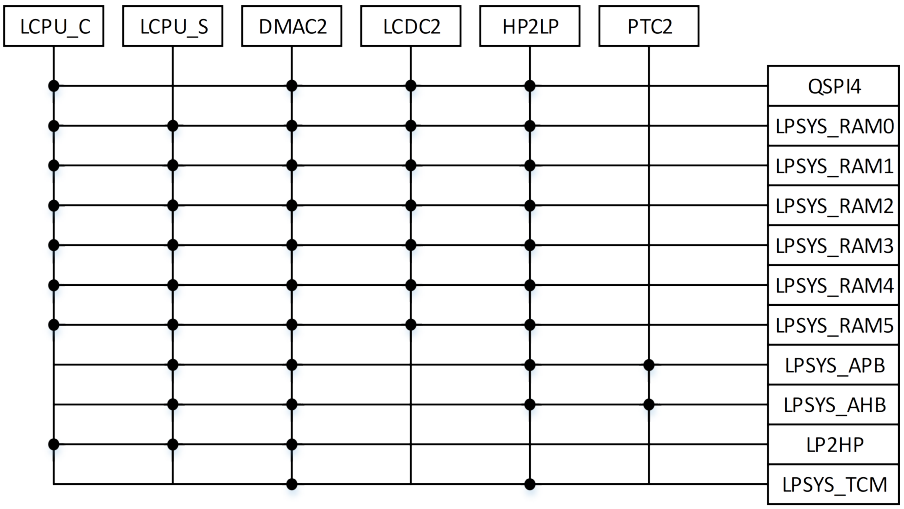


Figure 1-2: Bus Architecture of LPSYS

1.4.2 Memory Type

1.4.2.1 Cache

LCPU has 16KB 2-way I-Cache (Level 1 instruction cache) and 8KB 4-way D-Cache (Level 1 data cache).

1.4.2.2 TCM

LCPU has 16KB zero-wait-cycle I-TCM with address space 0x000F_C000-0x000F_FFFF. This TCM memory is exclusive to LCPU and can be initialized by HCPU via address segments 0x0B0F_C000 - 0x0B0F_FFFF. It is recommended to place codes and data with high real-time (or delay determinism) requirements.

LCPU also has 16KB zero-wait-cycle D-TCM with address space 0x200F_C000 - 0x200F_FFFF. This TCM memory is exclusive to LCPU and can be initialized by HCPU via address segment 0x2B0F_C000 - 0x2B0F_FFFF.

1.4.2.3 SRAM

There is a total of 224KB SRAM on LPSYS bus, which includes:

- 0x2010_0000-0x2010_7FFF, 32KB zero-wait-cycle SRAM, maximum frequency 48M
- 0x2010_8000-0x2013_7FFF, 192KB one-wait-cycle SRAM, maximum frequency 48M

All SRAMs can hold data in low-power mode.

1.4.2.4 Off-chip Flash

LPSYS supports external FLASH4 with address space 0x1200_0000-0x13FF_FFFF.

1.4.3 Address Mapping

Table 1-3: Address Mapping of LPSYS

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
LPSYS_ITCM	ROM	384KB	0x0B00_0000	0x0B05_FFFF	0x0000_0000	0x0005_FFFF
	RAM	16KB	0x0B0F_C000	0x0B0F_FFFF	0x000F_C000	0x000F_FFFF
LPSYS_DTCM	RAM	16KB	0x2B0F_C000	0x2B0F_FFFF	0x200F_C000	0x200F_FFFF
XIP Memory	QSPI4 Memory	32MB	0x1200_0000	0x13FF_FFFF	0x1200_0000	0x13FF_FFFF
LPSYS_RAM (128KB) 0X2010_0000 0x2011_FFFF	LPSYS_RAM0	16KB	0x2010_0000	0x2010_3FFF	0x2010_0000	0x2010_3FFF
	LPSYS_RAM1	16KB	0x2010_4000	0x2010_7FFF	0x2010_4000	0x2010_7FFF
	LPSYS_RAM2	32KB	0x2010_8000	0x2010_FFFF	0x2010_8000	0x2010_FFFF
	LPSYS_RAM3	64KB	0x2011_0000	0x2011_FFFF	0x2011_0000	0x2011_FFFF
	LPSYS_RAM4	64KB	0x2012_0000	0x2012_FFFF	0x2012_0000	0x2012_FFFF
	LPSYS_RAM5	32KB	0x2013_0000	0x2013_7FFF	0x2013_0000	0x2013_7FFF

Continued on the next page...

Table 1-3: Address Mapping of LPSYS (continued)

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
LPSYS_APB1 (192KB) 0x4004_0000 0x4006_FFFF	RCC2	4KB	0x4004_0000	0x4004_0FFF	0x4004_0000	0x4004_0FFF
	DMAC2	4KB	0x4004_1000	0x4004_1FFF	0x4004_1000	0x4004_1FFF
	MAILBOX2	4KB	0x4004_2000	0x4004_2FFF	0x4004_2000	0x4004_2FFF
	PINMUX2	4KB	0x4004_3000	0x4004_3FFF	0x4004_3000	0x4004_3FFF
	PATCH	4KB	0x4004_4000	0x4004_4FFF	0x4004_4000	0x4004_4FFF
	UART3	4KB	0x4004_5000	0x4004_5FFF	0x4004_5000	0x4004_5FFF
	UART4	4KB	0x4004_6000	0x4004_6FFF	0x4004_6000	0x4004_6FFF
	UART5	4KB	0x4004_7000	0x4004_7FFF	0x4004_7000	0x4004_7FFF
	Reserved	4KB	0x4004_8000	0x4004_8FFF	0x4004_8000	0x4004_8FFF
	SPI3	4KB	0x4004_9000	0x4004_9FFF	0x4004_9000	0x4004_9FFF
	SPI4	4KB	0x4004_A000	0x4004_AFFF	0x4004_A000	0x4004_AFFF
	Reserved	4KB	0x4004_B000	0x4004_BFFF	0x4004_B000	0x4004_BFFF
	I2C4	4KB	0x4004_C000	0x4004_CFFF	0x4004_C000	0x4004_CFFF
	I2C5	4KB	0x4004_D000	0x4004_DFFF	0x4004_D000	0x4004_DFFF
	I2C6	4KB	0x4004_E000	0x4004_EFFF	0x4004_E000	0x4004_EFFF
	SYSCFG2	4KB	0x4004_F000	0x4004_FFFF	0x4004_F000	0x4004_FFFF
	GPTIM3	4KB	0x4005_0000	0x4005_0FFF	0x4005_0000	0x4005_0FFF
	GPTIM4	4KB	0x4005_1000	0x4005_1FFF	0x4005_1000	0x4005_1FFF
	GPTIM5	4KB	0x4005_2000	0x4005_2FFF	0x4005_2000	0x4005_2FFF
	BTIM3	4KB	0x4005_3000	0x4005_3FFF	0x4005_3000	0x4005_3FFF
	BTIM4	4KB	0x4005_4000	0x4005_4FFF	0x4005_4000	0x4005_4FFF
	WDT2	4KB	0x4005_5000	0x4005_5FFF	0x4005_5000	0x4005_5FFF
	GPADC	4KB	0x4005_6000	0x4005_6FFF	0x4005_6000	0x4005_6FFF
	SDADC	4KB	0x4005_7000	0x4005_7FFF	0x4005_7000	0x4005_7FFF
	Reserved	4KB	0x4005_8000	0x4005_8FFF	0x4005_8000	0x4005_8FFF
	LPCOMP	4KB	0x4005_9000	0x4005_9FFF	0x4005_9000	0x4005_9FFF
	TSEN	4KB	0x4005_A000	0x4005_AFFF	0x4005_A000	0x4005_AFFF
	PTC2	4KB	0x4005_B000	0x4005_BFFF	0x4005_B000	0x4005_BFFF
	LCDC2	4KB	0x4005_C000	0x4005_CFFF	0x4005_C000	0x4005_CFFF
	BUSMON2	4KB	0x4005_D000	0x4005_DFFF	0x4005_D000	0x4005_DFFF
	Reserved	4KB	0x4005_E000	0x4005_EFFF	0x4005_E000	0x4005_EFFF
	Reserved	4KB	0x4005_F000	0x4005_FFFF	0x4005_F000	0x4005_FFFF
	Reserved	64KB	0x4006_0000	0x4006_FFFF	0x4006_0000	0x4006_FFFF
LPSYS_APB2 (64KB) 0x4007_0000 0x4007_FFFF	LPSYS_AON	4KB	0x4007_0000	0x4007_0FFF	0x4007_0000	0x4007_0FFF
	LPTIM2	4KB	0x4007_1000	0x4007_1FFF	0x4007_1000	0x4007_1FFF
	LPTIM3	4KB	0x4007_2000	0x4007_2FFF	0x4007_2000	0x4007_2FFF
	Reserved	4KB	0x4007_3000	0x4007_3FFF	0x4007_3000	0x4007_3FFF
	Reserved	24KB	0x4007_4000	0x4007_9FFF	0x4007_4000	0x4007_9FFF
	PMUC	4KB	0x4007_A000	0x4007_AFFF	0x4007_A000	0x4007_AFFF
	RTC	4KB	0x4007_B000	0x4007_BFFF	0x4007_B000	0x4007_BFFF
	IWDT	4KB	0x4007_C000	0x4007_CFFF	0x4007_C000	0x4007_CFFF
	Reserved	12KB	0x4007_D000	0x4007_FFFF	0x4007_D000	0x4007_FFFF

Continued on the next page...

Table 1-3: Address Mapping of LPSYS (continued)

Category	Memory/IP	Address Space	HCPU Address Space		LCPU Address Space	
			Starting Address	Ending Address	Starting Address	Ending Address
LPSYS_AHB (256KB) 0x5004_0000 0x5007_FFFF	GPIO2	4KB	0x5004_0000	0x5004_0FFF	0x5004_0000	0x5004_0FFF
	QSPI4	4KB	0x5004_1000	0x5004_1FFF	0x5004_1000	0x5004_1FFF
	RFC	8KB	0x5004_2000	0x5004_3FFF	0x5004_2000	0x5004_3FFF
	PHY	4KB	0x5004_4000	0x5004_4FFF	0x5004_4000	0x5004_4FFF
	Reserved	44KB	0x5004_5000	0x5004_FFFF	0x5004_5000	0x5004_FFFF
	MAC	64KB	0x5005_0000	0x5005_FFFF	0x5005_0000	0x5005_FFFF
	Reserved	128KB	0x5006_0000	0x5007_FFFF	0x5006_0000	0x5007_FFFF
PHY_DUMP	PHY_DUMP	64KB	0x5008_0000	0x5008_FFFF	0x5008_0000	0x5008_FFFF

1.4.4 Interrupt List

Table 1-4: Interrupt List of LCPU

IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source
LCPU_NMI	**WDT2**	LCPU_IRQ[12]	**UART3**	LCPU_IRQ[25]	**BTIM3**	LCPU_IRQ[38]	**Reserved**
LCPU_IRQ[0]	**AON2**	LCPU_IRQ[13]	**UART4**	LCPU_IRQ[26]	**BTIM4**	LCPU_IRQ[39]	**Reserved**
LCPU_IRQ[1]	**BT_MAC**	LCPU_IRQ[14]	**UART5**	LCPU_IRQ[27]	**Reserved**	LCPU_IRQ[40]	**Reserved**
LCPU_IRQ[2]	**DMAC2_CH1**	LCPU_IRQ[15]	**Reserved**	LCPU_IRQ[28]	**GPADC**	LCPU_IRQ[41]	**LPCOMP**
LCPU_IRQ[3]	**DMAC2_CH2**	LCPU_IRQ[16]	**SPI3**	LCPU_IRQ[29]	**SDADC**	LCPU_IRQ[42]	**LPTIM2**
LCPU_IRQ[4]	**DMAC2_CH3**	LCPU_IRQ[17]	**SPI4**	LCPU_IRQ[30]	**Reserved**	LCPU_IRQ[43]	**LPTIM3**
LCPU_IRQ[5]	**DMAC2_CH4**	LCPU_IRQ[18]	**Reserved**	LCPU_IRQ[31]	**Reserved**	LCPU_IRQ[44]	**Reserved**
LCPU_IRQ[6]	**DMAC2_CH5**	LCPU_IRQ[19]	**I2C4**	LCPU_IRQ[32]	**TSEN**	LCPU_IRQ[45]	**Reserved**
LCPU_IRQ[7]	**DMAC2_CH6**	LCPU_IRQ[20]	**I2C5**	LCPU_IRQ[33]	**PTC2**	LCPU_IRQ[46]	**HCPU2LCPU**
LCPU_IRQ[8]	**DMAC2_CH7**	LCPU_IRQ[21]	**I2C6**	LCPU_IRQ[34]	**LCDC2**	LCPU_IRQ[47]	**RTC**
LCPU_IRQ[9]	**DMAC2_CH8**	LCPU_IRQ[22]	**GPTIM3**	LCPU_IRQ[35]	**GPIO2**	\	\
LCPU_IRQ[10]	**PATCH**	LCPU_IRQ[23]	**GPTIM4**	LCPU_IRQ[36]	**QSPI4**	\	\
LCPU_IRQ[11]	**Reserved**	LCPU_IRQ[24]	**GPTIM5**	LCPU_IRQ[37]	**Reserved**	\	\

1.5 Bus Access Permissions

Table 1-5: Bus Access Permissions

	AHB Slave device										
AHB Master controller	HP_ITCM	HP_RAM 0~5	PSRAMC	QSPI1	QSPI2~3	HP_AHB	HP_APB	LP_DTCM LP_ITCM	LP_RAM 0~5	QSPI4	LP_AHB LP_APB
HCPU	√	√	√	√	√	√	√	√(1)	√	√	√
DMAC1	x	√	√	√	√	√	√	√(1)	√	√	√
EXTDMA	x	√	√	√	√	x	x	x	x	x	x
AES	x	√	√	√	√	x	x	x	x	x	x
LCDC1	x	√	√	√	√	x	x	x	x	x	x
EZIP	x	√	√	√	√	x	x	x	x	x	x
EPIC	x	√	√	√	√	x	x	x	x	x	x
USBC	x	√	√	x	√	x	x	x	x	x	x
NNACC	x	√	√	x	√	x	x	x	x	x	x
SDMMC1	x	√	√	x	√	x	x	x	x	x	x
SDMMC2	x	√	√	x	√	x	x	x	x	x	x
PTC1	x	x	x	x	x	√	√	x	x	x	x
LCPU	x	√(2)	x	√	√	x	√	√	√(3)	√	√
DMAC2	x	√(2)	x	√	√	x	√	√	√	√	√
LCDC2	x	x	x	x	x	x	x	x	√	√	x
PTC2	x	x	x	x	x	x	x	x	x	x	√

* (1)The master controller of HPSYS accesses the ITCM of LPSYS, starting at address 0x0b000000, and accesses the DTCM of LPSYS, starting at address 0x2b000000, which differs from the address accessed by LCPU by an offset of 0x0b000000.
(2)The master controller of LPSYS accesses the SRAM of HPSYS, starting at address 0x2a000000, which corresponds to an added offset of 0x0a000000.
(3)LCPU accesses the SRAM of LPSYS, with the starting address either at 0x00100000 or at 0x20100000.

2 Clock and Reset

2.1 Introduction

The Clock and Reset Module controls the chip's clock and reset functions, supporting clock selection, clock division, module enablement, and module reset operations.

2.2 Reset Sources

The chip's reset sources are primarily divided into four categories: board-level reset, watchdog reset, software reset, and wake-up reset. Each category comprises several reset sources, each with a distinct scope of effect.

2.2.1 Board-Level Reset Sources

Board-level reset sources mainly include:

Power-On Reset POR (Power-On Reset). A reset automatically generated upon chip power-up, capable of initializing the entire chip and resetting all module states to their default values.

Brown-Out Reset BOR (Brown-Out Reset). A reset automatically triggered when the chip supply voltage falls below a specifed threshold, capable of initializing the entire chip and resetting all module states to their default values.

Reset Button (RSTN) Reset. If the reset button RSTN pin level is low, a reset will occur, initializing the entire chip and resetting all module states to their default values.

2.2.2 Watchdog Reset Sources

Watchdog reset sources primarily include:

Global Watchdog IWDT. If this watchdog times out, it generates an IWDT reset, resetting all modules except PMUC, RTC, and IWDT.

HPSYS Watchdog WDT1. If this watchdog times out, it generates a WDT1 reset, resetting the HCPU and all peripherals in HPSYS except for HPSYS_AON.

LPSYS Watchdog WDT2. If this watchdog times out, it generates a WDT2 reset, resetting the LCPU and all peripherals in LPSYS except for LPSYS_AON.

2.2.3 Software Reset Sources

Software Reboot. Software can trigger a reboot by writing 1 to the CR_REBOOT bit of the PMUC. After reboot, all modules except PMUC,RTC, and IWDT are reset. After a software reboot, the PMUC's CR_REBOOT remains set to 1, serving as an indicator that a software reboot has occurred. To trigger a software reboot again, CR_REBOOT must first be cleared to 0.

HCPU system reset. By configuring the internal registers of HCPU to send SYSRESETREQ, software can reset all modules within HPSYS except for HPSYS_AON, including HCPU, EPIC, DMAC1, and others. A system reset issued by an external debugger connected to HCPU is equivalent to an HCPU system reset.

LCPU system reset. By configuring the internal registers of LCPU to send SYSRESETREQ, software can reset all modules within LPSYS except for LPSYS_AON, including LCPU, DMAC2, MAC, and others. The system reset triggered after the external debugger connects to the LCPU is equivalent to the LCPU system reset.

Module RCC reset. Individual module reset can be performed via the RSTRx registers within HPSYS_RCC or LPSYS_RCC.

2.2.4 Wakeup Reset Sources

Hibernate Wakeup. When the chip enters hibernate mode, all modules except PMUC,RTC, and IWDT are reset upon wakeup.

HPSYS Standby Wakeup. When HPSYS enters standby mode, all internal modules of HPSYS except HPSYS_AON are reset upon wakeup, including HCPU, EPIC, DMAC1, and others.

LPSYS standby wake-up. LPSYS enters standby mode; upon wake-up, it resets all internal modules of LPSYS except for LPSYS_AON, including LCPU, DMAC2, MAC, etc.

Table 2-1: Main Reset Sources of the Chip

Reset Scope		Board-Level Reset			Watchdog Reset		
		POR	BOR	PIN RSTN	IWDT	WDT1	WDT2
HPSYS	HCPU	√	√	√	√	√	x
	SRAM ret	√	√	√	√	x	x
	SRAM noret	√	√	√	√	x	x
	HPSYS Peri	√	√	√	√	√	x
	HPAON	√	√	√	√	x	x
LPSYS	LCPU	√	√	√	√	x	√
	SRAM ret	√	√	√	√	x	x
	SRAM noret	√	√	√	√	x	x
	LPSYS Peri	√	√	√	√	x	√
	LPAON	√	√	√	√	x	x
AON	PMUC	√	√	√	x	x	x
	RTC	√	√	√	x	x	x
	IWDT	√	√	√	x	x	x

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 Peri	√	√	x	√	√	x
	HPAON	√	x	x	√	x	x
LPSYS	LCPU	√	x	√	√	x	√
	SRAM ret	√	x	x	√	x	x
	SRAM noret	√	x	x	√	x	√
	LPSYS Peri	√	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 Sources

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

Table 2-3: Clock Sources

Clock	Frequency	Dependency
clk_lrc10	~10kHz	None
clk_lxt32	32.768kHz	32k Crystal Oscillator
clk_hrc48	~48MHz	None
clk_hxt48	48MHz	48M Crystal Oscillator
dll1/2/3	48~384MHz	clk_hxt48

clk_lrc10 is a low-power RC oscillator clock internally generated by the chip, with a frequency of approximately 10kHz. This clock frequency is influenced by external environmental factors; the current frequency can be obtained through a measurement procedure during use. After chip startup, this clock is automatically enabled as the default operating clock for low-power modules (such as PMUC). The related configuration register for clk_lrc10 is the LRC_CR register within the PMUC.

clk_lxt32 is a low-power clock generated based on an external 32k crystal oscillator, with a frequency of 32.768kHz. This clock is optional and is recommended for scenarios requiring precise timing. The configuration register related to clk_lxt32 is LXT_CR within the PMUC.

clk_hrc48 is an RC oscillator clock generated internally by the chip. Upon chip startup, this clock is automatically enabled

as the default operating clock for HCPU; however, its frequency is uncalibrated at this stage and is an unknown value below 48MHz. Prior to calibration, the HCPU operating clock should be switched to another clock (e.g., clk_hxt48), followed by execution of the calibration procedure. After calibration, the frequency of clk_hrc48 is 48MHz. The configuration register for clk_hrc48 is HRC_CR within the PMUC, and the calibration-related registers are HRCCAL1 and HRCCAL2 in HPSYS_RCC. When the chip is in overall low-power mode, this clock is disabled by default.

clk_hxt48 is a clock generated from an external 48M crystal oscillator, with a frequency of 48MHz . This clock automatically starts when the chip powers on; it serves as the base clock for generating higher-frequency clocks and is also the fundamental clock required for Bluetooth operation. When the system does not require higher-frequency clocks and Bluetooth is in sleep mode, this clock can be disabled. When the chip is in low-power mode overall, this clock is disabled by default. The related configuration registers for clk_hrc48 are the HXT_CR1/2/3 registers within the PMUC .

clk_dll1/2/3 are high-frequency clocks generated by the internal DLL modules of the chip based on clk_hxt48 . These clocks are disabled by default and can be enabled individually as required, each capable of independently generating different frequencies. The clock frequency generated by the DLL Module can be configured in increments of 24MHz or 48MHz, with configuration registers DLL1CR, DLL2CR, and DLL3CR located within HPSYS_RCC. When the chip is in overall low-power mode, these clocks are disabled by default.

2.4 HPSYS Clock Structure

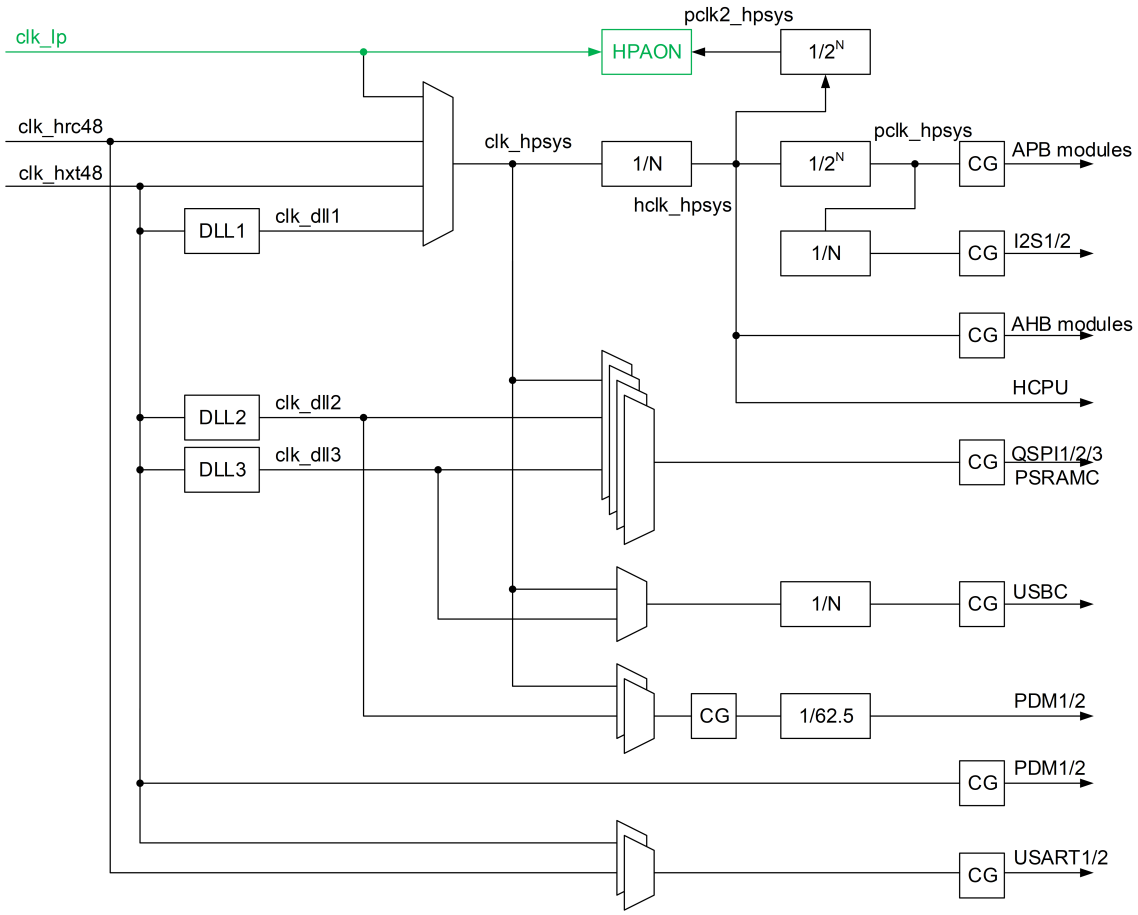


Figure 2-1: HPSYS Clock Structure

The system clock clk_hpsys of HPSYS can be selected from clk_hrc48, clk_hxt48, clk_lp, and clk_dll1. The selection register is located in HPSYS_RCC of CSR_SEL_SYS. The maximum frequency supported byclk_hpsys is 240MHz.

hclk_hpsys is generated by dividing clk_hpsys 1 by N, with the division ratio specifed by CFGR_HDIV in HPSYS_RCC. The maximum frequency supported by hclk is 240 MHz and it serves as the operating clock for AHB modules such as HCPU, EPIC, DMAC1, as well as the AHB bus and SRAM.

pclk_hpsys is generated by dividing hclk_hpsys 1 by 2N, with the division ratio specifed by CFGR_PDIV1. The maximum frequency supported by pclk_hpsys is240 MHz and it serves as the operating clock for APB modules such as GPTIM1/2, BTIM1/2, I2C1/2/3, SPI1/2, as well as the APB bus clock. When the frequency of hclk_hpsys or pclk_hpsys changes, the operating clock frequency of modules such asGPTIM1/2 ,BTIM1/2 ,I2C1/2/3 , andSPI1/2 changes accordingly, which affects their functionality. Therefore, during the operation of the related modules, the frequencies of hclk_hpsys and pclk_hpsys must remain constant.

pclk2_hpsys is generated by dividing hclk_hpsys by a ratio of 1 to 2^N, with a division ratio of 2^{CFGR_PDIV2}. pclk2_hpsys supports a maximum frequency of 7.5MHz and serves as the register access clock for the HPAON module.

clk_lp is a low-power clock generated by selecting between clk_lrc10 and clk_lxt32 (refer to the LPSYS Clock Structure Block Diagram) and serves as the operating clock for the HPAON Module.

The operating clocks of QSPI1/2/3 can be selected from clk_hpsys , clk_dll2 , and clk_dll3. The selection register is CSR_SEL_QSPI1/2/3 in HPSYS_RCC.

The operating clock of PSRAMC can be selected from clk_hpsys , clk_dll2 , and clk_dll3. The selection register is CSR_SEL_PSRAMC in HPSYS_RCC.

The operating clock of USB can be selected from clk_hpsys and clk_dll3 using the selection register CSR_SEL_USB and is divided by a 1-to-N clock division, where the division ratio is USBCR_DIV. It is necessary to ensure that after clock division, the USB operating clock is 60 MHz; otherwise, the USB will not function properly.

The operating clock of USART1/2 can be selected between clk_hrc48 and clk_hxt48. The selection register is CSR_SEL_USART1/2 within HPSYS_RCC. The operating clock of USART1/2 is independent of the system clock, thus it remains unaffected during dynamic frequency adjustments of the system.

The operating clock of I2S1/2 is generated by dividing pclk_hpsys 1 by N, where the division ratio is specifed by I2SR in HPSYS_RCC.

PDM1/2 can select one of two operating clocks for use; the selection register is located within PDM. One operating clock is generated by clk_hxt48; the other The system working clock is generated by selecting between clk_hpsys and clk_dll2 and then divided by 62.5; the selection register is CSR_SEL_PDM1/2 in HPSYS_RCC.

2.5 HPSYS Module Enable

The ENR1 and ENR2 registers in HPSYS_RCC control the enablement of each HPSYS module. When the bit corresponding to a module is 1, the module's registers are accessible and the module can operate. When the bit corresponding to a module is 0, the module's working clock and bus clock are both disabled, the module stops operating, and its registers are inaccessible; however, the register values are not reset.

2.6 HPSYS Module Reset

The RSTR1 and RSTR2 registers in HPSYS_RCC control the reset of each module within HPSYS. When the bit corresponding to a module is set to 1, the module's registers and internal states are reset. When the bit is set to 0, the module reset is released.

2.7 HPSYS_RCC Registers

Table 2-4: HPSYS_RCC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**RSTR1**	Reset Register 1
[31]	rw	1'h0	PTC1	0 - no reset; 1 - reset
[30]	rw	1'h0	DSIPHY	0 - no reset; 1 - reset
[29]	rw	1'h0	DSIHOST	0 - no reset; 1 - reset
[28]	rw	1'h0	I2C2	0 - no reset; 1 - reset
[27]	rw	1'h0	I2C1	0 - no reset; 1 - reset
[26]	rw	1'h0	PDM2	0 - no reset; 1 - reset
[25]	rw	1'h0	PDM1	0 - no reset; 1 - reset
[24]	rw	1'h0	NNACC	0 - no reset; 1 - reset
[23]	rw	1'h0	PSRAMC	0 - no reset; 1 - reset
[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]	rw	1'h0	WDT1	0 - no reset; 1 - reset
[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	CRC	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]	rw	1'h0	I2S2	0 - no reset; 1 - reset
[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	EZIP	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
[0]	rw	1'h0	DMAC1	0 - no reset; 1 - reset
0x04			**RSTR2**	Reset Register 2
[31:9]			RSVD	
[8]	rw	1'h0	I2C3	0 - no reset; 1 - reset
[7]	rw	1'h0	BUSMON1	0 - no reset; 1 - reset
[6]	rw	1'h0	USBC	0 - no reset; 1 - reset
[5]	rw	1'h0	SDMMC2	0 - no reset; 1 - reset
[4]	rw	1'h0	SDMMC1	0 - no reset; 1 - reset

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3]	rw	1'h0	QSPI3	0 - no reset; 1 - reset
[2]	rw	1'h0	QSPI2	0 - no reset; 1 - reset
[1]	rw	1'h0	QSPI1	0 - no reset; 1 - reset
[0]	rw	1'h0	GPIO1	0 - no reset; 1 - reset
0x08			**ENR1**	Enable Register 1
[31]	rw	1'h1	PTC1	0 - disabled; 1 - enabled
[30]	rw	1'h1	DSIPHY	0 - disabled; 1 - enabled
[29]	rw	1'h1	DSIHOST	0 - disabled; 1 - enabled
[28]	rw	1'h1	I2C2	0 - disabled; 1 - enabled
[27]	rw	1'h1	I2C1	0 - disabled; 1 - enabled
[26]	rw	1'h1	PDM2	0 - disabled; 1 - enabled
[25]	rw	1'h1	PDM1	0 - disabled; 1 - enabled
[24]	rw	1'h1	NNACC	0 - disabled; 1 - enabled
[23]	rw	1'h0	PSRAMC	0 - disabled; 1 - enabled
[22]	rw	1'h1	EXTDMA	0 - disabled; 1 - enabled
[21]	rw	1'h1	SPI2	0 - disabled; 1 - enabled
[20]	rw	1'h1	SPI1	0 - disabled; 1 - enabled
[19]	rw	1'h1	WDT1	0 - disabled; 1 - enabled
[18]	rw	1'h1	BTIM2	0 - disabled; 1 - enabled
[17]	rw	1'h1	BTIM1	0 - disabled; 1 - enabled
[16]	rw	1'h1	GPTIM2	0 - disabled; 1 - enabled
[15]	rw	1'h1	GPTIM1	0 - disabled; 1 - enabled
[14]	rw	1'h1	TRNG	0 - disabled; 1 - enabled
[13]	rw	1'h1	CRC	0 - disabled; 1 - enabled
[12]	rw	1'h1	AES	0 - disabled; 1 - enabled
[11]	rw	1'h1	EFUSEC	0 - disabled; 1 - enabled
[10]	rw	1'h1	SYSCFG1	0 - disabled; 1 - enabled
[9]	rw	1'h1	I2S2	0 - disabled; 1 - enabled
[8]	rw	1'h1	I2S1	0 - disabled; 1 - enabled
[7]	rw	1'h1	LCDC1	0 - disabled; 1 - enabled
[6]	rw	1'h1	EPIC	0 - disabled; 1 - enabled
[5]	rw	1'h1	EZIP	0 - disabled; 1 - enabled
[4]	rw	1'h1	USART2	0 - disabled; 1 - enabled
[3]	rw	1'h1	USART1	0 - disabled; 1 - enabled
[2]	rw	1'h1	PINMUX1	0 - disabled; 1 - enabled
[1]	rw	1'h1	MAILBOX1	0 - disabled; 1 - enabled
[0]	rw	1'h1	DMAC1	0 - disabled; 1 - enabled
0x0C			**ENR2**	Enable Register 2
[31:9]			RSVD	
[8]	rw	1'h1	I2C3	0 - disabled; 1 - enabled
[7]	rw	1'h1	BUSMON1	0 - disabled; 1 - enabled
[6]	rw	1'h1	USBC	0 - disabled; 1 - enabled
[5]	rw	1'h1	SDMMC2	0 - disabled; 1 - enabled
[4]	rw	1'h1	SDMMC1	0 - disabled; 1 - enabled
[3]	rw	1'h1	QSPI3	0 - disabled; 1 - enabled
[2]	rw	1'h1	QSPI2	0 - disabled; 1 - enabled
[1]	rw	1'h1	QSPI1	0 - disabled; 1 - enabled
[0]	rw	1'h1	GPIO1	0 - disabled; 1 - enabled
0x10			**CSR**	Clock Select Register
[31]	rw	1'h0	FORCE_BUS	for debug only
[30:15]			RSVD	

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14]	rw	1'b0	SEL_USBC	select USB source clock 0 - clk_hpsys; 1 - clk_dll3
[13]	rw	1'h0	SEL_PDM2	0 - clk_hpsys; 1 - clk_dll2
[12]	rw	1'h0	SEL_PDM1	0 - clk_hpsys; 1 - clk_dll2
[11]	rw	1'h0	SEL_USART2	0 - clk_hrc48; 1 - clk_hxt48
[10]	rw	1'h0	SEL_USART1	0 - clk_hrc48; 1 - clk_hxt48
[9:8]	rw	2'h0	SEL_QSPI3	selet QSPI3 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[7:6]	rw	2'h0	SEL_QSPI2	selet QSPI2 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[5:4]	rw	2'h0	SEL_QSPI1	selet QSPI1 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[3:2]	rw	2'h0	SEL_PSRAMC	selet PSRAMC function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[1:0]	rw	2'h0	SEL_SYS	select clk_hpsys source 0 - clk_hrc48; 1 - clk_hxt48; 2 - clk_lp; 3 - clk_dll1
0x14			**CFGR**	Clock Configuration Register
[31:15]			RSVD	
[14:12]	rw	3'b101	PDIV2	pclk2_hpsys = hclk_hpsys / (2^{PDIV2}), by default divided by 32
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk_hpsys = hclk_hpsys / (2^{PDIV1}), by default divided by 2
[7:0]	rw	8'h2	HDIV	hclk_hpsys = clk_hpsys / HDIV if HDIV=0, hclk_hpsys = clk_hpsys
0x18			**I2SR**	I2S Register
[31:5]			RSVD	
[4:0]	rw	5'h2	DIV	I2S clock = pclk_hpsys / DIV. After divider, I2S clock must be 12MHz.
0x1C			**USBCR**	USBC 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.
0x20			**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	

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:2]	rw	4'h0	STG	DLL lock freqency is decided by STG. DLL output frequency is: 0: 192MHz(if OUT_DIV2_EN=1) or 384MHz(if OUT_DIV2_EN=0) 1: 168MHz(if OUT_DIV2_EN=1) or 336MHz(if OUT_DIV2_EN=0) 2: 144MHz(if OUT_DIV2_EN=1) or 288MHz(if OUT_DIV2_EN=0) 3: 120MHz(if OUT_DIV2_EN=1) or 240MHz(if OUT_DIV2_EN=0) 4: 96MHz(if OUT_DIV2_EN=1) or 192MHz(if OUT_DIV2_EN=0)
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x24			**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	
[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: 0: 192MHz(if OUT_DIV2_EN=1) or 384MHz(if OUT_DIV2_EN=0) 1: 168MHz(if OUT_DIV2_EN=1) or 336MHz(if OUT_DIV2_EN=0) 2: 144MHz(if OUT_DIV2_EN=1) or 288MHz(if OUT_DIV2_EN=0) 3: 120MHz(if OUT_DIV2_EN=1) or 240MHz(if OUT_DIV2_EN=0) 4: 96MHz(if OUT_DIV2_EN=1) or 192MHz(if OUT_DIV2_EN=0)
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x28			**DLL3CR**	DLL3 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	

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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: 0: 192MHz(if OUT_DIV2_EN=1) or 384MHz(if OUT_DIV2_EN=0) 1: 168MHz(if OUT_DIV2_EN=1) or 336MHz(if OUT_DIV2_EN=0) 2: 144MHz(if OUT_DIV2_EN=1) or 288MHz(if OUT_DIV2_EN=0) 3: 120MHz(if OUT_DIV2_EN=1) or 240MHz(if OUT_DIV2_EN=0) 4: 96MHz(if OUT_DIV2_EN=1) or 192MHz(if OUT_DIV2_EN=0)
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x2C			**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
0x30			**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
0x34			**DBGCLKR**	Debug Clock Register
[31:28]			RSVD	
[27:26]	rw	2'b0	DLL3_OUT_STR	for debug only
[25]	rw	1'b0	DLL3_CG_EN	for debug only
[24]	rw	1'b0	DLL3_OUT_RSTB	for debug only
[23]	rw	1'b0	DLL3_LOOP_EN	for debug only
[22]	rw	1'b0	DLL3_OUT_EN	for debug only
[21]	rw	1'b0	DLL3_LDO_EN	for debug only
[20]	rw	1'b0	DLL3_DBG	for debug only
[19:18]	rw	2'b0	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'b0	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

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1:0]	rw	2'b0	CLK_SEL	for debug only

2.8 LPSYS Clock Structure

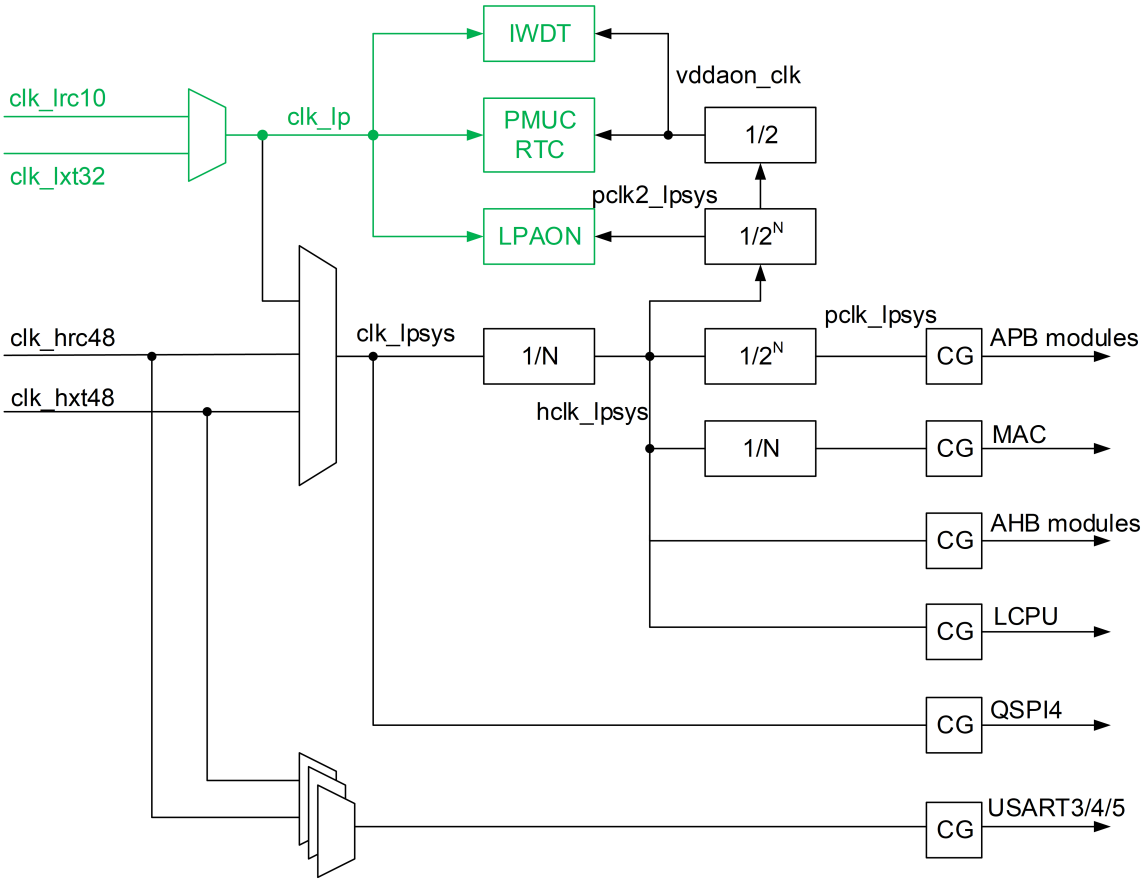


Figure 2-2: LPSYS Clock Structure

The system clock clk_lpsys of LPSYS can be selected from clk_hrc48, clk_hxt48, and clk_lp. The selection register is CSR_SEL_SYS within LPSYS_RCC. The maximum frequency supported by clk_lpsys is 48 MHz.

hclk_lpsys is generated by dividing clk_lpsys by N, with the division ratio specifed by CFGR_HDIV1 in LPSYS_RCC. The maximum frequency supported by hclk_lpsys is 48 MHz; it serves as the operating clock for AHB modules such as LCPU and DMAC2, as well as for the AHB bus and SRAM. CFGR_HDIV2 in LPSYS_RCC is used to adjust the wait cycles of SRAM. When hclk_lpsys exceeds 24 MHz, CFGR_HDIV2 should be set to 1; otherwise, it should be set to 0.

hclk_lpsys is divided by N, with the division ratio specifed by CFGR_MACDIV in LPSYS_RCC, to generate the clock required by the Bluetooth MAC.

pclk_lpsys is generated by dividing hclk_lpsys 1 by 2^N with a division ratio of 2^{CFGR_PDIV1} . The maximum frequency supported by pclk_lpsys is 48 MHz , serving as the operating clock for APB modules such as GPTIM3/4/5 ,BTIM3/4 ,I2C4/5/6 ,SPI3/4 ,GPADC , as well as the APB bus clock. When the frequency of hclk_lpsys or pclk_lpsys changes, the operating clock frequency of modules such as GPTIM3/4/5 , BTIM3/4 , I2C4/5/6 , SPI3/4 , GPADC also changes accord-

ingly, impacting their functionality. Therefore, during the operation of related modules, the frequencies of hclk_lpsys and pclk_lpsys must be kept constant.

pclk2_lpsys is generated by dividing hclk_lpsys by a factor of 1 to 2^N, with a division ratio of 2^{CFGR_PDIV2}. The maximum frequency supported by pclk2_lpsys is 6 MHz , which serves as the register access clock for modules such as LPAON, PMUC, IWDT, and RTC.

The low-power clock clk_lp can be selected from clk_lrc10 and clk_lxt32 , serving as the operating clock for low-power modules including HPAON, LPAON, PMUC, RTC, IWDT, and as the sleep clock for Bluetooth. The clk_lp selection register is CR_SEL_LPCLK in the PMUC module.

The operating clock of QSPI4 is generated by clk_lpsys.

The operating clock of USART3/4/5 can be selected between clk_hrc48 and clk_hxt48. The selection register is CSR_SEL_USART3/4/5 in LPSYS_RCC. The operating clock of USART3/4/5 is independent of the system clock and thus remains unaffected during dynamic system frequency adjustments.

2.9 LPSYS Module Enable

The ENR1 and ENR2 registers in LPSYS_RCC control the enablement of each module. When the corresponding module bit is set to 1, the module registers are accessible and the module can operate. When the bit corresponding to the module is 0, the module's working clock and bus clock are both disabled, the module ceases operation, and its registers become inaccessible; however, the register values are not reset.

2.10 LPSYS Module Reset

The RSTR1 and RSTR2 registers within LPSYS_RCC control the reset of each module. When the bit corresponding to the module is 1, the module's registers and internal state are reset. When the bit is 0, the module reset is deactivated.

2.11 LPSYS_RCC Registers

Table 2-5: LPSYS_RCC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**RSTR1**	Reset Register
[31:30]			RSVD	
[29]	rw	1'h0	BUSMON2	0 - no reset; 1 - reset
[28]	rw	1'h0	LCDC2	0 - no reset; 1 - reset
[27]	rw	1'h0	PTC2	0 - no reset; 1 - reset
[26]	rw	1'h0	TSEN	0 - no reset; 1 - reset
[25]	rw	1'h0	LPCOMP	0 - no reset; 1 - reset
[24]			RSVD	
[23]	rw	1'h0	SDADC	0 - no reset; 1 - reset
[22]	rw	1'h0	GPADC	0 - no reset; 1 - reset
[21]	rw	1'h0	WDT2	0 - no reset; 1 - reset
[20]	rw	1'h0	BTIM4	0 - no reset; 1 - reset
[19]	rw	1'h0	BTIM3	0 - no reset; 1 - reset

Continued on next page...

Table 2-5: LPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[18]	rw	1'h0	GPTIM5	0 - no reset; 1 - reset
[17]	rw	1'h0	GPTIM4	0 - no reset; 1 - reset
[16]	rw	1'h0	GPTIM3	0 - no reset; 1 - reset
[15]	rw	1'h0	SYSCFG2	0 - no reset; 1 - reset
[14]	rw	1'h0	I2C6	0 - no reset; 1 - reset
[13]	rw	1'h0	I2C5	0 - no reset; 1 - reset
[12]	rw	1'h0	I2C4	0 - no reset; 1 - reset
[11]			RSVD	
[10]	rw	1'h0	SPI4	0 - no reset; 1 - reset
[9]	rw	1'h0	SPI3	0 - no reset; 1 - reset
[8]			RSVD	
[7]	rw	1'h0	USART5	0 - no reset; 1 - reset
[6]	rw	1'h0	USART4	0 - no reset; 1 - reset
[5]	rw	1'h0	USART3	0 - no reset; 1 - reset
[4]	rw	1'h0	PATCH	0 - no reset; 1 - reset
[3]	rw	1'h0	PINMUX2	0 - no reset; 1 - reset
[2]	rw	1'h0	MAILBOX2	0 - no reset; 1 - reset
[1]	rw	1'h0	DMAC2	0 - no reset; 1 - reset
[0]	rw	1'h0	LCPU	0 - no reset; 1 - reset
0x04			**RSTR2**	Reset Register2
[31:5]			RSVD	
[4]	rw	1'h0	MAC	0 - no reset; 1 - reset
[3]	rw	1'h0	PHY	0 - no reset; 1 - reset
[2]	rw	1'h0	RFC	0 - no reset; 1 - reset
[1]	rw	1'h0	QSPI4	0 - no reset; 1 - reset
[0]	rw	1'h0	GPIO2	0 - no reset; 1 - reset
0x08			**ENR1**	Enable Register
[31:30]			RSVD	
[29]	rw	1'h1	BUSMON2	0 - disabled; 1 - enabled
[28]	rw	1'h1	LCDC2	0 - disabled; 1 - enabled
[27]	rw	1'h1	PTC2	0 - disabled; 1 - enabled
[26]	rw	1'h1	TSEN	0 - disabled; 1 - enabled
[25]	rw	1'h1	LPCOMP	0 - disabled; 1 - enabled
[24]			RSVD	
[23]	rw	1'h1	SDADC	0 - disabled; 1 - enabled
[22]	rw	1'h1	GPADC	0 - disabled; 1 - enabled
[21]	rw	1'h1	WDT2	0 - disabled; 1 - enabled
[20]	rw	1'h1	BTIM4	0 - disabled; 1 - enabled
[19]	rw	1'h1	BTIM3	0 - disabled; 1 - enabled
[18]	rw	1'h1	GPTIM5	0 - disabled; 1 - enabled
[17]	rw	1'h1	GPTIM4	0 - disabled; 1 - enabled
[16]	rw	1'h1	GPTIM3	0 - disabled; 1 - enabled
[15]	RW	1'h1	SYSCFG2	0 - disabled; 1 - enabled
[14]	rw	1'h1	I2C6	0 - disabled; 1 - enabled
[13]	rw	1'h1	I2C5	0 - disabled; 1 - enabled
[12]	rw	1'h1	I2C4	0 - disabled; 1 - enabled
[11]			RSVD	
[10]	rw	1'h1	SPI4	0 - disabled; 1 - enabled
[9]	rw	1'h1	SPI3	0 - disabled; 1 - enabled
[8]			RSVD	
[7]	rw	1'h1	USART5	0 - disabled; 1 - enabled
[6]	rw	1'h1	USART4	0 - disabled; 1 - enabled

Continued on next page...

Table 2-5: LPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	rw	1'h1	USART3	0 - disabled; 1 - enabled
[4]	rw	1'h1	PATCH	0 - disabled; 1 - enabled
[3]	rw	1'h1	PINMUX2	0 - disabled; 1 - enabled
[2]	rw	1'h1	MAILBOX2	0 - disabled; 1 - enabled
[1]	rw	1'h1	DMAC2	0 - disabled; 1 - enabled
[0]			RSVD	
0x0C			**ENR2**	Enable Register2
[31:5]			RSVD	
[4]	rw	1'h1	MAC	0 - disabled; 1 - enabled
[3]	rw	1'h1	PHY	0 - disabled; 1 - enabled
[2]	rw	1'h1	RFC	0 - disabled; 1 - enabled
[1]	rw	1'h1	QSPI4	0 - disabled; 1 - enabled
[0]	rw	1'h1	GPIO2	0 - disabled; 1 - enabled
0x10			**CSR**	Clock Select Register
[31]	rw	1'h0	FORCE_BUS	for debug only
[30]	rw	1'h0	FORCE_MAC	for debug only
[29:7]			RSVD	
[6]	rw	1'h0	SEL_USART5	0 - clk_hrc48; 1 - clk_hxt48
[5]	rw	1'h0	SEL_USART4	0 - clk_hrc48; 1 - clk_hxt48
[4]	rw	1'h0	SEL_USART3	0 - clk_hrc48; 1 - clk_hxt48
[3:2]			RSVD	
[1:0]	rw	2'h0	SEL_SYS	select clk_lpsys source 0 - clk_hrc48; 1 - clk_hxt48; 2 - clk_lp; 3 - RSVD
0x14			**CFGR**	Clock Configuration Register
[31:24]			RSVD	
[23:19]	rw	5'h8	MACFREQ	clock frequency of MAC clock
[18:16]	rw	3'h3	MACDIV	MAC clock divider MAC clock = hclk_lpsys / MACDIV
[15]			RSVD	
[14:12]	rw	3'b011	PDIV2	pclk2_lpsys = hclk_lpsys / (2^{PDIV2}), by default divided by 8
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk1_lpsys = hclk_lpsys / (2^{PDIV1}), by default divided by 2
[7]			RSVD	
[6]	rw	1'h0	HDIV2	should set to 0 if hclk_lpsys frequency no higher than 24MHz should set to 1 if hclk_lpsys frequency higher than 24MHz
[5:0]	rw	6'h2	HDIV1	hclk_lpsys = clk_lpsys / HDIV1 if HDIV1=0, hclk_lpsys = clk_lpsys

2.12 Measurement and Calibration of Clocks

2.12.1 Measurement and Calibration of clk_hxt48

clk_hxt48 is a clock generated based on an external 48MHz crystal oscillator, and its accuracy mainly depends on the external 48MHz crystal oscillator component. The frequency deviation caused by individual differences in crystal oscillator components is usually on the order of tens of ppm.

High-frequency clocks derived from clk_hxt48 include clk_dll1, clk_dll2, clk_dll3, clk_dbl96, clk_audpll, and Bluetooth RF clocks, whose accuracy is correlated with that of clk_hxt48.

clk_hxt48 does not require calibration during normal chip operation. If the product has higher frequency accuracy re-

quirements, a higher-precision crystal oscillator component can be selected, or a crystal oscillator calibration process can be added to the product's production phase.

The crystal oscillator calibration process mainly includes the following steps:

1. Read the HXT_CR1.CBANK_SEL register of the PMUC;
2. Measure the frequency of clk_hxt48;
3. Increase or decrease the CBANK_SEL register according to the measurement result to decrease or increase the frequency of clk_hxt48;
4. Wait for 100µs to stabilize clk_hxt48;
5. Repeat steps 2~4 until the measurement result meets the accuracy requirement;
6. Record the value of the CBANK_SEL register at this time and restore this value every time the chip starts up subsequently

The main methods for measuring the frequency of clk_hxt48 are as follows:

1. Lead out clk_hxt48 through the PA70 pin for direct measurement. The configuration method is:
   - HPSYS_PINMUX->PAD_PA70.FSEL=0xc;
   - HPSYS_RCC->DBGCLKR=5;
   - This measurement method is difficult to achieve high accuracy.
2. Lead out clk_hxt48 through the PB01 pin for direct measurement. The configuration method is:
   - LPSYS_PINMUX->PAD_PB01=0x9;
   - LPSYS_CFG->DBGR=0x9;
   - This measurement method is difficult to achieve high accuracy.
3. Measure the frequency of the RF carrier signal transmitted by Bluetooth and calculate the frequency of clk_hxt48.
4. Use ATIM or GPTIM with clk_hxt48 as the clock source to perform gating counting on the externally input high-precision PWM signal, and calculate the frequency of clk_hxt48 based on the count value. This method is applicable to the product's production phase.
5. Bluetooth receives the high-precision Bluetooth signal transmitted externally and calculates the frequency of clk_hxt48. This method is applicable to the product's production phase.

2.12.2 Measurement and Calibration of clk_hrc48

clk_hrc48 is an RC oscillator clock internally generated by the chip. Its default frequency is lower than 48MHz without calibration, and it becomes 48MHz after calibration. It is recommended to calibrate clk_hrc48 every time the chip starts up.

The calibration process of clk_hrc48 mainly includes the following steps:

1. Switch the system clock to a clock source other than clk_hrc48;
2. Measure the frequency of clk_hrc48;
3. Increase or decrease the HRC_CR.FREQ_TRIM register of the PMUC according to the measurement result to increase or decrease the frequency of clk_hrc48;
4. Wait for 100µs to stabilize clk_hrc48;
5. Repeat steps 2~4 until the measurement result meets the accuracy requirement;
6. Record the value of the FREQ_TRIM register at this time and restore this value every time the chip restarts subsequently.

The main methods for measuring the frequency of clk_hrc48 are as follows:

1. Lead out clk_hxt48 through the PA70 pin for direct measurement. The configuration method is:
   - HPSYS_PINMUX->PAD_PA70.FSEL=0xc;
   - HPSYS_RCC->DBGCLKR=4;
2. Lead out clk_hxt48 through the PB01 pin for direct measurement. The configuration method is:
   - LPSYS_PINMUX->PAD_PB01=0x9;
   - LPSYS_CFG->DBGR=0xA;
3. Calculate the frequency of clk_hrc48 using clk_hxt48. The configuration method is :
   - HPSYS_RCC->HRCCAL1 = 0x8000; // 0x8000 sets the measurement duration, counted in 48MHz cycles, which can be modified as needed;
   - HPSYS_RCC->HRCCAL1 |= HPSYS_RCC_HRCCAL1_CAL_EN; // Start measurement
   - Wait until HPSYS_RCC->HRCCAL1.CAL_DONE is 1;
   - Read HPSYS_RCC->HRCCAL2.HRC_CNT to obtain the cycle count of clk_hrc48 within the configured measurement duration, and the frequency of clk_hrc48 can be calculated from this.

2.12.3 Measurement of clk_lxt32

clk_lxt32 is a low-power clock generated based on an external 32kHz crystal oscillator, with a frequency of 32.768kHz. Its accuracy mainly depends on the external 32kHz crystal oscillator component. The frequency deviation caused by individual differences in crystal oscillator components is usually on the order of tens of ppm.

The frequency of clk_lxt32 cannot be adjusted, but it can be measured to obtain a more accurate frequency value.

The main methods for measuring the frequency of clk_lxt32 are as follows:

1. Lead out clk_lxt32 through pins PB01 for direct measurement. The configuration method is:
   - LPSYS_PINMUX->PAD_PB01=0x9;
   - LPSYS_CFG->DBGR=0x8;
   - RTC->CR_LPCKSEL=1; //Select clk_lxt32 as clk_rtc
2. Calculate the frequency of clk_lxt32 using clk_hxt48. This measurement must be performed when LPSYS is in active mode and Bluetooth is not working. The configuration method is:
   - RTC->CR_LPCKSEL=1; //Select clk_lxt32 as clk_rtc
   - BT_MAC->RCCAL_CTRL=0x80; // 0x80 sets the measurement duration, counted in clk_lxt32 cycles. It can be modified as needed, with a maximum value of 255;
   - BT_MAC->RCCAL_CTRL |=0x20000; // Start measurement
   - Wait until bit 31 of BT_MAC->RCCAL_RESULT is 1;
   - Read BT_MAC->RCCAL_RESULT. The lower 30 bits are the cycle count of clk_hxt48 within the configured measurement duration, and the frequency of clk_lxt32 can be calculated from this.

2.12.4 Measurement of clk_lrc10

clk_lrc10 is a low-power RC oscillator clock internally generated by the chip, with a frequency of approximately 10kHz. Its frequency cannot be adjusted, but it can be measured to obtain an accurate frequency value.

If clk_rtc is selected as clk_lrc10, it is recommended to periodically measure clk_lrc10 during the chip's operation.

The main methods for measuring the frequency of clk_lrc10 are as follows:

1. Lead out clk_lxt32 through pins PB01 for direct measurement. The configuration method is:

- LPSYS_PINMUX->PAD_PB01=0x9;
- LPSYS_CFG->DBGR=0x8;
- RTC->CR_LPCKSEL=0; //Select clk_lxt32 as clk_rtc

2. Calculate the frequency of clk_lrc10 using clk_hxt48. This measurement must be performed when LPSYS is in active mode. The configuration method is:
   - RTC->CR_LPCKSEL=0; // Select clk_lrc10 as clk_rtc
   - BT_MAC->RCCAL_CTRL = 0x80; // 0x80 sets the measurement duration, counted in clk_lrc10 cycles. It can be modified as needed, with a maximum value of 255;
   - BT_MAC->RCCAL_CTRL |= 0x20000; // Start measurement
   - Wait until bit 31 of BT_MAC->RCCAL_RESULT is 1;
   - Read BT_MAC->RCCAL_RESULT. The lower 30 bits are the cycle count of clk_hxt48 within the configured measurement duration, and the frequency of clk_lrc10 can be calculated from this.

3 Power Management

3.1 Introduction

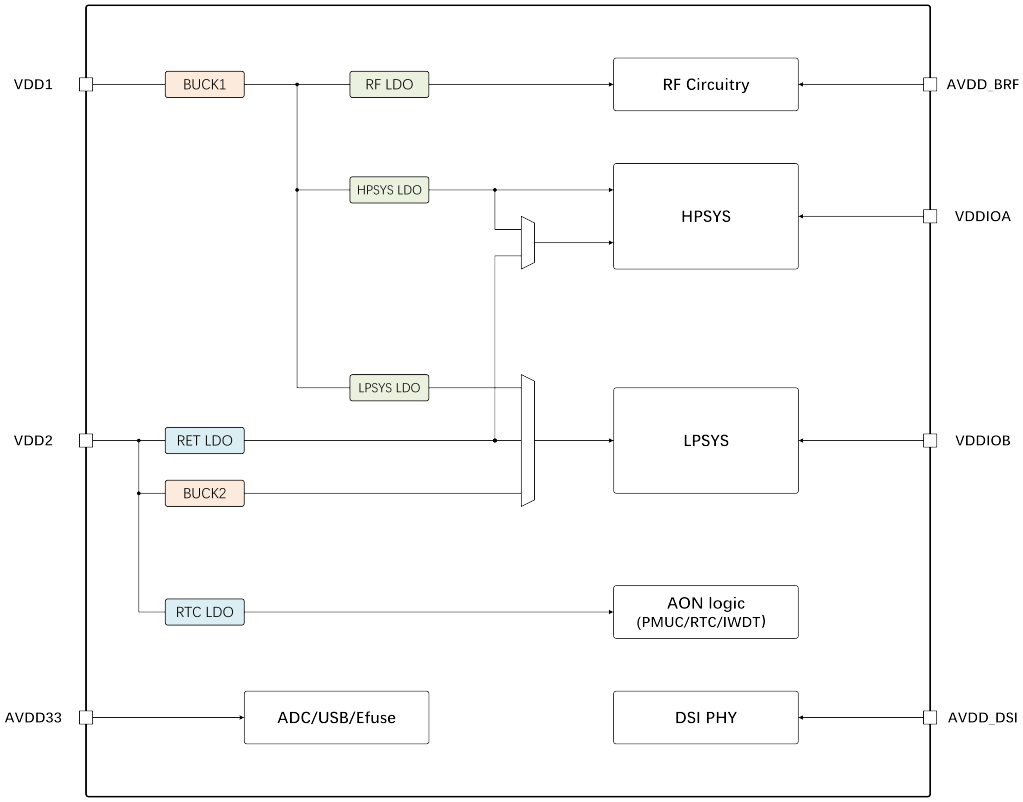


Figure 3-1: Power Management Architecture

3.2 PMUC Registers

Table 3-1: PMUC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31:22]			RSVD	
[21:19]	rw	3'h0	PIN5_MODE	
[18:16]	rw	3'h0	PIN4_MODE	
[15:13]	rw	3'h0	PIN3_MODE	
[12:10]	rw	3'h0	PIN2_MODE	
[9:7]	rw	3'h0	PIN1_MODE	
[6:4]	rw	3'h0	PIN0_MODE	0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
[3]			RSVD	
[2]	rw	1'h0	REBOOT	Write 1 to reboot; write 0 to clear after boot up
[1]	rw	1'h0	HIBER_EN	Write 1 to enter hibernate mode; write 0 to clear when exit from hibernate

Continued on next page...

Table 3-1: PMUC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	SEL_LPCLK	0 - LRC10, 1 - LXT32
0x04			**WER**	Wakeup Enable register
[31:10]			RSVD	
[9]	rw	1'b0	PIN5	
[8]	rw	1'b0	PIN4	
[7]	rw	1'b0	PIN3	
[6]	rw	1'b0	PIN2	
[5]	rw	1'b0	PIN1	
[4]	rw	1'b0	PIN0	
[3:2]			RSVD	
[1]	rw	1'b0	IWDT	Set 1 to enable IWDT as wakeup source
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x08			**WSR**	Wakeup Status register
[31:10]			RSVD	
[9]	r	1'b0	PIN5	
[8]	r	1'b0	PIN4	
[7]	r	1'b0	PIN3	
[6]	r	1'b0	PIN2	
[5]	r	1'b0	PIN1	
[4]	r	1'b0	PIN0	
[3:2]			RSVD	
[1]	r	1'b0	IWDT	Indicates the wakeup status from LPTIM2. 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
0x0C			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:10]			RSVD	
[9]	w1c	1'b0	PIN5	
[8]	w1c	1'b0	PIN4	
[7]	w1c	1'b0	PIN3	
[6]	w1c	1'b0	PIN2	
[5]	w1c	1'b0	PIN1	
[4]	w1c	1'b0	PIN0	Write 1 to clear the PIN0 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[3:0]			RSVD	
0x10			**VRTC_CR**	VRTC Control Register
[31:13]			RSVD	
[12:9]	rw	4'hf	BOR_VT_TRIM	
[8]	rw	1'h0	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	
[30:27]			RSVD	
[26:21]	rw	6'h20	DLY	VRET_LDO power up delay in number of CLK_LP cycles
[20:17]	r	4'h0	CAL_TRIM	
[16]	r	1'h0	CAL_RDY	
[15]	rw	1'h1	TRIM_RSTN	
[14]	rw	1'h0	TRIM_SEL	
[13:10]	rw	4'h7	TRIM	
[9:6]			RSVD	
[5:2]	rw	4'h7	VBIT	
[1]	rw	1'h0	BM	

Continued on next page...

Table 3-1: PMUC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h1	EN	
0x18			**LRC_CR**	RC10K Control Register
[31:9]			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			**LXT_CR**	XTAL32K Control Register
[31]	r	1'h0	RDY	
[30:15]			RSVD	
[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'h1	RSN	
[0]	rw	1'h1	EN	
0x20			**BG1_CR**	BG1 Control Register
[31:11]			RSVD	
[10:9]	rw	2'b01	BG1_DLY	Bandgap power up delay in CLK_LP cycles
[8:5]	rw	4'hd	BG1_VREF12	
[4:1]	rw	4'hd	BG1_VREF06	
[0]	rw	1'h0	BG1_EN	Force BG1 enable
0x24			**BG2_CR**	BG2 Control Register
[31:11]			RSVD	
[10:9]	rw	2'b01	BG2_DLY	Bandgap power up delay in CLK_LP cycles
[8:5]	rw	4'h6	BG2_VREF12	
[4:1]	rw	4'h6	BG2_VREF06	
[0]	rw	1'h0	BG2_EN	Force BG2 enable
0x28			**BUCK1_CR**	BUCK1 Control Register 1
[31:27]			RSVD	
[26]	r	1'h0	BUCK1_RDY	
[25]	rw	1'h1	BUCK1_ZCD_AON	
[24:22]	rw	3'h5	BUCK1_BM_ZCD	
[21:19]	rw	3'h4	BUCK1_BM_PWMCMP	
[18:16]	rw	3'h3	BUCK1_BM_COTCMP	
[15:13]	rw	3'h3	BUCK1_MOT	
[12]	rw	1'h0	BUCK1_SEL_LX22	
[11:8]	rw	4'h5	BUCK1_CS	
[7:4]	rw	4'h7	BUCK1_CCH	
[3:1]	rw	3'h5	BUCK1_ILIMIT	
[0]	rw	1'h1	BUCK1_EN	
0x2C			**BUCK2_CR**	BUCK2 Control Register 1
[31:27]			RSVD	
[26]	r	1'h0	BUCK2_RDY	
[25]	rw	1'h1	BUCK2_ZCD_AON	
[24:22]	rw	3'h5	BUCK2_BM_ZCD	
[21:19]	rw	3'h4	BUCK2_BM_PWMCMP	
[18:16]	rw	3'h3	BUCK2_BM_COTCMP	

Continued on next page...

Table 3-1: PMUC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:13]	rw	3'h3	BUCK2_MOT	
[12]	rw	1'h0	BUCK2_SEL_LX22	
[11:8]	rw	4'h5	BUCK2_CS	
[7:4]	rw	4'h7	BUCK2_CCH	
[3:1]	rw	3'h5	BUCK2_ILIMIT	
[0]	rw	1'h0	BUCK2_EN	
0x30			**LDOVCC1_CR**	LDOVCC1 Control Register
[31:3]			RSVD	
[2]	r	1'h0	LDOVCC1_RDY	
[1]	rw	1'h1	LDOVCC1_EN_SS	
[0]	rw	1'h1	LDOVCC1_EN	
0x34			**LDOVCC2_CR**	LDOVCC2 Control Register
[31:3]			RSVD	
[2]	r	1'h0	LDOVCC2_RDY	
[1]	rw	1'h1	LDOVCC2_EN_SS	
[0]	rw	1'h0	LDOVCC2_EN	
0x38			**LDO_CR**	LDO Control Register
[31]	rw	1'b0	LDOBG_EN	Force LDOBG enable
[30:28]			RSVD	
[27]	R	1'b0	LPSYS_LDO_RDY	
[26:21]	rw	6'h08	LPSYS_LDO_DLY	
[20:17]	rw	4'h5	LPSYS_LDO_VREF	
[16]	rw	1'b1	LPSYS_LDO_EN	
[15]	r	1'b0	HPSYS_LDO_RDY	
[14:9]	rw	6'h08	HPSYS_LDO_DLY	HPSYS_LDO power up delay in CLK_LP cycles
[8:5]	rw	4'ha	HPSYS_LDO_VREF2	Lower voltage if needed, selected by HPSYS AON
[4:1]	rw	4'ha	HPSYS_LDO_VREF	
[0]	rw	1'b1	HPSYS_LDO_EN	
0x3C			**HPSYS_SWR**	HPSYS Switch Register
[31]	r	1'b0	RDY	
[30:5]			RSVD	
[4]	rw	1'b0	NORET	Cut off power switch entirely during standby. No retention
[3:2]	rw	2'b11	DLY	wait for N cycles before asserting RDY
[1:0]	rw	2'b10	PSW	bit[0] - select RET_LDO; bit[1] - select HPSYS_LDO
0x40			**LPSYS_SWR**	LPSYS Switch Register
[31]	r	1'b0	RDY	
[30:5]			RSVD	
[4:3]	rw	2'b11	DLY	
[2:0]	rw	3'b100	PSW	bit[0] - select RET_LDO; bit[1] - select BUCK2; bit [2] - select LPSYS_LDO
0x44			**PMU_TR**	PMU Test Register
[31:9]			RSVD	
[8:6]	rw	3'h0	PMU_DC_MR	macro select
[5:3]	rw	3'h0	PMU_DC_BR	block select
[2:0]	rw	3'h0	PMU_DC_TR	test point select
0x48			**PMU_RSVD1**	PMU Reserved Register 1
[31:24]	r	8'h0	RESERVE3	
[23:16]	rw	8'h0	RESERVE2	
[15:8]	rw	8'h0	RESERVE1	
[7:0]	rw	8'h0	RESERVE0	
0x4C			**PMU_RSVD2**	PMU Reserved Register 2
[31:24]	r	8'h0	RESERVE3	
[23:16]	rw	8'h0	RESERVE2	

Continued on next page...

Table 3-1: PMUC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:8]	rw	8'h0	RESERVE1	
[7:0]	rw	8'h0	RESERVE0	
0x50			**HXT_CR1**	HXT48 Control Register 1
[31:27]			RSVD	
[26:17]	rw	10'h1ca	CBANK_SEL	
[16]	rw	1'b1	GM_EN	
[15:14]	rw	2'h3	LDO_FLT_RSEL	
[13:10]	rw	4'ha	LDO_VREF	
[9:8]	rw	2'h1	BUF_RF_STR	
[7:6]	rw	2'h1	BUF_DLL_STR	
[5:4]	rw	2'h1	BUF_DIG_STR	
[3]	rw	1'b0	BUF_DLL_EN	
[2]	rw	1'b1	BUF_DIG_EN	
[1]	rw	1'b1	BUF_EN	
[0]	rw	1'b1	EN	
0x54			**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	
0x58			**HXT_CR3**	HXT48 Control Register 3
[31:27]			RSVD	
[26:21]	rw	6'd31	DLY	
[20:11]	rw	10'h32	OPT_IDAC	
[10]	rw	1'b0	OPT_IDAC_EN	
[9:0]	rw	10'h2da	OPT_CBANK_SEL	
0x5C			**HRC_CR**	HRC48 Control Register
[31:24]			RSVD	
[23]	rw	1'b0	DLY	number of cycles for BG ready. 0 - one cycle of CLK_LP; 1 - two cycles of CLK_LP
[22:12]	rw	11'h400	CT	
[11]	rw	1'b0	MODE24M_EN	
[10:9]	rw	2'h1	BM	
[8:5]	rw	4'ha	LDO_VREF	
[4]	rw	1'b0	RST	
[3:2]	rw	2'h1	OUT_STR	
[1]	rw	1'b1	OUT_EN	
[0]	rw	1'b1	EN	
0x60			**RC1M_CR**	RC1M Control Register
[31:24]			RSVD	
[23]	rw	1'h0	RINGOSC_MODE	
[22:19]	rw	4'h4	RINGOSC_BM	
[18]	rw	1'h0	RINGOSC_EN	

Continued on next page...

Table 3-1: PMUC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[17]	rw	1'h0	SEL_SOURCE	
[16]	rw	1'h0	DLY	number of cycles for BG ready. 0 - one cycle of CLK_LP; 1 - two cycles of CLK_LP
[15:14]	rw	2'h1	RSEL	
[13:7]	rw	7'h27	CSEL	
[6:5]	rw	2'h1	BM_COMP	
[4:3]	rw	2'h1	BM_CHG	
[2]	rw	1'b0	RST	
[1]	rw	1'b1	OUT_EN	
[0]	rw	1'b0	EN	
0x64			**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	
0x68			**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	
0x6C			**CAU_RSVD**	CAU 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	

4 Low Power Modes

4.1 Introduction

The chip supports multiple low-power modes to satisfy low-power requirements across various scenarios. According to the task division between the performance processor and the low-power processor, as well as the demands for power consumption and wake-up latency, HPSYS and LPSYS can be configured into distinct low-power modes, each with independently configured wake-up sources, to optimize power consumption.

4.2 Summary of Main Operating Modes

The main operating modes of the chip are listed in the table below. Except for the active mode, all other modes are low-power modes.

Table 4-1: Chip Operating Modes

Mode	Subsys	CPU	Peri	SRAM	IO	HP_AON LP_AON LPTIM	Wake-up Source	Wake-up Time
Active		run	run	Accessible	Flippable	run		
Sleep		stop	run	Accessible	Flippable	run	Any Interrupt	<1us
Deepsleep	HPSYS	stop	stop	Inaccessible, Fully Reserved	Level Hold	run	RTC, Wake-up PIN(PA), LPTIM1, LPSYS, MAILBOX2	~250us
	LPSYS	stop	stop	Inaccessible, Fully Reserved	Level Hold	run	RTC, Wake-up PIN(PB), LPTIM2, HPSYS, MAILBOX1, Bluetooth	~250us
Standby	HPSYS	reset	reset	Inaccessible, Reserved 64KB	Level Hold	run	RTC, Wake-up PIN(PA), LPTIM1,LPSYS, MAILBOX2	~1.5ms
	LPSYS	reset	reset	Inaccessible, Fully Reserved	Level Hold	run	RTC, Wake-up PIN(PB), Bluetooth,LPTIM2, HPSYS, MAILBOX1	~1.5ms
Hibernate		reset	reset	Inaccessible, Not reserved	High impedance	reset	RTC,Wake-up PIN(IO)	>2ms

After entering low-power mode, the clocks and power supply of subsystems may be disabled, and access to certain modules is restricted.

Table 4-2: Access permissions in low-power mode

HPSYS	LPSYS	Main controller	HP_RAM QSPI1~3	HP_AHB HP_APB	LP_RAM QSPI4	LP_AHB LP_APB	PMUC RTC IWDT
active/sleep	active/sleep	HCPU/ACPU DMAC1/DMAC2	√	√	√	√	√
	deepsleep/standby		√	√	x	x	x
active/sleep	active/sleep	LCPU/DMAC3	√	√	√	√	√
deepsleep/standby			x	x	√	√	√

4.2.1 Active Mode

HPSYS and LPSYS can independently operate in Active mode. Within a subsystem in Active mode, the CPU and peripherals operate normally, SRAM is accessible, IO can toggle normally, and resources are accessible by other subsystems. 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 be activated only when the peripheral is operating.

All low-power modes can only be entered from the activemode, and upon exit, the system uniformly returns to the activemode.

4.2.2 Sleep Mode

HPSYS and LPSYS can independently enter Sleep mode. In a subsystem in Sleep mode, the CPU halts instruction execution, peripherals continue to operate normally, SRAM remains accessible, IO can toggle normally, and resources can be accessed by other subsystems.

When the PMR_MODE register of HPSYS_AON is set to 0, HCPU executing the WFI or WFE instruction can cause HPSYS to enter Sleep mode. After the HCPU receives any enabled interrupt request, it exits sleep mode and resumes execution from the point prior to entering sleep mode.

When the PMR_MODE register of LPSYS_AON is 0, the LCPU executes the WFI or WFE instruction to allow LPSYS to enter sleep mode. After the LCPU receives any enabled interrupt request, it exits sleep mode and resumes execution from the point prior to entering sleep mode.

4.2.3 Deepsleep Mode

The HPSYS subsystem and the LPSYS subsystem can independently enter deepsleep mode. In deepsleep mode, most clocks of the subsystem are disabled, retaining only low-power clocks. Both the CPU and peripherals cease operation. SRAM is inaccessible, but its contents are preserved. IO configured as output cannot be toggled, maintaining the level before entering deepsleep mode unchanged. Accessing subsystems in deepsleep mode by other subsystems will cause bus errors.

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

1. Ensure that all tasks of HPSYS peripherals (except LPTIM1) have been completed;
2. HCPU has processed all currently reported interrupts;
3. HCPU disables all interrupt enables via the NVIC register to prevent exceptions such as SysTick, SVCall, and PendSV, and sets PRIMASK to 1;
4. Switch the clock clk_hpsys to clk_hrc48 and disable high-speed clocks such as clk_dll1/2/3;
5. In Set the wake-up source in HPSYS_AON;
6. Set SetISSR_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.

The procedure for LPSYS to enter deepsleep mode is as follows:

1. Ensure that all tasks of LPSYS peripherals (except LPTIM2/3) have been completed;
2. LCPU has processed all currently reported interrupts;
3. LCPU disables all interrupt enables via the NVIC register to prevent exceptions such as SysTick, SVCall, and PendSV,

and sets PRIMASK.1;

4. Switch the clock clk_lpsys to clk_hrc48, and disable high-speed clocks such as clk_dbl96;
5. In Set wake-up sources in LPSYS_AON;
6. Set Set ISSR_LP_ACTIVE in LPSYS_AON to 0;
7. Set Configure the PMR_MODE register of LPSYS_AON to 2;
8. LCPU executes the WFI instruction.

After executing the above procedure, if interrupts remain uncleared causing interrupt or exception requests, entry into deepsleep mode fails, and the CPU will remain In in active mode and continue running. If this occurs, you may retry entering deepsleep mode after handling the current interrupt.

For debugging convenience, writing 0x80000002 to the PMR_MODE register in step 7 above allows ignoring the current interrupt status and forcibly entering deepsleep mode.

When HPSYS is in deepsleep mode, all master control modules of LPSYS, including LCPU, accessing any address space of HPSYS (including HPSYS_AON) will return a bus error.

When LPSYS is in deepsleep mode, all master control modules of HPSYS, including HCPU, accessing any address space of LPSYS (including LPSYS_AON), as well as any address space within the chip's low-power domain (such as PMUC, RTC, IWDT, etc.), will return a bus error.

When HPSYS is in deepsleep mode, the primary wake-up sources include RTC, wake-up PIN (PA), LPTIM1, and wake-up requests from LPSYS and MAILBOX2. Wake-up sources are configured via the WER register of HPSYS_AON.

When LPSYS is in deepsleep mode, the primary wake-up sources include RTC , wake-up PIN (PB) , LPTIM2 , Bluetooth, and wake-up requests from HPSYS and MAILBOX1. Wake-up sources are configured via the WER register of LPSYS_AON.

When a wake-up source is triggered, the subsystem exits deepsleep mode after an initialization period of approximately 250 µs and returns to active mode. CPU , peripherals, and memory states are preserved, and the CPU resumes execution from the position prior to entering deepsleep mode (after the WFI instruction).

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

After HPSYS exits deepsleep mode, the recommended recovery procedure for 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 clear PRIMASK to 0;
3. Within the AON interrupt, query the wake-up source through the WSR register of HPSYS_AON and perform the corresponding handling, then clear the AON interrupt flag via the WCR register of HPSYS_AON ;
4. Set ISSR_HP_ACTIVE of HPSYS_AON to 1;
5. Enable other HCPU interrupts via the NVIC register;
6. If necessary, restart the high-speed clock.。

After LPSYS exits deepsleep mode, the recommended recovery procedure for LCPU is as follows:

1. Set the PMR_MODE register of LPSYS_AON to 0;
2. Enable the AON interrupt via the NVIC register and clear PRIMASK to 0;
3. In Within the AON interrupt, query the wake-up source via the WSR register of LPSYS_AON and perform the corresponding handling, then clear the AON interrupt flag via the WCR register of LPSYS_AON;
4. Set Set ISSR_LP_ACTIVE of LPSYS_AON to 1 ;

5. Enable other LCPU interrupts via the NVIC register;
6. If necessary, restart the high-speed clock.

When the CPU clears the wake-up source flags, it should be noted that only the PIN wake-up flags are cleared via the WCR registers of HPSYS_AON or LPSYS_AON, whereas other wake-up source flags require configuring the corresponding module that generates the wake-up source to clear them.

From After exiting deepsleep mode, the subsystem clock is sourced from clk_hrc48; if switching to another clock is necessary, the required clock must be re-enabled source.

From After exiting deepsleep mode, the register contents of peripherals within the subsystem are not reset, and all peripherals retain the state prior to entering deepsleep mode state.

4.2.4 Standby Mode

HPSYS and LPSYS can independently enter standby mode. In a subsystem operating in standby mode, most clocks and power supplies are disabled, retaining only the clocks and power for low-power modules. CPU and peripherals are reset. SRAM is inaccessible; HPSYS can retain 64KB of SRAM, while LPSYS can retain all TCM and SRAM. Configured output IOs (except PBR) cannot be toggled and maintain the level prior to entering standby mode. Accessing a subsystem in standby mode from other subsystems will result in a bus error.

The procedure for HPSYS to enter standby mode is as follows:

1. Ensure that all tasks of HPSYS peripherals (except LPTIM1) have been completed;
2. Backup the contents of SRAM and peripheral states in the retainable SRAM of HPSYS;
3. HCPU handles all currently reported interrupts;
4. HCPU disables all interrupt enables via the NVIC register to prevent exceptions such as SysTick, SVCall, and PendSV, and sets PRIMASK to 1;
5. Switch the clock clk_hpsys to clk_hrc48;
6. In Set the wake-up source in HPSYS_AON;
7. Set ISSR_HP_ACTIVE of HPSYS_AON to 0;
8. Configure the ANACR register of HPSYS_AON to 1;
9. Configure the PMR_MODE register of HPSYS_AON to 3;
10. HCPU executes the WFI instruction.

The procedure for LPSYS to enter standby mode is as follows::

1. Ensure that all tasks of LPSYS peripherals (except LPTIM2/3) have been completed;
2. Backup peripheral status in the SRAM of LPSYS;
3. The LCPU completes processing the currently reported interrupts;
4. LCPU disables all interrupt enables via the NVIC register to prevent exceptions such as SysTick, SVCall, and PendSV, and sets PRIMASK to 1;
5. The clock clk_lpsys switches to clk_hrc48, disabling high-speed clocks such as clk_dbl96;
6. Set wake-up sources in LPSYS_AON;
7. Set ISSR_LP_ACTIVE of LPSYS_AON to 0;
8. Configure the ANACR register of LPSYS_AON to 3;
9. Configure the PMR_MODE register of LPSYS_AON to 3;
10. The LCPU executes the WFI instruction.

After executing the above procedure, if interrupts remain unmasked causing interrupt or exception requests, the attempt to enter standby mode fails, and the CPU will remain in active mode to continue operation. If this situation occurs, it is possible to retry entering the standby mode after processing the current interrupt.

For debugging convenience, writing 0x80000003 to the PMR_MODE register in step 9 above allows bypassing the current interrupt status and forcibly entering the standby mode.

When HPSYS is in standby mode, all master control modules of LPSYS, including LCPU, accessing any address space of HPSYS (including HPSYS_AON) will return a bus error.

When LPSYS is in standby mode, all master control modules of HPSYS, including HCPU, accessing any address space of LPSYS (including LPSYS_AON), as well as any address space within the chip's low-power domain (such as PMUC, RTC, IWDT, etc.), will return a bus error.

When HPSYS is in standby mode, the primary wake-up sources include RTC, wake-up PIN (PA), LPTIM1, as well as wake-up requests from LPSYS and MAILBOX2. Wake-up sources are configured via the WER registers.

When LPSYS is in standby mode, the primary wake-up sources include RTC, wake-up PIN (PB), LPTIM2, Bluetooth, and wake-up requests from HPSYS and MAILBOX1. Wake-up sources are configured via the WER registers.

When a wake-up source is triggered, the subsystem exits standby mode and returns to active mode after an initialization period of approximately 1.5 ms. The status of both the CPU and peripherals is reset, SRAM is partially retained, the CPU starts execution from address 0 in ROM, and queries the PMR_MODE register of HPSYS_AON or LPSYS_AON to select different execution branches.

When exiting standby mode, the subsystem generates an AON interrupt. After the CPU exits the ROM program, the corresponding recovery procedure should be executed first.

After HPSYS exits standby mode, the recommended recovery procedure for HCPU is as follows:

1. Set the PMR_MODE and ANACR registers of HPSYS_AON to 0;
2. Enable the AON interrupt via the NVIC register;
3. Within the AON interrupt, query the wake-up source through the WSR register of HPSYS_AON and perform the corresponding handling, then clear the AON interrupt flag via the WCR register of HPSYS_AON ;
4. Set ISSR_HP_ACTIVEof HPSYS_AON to 1;
5. Re-enable the high-speed clock;
6. Restore peripheral configuration and SRAM contents from the retainable SRAM of HPSYS;
7. Enable other HCPU interrupts via the NVIC registers.

After LPSYS exits standby mode, the recommended recovery procedure for LCPU is as follows:

1. Set the PMR_MODE and ANACR registers of LPSYS_AON to 0;
2. Enable the AON interrupt via the NVIC register;
3. Within the AON interrupt, query the wake-up source via the WSR register of LPSYS_AON and perform the corresponding handling, then clear the AON interrupt flag via the WCR register of LPSYS_AON;
4. Set ISSR_LP_ACTIVE of LPSYS_AON to 1;
5. If necessary, re-enable the high-speed clock.
6. Restore peripheral configuration from the LPSYS SRAM.
7. Enable other LCPU interrupts via the NVIC register.

When the CPU clears the wake-up source flags, it should be noted that only the PIN wake-up flags are cleared via the

WCR registers of HPSYS_AON or LPSYS_AON, whereas other wake-up source flags require configuring the corresponding module that generates the wake-up source to clear them.

From After exiting standby mode, the subsystem system clock originates from clk_hrc48. If switching to another clock source is required, the desired clock source should be re-enabled.

From After exiting standby mode, except for low-power peripherals (such as LPTIM), all peripherals within the subsystem are reset and must be reconfigured or reinitialized before use. Peripheral states can be backed up in retention-capable SRAM or PSRAM before entering standby mode; the backed-up contents will not be cleared during standby mode transitions.

4.2.5 Hibernate Mode

Hibernate Mode is the chip's lowest power consumption mode. In this mode, most clocks and power supplies of the chip are disabled, retaining only the clocks and power supplies of certain low-power modules. All registers of the CPU, peripherals, HPSYS (including HPSYS_AON), and LPSYS (including LPSYS_AON) are reset. All SRAM contents are lost. IO pins enter a high-impedance state.

The procedure to enter hibernate mode is as follows:

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

After entering hibernate mode, only the PMUC, RTC, and IWDT modules remain operational; the registers of these modules are preserved.

The wake-up sources forhibernatemode include the RTCand the wake-up PIN (PB).

When a wake-up source is triggered, the chip exits hibernate mode after an initialization period (>2 ms) and returns to active mode. HCPU begins execution from the address in ROM0 address. At this point, the CR_HIBER_EN register of the PMUC remains set to 1, enabling the CPU to detect that the current startup is an exit from hibernate mode; the CPU can then perform the corresponding processing and clear CR_HIBER_EN to 0.

From After exiting hibernatemode, the subsystem's system clock is sourced from clk_hrc48. If switching to another clock is required, the desired clock source must be re-enabled.

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

4.2.6 Cross-System Access Method

The HPSYS subsystem and the LPSYS subsystem can independently enter bus access-prohibited deepsleep or standby low-power modes; therefore, specific procedures must be followed during mutual access. The accessed party must implement bus protection mechanisms, while the access initiator must follow the pre-wake-up procedure.

A bus protection mechanism shall be configured when the accessed subsystem enters or exits low-power mode. Specifically, when HPSYS enters deepsleep or standby mode, HCPU shall set ISSR_HP_ACTIVE of HPSYS_AON to 0. Thereafter, all master modules of LPSYS, including LCPU, accessing any address space of HP-SYS (including HPSYS_AON) will receive a bus error. Normal access is only restored after HPSYS exits low-power mode and sets ISSR_HP_ACTIVE back to 1. When LPSYS enters deepsleep or standby mode, LCPU shall set LPSYS_AON of When ISSR_LP_ACTIVE is set to 0, subsequent

accesses by HCPU to any address space of LPSYS (including LPSYS_AON), as well as to any address space within the chip's low-power domain (such as PMUC, RTC, IWDT, etc.), will result in a bus error. Normal access is only possible after LPSYS exits low-power mode and ISSR_LP_ACTIVE is set to 1. This bus error is manifested as a HardFault exception on the CPU. If the aforementioned bus protection mechanism is not enabled, improper cross-system accesses by the main control module may cause unrecoverable exceptions such as bus data errors or bus lock-ups.

To support the pre-wake mechanism, the accessed party must also pre-enable cross-system wake-up sources by setting HPSYS_AON's WER_LP2HP_REQ and LPSYS_AON of WER_HP2LP_REQ to 1.

Subsystems initiating cross-system access must adhere to the pre-wake procedure. An exception occurs when the chip powers on or exits hibernate mode; at this time, both the HPSYS and LPSYS subsystems are in active mode, allowing mutual access to the address space. Otherwise, once the CPU program of the accessed subsystem configures a low-power mode, pre-wake must be performed prior to any cross-system access.

The pre-wake-up procedure for HCPU accessing the LPSYS subsystem and the low-power domain (e.g., PMUC, RTC, IWDT, etc.) is as follows:

1. Set ISSR_HP2LP_REQ of HPSYS_AON to 1;
2. Wait for 1 ms;
3. Check ISSR_LP_ACTIVE of HPSYS_AON; if it is 1, proceed to step 4; otherwise, repeat steps 2 and 3;
4. Begin accessing LPSYS or the low-power domain. Access duration and frequency are unrestricted. During this time, LPSYS will not enter deepsleep or standby mode;
5. After access is complete, clear ISSR_HP2LP_REQ of HPSYS_AON to 0, allowing LPSYS to enter deepsleep or standby mode.

The pre-wake-up procedure for the LCPU to access the HPSYS subsystem is as follows:

1. Set ISSR_LP2HP_REQ of LPSYS_AON to 1;
2. Wait for 1 ms;
3. Check ISSR_HP_ACTIVE of LPSYS_AON; if it is 1, proceed to step 4; otherwise, repeat steps 2 and 3;
4. Begin accessing the HPSYS subsystem. The access duration and frequency are unrestricted. During this period, HPSYS will not enter deepsleep or standby mode;
5. After access is complete, set ISSR_LP2HP_REQ of LPSYS_AON to 0, permitting HPSYS to enter deepsleep or standby mode.

4.2.7 Cross-System Data Interaction Method

When the HPSYS Subsystem and the LPSYS Subsystem perform data interaction, in addition to access through the pre-wake-up process, a cross-system data interaction mechanism based on the MAILBOX can also be employed system data interaction mechanism.

To support the cross-system data interaction mechanism, the accessed party must pre-enable the MAILBOX wake-up source by setting WER_LP2HP_IRQ of HPSYS_AON and WER_HP2LP_IRQ of LPSYS_AON to 1 .

The cross-system data interaction procedure is as follows:

1. The accessing party prepares the data signaling and stores it in the SRAM of its own system;
2. The accessing party configures the MAILBOX module of its own system, which issues a cross-system wake-up signal via the MAILBOX , and then may enter sleep mode;
3. After the recipient receives the MAILBOX interrupt, it obtains data signaling from the initiator's SRAM.

4. The recipient parses the signaling content, reads data from or writes data to the initiator's SRAM, and sets the data completion flag. Subsequently, it may re-enter low-power mode.
5. The initiator monitors the data completion flag and terminates the access.

4.2.8 Debugger Behavior in Low-power Mode

After the chip enters low-power mode, debugging becomes more difcult, primarily manifested as the debugger disconnecting or being unable to connect. By default, the debugger connects to the HCPU via two IOs, PB31 and PB34. When the register PMR_FORCE_LCPU of LPSYS_AON is set to 1, the connection switches to the LCPU. When the chip undergoes a reset or exits hibernate mode, causing the registers of LPSYS_AON to reset, the debugger will reconnect to the HCPU .

The debugger connection to HCPU will disconnect and fail to reconnect under the following conditions:

1. The debugger switches to LCPU;
2. The IO function of PB31 or PB34 is modifed;
3. LPSYS is in deepsleep or standby mode;
4. The ANACR_PB_ISO register of LPSYS_AON is set to 1 (configured by LCPU when LPSYS enters low-power mode);
5. HPSYS is in deepsleep or standby mode;
6. The chip is in hibernate mode.

When debugging HCPU, to prevent the debugger from being affected by LPSYS entering or exiting low-power mode, it is recommended to keep LPSYS in active or sleep mode.HPSYS after waking from deepsleep or standby mode and setting ISSR_HP_ACTIVE of HPSYS_AON to 1 , the debugger can reconnect to HCPU and access the HPSYS address space.

When the debugger is connected to LCPU, the following conditions will cause disconnection and prevent reconnection:

1. The debugger switches to HCPU;
2. The IO function of PB31 or PB34 is modifed;
3. LPSYS is in deepsleep or standby mode;
4. The ANACR_PB_ISO register of LPSYS_AON is set to 1 (configured by LCPU when LPSYS enters low-power mode);
5. The chip is in hibernate mode.

After LPSYS wakes from deepsleep or standby mode, and ANACR_PB_ISO of LPSYS_AON is set to 0 , and ISSR_LP_ACTIVE is set to 1 , The debugger can reconnect to LCPUand access the LPSYSaddress space.

4.2.9 Determination of the Current Low-power Mode

The current low-power mode can be determined by measuring the voltage at the chip' s power pins. When HPSYS is in active, sleep, or deepsleep mode, the LDO1_VOUT voltage remains at 1.1V. When HPSYS is in standby mode, the LDO1_VOUT voltage cannot be maintained and gradually decreases to 0V. When LPSYS is in active, sleep, or deepsleep mode, the LDO2_VOUT or BUCK2_VOUT voltage remains at 0.9V. When LPSYS is in standby mode, the voltage of LDO2_VOUT or BUCK2_VOUT cannot be maintained and will gradually drop to 0V. When the chip enters hibernate mode, the voltages of LDO1_VOUT, LDO2_VOUT, BUCK2_VOUT, and VDD_RET all drop to 0.

Table 4-3: Power Pin Voltages in Low-power Mode

HPSYS	LPSYS	LDO1_VOUT	LDO2_VOUT or BUCK2_VOUT	VDD_RET	VDD_RTC
active/sleep/deepsleep	active/sleep/deepsleep	1.1V	0.9V	0.7V	1.1V
	standby	1.1V	0V	0.7V	1.1V
standby	active/sleep/deepsleep	0V	0.9V	0.7V	1.1V
	standby	0V	0V	0.7V	1.1V
hibernate		0V	0V	0V	1.1V
power off		0V	0V	0V	0V

4.3 HPSYS_AON Registers

Table 4-4: HPSYS_AON Register Map

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
[30:2]			RSVD	
[1:0]	rw	2'h0	MODE	Power Mode: 2'h0 - active/idle; 2'h1 - light sleep; 2'h2 - deep sleep; 2'h3 - standby
0x04			**CR**	Control Register
[31:13]			RSVD	
[12]	rw	1'h0	GTIM_EN	Enable global timer
[11:9]	rw	3'h0	PIN3_MODE	mode for wakeup PIN3 (PA80)
[8:6]	rw	3'h0	PIN2_MODE	mode for wakeup PIN2 (PA79)
[5:3]	rw	3'h0	PIN1_MODE	mode for wakeup PIN1 (PA78)
[2:0]	rw	3'h0	PIN0_MODE	mode for wakeup PIN0 (PA77) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x08			**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
[29:6]			RSVD	
[5]	rw	1'b0	DS_ULPMEM	for debug only
[4]	rw	1'b0	PD_LPMEM	for debug only
[3]	rw	1'b0	EXTPWR_REQ	Request power for LPSYS during Active mode
[2]	rw	1'b1	PWR_REQ	Request power for HPSYS 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
0x0C			**LSCR**	Light Sleep Ctrl Register
[31:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	Request power for LPSYS during Light Sleep mode
[2]	rw	1'b1	PWR_REQ	Request power for HPSYS 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
0x10			**DSCR**	Deep Sleep Ctrl Register
[31:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	Request power for LPSYS during Deep Sleep mode
[2]	rw	1'b1	PWR_REQ	Request power for HPSYS 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

Continued on next page...

Table 4-4: HPSYS_AON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x14			**SBCR**	Standby Mode Ctrl Register
[31:4]			RSVD	
[3]	rw	1'b0	EXTPWR_REQ	Request power for LPSYS during Standby mode
[2]	rw	1'b0	PWR_REQ	Request power for HPSYS 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
0x18			**WER**	Wakeup Enable register
[31:10]			RSVD	
[9]	rw	1'b0	LP2HP_IRQ	Set 1 to enable MAILBOX2 as wakeup source
[8]	rw	1'b0	LP2HP_REQ	Set 1 to enable LPSYS request as wakeup source
[7:6]			RSVD	
[5]	rw	1'b0	PIN3	Set 1 to enable PA80 as wakeup source
[4]	rw	1'b0	PIN2	Set 1 to enable PA79 as wakeup source
[3]	rw	1'b0	PIN1	Set 1 to enable PA78 as wakeup source
[2]	rw	1'b0	PIN0	Set 1 to enable PA77 as wakeup source
[1]	rw	1'b0	LPTIM1	Set 1 to enable LPTIM1 as wakeup source
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x1C			**WSR**	Wakeup Status register
[31:10]			RSVD	
[9]	r	1'b0	LP2HP_IRQ	Indicates the wakeup status from MAILBOX2. Note: the status is masked by WER
[8]	r	1'b0	LP2HP_REQ	Indicates the wakeup status from LPSYS request. Note: the status is masked by WER
[7:6]			RSVD	
[5]	r	1'b0	PIN3	Indicates the wakeup status from PA80 request. Note: the status is masked by WER
[4]	r	1'b0	PIN2	Indicates the wakeup status from PA79 request. Note: the status is masked by WER
[3]	r	1'b0	PIN1	Indicates the wakeup status from PA78 request. Note: the status is masked by WER
[2]	r	1'b0	PIN0	Indicates the wakeup status from PA77 request. Note: the status is masked by WER
[1]	r	1'b0	LPTIM1	Indicates the wakeup status from LPTIM1. 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
0x20			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:6]			RSVD	
[5]	w1c	1'b0	PIN3	Write 1 to clear the PA80 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[4]	w1c	1'b0	PIN2	Write 1 to clear the PA79 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[3]	w1c	1'b0	PIN1	Write 1 to clear the PA78 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[2]	w1c	1'b0	PIN0	Write 1 to clear the PA77 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[1:0]			RSVD	
0x24			**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

Continued on next page...

Table 4-4: HPSYS_AON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x28			**ANACR**	Analog Control Register
[31:1]			RSVD	
[0]	rw	1'b0	PA_ISO	Set 1 to force IO(PA) into retention mode
0x2C			**GTIMR**	Global Timer Register
[31:0]	r	32'h0	CNT	Global timer value
0x30			**RESERVE0**	Reserved Register 0
[31:0]	rw	32'b0	DATA	for debug only
0x34			**RESERVE1**	Reserved Register 1
[31:0]	rw	32'b0	DATA	for debug only

4.4 LPSYS_AON Registers

Table 4-5: LPSYS_AON Register Map

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'b1	CPUWAIT	Stall CPU out of reset. Should be cleared before LCPU run
[29:2]			RSVD	
[1:0]	rw	2'h0	MODE	Power Mode: 2'h0 - active/idle; 2'h1 - light sleep; 2'h2 - deep sleep; 2'h3 - standby
0x04			**CR**	Control Register
[31:19]			RSVD	
[18]	rw	1'h0	GTIM_EN	Enable global timer
[17:15]	rw	3'h0	PIN5_MODE	mode for wakeup PIN5 (PB48)
[14:12]	rw	3'h0	PIN4_MODE	mode for wakeup PIN4 (PB47)
[11:9]	rw	3'h0	PIN3_MODE	mode for wakeup PIN3 (PB46)
[8:6]	rw	3'h0	PIN2_MODE	mode for wakeup PIN2 (PB45)
[5:3]	rw	3'h0	PIN1_MODE	mode for wakeup PIN1 (PB44)
[2:0]	rw	3'h0	PIN0_MODE	mode for wakeup PIN0 (PB43) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x08			**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
[29:15]			RSVD	
[14]	rw	1'b0	DS_CACHE	for debug only
[13]	rw	1'b0	DS_DTCM	for debug only
[12]	rw	1'b0	DS_ITCM	for debug only
[11]	rw	1'b0	DS_RAM5	for debug only
[10]	rw	1'b0	DS_RAM4	for debug only
[9]	rw	1'b0	DS_RAM3	for debug only
[8]	rw	1'b0	DS_RAM2	for debug only
[7]	rw	1'b0	DS_RAM1	for debug only
[6]	rw	1'b0	DS_RAM0	for debug only
[5]			RSVD	
[4]	rw	1'b0	EXTPWR_REQ	Request power for HPSYS during Active mode
[3]	rw	1'b1	PWR_REQ	Request power for LPSYS during Active mode
[2]	rw	1'b1	HXT48_REQ	Request hxt48 in active mode
[1]	rw	1'b1	HRC48_REQ	Request hrc48 in active mode
[0]			RSVD	

Continued on next page...

Table 4-5: LPSYS_AON Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x0C			**LSCR**	Light Sleep Ctrl Register
[31:5]			RSVD	
[4]	rw	1'b0	EXTPWR_REQ	Request power for HPSYS during Light Sleep mode
[3]	rw	1'b1	PWR_REQ	Request power for LPSYS during Light Sleep mode
[2]	rw	1'b1	HXT48_REQ	Request hxt48 in Light Sleep mode
[1]	rw	1'b1	HRC48_REQ	Request hrc48 in Light Sleep mode
[0]			RSVD	
0x10			**DSCR**	Deep Sleep Ctrl Register
[31:5]			RSVD	
[4]	rw	1'b0	EXTPWR_REQ	Request power for HPSYS during Deep Sleep mode
[3]	rw	1'b0	PWR_REQ	Request power for LPSYS during Deep Sleep mode
[2]	rw	1'b0	HXT48_REQ	Request hxt48 in Deep Sleep mode
[1]	rw	1'b0	HRC48_REQ	Request hrc48 in Deep Sleep mode
[0]			RSVD	
0x14			**SBCR**	Standby Mode Ctrl Register
[31:16]			RSVD	
[15:14]	rw	2'b0	PU_DLY	for debug only
[13]	rw	1'h0	PD_DTCM	for debug only
[12]	rw	1'h0	PD_ITCM	for debug only
[11]	rw	1'h0	PD_RAM5	for debug only
[10]	rw	1'h0	PD_RAM4	for debug only
[9]	rw	1'h0	PD_RAM3	for debug only
[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]			RSVD	
[4]	rw	1'b0	EXTPWR_REQ	Request power for HPSYS during Standby mode
[3]	rw	1'b0	PWR_REQ	Request power for LPSYS during Standby mode
[2]	rw	1'b0	HXT48_REQ	Request hxt48 in Standby mode
[1]	rw	1'b0	HRC48_REQ	Request hrc48 in Standby mode
[0]			RSVD	
0x18			**WER**	Wakeup Enable register
[31:13]			RSVD	
[12]	rw	1'b0	HP2LP_IRQ	Set 1 to enable MAILBOX1 as wakeup source
[11]	rw	1'b0	HP2LP_REQ	Set 1 to enable HPSYS request as wakeup source
[10]	rw	1'b0	PIN5	Set 1 to enable PB48 as wakeup source
[9]	rw	1'b0	PIN4	Set 1 to enable PB47 as wakeup source
[8]	rw	1'b0	PIN3	Set 1 to enable PB46 as wakeup source
[7]	rw	1'b0	PIN2	Set 1 to enable PB45 as wakeup source
[6]	rw	1'b0	PIN1	Set 1 to enable PB44 as wakeup source
[5]	rw	1'b0	PIN0	Set 1 to enable PB43 as wakeup source
[4]	rw	1'b0	BLE	Set 1 to enable BLE as wakeup source
[3]	rw	1'b0	LPCOMP2	Set 1 to enable LPCOMP2 as wakeup source
[2]	rw	1'b0	LPCOMP1	Set 1 to enable LPCOMP1 as wakeup source
[1]	rw	1'b0	LPTIM2	Set 1 to enable LPTIM2 as wakeup source
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x1C			**WSR**	Wakeup Status register
[31:13]			RSVD	
[12]	r	1'b0	HP2LP_IRQ	Indicates the wakeup status from MAILBOX1. Note: the status is masked by WER
[11]	r	1'b0	HP2LP_REQ	Indicates the wakeup status from HPSYS request. Note: the status is masked by WER

Continued on next page...

Table 4-5: LPSYS_AON Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	r	1'b0	PIN5	Indicates the wakeup status from PB48 request. Note: the status is masked by WER
[9]	r	1'b0	PIN4	Indicates the wakeup status from PB47 request. Note: the status is masked by WER
[8]	r	1'b0	PIN3	Indicates the wakeup status from PB46 request. Note: the status is masked by WER
[7]	r	1'b0	PIN2	Indicates the wakeup status from PB45 request. Note: the status is masked by WER
[6]	r	1'b0	PIN1	Indicates the wakeup status from PB44 request. Note: the status is masked by WER
[5]	r	1'b0	PIN0	Indicates the wakeup status from PB43 request. Note: the status is masked by WER
[4]	r	1'b0	BLE	Indicates the wakeup status from BLE. Note: the status is masked by WER
[3]	r	1'b0	LPCOMP2	Indicates the wakeup status from LPCOMP2. Note: the status is masked by WER
[2]	r	1'b0	LPCOMP1	Indicates the wakeup status from LPCOMP1. Note: the status is masked by WER
[1]	r	1'b0	LPTIM2	Indicates the wakeup status from LPTIM2. 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
0x20			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:11]			RSVD	
[10]	w1c	1'b0	PIN5	Write 1 to clear the PB48 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[9]	w1c	1'b0	PIN4	Write 1 to clear the PB47 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[8]	w1c	1'b0	PIN3	Write 1 to clear the PB46 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[7]	w1c	1'b0	PIN2	Write 1 to clear the PB45 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[6]	w1c	1'b0	PIN1	Write 1 to clear the PB44 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[5]	w1c	1'b0	PIN0	Write 1 to clear the PB43 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[4:0]			RSVD	
0x24			**ISSR**	Inter System Status Register
[31:6]			RSVD	
[5]	r	1'h1	HP_ACTIVE	read 1 indicates HPSYS is active
[4]	rw	1'h1	LP_ACTIVE	write 1 to indicates LPSYS is active
[3:2]			RSVD	
[1]	r	1'b0	HP2LP_REQ	indicate HPSYS request exists
[0]	rw	1'b0	LP2HP_REQ	write 1 to request HPSYS to stay in active mode
0x28			**DBGMUX**	Debug Mux Register
[31:12]			RSVD	
[11:10]	rw	2'b0	PB48_SEL	for debug only
[9:8]	rw	2'b0	PB47_SEL	for debug only
[7:6]	rw	2'b0	PB46_SEL	for debug only
[5:4]	rw	2'b0	PB45_SEL	for debug only
[3:2]	rw	2'b0	PB44_SEL	for debug only
[1:0]	rw	2'b0	PB43_SEL	for debug only
0x2C			**TARGET**	BLE sleep time target
[31:28]			RSVD	
[27:0]	rw	28'h0	SLEEP_TARGET	ble sleep time target in cycles of clk_lp
0x30			**ACTUAL**	BLE actual sleep time

Continued on next page...

Table 4-5: LPSYS_AON Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:28]			RSVD	
[27:0]	r	28'h0	SLEEP_CNT	ble actual sleep time in cycles of clk_lp. If not woken up by software or external interrupt, sleep_cnt counts up every low power clock cycle, until reaches sleep_target
0x34			**PRE_WKUP**	time before ble awake
[31:26]			RSVD	
[25:16]	rw	10'h18	WKUP_TIME	cycles of clk_lp for LPSYS ready before bt awake.
[15:10]			RSVD	
[9:0]	rw	10'h20	XTAL_TIME	cycles of clk_lp for hxt48 ready before bt awake.
0x38			**SLP_CFG**	ble sleep configuration
[31:4]			RSVD	
[3]	rw	1'h0	XTAL_FORCE_OFF	for debug only
[2]	rw	1'h0	XTAL_ALWAYS_ON	for debug only
[1:0]			RSVD	
0x3C			**SLP_CTRL**	ble sleep control
[31:7]			RSVD	
[6]	r	1'h1	BLE_WKUP	ble wakeup source. 1 means ble has not enter sleep or has enter wakeup procedure
[5]	r	1'h1	XTAL_REQ	xtal request status. 1 means ble is requiring xtal.
[4]	r	1'h0	SLEEP_STATUS	ble sleep status. 1 means ble is sleeping and sleep_cnt is counting up
[3:2]			RSVD	
[1]	w1s	1'h0	WKUP_REQ	software request to wakeup ble. Will be cleared automatically
[0]	w1s	1'h0	SLEEP_REQ	ble sleep request. Will be cleared automatically
0x40			**ANACR**	Analog Control Register
[31:2]			RSVD	
[1]	rw	1'b0	PB_AON_ISO	Set 1 to force off all LPSYS related analog modules
[0]	rw	1'b0	PB_ISO	Set 1 to force IO(PB) into retention mode
0x44			**GTIMR**	Global Timer Register
[31:0]	r	32'h0	CNT	Global timer value
0x48			**RESERVE0**	Reserved Register 0
[31:0]	rw	32'b0	DATA	for debug only
0x4C			**RESERVE1**	Reserved Register 1
[31:0]	rw	32'b0	DATA	for debug only
0x100			**SPR**	Stack Pointer Register
[31:0]	rw	32'h20120000	SP	LCPU stack pointer address
0x104			**PCR**	Pointer Counter Register
[31:0]	rw	32'h00100875	PC	LCPU PC pointer address

5 Input and Output

5.1 Introduction

The chip supports up to 119 configurable IO pins, including 71 general-purpose IOs located in HPSYS (PA00~PA15, PA17~PA38, PA41~PA68, PA70, PA77~PA80), and 48 general-purpose IOs located in LPSYS (PB00~PB41, PB43~PB48). The number of usable IO pins varies depending on the chip package. Each general-purpose IO can independently select input or output functionality and independently configure drive strength as well as pull-up and pull-down resistors. When configured as the GPIO function, each general-purpose IO can trigger an interrupt based on a high or low level or on toggling. Some IOs that support the wake-up PIN function can wake the system from low-power mode.

5.2 IO Structure

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

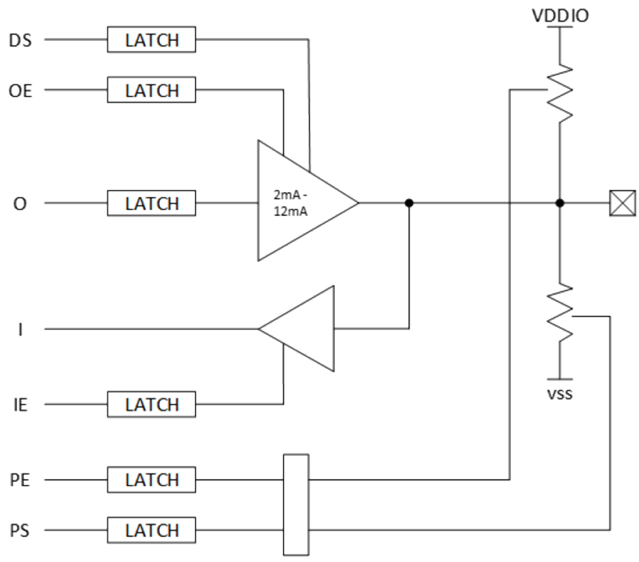


Figure 5-1: IO Structure

DS is used to configure the drive strength, specifed by the DS0 and DS1 registers of the corresponding IO in PINMUX. The values of {DS1, DS0} range from 0 to 3, with the drive strength increasing progressively.

IO output is determined by O and OE and selects different functions according to the FSEL register of the corresponding IO in PINMUX, automatically mapping to the output of the selected function.

IO input enable is controlled by the IE register corresponding to the IO in PINMUX. When IE is high, the input level I is automatically mapped to the input of the corresponding function according to the FSEL register of the IO in PINMUX. When IE is low, the corresponding function cannot acquire the input level of the IO.

PE and PS control the pull-up and pull-down resistors of the IO, as specifed by the PE and PS registers corresponding to

the IO in PINMUX. When PE is 0, both pull-up and pull-down resistors are disabled. When PE is 1 and PS is 0, the pull-down resistor is enabled. When PE is 1 and PS is 1, the pull-up resistor is enabled. The resistance value of the pull-up and pull-down resistors is approximately 10k~40k ohms, depending on the IO external circuitry and interface voltage levels.

5.3 Input and Output Selection

By configuring the PINMUX register corresponding to the IO FSEL field, the input and output of configurable IO can be mapped to one of several functions. The functions to which each IO can be mapped are listed in the GPIO pin list. If the function is input or bidirectional input/output, the corresponding IO IE register must be set to 1.

Each general-purpose IO can be mapped as a GPIO function, in which case the output of the IO is controlled by the GPIO module. If the IO is not mapped as a GPIO function, the IO input cannot be read from the GPIO module, nor will it generate IO interrupts.

5.4 IO High Impedance

After chip power-up, theIO has default pull-up or pull-down resistors (PE default value is 1) ; the IO must be configured to the desired state via software. To configure the IO as high impedance, set PE to 0 , configure the IO as a GPIO function, and disable the GPIO output enable.

5.5 I2C Mode

Certain IO pins can be configured to I2C mode. This mode is optimized for the I2C function, supports higher load current (20mA) , and incorporates a built-in analog filter to suppress glitches. Setting the MODE register of the corresponding IO in PINMUX to 1 enables I2C mode. The I2C mode should only be enabled when the IO is mapped to the I2C function.

IOs that can be configured in I2C mode include:

PA10, PA14, PA79, PA80, PB04, PB05, PB43, PB44, PB45, PB46.

5.6 GPIO Output

When the IO is configured as a GPIO function, the O and OE control signals of the IO are controlled by the GPIO registers, thereby generating the output of the IO. The general-purpose IO (PA) of HPSYS is controlled by the HPSYS_GPIO registers, while the general-purpose IO (PB) of LPSYS is controlled by the LPSYS_GPIO registers. Each bit of the GPIO register corresponds to a general-purpose IO; for example, bit 0 of DOER0 in HPSYS_GPIO corresponds to PA00, and bit 31 corresponds to PA31; Bit 0 of DOER1 corresponds to PA32, and bit 31 corresponds to PA63. Bit 0 of DOER2 corresponds to PA64, and bit 16 corresponds to PA80.

The DOERx registers directly control 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 registers or configure DOESRx or DOECRx for bit operations to avoid affecting other IOs.

The DORx registers directly control the O signal of the IO; when set to 1, the IO output is at a high level, and when set to 0, the IO output is at a low level. The software can directly configure the DORx registers, or configure DOSRx or DOCRx

for bitwise operations to avoid affecting other IO.

The configuration method for GPIOpush-pull output is as follows:

For push-pull output 0, the OE of the IO must be set to 1, O is 0, meaning the corresponding bit of DOERx is set to 1. The corresponding bit of DORx is set to 0.

Push-pull output 1 , the OE of the IO must be set to1 , O is 1, meaning the corresponding bit of DOERx is set to 1 , The corresponding bit of DORx is set to 1.

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

GPIO Open-drain output configuration method is as follows:

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

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

5.7 GPIO Input

Only when the IO is mapped to the GPIO function can the IO input be read from the GPIO register DIRx and generate GPIO interrupts according to the configuration.

IO interrupt enable should only be activated when necessary. Software can directly configure the IERx register or configure IESRx or IECRx for bit operations to avoid affecting other IO pins.

GPIO interrupt generation conditions include IOinput signal being 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	Both edges

ITR is used to select the interrupt condition type. Software can directly configure the ITRx register, or configure ITSRx or ITCRx for bit operations to avoid affecting other IO.

IPH is used to enable high level or rising edge interrupt conditions. Software can directly configure the IPHRx register, or configure IPHSRx or IPHCRx for bit operations to avoid affecting other IO.

IPL is used to enable low-level or falling-edge interrupt conditions. Software can directly configure the IPLRx registers, or configure IPLSRx or IPLCRx for bit operations to avoid affecting other IOs.

The IOs that generate interrupts can be queried through ISRx. Writing 1 to the corresponding bit of ISRx clears the interrupt status.

5.8 Big Core Domain GPIO (PA) List

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	K10	K10	K11	PA00	I/O	PD	0	GPIO_A0
							1	QSPI3_CLK
							Others	Reserved
35	L12	L12	N13	PA01	I/O	PD	0	GPIO_A1
							1	QSPI3_CS
							3	LPTIM1_IN
							4	PDM1_CLK
							5	I2S1_BCK
							11	#USB_DP
							Others	Reserved
-	-	-	L10	PA02	I/O	PD	0	GPIO_A2
							2	PSRAM_DQ0
							Others	Reserved
34	M13	M13	M12	PA03	I/O	PD	0	GPIO_A3
							1	QSPI3_DIO0
							3	LPTIM1_OUT
							4	PDM1_DATA
							5	I2S1_SDI
							11	#USB_DM
							Others	Reserved
-	-	-	H9	PA04	I/O	PD	0	GPIO_A4
							2	PSRAM_DQ1
							Others	Reserved
-	L11	L11	N12	PA05	I/O	PU	0	GPIO_A5
							1	QSPI3_DIO1
							2	PSRAM_DQ2
							3	UART2_RXD
							4	PDM1_DATA
							Others	Reserved
-	L10	L10	H11	PA06	I/O	PU	0	GPIO_A6
							2	PSRAM_DQ3
							Others	Reserved
-	M12	M12	M11	PA07	I/O	PU	0	GPIO_A7
							1	QSPI3_DIO2
							2	PSRAM_CS
							3	UART2_TXD
							4	PDM1_CLK
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	L9	L9	K10	PA08	I/O	PU	0	GPIO_A8
							1	QSPI3_DIO3
							2	PSRAM_CLK
							3	UART2_CTS
							4	PSRAM_CLKB
							5	I2S1_LRCK
							Others	Reserved
-	N13	N13	N11	PA09	I/O	PU	0	GPIO_A9
							2	PSRAM_DQ4
							3	UART2_RTS
							Others	Reserved
33	J10	J10	J10	PA10	I/O	PU	0	GPIO_A10
							3	I2C1_SCL
							Others	Reserved
-	-	-	M10	PA11	I/O	PD	0	GPIO_A11
							2	PSRAM_DQ5
							Others	Reserved
-	-	-	J9	PA12	I/O	PD	0	GPIO_A12
							2	PSRAM_DQ6
							Others	Reserved
-	-	-	N10	PA13	I/O	PD	0	GPIO_A13
							2	PSRAM_DQ7
							Others	Reserved
32	K9	K9	H10	PA14	I/O	PU	0	GPIO_A14
							3	I2C1_SDA
							10	GPTIM2_CH4
							Others	Reserved
-	-	-	J8	PA15	I/O	PU	0	GPIO_A15
							2	PSRAM_DQS0
							Others	Reserved
-	-	-	-	PA16	I/O	PU	0	GPIO_A16
							Others	Reserved
-	N12	N12	N9	PA17	I/O	PD	0	GPIO_A17
							3	UART1_TXD
							9	LCDC1_DPI_CLK
							Others	Reserved
-	-	-	G8	PA18	I/O	PD	0	GPIO_A18
							2	PSRAM_DQ8
							3	SPI2_DO
							4	SPI2_DIO
							7	LCDC1_SPI_DIO1
							9	LCDC1_DPI_DE
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	M11	M11	M8	PA19	I/O	PU	0	GPIO_A19
							3	UART1_RXD
							9	LCDC1_DPI_R0
							Others	Reserved
30	J9	J9	H8	PA20	I/O	PU	0	GPIO_A20
							2	PSRAM_CLK
							3	SPI2_CLK
							7	LCDC1_SPI_CLK
							8	LCDC1_8080_WR
							10	LCDC1_JDI_SCLK
							11	LCDC1_JDI_VCK
							Others	Reserved
-	M10	M10	N8	PA21	I/O	PU	0	GPIO_A21
							5	I2S2_SDO
							9	LCDC1_DPI_R1
							Others	Reserved
-	-	-	L9	PA22	I/O	PU	0	GPIO_A22
							2	PSRAM_DQ9
							5	SD2_DIO0
							6	SD1_DIO6
							7	LCDC1_SPI_CS
							9	LCDC1_DPI_R2
							Others	Reserved
-	N11	N11	L7	PA23	I/O	PU	0	GPIO_A23
							4	PDM2_CLK
							5	I2S2_BCK
							9	LCDC1_DPI_R3
							Others	Reserved
-	-	-	M9	PA24	I/O	PD	0	GPIO_A24
							2	PSRAM_DQ10
							5	SD2_DIO2
							6	SD1_DIO7
							9	LCDC1_DPI_R4
							Others	Reserved
-	M9	M9	J7	PA25	I/O	PU	0	GPIO_A25
							5	I2S2_LRCK
							9	LCDC1_DPI_R5
							Others	Reserved
-	-	-	K8	PA26	I/O	PD	0	GPIO_A26
							2	PSRAM_DQS1
							5	SD2_CLK
							9	LCDC1_DPI_R6
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	N8	N8	H7	PA27	I/O	PD	0	GPIO_A27
							4	PDM2_DATA
							5	I2S2_SDI
							9	LCDC1_DPI_R7
							Others	Reserved
-	-	K8	K7	PA28	I/O	PU	0	GPIO_A28
							2	PSRAM_DQ0
							6	SD1_DIO0
							9	LCDC1_DPI_G0
							Others	Reserved
-	-	M8	H6	PA29	I/O	PU	0	GPIO_A29
							2	PSRAM_DQ1
							6	SD1_DIO1
							9	LCDC1_DPI_G1
							Others	Reserved
-	-	J8	L8	PA30	I/O	PU	0	GPIO_A30
							2	PSRAM_DQ2
							6	SD1_DIO2
							9	LCDC1_DPI_G2
							Others	Reserved
29	N7	N7	J6	PA31	I/O	PU	0	GPIO_A31
							2	PSRAM_DQ3
							6	SD1_DIO3
							7	LCDC1_SPI_CS
							8	LCDC1_8080_CS
							9	LCDC1_DPI_G3
							10	LCDC1_JDI_SCS
							11	LCDC1_JDI_VST
							Others	Reserved
-	-	-	M7	PA32	I/O	PU	0	GPIO_A32
							2	PSRAM_DQ11
							3	SPI2_DI
							7	LCDC1_SPI_DIO0
							9	LCDC1_DPI_G4
							Others	Reserved
-	-	-	N7	PA33	I/O	PU	0	GPIO_A33
							2	PSRAM_DQ12
							3	SPI2_CS
							7	LCDC1_SPI_DIO2
							9	LCDC1_DPI_G5
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
28	L8	L8	J5	PA34	I/O	PU	0	GPIO_A34
							2	PSRAM_DQ4
							6	SD1_CLK
							7	LCDC1_SPI_DIO0
							8	LCDC1_8080_RD
							9	LCDC1_DPI_G6
							10	LCDC1_JDI_SO
							11	LCDC1_JDI_XRST
							Others	Reserved
-	M7	M7	M6	PA35	I/O	PU	0	GPIO_A35
							2	PSRAM_DQS0
							4	PDM1_CLK
							5	I2S1_BCK
							7	LCDC1_SPI_TE
							8	LCDC1_8080_TE
							9	LCDC1_DPI_G7
							Others	Reserved
27	L7	L7	K6	PA36	I/O	PD	0	GPIO_A36
							2	PSRAM_DQ5
							6	SD1_CMD
							7	LCDC1_SPI_DIO1
							8	LCDC1_8080_DC
							9	LCDC1_DPI_B0
							10	LCDC1_JDI_DISP
							11	LCDC1_JDI_HCK
							Others	Reserved
-	M6	M6	L6	PA37	I/O	PU	0	GPIO_A37
							2	PSRAM_CS
							3	SPI2_CLK
							5	I2S1_LRCK
							7	LCDC1_SPI_RSTB
							8	LCDC1_8080_RSTB
							Others	Reserved
26	M5	M5	L5	PA38	I/O	PU	0	GPIO_A38
							2	PSRAM_DQ6
							7	LCDC1_SPI_DIO2
							8	LCDC1_8080_DIO0
							9	LCDC1_DPI_B1
							10	LCDC1_JDI_EXTCOMIN
							11	LCDC1_JDI_HST
							Others	Reserved
-	-	-	-	PA39	I/O	PD	0	GPIO_A39
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	-	-	-	PA40	I/O	PD	0	GPIO_A40
							Others	Reserved
-	L6	L6	M5	PA41	I/O	PD	0	GPIO_A41
							4	SPI2_DI
							9	LCDC1_DPI_B2
							Others	Reserved
25	L5	L5	N5	PA42	I/O	PD	0	GPIO_A42
							2	PSRAM_DQ7
							7	LCDC1_SPI_DIO3
							8	LCDC1_8080_DIO1
							9	LCDC1_DPI_B3
							11	LCDC1_JDI_ENB
							Others	Reserved
-	K6	K6	K5	PA43	I/O	PD	0	GPIO_A43
							9	LCDC1_DPI_B4
							10	GPTIM1_ETR
							Others	Reserved
24	M4	M4	H5	PA44	I/O	PD	0	GPIO_A44
							1	QSPI3_CLK
							4	SPI2_CLK
							5	I2S2_BCK
							6	SD2_CLK
							8	LCDC1_8080_DIO2
							9	LCDC1_DPI_B5
							10	GPTIM1_CH1
							11	LCDC1_JDI_FRP
							Others	Reserved
22	N3	N3	G7	PA45	I/O	PU	0	GPIO_A45
							1	QSPI3_CS
							4	SPI2_CS
							5	I2S2_LRCK
							6	SD2_CMD
							8	LCDC1_8080_DIO3
							9	LCDC1_DPI_B6
							10	GPTIM1_CH2
							11	LCDC1_JDI_XFRP
							Others	Reserved
-	K5	K5	F7	PA46	I/O	PU	0	GPIO_A46
							3	I2C2_SCL
							4	UART1_CTS
							9	LCDC1_DPI_B7
							10	GPTIM1_CH3
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
21	J6	J6	G6	PA47	I/O	PU	0	GPIO_A47
							1	QSPI3_DIO0
							2	QSPI2_DIO4
							4	SPI2_DI
							5	I2S2_SDI
							6	SD2_DIO0
							7	SD1_DIO4
							8	LCDC1_8080_DIO4
							10	GPTIM1_CH4
							11	LCDC1_JDI_VCOM
							Others	Reserved
-	H6	H6	F6	PA48	I/O	PD	0	GPIO_A48
							3	UART2_TXD
							4	UART1_RTS
							5	I2C2_SDA
							Others	Reserved
20	L4	L4	N4	PA49	I/O	PD	0	GPIO_A49
							1	QSPI3_DIO1
							2	QSPI2_DIO5
							3	UART2_RXD
							4	SPI2_DO
							5	SPI2_DIO
							6	SD2_DIO1
							7	SD1_DIO5
							8	LCDC1_8080_DIO5
							11	LCDC1_JDI_R1
							12	UART1_TXD
							Others	Reserved
-	N2	N2	M4	PA50	I/O	PD	0	GPIO_A50
							3	I2C3_SCL
							4	UART1_TXD
							Others	Reserved
19	M3	M3	F5	PA51	I/O	PD	0	GPIO_A51
							1	QSPI3_DIO2
							2	QSPI2_DIO6
							4	UART1_RXD
							6	SD2_DIO2
							7	SD1_DIO6
							8	LCDC1_8080_DIO6
							11	LCDC1_JDI_R2
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	N1	N1	L4	PA52	I/O	PD	0	GPIO_A52
							2	QSPI2_DIO4
							3	SPI1_CLK
							10	GPTIM2_CH1
							Others	Reserved
-	M2	M2	G5	PA53	I/O	PD	0	GPIO_A53
							2	QSPI2_DIO5
							3	SPI1_CS
							4	UART1_TXD
							10	GPTIM2_CH2
							Others	Reserved
-	J5	J5	K4	PA54	I/O	PD	0	GPIO_A54
							2	QSPI2_DIO6
							3	SPI1_DI
							4	UART1_RXD
							10	GPTIM2_CH3
							Others	Reserved
18	H5	H5	J4	PA55	I/O	PD	0	GPIO_A55
							1	QSPI3_DIO3
							2	QSPI2_DIO7
							6	SD2_DIO3
							7	SD1_DIO7
							8	LCDC1_8080_DIO7
							11	LCDC1_JDI_G1
							Others	Reserved
-	K4	K4	L3	PA56	I/O	PD	0	GPIO_A56
							2	QSPI2_DIO7
							3	SPI1_DO
							4	SPI1_DIO
							10	GPTIM2_CH4
							Others	Reserved
-	L3	L3	N3	PA57	I/O	PD	0	GPIO_A57
							3	I2C3_SDA
							10	GPTIM2_ETR
							Others	Reserved
17	M1	M1	L2	PA58	I/O	PD	0	GPIO_A58
							4	LPTIM1_ETR
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	-	-	N2	PA59	I/O	PD	0	GPIO_A59
							2	PSRAM_DQ13
							5	SD2_CMD
							7	LCDC1_SPI_DIO3
							9	LCDC1_DPI_HSYNC
							Others	Reserved
16	L2	L2	M2	PA60	I/O	PU	0	GPIO_A60
							1	QSPI2_CLK
							6	SD1_CLK
							Others	Reserved
15	L1	L1	N1	PA61	I/O	PU	0	GPIO_A61
							1	QSPI2_CS
							6	SD1_CMD
							Others	Reserved
-	-	-	M1	PA62	I/O	PD	0	GPIO_A62
							2	PSRAM_DQ14
							5	SD2_DIO1
							6	SD1_DIO4
							9	LCDC1_DPI_VSYNC
							Others	Reserved
14	H1	H1	H4	PA63	I/O	PD	0	GPIO_A63
							1	QSPI2_DIO0
							3	SPI2_CS
							6	SD1_DIO0
							10	GPTIM1_ETR
							Others	Reserved
-	-	-	H1	PA64	I/O	PD	0	GPIO_A64
							2	PSRAM_DQ15
							5	SD2_DIO3
							6	SD1_DIO5
							9	LCDC1_DPI_SD
							Others	Reserved
13	G2	G2	G4	PA65	I/O	PD	0	GPIO_A65
							1	QSPI2_DIO1
							3	SPI2_DI
							6	SD1_DIO1
							10	GPTIM1_CH1
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
12	G1	G1	F4	PA66	I/O	PD	0	GPIO_A66
							1	QSPI2_DIO2
							3	SPI2_DO
							4	SPI2_DIO
							6	SD1_DIO2
							10	GPTIM1_CH2
							Others	Reserved
-	-	-	G3	PA67	I/O	PD	0	GPIO_A67
							9	LCDC1_DPI_CM
							Others	Reserved
11	F4	F4	G1	PA68	I/O	PD	0	GPIO_A68
							1	QSPI2_DIO3
							6	SD1_DIO3
							10	GPTIM1_CH3
							Others	Reserved
-	-	-	-	PA69	I/O	PD	0	GPIO_A69
							Others	Reserved
10	F3	F3	G2	PA70	I/O	PD	0	GPIO_A70
							2	PSRAM_CLKB
							4	PDM1_DATA
							5	I2S1_SDI
							10	GPTIM1_CH4
							Others	Reserved
-	-	-	-	PA71	I/O	PU	0	GPIO_A71
							Others	Reserved
-	-	-	-	PA72	I/O	PU	0	GPIO_A72
							Others	Reserved
-	-	-	-	PA73	I/O	PU	0	GPIO_A73
							Others	Reserved
-	-	-	-	PA74	I/O	PD	0	GPIO_A74
							Others	Reserved
-	-	-	-	PA75	I/O	PD	0	GPIO_A75
							Others	Reserved
-	-	-	-	PA76	I/O	PD	0	GPIO_A76
							Others	Reserved

Continued on next page...

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

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
8	F2	F2	F3	PA77	I/O	PD	0	GPIO_A77
							1	#WKUP_A0
							3	I2C3_SCL
							4	SPI2_CLK
							7	LCDC1_SPI_TE
							8	LCDC1_8080_TE
							10	GPTIM2_ETR
							11	LCDC1_JDI_G2
							Others	Reserved
7	E2	E2	E3	PA78	I/O	PD	0	GPIO_A78
							1	#WKUP_A1
							3	I2C3_SDA
							4	SPI2_CS
							7	LCDC1_SPI_RSTB
							8	LCDC1_8080_RSTB
							10	GPTIM2_CH1
							11	LCDC1_JDI_B1
							Others	Reserved
6	D2	D2	E4	PA79	I/O	PD	0	GPIO_A79
							1	#WKUP_A2
							3	I2C2_SCL
							4	SPI2_DO
							5	SPI2_DIO
							10	GPTIM2_CH2
							11	LCDC1_JDI_B2
							Others	Reserved
5	C2	C2	E5	PA80	I/O	PD	0	GPIO_A80
							1	#WKUP_A3
							3	I2C2_SDA
							10	GPTIM2_CH3
							Others	Reserved

5.9 Little Core Domain GPIO(PB) Pin List

Table 5-2: Little Core Domain GPIO(PB) Pin List

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	-	-	-	PB00	I/O	PD	0	GPIO_B0
							4	SPI4_CLK
							7	QSPI4_CLK
							Others	Reserved
36	K11	K11	-	PB01	I/O	PU	0	GPIO_B1
							4	SPI4_CS
							5	LPTIM3_IN
							7	QSPI4_CS
							10	#LPCOMP1_P
							Others	Reserved
-	-	-	-	PB02	I/O	PD	0	GPIO_B2
							4	SPI4_DI
							7	QSPI4_DIO0
							Others	Reserved
37	L13	L13	-	PB03	I/O	PD	0	GPIO_B3
							1	GPTIM4_CH1
							4	SPI4_DO
							5	SPI4_DIO
							7	QSPI4_DIO1
							10	#LPCOMP1_N
							Others	Reserved
38	K12	K12	G9	PB04	I/O	PU	0	GPIO_B4
							1	GPTIM4_CH2
							2	I2C4_SCL
							10	#LPCOMP2_P
							Others	Reserved
39	J11	J11	F8	PB05	I/O	PU	0	GPIO_B5
							1	GPTIM4_CH3
							2	I2C4_SDA
							10	#LPCOMP2_N/ATEST
							Others	Reserved
-	H10	H10	F9	PB06	I/O	PU	0	GPIO_B6
							3	UART5_TXD
							4	SPI3_DI
							Others	Reserved
-	-	H9	-	PB07	I/O	PU	0	GPIO_B7
							3	UART5_RXD
							6	LCDC2_SPI_DIO3
							7	QSPI4_DIO3
							Others	Reserved

Continued on next page...

Table 5-2: Little Core Domain GPIO(PB) Pin List (Continued)

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
40	J12	J12	D9	PB08	I/O	PU	0	GPIO_B8
							1	GPTIM4_ETR
							4	SPI3_DO
							5	SPI3_DIO
							10	#GPADC_CH0
							Others	Reserved
-	H11	H11	G10	PB09	I/O	PU	0	GPIO_B9
							1	GPTIM5_CH1
							3	UART3_CTS
							Others	Reserved
41	G11	G11	E10	PB10	I/O	PU	0	GPIO_B10
							2	I2C4_SDA
							5	LPTIM3_OUT
							10	#GPADC_CH1
							Others	Reserved
-	F11	F11	C10	PB11	I/O	PU	0	GPIO_B11
							2	I2C4_SCL
							3	UART5_RXD
							5	LPTIM3_ETR
							Others	Reserved
-	F9	F9	K13	PB12	I/O	PU	0	GPIO_B12
							1	GPTIM3_CH1
							3	UART4_TXD
							4	SPI3_CLK
							10	#GPADC_CH2
							Others	Reserved
42	E9	E9	J11	PB13	I/O	PD	0	GPIO_B13
							1	GPTIM3_CH2
							4	SPI3_CLK
							10	#GPADC_CH3
							Others	Reserved
-	D8	D8	D10	PB14	I/O	PU	0	GPIO_B14
							1	GPTIM3_CH3
							3	UART4_RXD
							4	SPI3_CS
							Others	Reserved
-	C8	C8	F11	PB15	I/O	PU	0	GPIO_B15
							1	GPTIM3_CH4
							3	UART4_CTS
							Others	Reserved

Continued on next page...

Table 5-2: Little Core Domain GPIO(PB) Pin List (Continued)

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
43	D9	D9	G11	PB16	I/O	PU	0	GPIO_B16
							3	UART3_RTS
							4	SPI3_DO
							5	SPI3_DIO
							10	#GPADC_CH4
							Others	Reserved
-	-	F10	J12	PB17	I/O	PU	0	GPIO_B17
							1	GPTIM5_CH2
							3	UART5_RTS
							6	LCDC2_SPI_RSTB
							10	#GPADC_CH5
							Others	Reserved
-	-	E10	J13	PB18	I/O	PU	0	GPIO_B18
							3	UART5_CTS
							6	LCDC2_SPI_TE
							10	#GPADC_CH6
							Others	Reserved
44	C9	C9	F10	PB19	I/O	PD	0	GPIO_B19
							1	GPTIM5_ETR
							4	SPI3_DI
							10	#GPADC_CH7
							Others	Reserved
-	-	-	E11	PB20	I/O	PD	0	GPIO_B20
							1	GPTIM3_ETR
							Others	Reserved
-	-	-	-	PB21	I/O	PU	0	GPIO_B21
							1	GPTIM4_CH4
							7	QSPI4_DIO2
							Others	Reserved
-	D10	D10	C11	PB22	I/O	PU	0	GPIO_B22
							3	UART4_RTS
							Others	Reserved
45	C10	C10	H12	PB23	I/O	PU	0	GPIO_B23
							1	GPTIM5_CH1
							4	SPI3_CS
							10	#SDADC_CH0
							Others	Reserved
46	B12	B12	B12	PB24	I/O	PD	0	GPIO_B24
							1	GPTIM5_CH2
							5	LPCOMP1_OUT
							10	#SDADC_CH1/ACTEST_P
							Others	Reserved

Continued on next page...

Table 5-2: Little Core Domain GPIO(PB) Pin List (Continued)

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
47	A12	A12	A12	PB25	I/O	PD	0	GPIO_B25
							1	GPTIM5_CH3
							5	LPCOMP2_OUT
							10	#SDADC_CH2/ACTEST_N
							Others	Reserved
-	A13	A13	A13	PB26	I/O	PU	0	GPIO_B26
							1	GPTIM5_CH4
							3	UART5_CTS
							10	#SDADC_CH3
							Others	Reserved
-	B13	B13	B13	PB27	I/O	PU	0	GPIO_B27
							3	UART5_RTS
							Others	Reserved
-	B10	B10	B9	PB28	I/O	PU	0	GPIO_B28
							3	UART3_CTS
							Others	Reserved
54	A11	A11	C9	PB29	I/O	PU	0	GPIO_B29
							2	I2C6_SCL
							3	UART3_RTS
							Others	Reserved
-	B11	B11	D8	PB30	I/O	PU	0	GPIO_B30
							2	I2C6_SDA
							Others	Reserved
55	C11	C11	C8	PB31	I/O	PD	0	SWCLK
							1	GPIO_B31
							3	UART3_TXD
							Others	Reserved
-	-	E8	B8	PB32	I/O	PD	0	GPIO_B32
							4	SPI4_CLK
							6	LCDC2_SPI_CLK
							7	QSPI4_CLK
							Others	Reserved
-	-	C7	A8	PB33	I/O	PU	0	GPIO_B33
							4	SPI4_CS
							6	LCDC2_SPI_CS
							7	QSPI4_CS
							Others	Reserved
56	B6	B6	C7	PB34	I/O	PD	0	SWDIO
							1	GPIO_B34
							3	UART3_RXD
							Others	Reserved

Continued on next page...

Table 5-2: Little Core Domain GPIO(PB) Pin List (Continued)

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
-	-	E6	E8	PB35	I/O	PD	0	GPIO_B35
							4	SPI4_DI
							6	LCDC2_SPI_DIO0
							7	QSPI4_DIO0
							Others	Reserved
-	-	D6	D7	PB36	I/O	PD	0	GPIO_B36
							4	SPI4_DO
							5	SPI4_DIO
							6	LCDC2_SPI_DIO1
							7	QSPI4_DIO1
							Others	Reserved
-	-	E5	E7	PB37	I/O	PU	0	GPIO_B37
							3	UART5_TXD
							6	LCDC2_SPI_DIO2
							7	QSPI4_DIO2
							Others	Reserved
-	-	-	B5	PB38	I/O	PD	0	GPIO_B38
							Others	Reserved
-	-	-	B7	PB39	I/O	PU	0	GPIO_B39
							2	I2C5_SCL
							3	UART4_TXD
							Others	Reserved
-	-	-	D4	PB40	I/O	PU	0	GPIO_B40
							2	I2C5_SDA
							3	UART4_RXD
							Others	Reserved
-	-	-	C4	PB41	I/O	PU	0	GPIO_B41
							7	QSPI4_DIO3
							Others	Reserved
-	-	-	-	PB42	I/O	PD	0	GPIO_B42
							1	GPTIM5_ETR
							Others	Reserved
57	C6	C6	B6	PB43	I/O	PU	0	GPIO_B43
							1	GPTIM5_CH3
							2	I2C5_SCL
							10	#WKUP_B0
							Others	Reserved
58	D5	D5	C6	PB44	I/O	PU	0	GPIO_B44
							1	GPTIM5_CH4
							2	I2C5_SDA
							10	#WKUP_B1
							Others	Reserved

Continued on next page...

Table 5-2: Little Core Domain GPIO(PB) Pin List (Continued)

Pin Number				Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB551 QFN68L	SF32LB553 BGA125	SF32LB555 BGA145	SF32LB557 BGA169					
59	C5	C5	E6	PB45	I/O	PU	0	GPIO_B45
							2	I2C6_SCL
							3	UART3_TXD
							10	#WKUP_B2
							Others	Reserved
60	F5	F5	D6	PB46	I/O	PU	0	GPIO_B46
							2	I2C6_SDA
							3	UART3_RXD
							10	#WKUP_B3
							Others	Reserved
61	E4	E4	D5	PB47	I/O	PU	0	GPIO_B47
							3	UART3_CTS
							10	#WKUP_B4
							Others	Reserved
62	D4	D4	C5	PB48	I/O	PU	0	GPIO_B48
							3	UART3_RTS
							10	#WKUP_B5
							Others	Reserved

5.10 Package Integration IO

Package Integration IO is not included in the count of configurable IOs and is dedicated to accessing the package integration (SiP) NOR flash or pSRAM. Package Integration IO shares the same structure as general-purpose IOs, allowing independent configuration of each IO's function and pull-up/down settings. Package Integration IO (SIP) is configured within HPSYS_PINMUX.

IO (SIP) includes SIP00~SIP18 and is used by QSPI1 to access the package integration's 8-line pSRAM or NOR Flash.

When QSPI4 accesses the package integration's 4-line pSRAM, package integration IOs are not used; instead, pins PB00~PB03, PB07, and PB21 are utilized.

5.11 IO Power Supply

The voltage level standard of the IOdepends on its power supply voltage.

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

The power supplies for IO(PA) and IO(PB) are independent.

The 4-wire pSRAM encapsulated within the SF32LB557 chip is powered by VDD_SIP, while its IO power supply is VDDIOB; both power rails should be 1.8V.

IO Category	Power Supply
SIP	VDD_SIP
PA	VDDIOA
PB	VDDIOB

5.12 Wake-up PIN

The chip supports up to 10 wake-up PINs, including 4 located in HPSYS (PA77~PA80), and 6 located in LPSYS (PB43~PB48).PA77~PA80 can wake the HPSYS from standby mode or deepsleep mode, but cannot wake the LPSYS from standby mode or deepsleep mode. PB43~PB48 can wake the LPSYS from standby mode or deepsleep mode, and can also wake the chip from hibernate mode, but cannot wake the HPSYS from standby mode or deepsleep mode. 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 does not depend on the IO function selection; that is, the FSEL register does not affect the wake-up PIN function.

The general-purpose IOs (PB43~PB48, PA77~PA80) supporting the wake-up PIN function have, in addition to the original IO input path, a permanently enabled input path dedicated to the wake-up PIN function. Therefore, regardless of whether the chip is in low-power mode, it is necessary to ensure that these IOs are not in a floating state when not toggled; otherwise, leakage current may occur. When the chip is not in hibernate mode, the pull-up and pull-down resistors can be configured via the PINMUX register to ensure the IO is at a high or low logic level.

When the chip is in hibernate mode, all IO pull-up and pull-down resistor configurations in the PINMUX registers become ineffective. Therefore, it is necessary to ensure via external circuitry that the IOs (PB43~PB48, PA77~PA80) supporting the wake-up PIN function maintain a defined high or low level; otherwise, leakage current may occur.

5.13 IO Status in Low-power Mode

In active and sleep modes, HPSYS IO(PA) and IO(SIP) operate normally, outputs can toggle correctly, and IO interrupts can be generated.

After HPSYS enters standby or deepsleep modes, IO(PA) and IO(SIP) retention mode configurations—including input enable, output enable, and pull-up/pull-down resistors—remain unchanged; outputs hold their previous state, stop toggling, and IO interrupts cannot be generated. IO (PA) supporting wake-up PIN functionality can generate PIN wake-up. The operation of IO (PB) is not affected.

In active mode and sleep mode, LPSYS allows IO (PB) to operate normally; output can toggle normally, and IO interrupts can be generated.

After LPSYS enters standby or deepsleep mode, IO (PB) enters retention mode; input enable, output enable, and pull-up/pull-down resistor configurations remain unchanged, output holds the previous value, toggling stops, and IO interrupts cannot be generated. IO (PB) supporting wake-up PIN functionality can generate PIN wake-up. The operation of IO (PA) is not affected.

After the chip enters hibernate mode, all IOs enter shutdown mode; input enable and output enable are disabled, pull-up and pull-down resistors are turned off, and the IOs exhibit high impedance, preventing generation of IO interrupts. IOs (PA) and IOs (PB) that support the wake-up PIN function can generate PIN wake-up events.

Table 5-3: IO Operating Status

IO	Function	active	sleep	deepsleep	standby	hibernate
PA00~PA70 PB00~PB41	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Flippable	Flippable	Hold	Hold	High impedance
	Pull-up/Pull-down Resistor	Enabled	Enabled	Hold	Hold	Invalid
	GPIO Interrupt	Present	Present	None	None	None
	PIN Wake-up	None	None	None	None	None
PA77~PA80 PB43~PB48	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Flippable	Flippable	Hold	Hold	High impedance
	Pull-up/Pull-down Resistor	Enabled	Enabled	Hold	Hold	Invalid
	GPIO Interrupt	Present	Present	None	None	None
	PIN Wake-up	Present	Present	Present	Present	Present
SIP00~SIP18	Input Enable	Active	Active	Hold	Hold	Invalid
	Output Enable	Active	Active	Hold	Hold	Invalid
	Output Level	Flippable	Flippable	Hold	Hold	High impedance
	Pull-up/Pull-down Resistor	Enabled	Enabled	Hold	Hold	Invalid
	GPIO Interrupt	None	None	None	None	None
	PIN Wake-up	None	None	None	None	None

5.14 Avoiding IO Leakage Current

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

- Short-circuit leakage. An IO outputting a high level IO short-circuited through external circuitry of the chip with another IO outputting a low level IO (possibly from another device) will generate a large current, which may cause chip damage in severe cases.
- Output conduction leakage. When an IO outputs a high level, the presence of a pull-down resistor inside or outside the chip generates conduction current. When an IO outputs a low level, the presence of a pull-up resistor inside or outside the chip generates conduction current.
- Pull-up and pull-down resistor conduction leakage. A certain IO has both pull-up and pull-down resistors inside or outside the chip simultaneously, generating conduction current.
- Input intermediate-level leakage current. When the IO input path is conducting, the input terminal is at an intermediate-level voltage, causing conduction current within the IO internally. This type of leakage current is relatively concealed and may manifest as fluctuating leakage current.

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

For the third type of leakage, the chip's external circuitry should be inspected to ensure no conflict exists between pull-up and pull-down resistors.

For the fourth type of leakage, the following configuration method should be adopted:

- For general-purpose IOs without input functionality, disable input enable (IE set to 0).
- General-purpose IOwith input functionality requires configuration of pull-up or pull-down resistors or ensuring the

input level is high or low via external circuitry.

- General-purpose IO (PB43~PB48, PA77~PA80) supporting wake-up PIN function requires ensuring the input level is high or low via external circuitry.

If IO leakage is suspected, the following checks can be performed on each IO.

Table 5-4: IO Leakage Check Table

IO	If configured as a GPIO function with output enable activated, or configured as the output signal of other modules	When input enable is active	When both input enable and output enable are active When both are disabled
PA00~PA70 PB00~PB41	Verify whether the output value matches the pull up or pull-down configuration in the PINMUX. Verify whether the output value matches the pull-up and pull-down configuration of the external circuit. Verify if a high level is output while the external device connected to the IO is powered off.	Verify whether the external device connected to the IO can provide a defined logic level. Verify whether the PINMUX is configured with appropriate pull-up and pull-down settings.	Verify the pull-up and pull-down configuration of the PINMUX. Verify whether the logic levels conflict with those of the external circuit.
PA77~PA80 PB43~PB48	Verify whether the output value matches the pull-up and pull-down configuration of the PINMUX. Verify whether the output value matches the pull-up and pull-down configuration of the external circuit. Verify if a high level is output while the external device connected to the IO is powered off.	Verify whether the external device connected to the IO can provide a defined logic level. Verify whether the PINMUX is configured with appropriate pull-up and pull-down settings.	Verify whether the external device connected to the IO can provide a defined logic level. Verify whether the pull-up and pull-down configuration of the PINMUX conflicts with the logic levels of the external circuit.

The chip provides numerous IOs. When it is unclear which IOs are leaking current, all general-purpose IOs (PB00~PB48, PA00~PA80) can first be configured in the PINMUX as non-input high-impedance state (IE=0, PE=0, OE=0), ensuring that all general-purpose IOs (PB43~PB48, PA77~PA80) supporting wake-up PIN functionality have correct logic levels. Set appropriate pull-up and pull-down resistors in the RTC registers. Under this condition, the chip's IOs will not cause leakage current. Restore the IO configuration in batches, identify the IO causing leakage current, and eliminate the leakage factors.

In Hibernate mode, leakage current in the chip IO occurs only on the wake-up PIN; therefore, only the IO configuration of the wake-up PIN needs to be inspected.

5.15 Avoid leakage current in package-integrated IO

The power supply for package-integrated IO also powers the corresponding package-integrated memory; thus, leakage current in the package-integrated memory will also appear on the power supply of the package-integrated IO. The common types of package-integrated IO leakage current are as follows:

1. The chip enters low-power mode, but the package-integrated memory does not enter a low-power state. The

embedded NOR flash or pSRAM consumes current even in an idle standby state without access (dependent on the memory model, typically on the order of 10~200 µA). This leakage current becomes more pronounced after the chip enters lowpower mode. Therefore, it is recommended that before the chip enters deepsleep or standby mode, the embedded memory be placed into a low-power state (leakage approximately on the order of 1~10 µA), and after the chip wakes up, the embedded memory should be brought out of the low-power state. The embedded pSRAM can enter half sleep mode via commands, and the embedded NOR flash can enter deep power down mode via commands. These modes can signifcantly reduce the memory leakage current.

2. System-in-Package (SiP) IO uncontrollably causes leakage current in the SiP memory. When the chip enters hibernate mode, the SiP IO exhibits a high-impedance floating level. If the power supply to the SiP IO remains active at this time, it may cause the SiP memory to enter a high leakage current state (for example, if the CS pin is floating at a low or intermediate voltage level, leakage current ranging from hundreds of uA to several mA may occur). Therefore, before the chip enters hibernate mode, the power supply to the SiP IO should be turned off to prevent leakage current in the SiP memory. The chip reserves PA58 for controlling the package power supply. When the chip powers on or wakes up from hibernate , PA58 automatically outputs a high level; after entering hibernate mode, it switches to high impedance. PA58 can be used to enable the external power switch, thereby controlling the on/off state of the package power supply to prevent leakage current. If the power supply for the package IO can only maintain constant power without control, the chip's hibernate mode should be avoided; instead, the standby mode can be used as an alternative.

5.16 Known Issues

There are several limitations when using PA01 ; it should be avoided in low-power applications and configured as non-input high impedance. If its use is necessary, please contact after-sales support for further details.

5.17 HPSYS_PINMUX Registers

Table 5-5: HPSYS_PINMUX Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**PAD_SIP00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x04			**PAD_SIP01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x08			**PAD_SIP02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x0C			**PAD_SIP03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x10			**PAD_SIP04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x14			**PAD_SIP05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x18			**PAD_SIP06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x1C			**PAD_SIP07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x20			**PAD_SIP08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x24			**PAD_SIP09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x28			**PAD_SIP10**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x2C			**PAD_SIP11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x30			**PAD_SIP12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x34			**PAD_SIP13**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x38			**PAD_SIP14**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x3C			**PAD_SIP15**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x40			**PAD_SIP16**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x44			**PAD_SIP17**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x48			**PAD_SIP18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x4C			**PAD_PA00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x50			**PAD_PA01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x54			**PAD_PA02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x58			**PAD_PA03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x5C			**PAD_PA04**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x60			**PAD_PA05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x64			**PAD_PA06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x68			**PAD_PA07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x6C			**PAD_PA08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x70			**PAD_PA09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x74			**PAD_PA10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x78			**PAD_PA11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x7C			**PAD_PA12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x80			**PAD_PA13**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x84			**PAD_PA14**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x88			**PAD_PA15**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x8C			**PAD_PA16**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x90			**PAD_PA17**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x94			**PAD_PA18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x98			**PAD_PA19**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x9C			**PAD_PA20**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA0			**PAD_PA21**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA4			**PAD_PA22**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA8			**PAD_PA23**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xAC			**PAD_PA24**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB0			**PAD_PA25**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB4			**PAD_PA26**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB8			**PAD_PA27**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xBC			**PAD_PA28**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xC0			**PAD_PA29**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xC4			**PAD_PA30**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xC8			**PAD_PA31**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xCC			**PAD_PA32**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xD0			**PAD_PA33**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xD4			**PAD_PA34**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xD8			**PAD_PA35**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xDC			**PAD_PA36**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xE0			**PAD_PA37**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xE4			**PAD_PA38**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xE8			**PAD_PA39**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xEC			**PAD_PA40**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xF0			**PAD_PA41**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xF4			**PAD_PA42**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xF8			**PAD_PA43**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xFC			**PAD_PA44**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x100			**PAD_PA45**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x104			**PAD_PA46**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x108			**PAD_PA47**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x10C			**PAD_PA48**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x110			**PAD_PA49**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x114			**PAD_PA50**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x118			**PAD_PA51**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x11C			**PAD_PA52**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x120			**PAD_PA53**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x124			**PAD_PA54**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x128			**PAD_PA55**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x12C			**PAD_PA56**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x130			**PAD_PA57**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x134			**PAD_PA58**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x138			**PAD_PA59**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x13C			**PAD_PA60**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x140			**PAD_PA61**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x144			**PAD_PA62**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x148			**PAD_PA63**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x14C			**PAD_PA64**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x150			**PAD_PA65**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x154			**PAD_PA66**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x158			**PAD_PA67**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x15C			**PAD_PA68**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x160			**PAD_PA69**	
[31:12]			RSVD	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x164			**PAD_PA70**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x168			**PAD_PA71**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x16C			**PAD_PA72**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x170			**PAD_PA73**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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 next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h1	IS	Input select. Logic LOW selects CMOS input, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x174			**PAD_PA74**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x178			**PAD_PA75**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x17C			**PAD_PA76**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h0	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x180			**PAD_PA77**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.

Continued on next page...

Table 5-5: HPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x184			**PAD_PA78**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x188			**PAD_PA79**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x18C			**PAD_PA80**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select

5.18 HPSYS_GPIO Registers

Table 5-6: HPSYS_GPIO Register Map

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			**DOER0**	Data Output Enable Register
[31:0]	rw	32'h0	DOE	GPIO[31:0] output enable

Continued on next page...

Table 5-6: HPSYS_GPIO Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x0C			**OESR0**	Data Output Enable Set Register
[31:0]	w	32'h0	OES	set 1 to enable output of corresponding GPIO[31:0]
0x10			**OECR0**	Data Output Enable Clear Register
[31:0]	w	32'h0	OEC	set 1 to disable output of corresponding GPIO[31:0]
0x14			**IER0**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[31:0] interrupt enable
0x18			**IESR0**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[31:0]
0x1C			**IECR0**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[31:0]
0x20			**ITR0**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[31:0] interrupt type
0x24			**ITSR0**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[31:0]
0x28			**ITCR0**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[31:0]
0x2C			**IPSR0**	Interrupt Polarity Set Register
[31:0]	rw	32'h0	IPS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x30			**IPCR0**	Interrupt Polarity Clear Register
[31:0]	rw	32'h0	IPC	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x34			**ISR0**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[31:0]
0x38			**DIR1**	Data Input Register
[31:0]	r	32'h0	IN	GPIO[63:32] input value
0x3C			**DOR1**	Data Output Register
[31:0]	rw	32'h0	OUT	GPIO[63:32] output value if output enabled
0x40			**DOER1**	Data Direction Register
[31:0]	rw	32'h0	DOE	GPIO[63:32] output enable
0x44			**OESR1**	Data Output Enable Set Register
[31:0]	w	32'h0	OES	set 1 to enable output of corresponding GPIO[63:32]
0x48			**OECR1**	Data Output Enable Clear Register
[31:0]	w	32'h0	OEC	set 1 to disable output of corresponding GPIO[63:32]
0x4C			**IER1**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[63:32] interrupt enable
0x50			**IESR1**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[63:32]
0x54			**IECR1**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[63:32]
0x58			**ITR1**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[63:32] interrupt type
0x5C			**ITSR1**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[63:32]
0x60			**ITCR1**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[63:32]
0x64			**IPSR1**	Interrupt Polarity Set Register
[31:0]	rw	32'h0	IPS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[63:32]
0x68			**IPCR1**	Interrupt Polarity Clear Register
[31:0]	rw	32'h0	IPC	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[63:32]

Continued on next page...

Table 5-6: HPSYS_GPIO Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x6C			**ISR1**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[63:32]
0x70			**DIR2**	Data Input Register
[31:17]			RSVD	
[16:0]	r	17'h0	IN	GPIO[80:64] input value
0x74			**DOR2**	Data Output Register
[31:17]			RSVD	
[16:0]	rw	17'h0	OUT	GPIO[80:64] output value if output enabled
0x78			**DOER2**	Data Output Enable Register
[31:17]			RSVD	
[16:0]	rw	17'h0	DOE	GPIO[80:64] output enable
0x7C			**OESR2**	Data Output Enable Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	OES	set 1 to enable output of corresponding GPIO[80:64]
0x80			**OECR2**	Data Output Enable Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	OEC	set 1 to disable output of corresponding GPIO[80:64]
0x84			**IER2**	Interrupt Enable Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IER	GPIO[80:64] interrupt enable
0x88			**IESR2**	Interrupt Enable Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	IES	set 1 to enable interrupt of corresponding GPIO[80:64]
0x8C			**IECR2**	Interrupt Enable Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	IEC	set 1 to disable interrupt of corresponding GPIO[80:64]
0x90			**ITR2**	Interrupt Type Register
[31:17]			RSVD	
[16:0]	rw	17'h0	ITR	GPIO[80:64] interrupt type
0x94			**ITSR2**	Interrupt Type Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[80:64]
0x98			**ITCR2**	Interrupt Type Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[80:64]
0x9C			**IPSR2**	Interrupt Polarity Set Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IPS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[80:64]
0xA0			**IPCR2**	Interrupt Polarity Clear Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IPC	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[80:64]
0xA4			**ISR2**	Interrupt Status Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[80:64]

5.19 LPSYS_PINMUX Registers

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**PAD_PB00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x04			**PAD_PB01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x08			**PAD_PB02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x0C			**PAD_PB03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x10			**PAD_PB04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS	Drive select. Logic LOW selects 4mA drive, logic HIGH selects 20mA drive.

Continued on next page...

Table 5-7: **LPSYS_PINMUX Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x14			**PAD_PB05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x18			**PAD_PB06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x1C			**PAD_PB07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x20			**PAD_PB08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and logic HIGH selects Schmitt input.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x24			**PAD_PB09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x28			**PAD_PB10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x2C			**PAD_PB11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x30			**PAD_PB12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x34			**PAD_PB13**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x38			**PAD_PB14**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x3C			**PAD_PB15**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x40			**PAD_PB16**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x44			**PAD_PB17**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x48			**PAD_PB18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x4C			**PAD_PB19**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x50			**PAD_PB20**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x54			**PAD_PB21**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and logic HIGH selects Schmitt input.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (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'h1	PS	Pull select. Logic HIGH selects pull-up, logic LOW selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x58			**PAD_PB22**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x5C			**PAD_PB23**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x60			**PAD_PB24**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x64			**PAD_PB25**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x68			**PAD_PB26**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x6C			**PAD_PB27**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x70			**PAD_PB28**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x74			**PAD_PB29**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x78			**PAD_PB30**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x7C			**PAD_PB31**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x80			**PAD_PB32**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x84			**PAD_PB33**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x88			**PAD_PB34**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and logic HIGH selects Schmitt input.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x8C			**PAD_PB35**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x90			**PAD_PB36**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x94			**PAD_PB37**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0x98			**PAD_PB38**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x9C			**PAD_PB39**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA0			**PAD_PB40**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA4			**PAD_PB41**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xA8			**PAD_PB42**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xAC			**PAD_PB43**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS	Drive select. Logic LOW selects 4mA drive, logic HIGH selects 20mA drive.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB0			**PAD_PB44**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB4			**PAD_PB45**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xB8			**PAD_PB46**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xBC			**PAD_PB47**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and logic HIGH selects Schmitt input.

Continued on next page...

Table 5-7: LPSYS_PINMUX Register Map (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'h1	PS	Pull select. Logic HIGH selects pull-up, logic LOW selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select
0xC0			**PAD_PB48**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Parametric output enable. Logic LOW forces PO HIGH.
[10]	rw	1'h0	DS1	Drive select 1. Used to select output drive strength.
[9]	rw	1'h1	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, and 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 selected pull-down.
[4]	rw	1'h1	PE	Pull enable. Logic HIGH enables weak pull device.
[3:0]	rw	4'h0	FSEL	Function Select

5.20 LPSYS_GPIO Registers

Table 5-8: LPSYS_GPIO Register Map

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			**DOER0**	Data Output Enable Register
[31:0]	rw	32'h0	DOE	GPIO[31:0] output enable
0x0C			**OESR0**	Data Output Enable Set Register
[31:0]	w	32'h0	OES	set 1 to enable output of corresponding GPIO[31:0]
0x10			**OECR0**	Data Output Enable Clear Register
[31:0]	w	32'h0	OEC	set 1 to disable output of corresponding GPIO[31:0]
0x14			**IER0**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[31:0] interrupt enable
0x18			**IESR0**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[31:0]
0x1C			**IECR0**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[31:0]
0x20			**ITR0**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[31:0] interrupt type
0x24			**ITSR0**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[31:0]
0x28			**ITCR0**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[31:0]
0x2C			**IPSR0**	Interrupt Polarity Set Register
[31:0]	rw	32'h0	IPS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x30			**IPCR0**	Interrupt Polarity Clear Register
[31:0]	rw	32'h0	IPC	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]

Continued on next page...

Table 5-8: LPSYS_GPIO Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x34			**ISR0**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[31:0]
0x38			**DIR1**	Data Input Register
[31:17]			RSVD	
[16:0]	r	17'h0	IN	GPIO[48:32] input value
0x3C			**DOR1**	Data Output Register
[31:17]			RSVD	
[16:0]	rw	17'h0	OUT	GPIO[48:32]output value if output enabled
0x40			**DOER1**	Data Direction Register
[31:17]			RSVD	
[16:0]	rw	17'h0	DOE	GPIO[48:32] output enable
0x44			**OESR1**	Data Output Enable Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	OES	set 1 to enable output of corresponding GPIO[48:32]
0x48			**OECR1**	Data Output Enable Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	OEC	set 1 to disable output of corresponding GPIO[48:32]
0x4C			**IER1**	Interrupt Enable Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IER	GPIO[48:32] interrupt enable
0x50			**IESR1**	Interrupt Enable Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	IES	set 1 to enable interrupt of corresponding GPIO[48:32]
0x54			**IECR1**	Interrupt Enable Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	IEC	set 1 to disable interrupt of corresponding GPIO[48:32]
0x58			**ITR1**	Interrupt Type Register
[31:17]			RSVD	
[16:0]	rw	17'h0	ITR	GPIO[48:32] interrupt type
0x5C			**ITSR1**	Interrupt Type Set Register
[31:17]			RSVD	
[16:0]	w	17'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[48:32]
0x60			**ITCR1**	Interrupt Type Clear Register
[31:17]			RSVD	
[16:0]	w	17'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[48:32]
0x64			**IPSR1**	Interrupt Polarity Set Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IPS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[48:32]
0x68			**IPCR1**	Interrupt Polarity Clear Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IPC	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[48:32]
0x6C			**ISR1**	Interrupt Status Register
[31:17]			RSVD	
[16:0]	rw	17'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[48:32]

6 DMA

6.1 DMAC

The chip contains 2 DMACs, with DMAC1 located in HPSYS and DMAC2 located in LPSYS.

6.1.1 Introduction

DMAC (Direct Memory Access Controller) is used to perform data transfers between two different address regions on the bus. The DMAC comprises 8 independent channels. Each channel can configure source and destination address regions, mapped respectively to the address ranges of various memories or peripherals, enabling high-effciency transfers between memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral, thereby effectively reducing the CPU workload. DMAC supports Peripheral Response Mode and Memory Transfer Mode: In Peripheral Response Mode, DMAC performs transfers based on peripheral DMA requests, thereby adapting to the peripheral's bandwidth; In Memory Transfer Mode, DMAC does not wait for peripheral DMA requests and completes data transfers as quickly as possible. When multiple channels are enabled simultaneously, DMAC performs transfers sequentially according to priority from highest to lowest; and during the transfer process of a lower-priority channel, a higher-priority channel can preempt the transfer. Each channel can generate an interrupt or trigger a PTC when more than half of the transfer is completed or when the transfer is fnished.

6.1.2 Main Features

- Single AHB master bus capable of accessing SRAM, PSRAM, FLASH, AHB, and APB peripherals.
- 8 independent configurable channels
- Each channel's DMA request can select one request from up to 64 peripheral DMA requests or be triggered by software.
- Each channel supports 4 priority levels; when priorities are equal, arbitration is determined by channel number.
- Supports data transfers from peripheral to memory, memory to peripheral, memory to memory, and peripheral to peripheral.
- Source and destination addresses independently support byte, half-word, and word accesses. The source and destination addresses must be aligned according to the size of the data transfer unit and support automatic address increment.
- The number of units per single transfer can be configured from 0 to 65535.
- Supports circular buffer mode, automatically restarting upon completion of each single transfer.
- Each channel supports 3 types of event flags: transfer complete, half-transfer, or transfer error, and can independently generate interrupt requests and PTC triggers.
- Each channel supports block transfer mode with configurable block size.

6.1.3 Peripheral Requests

Each channel selects any one of the 32 peripheral request sources as the bound request source by configuring CSELR1/2_CxS. The response signal of the selected peripheral is correspondingly bound to the channel. The DMAC peripheral request

table is as follows.

Table 6-1: DMAC Peripheral Request Table

CSELR1/2_CxS	DMAC1	DMAC2
0	qspi1	usart3_tx
1	qspi2	usart3_rx
2	qspi3	usart4_tx
3	/	usart4_rx
4	usart1_tx	usart5_tx
5	usart1_rx	usart5_rx
6	usart2_tx	btim3
7	usart2_rx	btim4
8	gptim1_update	gptim3_update
9	gptim1_trigger	gptim3_trigger
10	gptim1_cc1	gptim3_cc1
11	gptim1_cc2	gptim3_cc2
12	gptim1_cc3	gptim3_cc3
13	gptim1_cc4	gptim3_cc4
14	btim1	gptim5_update
15	btim2	gptim5_trigger
16	/	spi3_tx
17	i2c3	spi3_rx
18	i2s1_rx/pdm1_r	spi4_tx
19	i2s1_rx/pdm1_r	spi4_rx
20	i2s2_tx/pdm2_l	qspi4
21	i2s2_rx/pdm2_r	i2c4
22	i2c1	i2c5
23	i2c2	i2c6
24	gptim2_update	gptim4_update
25	gptim2_trigger	gptim4_trigger
26	gptim2_cc1	gptim4_cc1
27	gptim2_cc2	gptim4_cc2
28	spi1_tx	gptim4_cc3
29	spi1_rx	gptim4_cc4
30	spi2_tx	gpadc
31	spi2_rx	sdadc

6.1.4 DMAC Function Description

6.1.4.1 DMAC Block Diagram

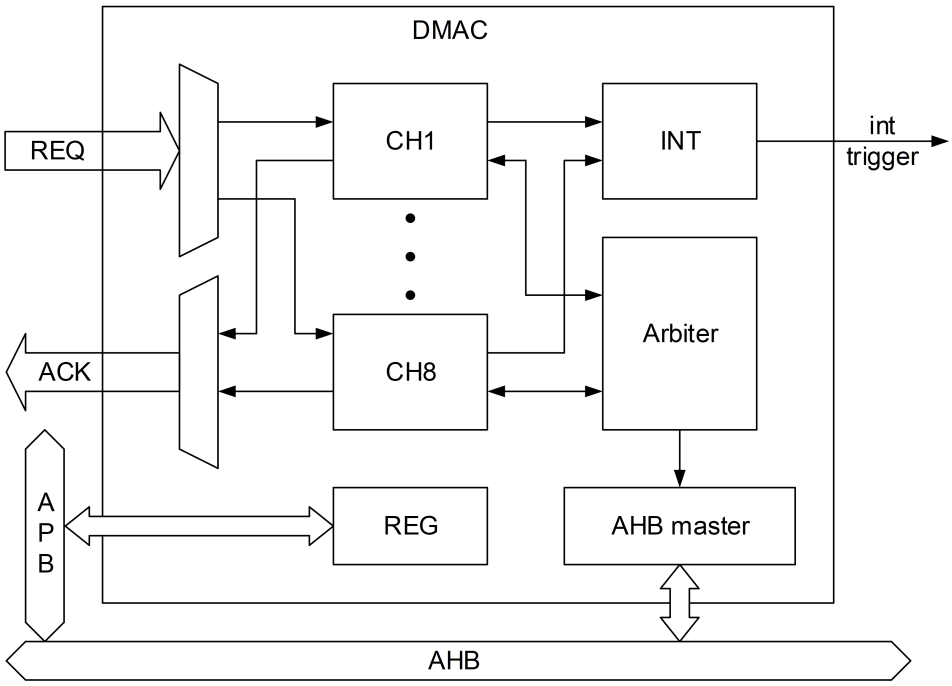


Figure 6-1: **DMAC Block Diagram**

6.1.4.2 Transfer Effciency

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 suspend the CPU's access to the system bus. The bus arbiter performs round-robin scheduling to ensure that the CPU receives at least half of the bus bandwidth. Similarly, when other master devices access the same AHB target as the DMAC, the DMAC's access bandwidth will also decrease. When the bus has no other accesses, DMAC requires a minimum of 2 HCLK cycles to complete a single unit transfer, specifically depending on the AHB wait cycles of the access target

6.1.4.3 Transfer Mode

Each DMAC channel can be independently configured to Peripheral Transfer Mode or Memory Transfer Mode, controlled by the CCRx_MEM2MEM register. The main distinction is whether the peripheral request/acknowledge mechanism is enabled. Peripheral Transfer Mode adapts to the peripheral's data bandwidth via the request/acknowledge mechanism, initiating transfers only after the peripheral is ready. It is typically used in scenarios such as UART transmission/reception, I2S audio input/output, and FLASH programming. The memory transfer mode does not require waiting for a request signal and transfers data at the maximum possible bandwidth. It is typically used in scenarios such as SRAM handling and CRC verifcation.

6.1.4.4 Transfer Procedure

In peripheral transfer mode, the DMAC transfer employs a peripheral request/acknowledge mechanism. Transfers are initiated by peripheral requests and proceed according to the following steps:

1. When the peripheral requires data transfer (e.g., receive buffer full or transmit buffer 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 holds the highest priority among all request channels, it initiates the single transfer for this channel.Unit transfer or block transfer;
3. Upon completion of unit transfer or block transfer, DMAC sends an acknowledgment signal to the peripheral;
4. After the peripheral receives the acknowledgment signal from DMAC, it releases the request;
5. Upon detecting the release of the peripheral request, DMAC releases the acknowledgment signal;
6. After the peripheral detects the release of the acknowledgment signal, if there is still a transfer demand, it may continue to send requests to restart DMAC unit transfer or block transfer.

In memory transfer mode, transfers are initiated by software and repeatedly perform unit transfers until completion when the channel priority is suffcient.

6.1.4.5 Transfer Enable

The DMAC's 8 channels are enabled independently. When CCRx_EN is 1 and CNDTRx is not 0 , the channel transfer initiates. In peripheral transfer mode, it begins responding to peripheral requests; in memory transfer mode, the transfer starts immediately. It should be noted that when the channel is enabled, even if the previous transfer has completed, rewriting a non- 0 value to CNDTRx will immediately start the DMAC transfer. Prior to this, all other parameters must be fully configured, or ensure CCRx_EN is 0 when writing to CNDTRx.

6.1.4.6 Transfer Unit

The basic unit of DMAC transfer is a transfer unit, which consists of 3 steps:

1. Single read of source address data via the bus, configurable as single-byte/double-byte/four-byte;
2. The data read is written once to the target address; single-byte/double-byte/four-byte transfers are configurable;
3. Update the count, stop the transfer based on the count result, or calculate the address for the next transfer.

6.1.4.7 Transfer quantity

DMAC The transfer quantity per channel is controlled by CNDTRx in units of transfer elements, supporting a range of 0~65535. For example, if configured for four-byte transfers, the maximum data amount per single channel transfer is 65535×4 = 262140 bytes. After configuring CNDTRx and starting the unit transfer (CCRx_EN=1), each completed transfer unit decrements the CNDTRx register value by 1. In non-circular mode (CCRx_CIRC=0), the transfer stops when CNDTRx decrements to 0. In circular mode (CCRx_CIRC=1) , after CNDTRx decrements to 0 , it immediately reloads the initial value configured by software and continues transmission.

6.1.4.8 Circular mode

In circular mode, transmission does not stop automatically. In circular mode, after the fnal data transmission is completed, the CNDTRx register automatically reloads the initially programmed value. The current internal address register reloads the base address values in the CPARx and CM0ARx registers. In circular mode, by monitoring the half-transfer and transfer-complete flags, the ping-pong buffer functionality can be realized.

6.1.4.9 Transfer direction

The transfer direction of the DMAC is determined by CCRx_DIR.

Table 6-2: DMAC Transfer Direction

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

6.1.4.10 Transfer Data Width

During DMAC transfer, bus accesses to source and destination addresses can be independently configured as byte/double-byte/four-byte types via CCRx_MSIZE and CCRx_PSIZE. DMAC does not provide data packing or unpacking functionality; therefore, when the data widths of source and destination address accesses differ, data will be truncated or compensated, as illustrated in the example below.

Table 6-3: DMAC Transfer Data 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 0xB7B6 @0x6	0xB0 @0x0 0xB2 @0x1 0xB4 @0x2 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	0xB3B2B1B0 @0x0 0xB7B6B5B4 @0x4 0xBBBAB9B8 @0x8

Continued on next page...

Table 6-3: DMAC Transfer Data Width (continued)

Source Size	Destination Size	Source Data	Destination Data
		0xBFBEBDBC @0xC	0xBFBEBDBC @0xC

6.1.4.11 Transfer Address

The start source and destination addresses for transfer are determined by CPARx,CM0ARx, and CCRx_DIR. If increment mode is enabled (CCRx_MINC and CCRx_PINC configured independently), after completing one unit transfer, the next transfer address will be the previous transfer address plus 1 , 2 , or 4 (matching the transfer data width).

When configuring addresses, it must be noted that during DMAC transfers, address increments must not cross a 1M byte boundary. For example, if the transfer address increments from 0x600FFFFC to 0x60100000 , the transfer will result in an error. Therefore, if data transfer crosses a 1M byte boundary, it should be split into two separate transfers.

During the transfer process, these registers retain their initially programmed values. Software cannot retrieve the current transfer address. Typically, the transfer address configuration for peripheral FIFO is non-incrementing, whereas the transfer address configuration for memory is incrementing.

6.1.4.12 Channel Arbitration

The DMAC arbiter manages the priority among different channels. When the arbiter grants priority to a valid channel (hardware request or software trigger), it initiates the unit transfer or block transfer for that channel. Subsequently, the arbiter re-arbitrates among valid channels and selects the channel with the highest priority.

The arbiter determines priority based on the following criteria:

1. Channels in peripheral transfer mode have priority over channels in memory transfer mode.
2. For channels both in peripheral transfer mode or both in memory transfer mode, the channel with the bigger CCRx_PL has higher priority.
3. The transfer mode is identical to CCRx_PL,with lower channel numbers having higher priority (for example, channel 1 has priority over channel 2).

During the transfer process of a given channel, if another channel with higher priority issues a transfer request, the arbiter will switch the transfer to the higher priority channel after the current channel completes its unit or block transfer.

6.1.4.13 Block Transfer

Block transfer is effective only in peripheral transfer mode. CBSRx defines the number of consecutive unit transfers the DMAC must complete for each peripheral request. For example, when CBSRx is set to 4, whenever the peripheral initiates a request, the DMAC will complete 4 consecutive unit transfers before issuing an acknowledgment signal, during which CNDTRx will decrement by 4 consecutively. If the remaining units to be transferred CNDTRx are less than CBSRx, only the remaining CNDTRx unit transfers will be executed.

6.1.4.14 Notifcation Mechanism

Each DMAC channel can independently generate Interrupt and PTC trigger signals, with the enablement of each interrupt type configurable separately. When at least half of the channel transfer is completed, an HTIFx interrupt and trigger are generated. Upon full completion of the channel transfer, a TCIFx interrupt and trigger are generated. When a bus access

error occurs during channel transfer, a TEIFx interrupt and trigger are generated. The occurrence of any of these three interrupts triggers a GIFx interrupt. Channel interrupts can be cleared individually or collectively via CGIFx.

6.1.4.15 Channel Configuration Procedure

When configuring the DMACchannel, the following steps must be performed:

1. Set the peripheral register address in the CPARx register.
   After a peripheral request occurs, or when the channel is enabled 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 when the channel is enabled in memory mode, data will be written to or read from memory.
3. Ensure that CCRx_EN is 0before writing the number of data units to be transferred into the CNDTRx register.
   This value decrements after each data transfer.
4. In The following parameters are configured in the CCRx register:
   - Channel Priority
   - Data Transfer Direction
   - Circular mode
   - Peripheral and Memory Increment Mode
   - Peripheral and Memory Data Size
   - Interrupt enable for half-transfer completion and/or full completion, as well as/or transfer error occurrence
5. Set Set CCRx_EN to 1 to activate the channel.
   Once enabled, the channel can process requests from the peripheral connected to it or initiate memory-to-memory transfers.

6.1.4.16 Transfer Completion Handling

In non-circular mode, after a DMAC channel transfer completes, CNDTRx automatically resets to zero. Software must write CCRx_EN to 0 for that channel to prevent false triggering during subsequent configurations. If an interrupt flag occurs, the software shall clear the interrupt flag after performing the corresponding handling.

In loop mode, DMAC does not automatically stop transmission. When the software intends to stop transmission, it shall write 0 to the channel's CCRx_EN and clear the interrupt flag.

6.1.5 DMAC Registers

Table 6-4: DMAC Register Map

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

Continued on next page...

Table 6-4: DMAC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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

Continued on next page...

Table 6-4: DMAC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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 next page...

Table 6-4: DMAC Register Map (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
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).

Continued on next page...

Table 6-4: DMAC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
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
[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 next page...

Table 6-4: DMAC Register Map (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
0x20			**CNDTR2**	
[31:16]			RSVD	

Continued on next page...

Table 6-4: DMAC Register Map (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).
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**	
[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

Continued on next page...

Table 6-4: DMAC Register Map (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 next page...

Table 6-4: DMAC Register Map (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
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).
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

Continued on next page...

Table 6-4: DMAC Register Map (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 next page...

Table 6-4: DMAC Register Map (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
0x48			**CNDTR4**	
[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).
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**	

Continued on next page...

Table 6-4: DMAC Register Map (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.
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
[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 next page...

Table 6-4: DMAC Register Map (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
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).

Continued on next page...

Table 6-4: DMAC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
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
[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 next page...

Table 6-4: DMAC Register Map (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
0x70			**CNDTR6**	
[31:16]			RSVD	

Continued on next page...

Table 6-4: DMAC Register Map (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).
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**	
[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

Continued on next page...

Table 6-4: DMAC Register Map (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 next page...

Table 6-4: DMAC Register Map (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
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).
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

Continued on next page...

Table 6-4: DMAC Register Map (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 next page...

Table 6-4: DMAC Register Map (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
0x98			**CNDTR8**	
[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).
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**	

Continued on next page...

Table 6-4: DMAC Register Map (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.
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

ExtDMA is located within HPSYS.

6.2.1 Introduction

ExtDMA (Extended Direct Memory Access) enables effcient data transfer between two distinct address regions on the bus and integrates the TurboPixel image frame compression engine, which performs image compression concurrently with data transfer. Compared to DMAC , ExtDMA is more effcient when accessing external memory (such as FLASH , 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 AHB master capable of accessing SRAM, PSRAM, FLASH, etc., supporting BURST transfers
- Single transfer channel with a built-in FIFO of depth 16 and width 32 bits

- Source and destination addresses both support 4-byte access with automatic address increment
- Maximum transfer unit per configuration is 2^{20}-1 units, each fixed at 4 bytes, resulting in a maximum single transfer of 4M bytes
- The channel supports transfer complete, half-transfer, and transfer error event flags, and can generate interrupt requests and PTC triggers.
- Integrated TurboPixel image frame compression engine, supporting RGB565/RGB888/ARGB8888 format input, with a maximum of 512 pixels per line.

6.2.3 Functional 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, the ExtDMA request may suspend the CPU's access to the system bus. The bus arbiter performs round-robin scheduling to ensure that the CPU receives at least partial bus bandwidth. Similarly, when other main control devices and ExtDMA access the same AHB target, the ExtDMA access bandwidth will also decrease. ExtDMA preferentially adopts the AHB Burst transmission mode; therefore, when accessing memory optimized for Burst transmission, the transmission effciency will exceed that of DMAC.

6.2.3.2 Operating Modes

ExtDMA cannot respond to peripheral requests; its operating modes include transfer mode and compression mode.

When CMPRCR_CMPREN is 0, ExtDMA operates in transfer mode, capable of moving a specifed amount of memory data from the source address to the destination address.

When CMPRCR_CMPREN is 1, ExtDMA operates in compression mode, capable of initiating the TurboPixel compression engine to compress image data of a specifed size stored at the source address and save the compressed result to the destination address.

6.2.3.3 Data Address and Bit Width

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

If address increment is enabled (source and destination addresses are configured respectively by CCR_SRCINC and CCR_DSTINC), ExtDMA increments the address by 4 after each source data read or destination data write operation; otherwise, the address remains unchanged. The non-increment address mode is primarily used to access FIFOs with fixed entry addresses, such as the data input of CRC.

6.2.3.4 Transfer Enable

CCR_EN is the channel enable register. When CCR_EN is set to 1 and CNDTR is not 0, data transfer will commence. It should be noted that when the channel is enabled, even if the previous transfer has completed, rewriting a non-0 value to CNDTR will immediately initiate the ExtDMA transfer; therefore, all other parameters must be fully configured beforehand, or the channel enable must be disabled first.

6.2.3.5 Transfer Quantity

In transfer mode, the quantity of data to be transferred is configured via CNDTR , counted in units of four bytes, supporting a range from 0 to 2^{20}-1. For example, when CNDTR is configured to 1000 , the amount of data transferred is 4000 bytes. After initiating data transfer, each completed four-byte read decrements CNDTR by 1 .CMPRNDTR indicates the amount of data to be written and does not require software configuration in transfer mode. After initiating data transfer,CMPRNDTR automatically copies the initial CNDTR value; thereafter, after each 4-byte write completion,CMPRNDTR decrements by 1 . When both CNDTR andCMPRNDTR decrement to 0 , all data reading and writing operations are complete, and ExtDMA ceases transmission.

In compression mode, the amount of image data to be compressed is configured via CNDTR , counted in units of 4 bytes, supporting a range from 0 to 2^{20}-1. For example, if CNDTR is configured as 1000 , the image data to be compressed is 4000 bytes. After initiating data transfer, each completed four-byte read decrements CNDTR . CMPRNDTR indicates the amount of data after image compression; in compression mode, software must calculate and write it according to the formula. The calculation formula for CMPRNDTR is TGTSIZE*6*total number of rows of the image to be compressed/4 , where at least one of TGTSIZE or the total number of rows must be even. If the software configuration of CMPRNDTR is incorrect, ExtDMA may fail to complete the transfer properly. After initiating data transfer, each completed four-byte write decrements CMPRNDTR by 1 . When both CNDTR and CMPRNDTR decrement to 0 , all data reading and writing are complete, and ExtDMA stops the transfer.

6.2.3.6 TurboPixel Compression

ExtDMA embedded TurboPixel image frame compression engine, capable of compressing raw images and outputting them to the target address;the compression process does not require additional memory space.

Raw images support RGB565/RGB888/ARGB8888 formats, configured via the CMPRCR_SRCFMT register. The alignment of the starting address for storing raw images is configured via the CMPRCR_SRCPOS register. The starting address for storing ARGB8888 images must be four-byte aligned. The starting address for storing RGB565 images must be two-byte aligned. The starting address for storing RGB888 images has no alignment requirements. It should be noted that although the image start address supports non-four-byte alignment, the ExtDMA transfer source address SRCAR must be configured to the four-byte aligned address corresponding to the image start address.

The number of pixels per line in the original image is configured by CMPRSR_LINESIZE, supporting up to 512.

The image compression ratio is configured through two associated registers, CMPRSR_TGTSIZE and CMPRNDTR. The ratio of data volume before and after compression is approximately equal to the ratio of CNDTR to CMPRNDTR, due to possible unalignment of the original image. Image compression parameters are configured via the CMPRCFG0 and CMPRCFG1 registers and must be adjusted according to the compression ratio and image format.

The image compression quality can be evaluated via the CMPRQR and CMPRDR registers. It should be noted that under higher compression rate (smaller data size after compression) configurations, abnormal compression quality may occur for certain special image patterns, which will trigger an overflow error interrupt OFIF . When such an anomaly occurs, it is recommended to avoid using images with similar patterns or to reduce the compression rate (increase the data size after compression).

6.2.3.7 Notifcation Mechanism

ExtDMA can generate interrupts and PTC trigger signals, with independent enable configuration for each interrupt type. When at least half of the transfer is complete, an HTIF interrupt and trigger are generated. Upon full completion of the

transfer, a TCIF interrupt and trigger are generated. When a bus access error occurs during transfer, a TEIF interrupt and trigger are generated. In compression mode, when an overflow error occurs, an OFIF interrupt and trigger are generated.

Interrupts are enabled via control bits such as TCIE, HTIE, TEIE, and OFIE in the CCR register. Interrupt status can be accessed through the ISR register and cleared via the IFCR register.

6.2.3.8 Exception Handling

If an ExtDMA transfer cannot complete due to configuration errors, setting CCR_RESET to 1 resets the ExtDMA logic. After reset, CCR_RESET automatically clears to 0. Other configuration registers are not reset by this operation.

6.2.3.9 Recommended configuration Procedure for Transfer Mode

1. Set the transfer mode by setting CMPRCR_CMPREN to 0.
2. Set the four-byte aligned source address SRCAR and destination address DSTAR.
3. Ensure CCR_EN is 0, then write the amount of data to be transferred, in units of four bytes, into the CNDTR register.
4. Configure the following parameters in the CCR register:
   - Address increment modes CCR_SRCINC and CCR_DSTINC
   - Set source data width CCR_SRCSIZE and destination data width CCR_DSTSIZE to four bytes
   - Enable interrupt
5. Set CCR_EN to 1 to start data transfer.
6. Wait for the interrupt to occur and be handled, then set CCR_EN to 0.

6.2.3.10 Recommended configuration Procedure for Compression Mode

1. Set compression mode by setting CMPRCR_CMPREN to 1.
2. Set the source image format and start address alignment method in CMPRCR .
3. Set the four-byte aligned source address SRCAR and destination address DSTAR.
4. Ensure CCR_EN is 0, then write the source image data size in units of four bytes into the CNDTR register.
5. Configure the compression parameters CMPRSR, CMPRCFG0, and CMPRCFG1.
6. Calculate the compressed data size and configure CMPRNDTR accordingly.
7. At configure the following parameters in the CCR register:
   - Address increment modes CCR_SRCINC and CCR_DSTINC
   - Set source data width CCR_SRCSIZE and destination data width CCR_DSTSIZE to four bytes
   - Enable interrupt
8. Set CCR_EN to 1 to initiate image compression.
9. Wait for the interrupt to occur and be handled, then set CCR_EN to 0.

6.2.4 ExtDMA Registers

Table 6-5: ExtDMA Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	interrupt status register
[31:5]			RSVD	
[4]	r	1'h0	OFIF	OFIF, overflow flag
[3]	r	1'h0	TEIF	TEIF, transfer error flag

Continued on next page...

Table 6-5: ExtDMA Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	r	1'h0	HTIF	HTIF, half transfer flag
[1]	r	1'h0	TCIF	TCIF, transfer complete flag
[0]	r	1'h0	GIF	GIF, global interrupt flag
0x04			**IFCR**	interrupt clear register
[31:5]			RSVD	
[4]	w1s	1'h0	COFIF	COFIF, overflow flag clear
[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]			RSVD	
[4]	rw	1'h0	OFIE	overflow interrupt enable 0: disabled 1: enabled
[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

Continued on next page...

Table 6-5: ExtDMA Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x0C			**CNDTR**	number of data register
[31:20]			RSVD	
[19:0]	rw	20'h0	NDT	number of data to transfer (0 to 2^{20} - 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
0x20			**CMPRCR**	Image Compression control register
[31:8]			RSVD	
[7:6]	rw	2'h0	SRCPOS	Starting byte position in the first word of the source frame Valid starting positon includes: RGB565: 0x0,0x2 RGB888: 0x0, 0x1, 0x2, 0x3 ARGB8888: 0x0
[5:4]	rw	2'h0	SRCFMT	Source frame format 00: 16-bit RGB 5:6:5 01: 24-bit RGB 8:8:8 10: 32-bit ARGB 8:8:8:8 11: Reserved
[3:1]			RSVD	
[0]	rw	1'h0	CMPREN	Compression enable
0x24			**CMPRSR**	Compression size register
[31:28]			RSVD	
[27:16]	rw	12'h0	TGTSIZE	output target size of each line after compression. output data size of each line is tgtsize*3*2 bytes
[15:12]			RSVD	
[11:0]	rw	12'h0	LINESIZE	column number (pixel number) of each line. Input data size of each line is line-size*(size per pixel)
0x28			**CMPRNDTR**	number of compression output data register
[31:20]			RSVD	
[19:0]	rw	20'h0	CMPRNDT	If compression enabled, cmprndt is the number of data to transfer after compression (0 to 2^{20} - 1) and have to be written by software before compression. The value should be TGTSIZE*6*line_number/4 in compression mode. If compression disabled, it reads the number of data that extdma has written into destination address. Software do not need to write this field. This field is updated by hardware when extdma enabled
0x2C			**CMPRCFG0**	Compression configuration 0
[31:0]	rw	32'h80023307	CFG	Compression configuration
0x30			**CMPRCFG1**	Compression configuration 1
[31:0]	rw	32'h01055982	CFG	Compression configuration
0x34			**CMPRQR**	Compression quality register
[31:29]			RSVD	
[28:16]	r	13'h0	DUMMY	line least dummy word in one frame, update every frame.

Continued on next page...

Table 6-5: ExtDMA Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:8]	r	8'h0	LQB	low quality block number. the bigger of this number, the worse compressed image quality
[7:0]	r	8'h0	LQR	quality sum to low qulity block number ratio. the bigger of this number, the worse compressed image quality.
0x38			**CMPRDR**	Compression debug register
[31:7]			RSVD	
[6:0]	r	7'h0	MAXBUF	record of max used buffer during compression output

7 Peripheral Connection

7.1 I2C

The chip contains a total of 6 I2C interfaces, of which I2C1 , I2C2 , and I2C3 are located in HPSYS , with input/output connected to IO(PA) , capable of sending requests to DMAC1. I2C4 , I2C5 , and I2C6 are located in LPSYS , with input/output connected to IO(PB) , capable of sending requests to DMAC2.

7.1.1 Introduction

The I2C (Inter-Integrated Circuit) interface supports both master and slave device roles simultaneously; it can operate as a master device to communicate with I2C peripherals or as a slave device to respond to external I2C master devices. The I2C module includes an 8-byte FIFO, supporting single read/write operations as well as batch data transfers via DMA. I2C supports standard-mode, fast-mode, fast-mode plus, and high-speed-mode, with a maximum data rate of up to 3.4 Mbps.

7.1.2 Main Features

Can operate simultaneously as both master and slave device

- Supports multi-master bus configuration
- Supports Standard Mode (up to 100 kbps)
- Supports Fast Mode (up to 400 kbps)
- Supports Fast Mode Plus (up to 1 Mbps)
- Supports High-Speed Mode (up to 3.4 Mbps)
- Supports 7-bit or 10-bit addressing as Master device
- Supports 7-bit addressing as Slave device
- Configurable bus timing
- Supports clock stretching
- 8-byte FIFO, supports DMA
- Configurable digital debounce circuit

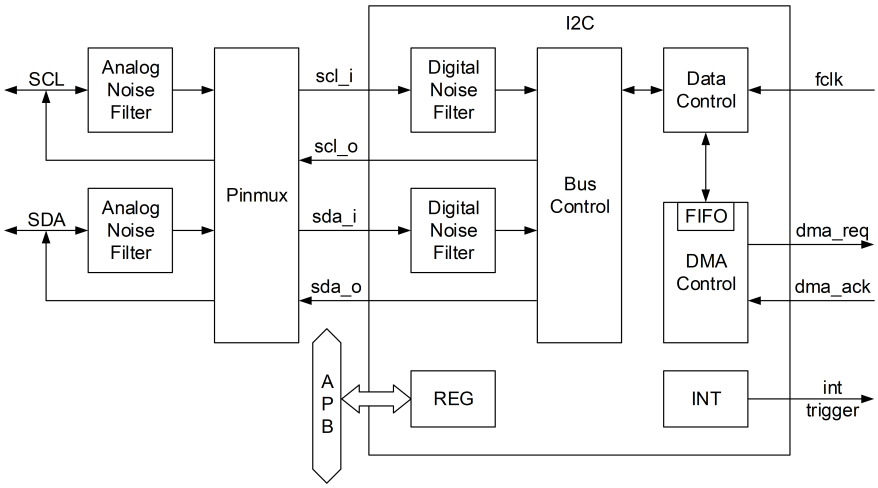


Figure 7-1: I2C Block Diagram

7.1.3 I2C Function Description

7.1.3.1 Two-wire Transmission

The I2C bus utilizes two lines, SCL and SDA, for communication, where SCL is commonly referred to as the clock line, and SDA as the data line. Both lines support bidirectional transmission and operate in open-drain output mode; therefore, external pull-up resistors must be added outside the chip, with resistance values determined based on the maximum transmission rate. When the I2C bus is idle, both SCL andSDA are pulled high. The current I2C bus signal levels can be monitored via BMR_SCL and BMR_SDA.

7.1.3.2 Input Filter

The SCL and SDA input signals can be filtered for glitches using analog and digital filters. The analog filter, configurable in the PIN-MUX module, can filter glitches shorter than 50ns. The digital filter can configure CR_DNF to select the upper limit of the glitch width to be filtered, with a maximum configurable value of 7 fclk cycles.

7.1.3.3 Transmission Rate

The transmission rate of I2C is primarily determined by the master device, but when the slave device supports clock stretching, clock stretchingcan also influence it.

The I2C interface timing is generated based onfclk.fclk is equal to the system pclk frequency, and changes in the pclk frequency will cause the I2C output clock to change accordingly; therefore, the pclk frequency should remain stable during I2C operation.

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

The base bit rate of the standard mode can be approximately calculated using the following formula:

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

Where, Ffclk is the frequency of fclk. WCR_CNT is used to adjust the offset between the edges of SDA and SCL to ensure

that the signal setup and hold times comply with the I2C protocol; when configured properly, it does not affect the bit rate.

The base bit rate of fast mode/fast-mode plus can be approximately calculated using the following formula:

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

7.1.3.4 Transmission sequence

A complete I2Ctransmission shall be based on the following transmission sequence:

1. Start bit. Sent by the master device to initiate transmission.
2. 7 bit address. Sent by the master device to select the slave device.
3. R/nW bit. Sent by the master device to indicate the direction of reception or transmission.
4. ACK bit. Sent by the slave device in response to the master device's request.ACK=0 indicates successful acknowledgment.ACK=1 indicates transmission failure.
5. 8 bit data. Sent 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 register addresses or data.
6. ACK bit. Sent by the master device during reception and by the slave device during transmission, it is a response to the previous 8bit data.
7. Repeat steps 5-6 until data transmission is complete or ACK=1 is detected.
8. Repeat the start bit (return to step 1) or issue the stop bit. The master device initiates either a restart or a stop to control the transmission.

7.1.3.5 Operating Modes and Status

I2C defaults to master-slave mode, enabling it to initiate transmissions actively while concurrently monitoring address transmissions on the bus. When transmission is initiated by software, I2C enters either master transmit or master receive state. When a 7-bit address following the start bit matches SAR_ADDR, I2C enters slave transmit or slave receive state according to the R/nW bit.

7.1.3.6 I2C Initialization Procedure

1. Configure the I2Crate mode CR_MODE and set the corresponding timing registers such as LCR and WCR according to the selected rate.
2. Configure IER to enable the required interrupts.
3. Enable I2C, set CR_SCLE=1 and CR_IUE=1.

7.1.3.7 Master Transmission Procedure

1. Left-shift the slave device address 1 bits, append the least signifcant bit 0 , and write it to DBR.
2. TCR=TCR_START;TCR|=TCR_TB.
3. Poll SR_TE until it is set 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 a stop bit and abort the transmission.
5. Write the data to be transmitted into DBR.
6. TCR=TCR_TB。
7. Poll SR_TE until it is set 1, or wait for the TE interrupt.

8. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, send a stop bit and abort the transmission.
9. Repeat steps 5-8; if it is the last data byte, set TCR = TCR_TB | TCR_STOP. The stop bit will be automatically generated after transmission completion.

7.1.3.8 Main Reception Process

1. Left-shift the slave device address 1 Bit combined with the least significant bit 1, written into DBR.
2. TCR=TCR_START;TCR|=TCR_TB.
3. Poll SR_TE until it is set1, or wait for the TE interrupt.
4. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, send a stop bit and abort the transmission.
5. TCR=TCR_TB.
6. Poll SR_RFuntil it equals 1, or wait for the reception complete RF interrupt.
7. Write 1 to SR_RF to clear the flag. Retrieve the received data from DBR .
8. Repeat steps 5-7; if it is the last data, setTCR = TCR_TB | TCR_NACK.
9. Set TCR = TCR_MA to send the stop bit.
10. Poll SR_UB until it equals 0, indicating the stop bit has been sent, then set TCR = 0.

7.1.3.9 Reception Process

1. When the address detection interrupt SADis triggered, automatically respond with ACK.
2. Write 1 to SR_SAD to clear the flag. Read SR_RWM; 1 indicates slave transmit, 0 indicates slave receive. Assuming the current value of SR_RWM is 1, enter the slave transmit state.
3. Write the data to be transmitted into DBR.
4. TCR=TCR_TB.
5. Poll SR_TE until it becomes 1,or wait for the TEinterrupt.
6. Write 1 to SR_TE to clear the flag. Check SR_NACK; if it is 1, abort the transmission.
7. Repeat steps 3-6 until the stop bit interrupt SSD is detected.
8. Write 1 to SR_SSD to clear the flag.

7.1.3.10 Slave Receive Procedure

1. When the address detection interrupt SADis triggered, automatically respond with ACK.
2. Write 1 to SR_SAD to clear the flag. Read SR_RWM; 1 indicates slave transmit, 0 indicates slave receive. Assuming the current value of SR_RWM is 0, enter the slave receive state.
3. TCR=TCR_TB.
4. Poll SR_RF until it equals 1, or wait for the reception complete RF interrupt.
5. Write 1 to SR_RF to clear the flag. Retrieve the received data from DBR.
6. Repeat steps 3-5; if the buffer is about to be full, set TCR = TCR_TB | TCR_NACK until the stop bit interrupt SSD is detected.
7. Write 1 to SR_SSD to clear the flag.

7.1.3.11 DMA Transfer

DMA transfer can be enabled when I2C is in master transmit or master receive mode. DMA operates only on the data segment and cannot be used to transfer the slave device address and the R/nW bit. I2C supports a maximum single DMA continuous data transfer of 511 bytes. If additional data transfer is required, DMA can be restarted.

After DMA is started, the transfer process does not require CPU involvement; the transfer is completed through direct interaction between the I2C and DMAC modules, generating the DMADONE interrupt. The I2C module includes a built-in 8-byte FIFO used to buffer data during DMA transfers. If a FIFO overflow occurs, an overflow interrupt OF or underflow interrupt UF will be generated. If an ACK=1 is received from the slave device during master transmission, the data transfer will be aborted, and the DMADONE interrupt will be generated.

The number of bytes transferred by DMA is configured by writing to DNR_NDT . The remaining number of bytes to be transferred can be obtained by reading DNR_NDT . When the DMADONE interrupt occurs, reading DNR_NDT and SR_NACK allows determination of whether the DMA transfer completed successfully.

Set Setting CR_LASTSTOP to 1 enables automatic transmission of a stop bit upon completion of the current DMA transfer. Setting CR_LASTNACK to 1 enables automatic acknowledgment with ACK=1 after the current DMA reception completes.

The procedure for using DMA for master transmission or master reception is as follows:

1. Left-shift the slave device address by 1 bit, append the least signifcant bit R/nW, and write it into DBR.
2. TCR=TCR_START;TCR|=TCR_TB。
3. Poll SR_TEuntil it becomes 1,or wait for the transmission complete TEinterrupt.
4. Write 1to SR_TE to clear the flag. Check SR_NACK; if it is 1, send a stop bit and abort the transmission.
5. Configure the DMAC module. Set the channel peripheral selection to the current I2C, data width to a single byte, and peripheral address to the current I2C FIFO register, then start the DMAC channel.
6. Configure the I2C DMA. Set DNR_NDT to the number of bytes to be transferred. If automatic stop bit transmission is required after DMA transfer completion, set CR_LASTSTOP to 1. If automatic ACK=1 response is required after DMA transfer completion, set CR_LASTNACK to 1. Set CR_DMAEN to 1 to enable DMA.
7. Wait for the DMADONE interrupt.
8. Write 1 to SR_DMADONE to clear the flag.
9. If DMA needs to be restarted, repeat steps 5–8.
10. Poll SR_UB until it is 0, indicating the stop bit has been transmitted.

7.1.3.12 Bus Exception Recovery

Due to electrical interference or device anomalies, the I2C bus may occasionally become stuck, resulting in continuous failures in I2C transmission or reception. When bus lock-up is suspected, the following methods can be used to attempt recovery.

1. Reset the I2C Module. Set CR_UR to 1, wait for 100us, then set it to 0.
2. If both BMR_SCL and BMR_SDA are 1, set CR_RSTREQ to 1, and poll CR_RSTREQ until it returns to 0. During this period, the I2C continuously transmits RCCR_RSTCYC clock cycles. The slave device may recover from an abnormal state upon detecting such bus signals.
3. If either BMR_SCL or BMR_SDA remains at 0 , it is suspected that the pull-up resistor has failed or the slave device is unresponsive; inspect the hardware circuitry or reset the slave device.

7.1.4 I2C Registers

Table 7-1: I2C Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			CR	Control register

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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 Note: Digital filter is added to analog filter. Digital filter will introduce delay on SCL and SDA processing, which is essential in hs-mode.
[11:10]			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.
[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

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
[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.

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	
[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.

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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.

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
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)
[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'h1B	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. 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'h75	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. 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
[31:5]			RSVD	

Continued on next page...

Table 7-1: I2C Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4:0]	rw	5'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

The chip contains a total of 4 SPI interfaces, among which SPI1 and SPI2 are located in HPSYS, with input/output connected to IO(PA), and can send requests to DMAC1. SPI3 and SPI4 are located in LPSYS, with input/output connected to IO(PB), and can send requests to DMAC2

7.2.1 Introduction

SPI supports 3 communication formats: SSP, SPI, and Microwire. SSP/SPI is a full-duplex communication protocol; the controller can be configured as either Master or Slave mode. Microwire is a half-duplex communication protocol; the controller can only be configured as Master mode. The SPI controller has built-in transmit and receive FIFO. The transmit FIFO and receive FIFO share the same address; reading from this address accesses the receive FIFO, while writing to it accesses the transmit FIFO. FIFO supports software access Mode and DMA access Mode.

7.2.2 Main Features

- Supports three communication formats: SSP, SPI, and Microwire
- Supports data widths from 8 to 32 bits
- In SPI format, clock polarity and phase can be configured via the SPO and SPH registers
- Chip select signal polarity is configurable
- FIFO depth is 32 bits×32 entries
- Both receive and transmit support DMA mode
- The maximum clock frequency of SPI in HPSYS is 48 MHz; The maximum clock frequency of SPI in LPSYS is 24MHz.

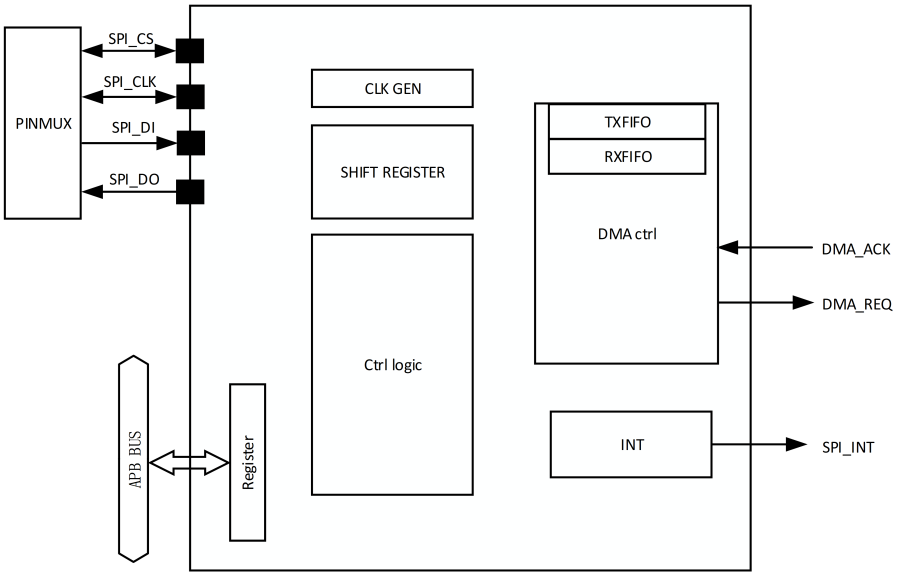


Figure 7-2: **SPI Block Diagram**

7.2.3 SPI Function Description

7.2.4 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. A transition of SPI_CS from 1 to 0 indicates the start of a data transfer, and a transition from 0 to 1 indicates the end of the data transfer. 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 configured for communication under the SSP protocol, SPI_CS indicates the start of a data frame transfer. A high pulse signals the start of a data frame transfer, with the pulse width corresponding to the duration of one bit transfer. This occurs before the start of each data frame transfer. 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 configured for communication under the Microwire protocol, SPI_CS serves as the chip select signal; a transition of SPI_CS from 1 to 0 indicates the start of a data transfer, and a transition from 0 to 1 indicates the end of the data transfer. Under the Microwire protocol, the SPI can only operate as a Master for communication, during which SPI_CS is an output signal driven internally.

SPI_CLK is the clock signal for serial communication; it is an output signal when communicating as a Master, and an input signal when communicating as a Slave.

SPI_DO is the data output signal. Data in theTX_FIFOis transmitted through SPI_DOinMSB firstorder. Whether configured as a Master or a Slave, it is always an output signal.

SPI_DI is the data input signal from external transmission. Data received on SPI_DI is stored in the RX_FIFO and read by the CPU or DMA . Whether configured as a Master or a Slave , it is always an input signal.

7.2.5 FIFO

The SPIcontroller contains TXFIFOand RXFIFO, each with a bit width of 32bits and a depth of 16 entries. Both FIFO scan be accessed by the CPU or DMA. From the perspective of address mapping, both FIFOs share the same address. Writing to this address writes data to the TXFIFO , while reading from this address retrieves the earliest data from the RX_FIFO .

A single access to the FIFO can only write or read one data entry (regardless of data bit width), and the data bit width for accessing the FIFO must be 32 bits. If the configured bit width is not 32 bits, the SPI controller will ignore the bits above the configured width within the 32-bit TXFIFO during transmission, and during reception, the SPI controller will pad zeros to the bits above the configured width before writing to the RXFIFO .

The FIFO can interact with the CPU via interrupts or software, or the CPU can poll the FIFO status registers to achieve interaction. The CPU writes data to the TXFIFO or reads data from the RXFIFO based on the interaction results.

FIFO interacts with the DMA controller via DMA_REQ to notify the DMA to write data to the TXFIFO or read data from the RXFIFO.

When FIFO interacts with the CPU via interrupts, the interrupt conditions are as follows:

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

When FIFO interacts with the DMA via DMA_REQ, the conditions for generating DMA_REQ are as follows:

- When the number of data items in the RXFIFO exceeds the value of RFT in the FIFO_CTRL register, the SPI controller generates a DMA_REQ to notify the DMA to read data from the RXFIFO.
- When the number of data items in the TXFIFO is less than the value of TFT plus 1 in the FIFO_CTRL register, the SPI controller generates a DMA_REQ to notify the DMA to write data into the TXFIFO.

In both cases, it is essential to configure settings appropriately to prevent RXFIFO overflow or TXFIFO underrun conditions.

7.2.6 Data Format

- SPI Format SPI is a full-duplex synchronous serial communication protocol, divided into four sub-modes based on the SPO and SPH settings in the TOP_CTRL register. SPO configures the polarity of SPI_CLK when the SPI controller is either disabled or in the IDLE state. When SPO is 0, the SPI_CLK polarity is low level, and the first edge is a rising edge; When SPO is 1, the SPI_CLK polarity is high level, and the first edge is a falling edge.SPH sets the timing of the clock edge for driving data and sampling data. When SPH is 0, the SPI controller samples data on SPI_DI at the first edge of SPI_CLK and drives SPI_DO at the second edge of SPI_CLK (by default, the MSB of the data to be transmitted is on SPI_DO); When SPH 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. Detailed communication timing is shown in Figures 7-3 and 7-4.

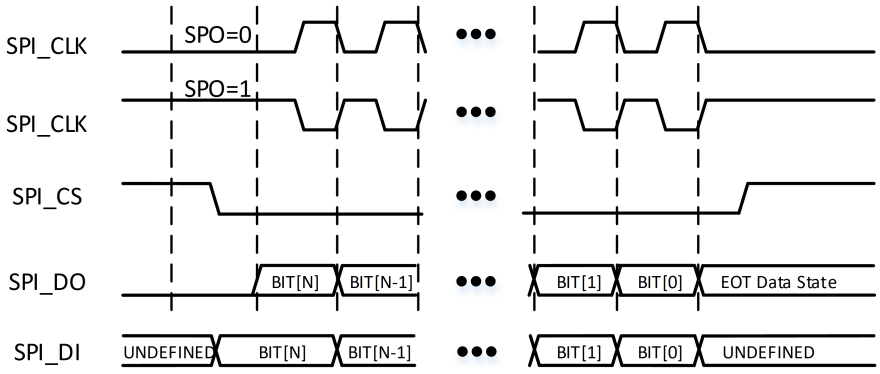


Figure 7-3: SPI communication when SPH is 0

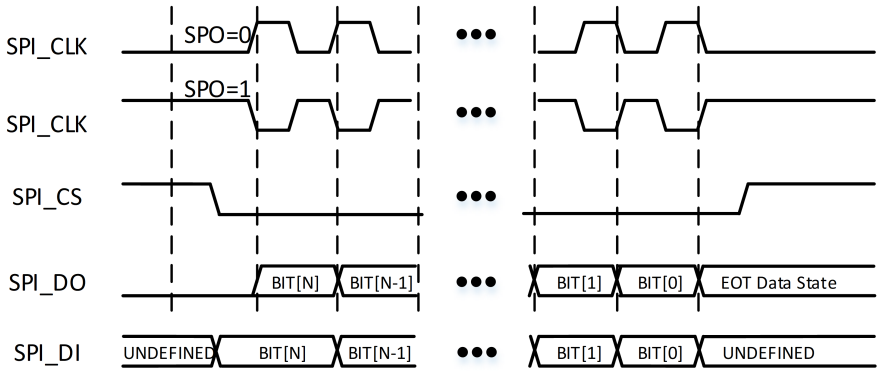


Figure 7-4: SPI communication when SPH is 1

The following describes the SPI protocol communication process with reference to Figure 7-3 under the conditions SPH=0 and SPO=0. When the SPI controller is disabled or in the 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 transfer and remains low until the current data frame transmission is complete. After half a SPI_CLK cycle, the data for this transfer of MSB is driven onto SPI_DO , and after another half SPI_CLK cycle, the SPI_CLK rises to generate the first rising edge and sample SPI_DI . Before all data for this transfer is fully transmitted, the SPI_CLK continues toggling at the configured frequency. After the final sampling, the SPI_CLK returns to low level, and after half a SPI_CLK cycle, the SPI_CS is pulled high to end this transfer.

When operating as the MASTER in communication, if multiple data entries are pending in the TXFIFO for transmission, theSPI controller will transmit the data continuously through consecutive transfers. During the process, the SPI_CS signal remains low. The transmission timing is shown in Figure 7-5。

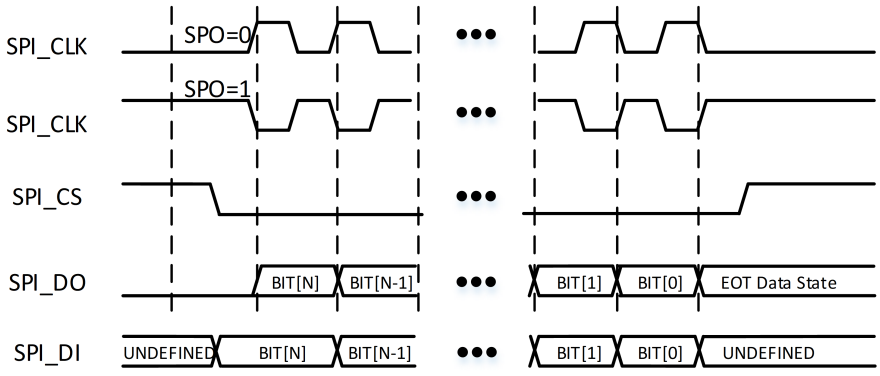


Figure 7-5: SPI Protocol Continuous Transmission Timing

In practical applications, scenarios exist where SPI_CS is pulled low once to transmit multiple data frames. In such cases, if the TXFIFOdata is not filled promptly, it may cause the SPI_CS line to be pulled high during transmission, resulting in communication failure. In these scenarios, special software control is required to maintain a stable low level on SPI_CS. Configuration details are provided in subsequent chapters.

- TI-SSP Format

  TI-SSP is a full-duplex synchronous serial port communication protocol. During data transmission, a high pulse on SPI_CS with a width of one clock cycle indicates the start of transmission. Subsequently, the data to be sent is driven onto SPI_DO at a rate of one bit per clock cycle in MSB first order. Data is driven onto the data line at the rising edge of SPI_CLK and sampled by the SPI controller at the falling edge of SPI_CLK. The single communication timing diagram is shown in Figure 7-6。

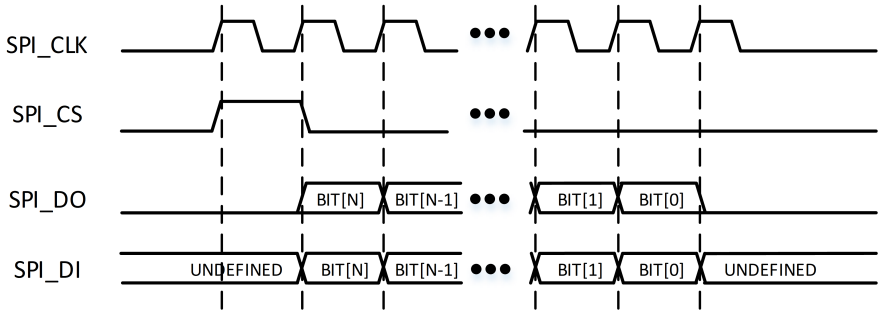


Figure 7-6: TI-SSP Protocol Communication Timing

When operating as a MASTER in communication, if multiple data entries are pending transmission in the TXFIFO, the SPI controller will transmit the data continuously through consecutive transfers. The transmission timing is shown in Figure 7-7.

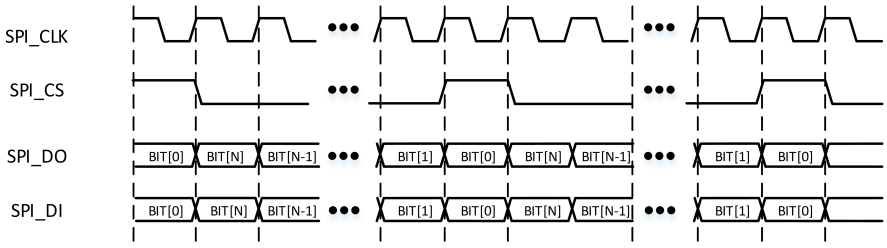


Figure 7-7: TI-SSP Protocol Continuous Communication Timing

If multiple data entries are transmitted consecutively, and data B immediately follows data A, the SPI_CS line is

pulled high during the last bit transmission of data A. One cycle, and the MSB of data B begins transmission at the next rising edge of SPI_CLK.

- Microwire protocol

  The Microwire protocol is a half-duplex protocol. The SPI controller supports communication only as a Master. During communication, the Master first transmits an 8-bit or 16-bit command word on SPI_DO. After one SPI_CLK cycle, the Slave returns data on SPI_DI. The timing for a single transfer is shown in Figure 7-8

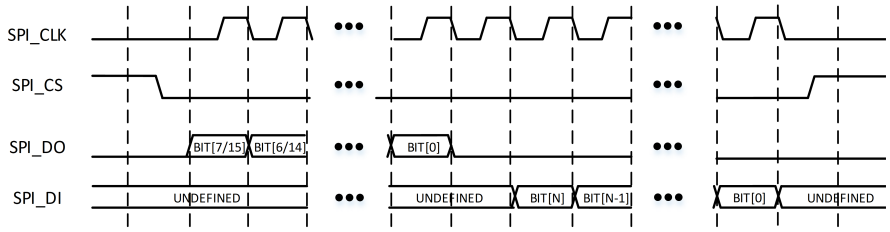


Figure 7-8: Timing diagram of a single communication in the Microwireprotocol

For consecutive transfers, the next command word is output on SPI_DO immediately after the last bit returned by the Slave, and SPI_CS remains low throughout all transfers. The timing for continuous transfers is shown in Figure 7-9.

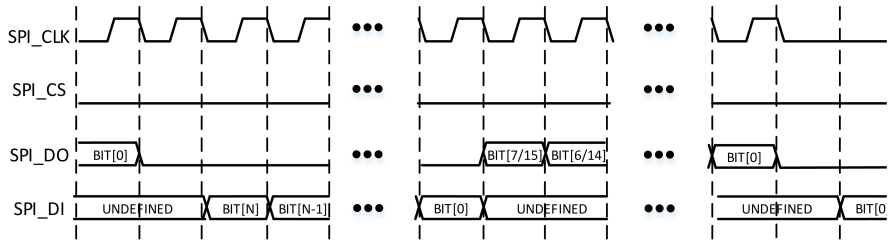


Figure 7-9: Microwire Protocol Continuous Transmission Timing

7.2.6.1 Related System Resources

SPI controller communication requires correct configuration of the PINMUX. Interfaces requiring PINMUX configuration include SPI_CS, SPI_CLK, SPI_DI, and SPI_DO. It should be noted that in three-wire half-duplex communication, SPI_DI and SPI_DO are configured on the same PIN , and the function of this SPI_DO PIN is set to SPI_DIO .

7.2.6.2 Communication Process

The following operations are required to perform SPI controller communication.

1. Configure the PINMUX to assign the corresponding PIN to the SPI communication function.
2. Communication Protocol Settings:
   - Configure the communication protocol via the FRF field in the TOP_CTRL register.
   - FRF=0:SPI Protocol
   - FRF=1:TI-SSP Protocol
   - FRF=2:Microwire Protocol
3. Master/Slave Mode Configuration:
   - SPI_CS/SPI_CLK can be configured independently for Master/Slave Mode. Master Mode indicates that the SPI controller drives the signals, whereas Slave Mode indicates that the SPI controller receives signals driven externally.

- Configure the Master/Slave Mode of SPI_CS via the SFRMDIR field in the TOP_CTRL Registers:
- SFRMDIR=0:Master Mode, SFRMDIR=1:Slave Mode
- Configure the Master/Slave Mode of SPI_CLK via the SCLKDIR field in the TOP_CTRL Registers:
- SCLKDIR=0:Master Mode,SCLKDIR=1:SlaveMode
- Under normal circumstances,SFRMDIR and SCLKDIR are configured to the same mode.

4. Clock frequency setting:
   - The procedure for setting the SPIclock is as follows:
     (a) Configure the clock division ratio in the CLK_DIV field of the CLK_CTRL register.
     (b) Select the clock source in the CLK_SEL field of the CLK_CTRL register, where CLK_SEL=0 selects the divided clock, and CLK_SEL=1 selects the source clock. Select the source clock.
   - The SPI clock is derived from either the source clock or the divided source clock. The frequency of the divided clock equals the source clock frequency divided by CLK_DIV. The source clock frequency is PCLK_HPSYS in the HPSYS and PCLK_LPSYS in the LPSYS.
5. Data Width Configuration:
   - Supports 8-32 bit data width. The data bit width can be configured by setting the DSS field in the TOP_CTRL register, where data bit width = DSS + 1.
6. Data Operation:
   - Data is divided into transmit data and receive data. Data to be transmitted is written into the TXFIFO, and received data is read from the RXFIFO.
   - Data operations can be performed either by the CPU running software to directly access the FIFO, or via DMA.
   - When data is operated by the CPU, generally the SPI controller notifes the CPU via interrupts to write to theTXFIFO or read from the RXFIFO. The corresponding interrupt is enabled by setting the TIE bit in the register to 1. Once enabled, when the number of data entries in the TXFIFO is less than or equal to the value in the register of When the TFT value is reached, the SPI controller issues a TX interrupt notifcation to the CPU. The interrupt corresponding to RXFIFO is enabled by setting the RIE bit in the register to 1. Once enabled, when the number of data items in the RXFIFO exceeds the RFT value in the register, theSPI controller issues an RX interrupt notifcation to the CPU.
   - The CPU can also operate the FIFOby polling the FIFO status.FIFO STATUS register

STATUS register	Meaning
TNF	0：TXFIFO is full；1：TXFIFO is not full.
TFL	Number of data items in the TXFIFO. When the read value is 0, the TXFIFO is either full or empty; it is necessary to determine this in conjunction with the value of TNF
TUR	1:TXFIFO underrun occurred, meaning the TXFIFO was empty when the SPI controller performed a TXFIFO read operation.
RNE	0: RXFIFO is empty; 1: RXFIFO is not empty.
RFL	Number of data items in the RXFIFO. When the read value is 0x1F, the RXFIFO is either empty or full; determination requires combining this with the value of RNE.
ROR	1: RXFIFO overrun occurred, meaning the RXFIFO was full when the SPI controller performed a RXFIFO write operation.

- The CPU can add data to be transmitted into the TXFIFO by reading the STATUS register, provided that the TXFIFO is not full; Received data can be read from the RXFIFO when it is not empty.
- When using DMA to operate data, the SPI controller initiates data operations by issuing a request to DMA to start the DMA. Write to RSRE. 1 Enable the DMA function for RX data by writing to RSRE. When the data in the RX FIFO exceeds RFT,the SPI will issue a DMA request. Write to TSRE. 1 Enable the DMA function for TX data by writing to TSRE. When the number of data items in the TXFIFO is less than or equal to the TFT value in the register, the SPI will issue a DMA request.

7. Enable the SPIcontroller
    - Set SSEin the TOP_CTRL register to 1 to enable the SPI controller.
8. Disable the SPI controller
    - Check the BUSY bit in the STATUS register. If BUSY is 0, indicating that the SPI controller is currently idle, then write 0 to SSE. The SPI controller can be disabled.
9. SPI_CS Signal Control
    - In certain communication scenarios, the SPI_CS signal must remain low throughout the transmission of multiple data frames. By default, if the TXFIFO becomes empty during the process, it may cause the SPI_CS to be pulled high intermittently, then pulled low again once the TXFIFO is no longer empty. Software intervention can be employed to ensure the SPI_CS signal remains stably low during communication. Before initiating communication, set the HOLD_FRAME_LOW bit in the TOP_CTRL register to 1 to ensure the SPI_CS signal remains low throughout the communication. Note to reset HOLD_FRAME_LOW to 0 after the communication completes to prepare for subsequent communications.

- configure the SPIcontroller according to the following procedure:
    1. Configure pinmux
    2. Configure the communication protocol
    3. Configure the clock frequency
    4. Configure the data bit width
    5. Configure the master/slave mode
    6. Enable the SPI controller
    7. Access the FIFO to start transmission and reception
    8. Complete transmission and reception, then disable the SPI controller

7.2.6.3 Receive-Only Mode

The SPI controller supports Receive-Only Mode. In this mode, when the data format is SPI or TI-SSP, whether or not there is data pending transmission in the TXFIFO, the SPI controller will drive the SPI_CLK to toggle, and the data received on SPI_DI will be stored in the RXFIFO. The configuration for using Receive-Only Mode is as follows:

- Configure the RWOT_CCM register, where the register value specifes the required number of SPI_CLK cycles.
- Set both SET_RWOT_CYCLE and RWOT_CYCLE bits in the RWOT_CTRL register to 1 to enable the RWOT counter.
- Set the RWOT bit in the RWOT_CTRL register to 1 to enable Receive-Only mode.

After completing the above configurations, 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 the RXFIFO.

7.2.6.4 Three-Wire Mode

The SPIcontroller supports three-wire mode via software assistance. Three-wire mode enables half-duplex communication. The required configuration and usage procedure are as follows:

1. Configure the PINMUX to communicate via a PIN with SPI_DIO functionality, and select the PIN function as SPI_DIO.
2. Select the SPI protocol, configure the clock frequency, set the data bit width, and configure the master/slave mode.
3. Set Set SPI_TRI_WIRE_EN in the SPI controller register TRIWIRE_CTRL to 1 to enable the SPI controller.
4. When transmitting data, set TXD_OEN in the register TRIWIRE_CTRL to 0. Write the data to be transmitted into the TXFIFO . In three-wire mode, during data transmission, the SPI controller does not write to the RXFIFO ; therefore, software does not need to process the RXFIFO after transmission.

5. When receiving data, set TXD_OEN in the register TRIWIRE_CTRL 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 to TXFIFOas the data to be received in order to drive SPI_CLK.
   - UseReserve-Only Mode; the procedure is as follows:
     (a) Disable the enable of the SPI controller.
     (b) Set the RWOT_CCM register, whose value specifes the required number of SPI_CLK cycles.
     (c) Set both SET_RWOT_CYCLE and RWOT_CYCLE bits in the RWOT_CTRL register to 1 to enable the RWOT counter.
     (d) Set the RWOT bit in the RWOT_CTRL register to 1 to enable Receive-Only mode.
     (e) Enable the SPI controller after completing the above configurations.
     (f) Read data from the RXFIFO.
6. Disable the SPIcontroller after completing transmission and reception.

Generally, this half-duplex communication mode requires the SPI_CS line to remain steadily low during the operation. To maintain a steady low level, please refer to the SPI_CS signal control settings.

7.2.7 SPI Registers

Table 7-2: SPI Register Map

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

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	
[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

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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.

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[19:15]	r	5'h0	RFL	Receive FIFO Level This field is the number of entries minus one in RXFIFO. When the value 0x1F 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
[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)
[11:7]	r	5'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	

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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
[31:8]			RSVD	
[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

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	SPI_TRI_WIRE_EN	SPI_THREE_WIRE_MODE_EN 0: normal mode 1: enable TRI-WIRE mode

7.3 USART

The chip contains a total of 5 USARTs, among which USART 1 and USART 2 are located in HPSYS, with input/output connected to IO(PA), and can send requests to DMAC1; USART3, USART4, and USART5 are located in LPSYS, with input/output connected to IO(PB), capable of sending requests to DMAC2.

The Universal Asynchronous Receiver/Transmitter supports full-duplex mode, providing baud rates up to 6 Mbps and multiple configurable data formats, offering a flexible and effcient data exchange method for communication with external standardized devices. It also supports DMA, enabling multi-packet transmission and reception.

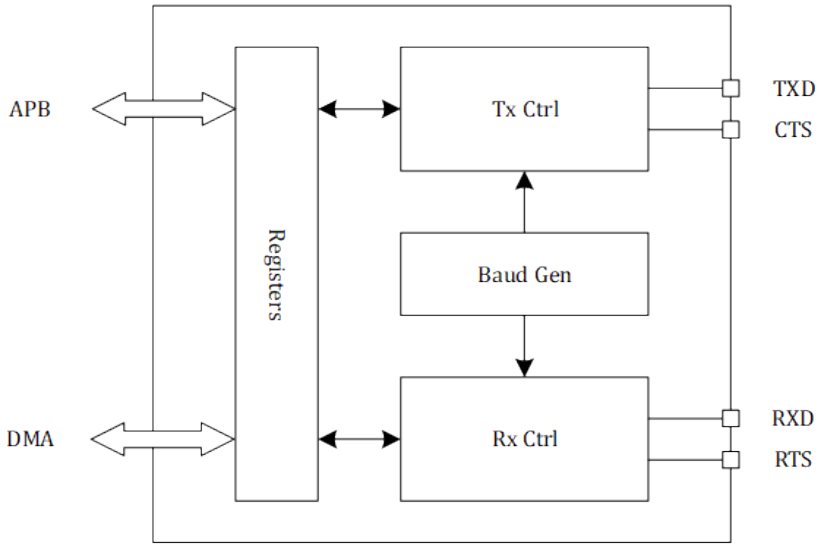


Figure 7-10: Universal Asynchronous Receiver/Transmitter

Key features of the Universal Asynchronous Receiver/Transmitter:

- Full-duplex asynchronous communication
- Configurable 16× or 8× oversampling, with options prioritizing frequency or clock tolerance
- Flexible baud rate configuration; when the input clock is 48 MHz and the oversampling rate is 16, the baud rate is 3 Mbps
- Configurable packet length (7/8/9 bits)
- Configurable stop bits (1/2 bits)
- Hardware flow control (CTS/RTS)
- DMA multi-packet transmission and reception
- Reception parity check and transmission parity generation
- Reception and transmission interrupts, as well as other error interrupts

Baud rate calculation description

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

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

7.3.1 USART Registers

Table 7-3: USART Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	Control Register 1
[31:29]			RSVD	
[28:27]	rw	2'h0	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

Continued on next page...

Table 7-3: USART Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	
[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

Continued on next page...

Table 7-3: USART Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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	

Continued on next page...

Table 7-3: USART Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
[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.

Continued on next page...

Table 7-3: USART Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
[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
[31:8]			RSVD	
[7:4]	rw	4'h2	RTSBIT	assert RTS ahead of the frame completion (in number of bits)
[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

7.4 PTC

The chip contains a total of 2 PTCs, with PTC1 located in HPSYS and PTC2 located in LPSYS.

7.4.1 Introduction

PTC (Peripheral Task Controller) is an independent peripheral controller capable of automatically coordinating and controlling various peripherals without waking the CPU. Based on event triggers from selected peripherals, PTC can automatically modify the operating modes or states of peripherals and chain these tasks into an automatically triggered task sequence, thereby executing complex and fast-response task chains. During the execution of the task chain, the CPU can remain in sleep mode continuously, thereby effectively reducing power consumption.

PTC comprises a total of 8 channels, each capable of selecting an independent trigger source and configuring independent tasks. Executable tasks include two categories: directly writing specifed data to a specifed address; Reading the content of a specifed address, performing XOR / AND / OR / ADD operations with the specifed data, and then writing it back. Upon completion of each channel task, a trigger signal can be generated to trigger tasks on other channels.

7.4.2 Main Features

- 8 independently configurable channels can operate simultaneously
- Each channel trigger can be selected from 128 trigger sources, including the PTC's own trigger sources.
- Accessible AHB and APB peripheral address space, supporting only word-aligned access.
- Supports direct data writing or read-modify-write operations.

- Supports 32-bit XOR / AND / OR / ADD operations.
- Fixed priority arbitration: the smaller the channel number, the higher the priority.
- 4 word register space allocated for data buffering.

7.4.3 Function Description

7.4.3.1 Channel Trigger

Each channel can select one trigger source from 128 available sources; the selection register is TCRx_TRIGSEL. Trigger sources are typically generated by various peripherals to indicate specific peripheral events, such as DMA transfer completion, IO toggle, timer update, etc., and also include the PTC's own channel completion events. When selecting IO input signals as trigger sources, only 4 out of every 32 IOs can be simultaneously selected as trigger sources. The specific selection is configured via PTC registers such as GPIO31_0 and GPIO63_32. IO trigger sources do not support debounce.

When TCRx_TRIGSEL is 0, the channel is disabled.

Table 7-4: 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	AES_DONE	AES_BUSERR	I2S2_OF	I2S2_UF	I2S1_OF	TRNG_RANDGEN	TRNG_SEEDGEN	TRNG_LOCKUP	112
111	EPIC_LINEHIT	EPIC_DONE	/	LCDC1_UDR	LCDC1_LINEHIT	LCDC1_OF	LCDC1_FMARK	LCDC1_DONE	104
103	/	NNACC_DONE	EZIP_ROW	EZIP_DONE	EXTDMA_OF	EXTDMA_TE	EXTDMA_HT	EXTDMA_TC	96
95	/	USB_RX	USB_TX	USB_INT	USART2_TXBYTE	USART2_RXBYTE	USART1_TXBYTE	USART1_RXBYTE	88
87	/	/	/	/	SPI2_START	SPI2_DONE	SPI1_START	SPI1_DONE	80
79	I2C3_RF	I2C3_TE	I2C3_DMADONE	I2C3_INT	I2C2_RF	I2C2_TE	I2C2_DMADONE	I2C2_INT	72
71	I2C1_RF	I2C1_TE	I2C1_DMADONE	I2C1_INT	PA95_64_D	PA95_64_C	PA95_64_B	PA95_64_A	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_C1INT7	MAILBOX1_C1INT6	MAILBOX1_C1INT5	MAILBOX1_C1INT4	MAILBOX1_C1INT3	MAILBOX1_C1INT2	MAILBOX1_C1INT1	MAILBOX1_C1INT0	48
47	BUSMON1_OF8	BUSMON1_OF7	BUSMON1_OF6	BUSMON1_OF5	BUSMON1_OF4	BUSMON1_OF3	BUSMON1_OF2	BUSMON1_OF1	40
39	DMAC1_TC8	DMAC1_HT8	DMAC1_TC7	DMAC1_HT7	DMAC1_TC6	DMAC1_HT6	DMAC1_TC5	DMAC1_HT5	32
31	DMAC1_TC4	DMAC1_HT4	DMAC1_TC3	DMAC1_HT3	DMAC1_TC2	DMAC1_HT2	DMAC1_TC1	DMAC1_HT1	24
23	SDMMC2_ERR	SDMMC2_INT	SDMMC2_TC	SDMMC2_CC	SDMMC1_ERR	SDMMC1_INT	SDMMC1_TC	SDMMC1_CC	16
15	BTIM2_UPDATE	BTIM1_UPDATE	GPTIM2_CH4	GPTIM2_CH3	GPTIM2_CH2	GPTIM2_CH1	GPTIM2_TRIG	GPTIM2_UPDATE	8
7	GPTIM1_CH4	GPTIM1_CH3	GPTIM1_CH2	GPTIM1_CH1	GPTIM1_TRIG	GPTIM1_UPDATE	/	0	0

Table 7-5: PTC2 Trigger Source

TRIGSEL	PTC1 trigger source								TRIGSEL
127	PTC2_CH8	PTC2_CH7	PTC2_CH6	PTC2_CH5	PTC2_CH4	PTC2_CH3	PTC2_CH2	PTC2_CH1	120
119	LPCOMP2_SENS	LPCOMP1_SENS	SDADC_DSAMPLE	TSEN_DONE	/	/	RF_UNLOCK	RF_PKTDET	112
111	/	BLE_RCCALDONE	BLE_SLPDONE	BLE_SLPSTART	/	/	BLE_PKTDET	BLE_CRCERR	104
103	BLE_RXDONE	BLE_PHYRXSTART	BLE_RFRXSTART	BLE_TXDONE	BLE_PHYTXSTART	BLE_RFTXSTART	BLE_EVTDONE	BLE_EVTSTART	96
95	LCDC2_FMARK	LCDC2_DONE	USART5_TXBYTE	USART5_RXBYTE	USART4_TXBYTE	USART4_RXBYTE	USART3_TXBYTE	USART3_RXBYTE	88
87	/	/	/	/	SPI4_START	SPI4_DONE	SPI3_START	SPI3_DONE	80
79	I2C6_RF	I2C6_TE	I2C6_DMADONE	I2C6_INT	I2C5_RF	I2C5_TE	I2C5_DMADONE	I2C2_INT	72
71	I2C4_RF	I2C4_TE	I2C4_DMADONE	I2C4_INT	/	/	/	/	64
63	PB63_32_D	PB63_32_C	PB63_32_B	PB63_32_A	PB31_0_D	PB31_0_C	PB31_0_B	PB31_0_A	56
55	MAILBOX2_C1INT7	MAILBOX2_C1INT6	MAILBOX2_C1INT5	MAILBOX2_C1INT4	MAILBOX2_C1INT3	MAILBOX2_C1INT2	MAILBOX2_C1INT1	MAILBOX2_C1INT0	48
47	BUSMON2_OF8	BUSMON2_OF7	BUSMON2_OF6	BUSMON2_OF5	BUSMON2_OF4	BUSMON2_OF3	BUSMON2_OF2	BUSMON2_OF1	40
39	DMAC2_TC8	DMAC2_HT8	DMAC2_TC7	DMAC2_HT7	DMAC2_TC6	DMAC2_HT6	DMAC2_TC5	DMAC2_HT5	32
31	DMAC2_TC4	DMAC2_HT4	DMAC2_TC3	DMAC2_HT3	DMAC2_TC2	DMAC2_HT2	DMAC2_TC1	DMAC2_HT1	24
23	/	/	GPTIM5_CH4	GPTIM5_CH3	GPTIM5_CH2	GPTIM5_CH1	GPTIM5_TRIG	GPTIM5_UPDATE	16
15	BTIM4_UPDATE	BTIM3_UPDATE	GPTIM4_CH4	GPTIM4_CH3	GPTIM4_CH2	GPTIM4_CH1	GPTIM4_TRIG	GPTIM4_UPDATE	8
7	GPTIM3_CH4	GPTIM3_CH3	GPTIM3_CH2	GPTIM3_CH1	GPTIM3_TRIG	GPTIM3_UPDATE	/	0	0

7.4.3.2 Channel Tasks

The tasks executable by the channel comprise two categories, configured via TCRx_OP. When TCRx_OP is 0, after the channel is triggered, the data in the TDRx register is directly written to the address pointed to by the TARx register. When

TCRx_OP is 0x4 to 0x7, after the channel is triggered, the data at the address pointed to by the TARx register is first read out, then an XOR, AND, OR, or ADD operation is performed with the data in the TDRx register before writing back to the address pointed to by the TARx register. The TARx register typically points to the peripheral's register address; therefore, the primary purpose of the channel task is to automatically configure the peripheral upon the occurrence of a specific event from the trigger source, thereby altering the peripheral's operating mode or state.

After the channel writes to the address pointed to by the TARx register, a channel completion event is generated, which can serve as the PTC trigger source for another channel. Simultaneously, the ISR_TCIFx flag is set, and when IER_TCIEx is 1, an interrupt is generated.

If the address pointed to by the TARx register is a bus address inaccessible by the PTC, a bus error occurs during channel access, setting the ISR_TEIFx flag, and when IER_TEIE is 1, an interrupt is generated.

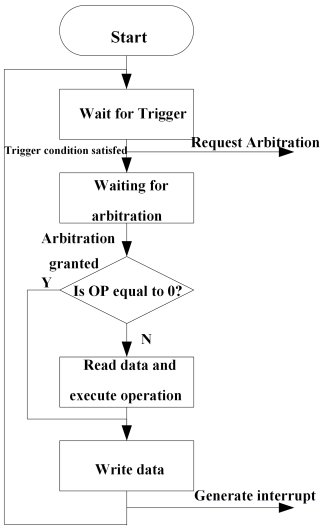


Figure 7-11: PTC Channel Execution Flowchart

7.4.3.3 Channel Arbitration

PTC comprises a total of 8 channels, arbitrated based on the principle that a smaller channel number corresponds to a higher priority. Each channel enters arbitration upon being triggered. When multiple channels enter arbitration simultaneously, the channel with the smallest number is granted permission to initiate task execution first, while the remaining channels remain suspended until the task of the smallest-numbered channel is completed, after which arbitration is conducted again.

7.4.4 PTC Registers

Table 7-6: PTC Register Map

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

Continued on next page...

Table 7-6: PTC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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
0x0C			**TCR1**	task 1 control register
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x10			**TAR1**	task 1 address register
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x14			**TDR1**	task 1 data register

Continued on next page...

Table 7-6: PTC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	DATA	data value for task operation
0x18			**TCR2**	
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x1C			**TAR2**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x20			**TDR2**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x24			**TCR3**	
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x28			**TAR3**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x2C			**TDR3**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x30			**TCR4**	
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x34			**TAR4**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x38			**TDR4**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x3C			**TCR5**	
[31:19]			RSVD	

Continued on next page...

Table 7-6: PTC Register Map (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 0: task will not be triggered others: task will be triggered by selected source
0x40			**TAR5**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x44			**TDR5**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x48			**TCR6**	
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x4C			**TAR6**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x50			**TDR6**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x54			**TCR7**	
[31:19]			RSVD	
[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 not be triggered others: task will be triggered by selected source
0x58			**TAR7**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x5C			**TDR7**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x60			**TCR8**	
[31:19]			RSVD	

Continued on next page...

Table 7-6: PTC Register Map (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 0: task will not be triggered others: task will be triggered by selected source
0x64			**TAR8**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x68			**TDR8**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x70			**MEM1**	temporary memory 1
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0x74			**MEM2**	temporary memory 2
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0x78			**MEM3**	temporary memory 3
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0x7C			**MEM4**	temporary memory 4
[31:0]	rw	32'h0	DATA	memory to store temporary variables
0x80			**GPIO31_0**	
[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
0x84			**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
0x88			**GPIO95_64**	
[31:29]			RSVD	
[28:24]	rw	5'h0	SELD	select trigger D of GPIO 95~64
[23:21]			RSVD	

Continued on next page...

Table 7-6: PTC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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

7.5 USB

USB is located in HPSYS.

This chip integrates a full-speed (FS) USB 2.0 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

GPADC is located in LPSYS and can send requests to DMAC2.

8.1.1 Introduction

GPADC is a 10-bit precision SAR ADC that supports an input voltage range of 0-1.1V and outputs 10-bit data. The input voltage can be single-ended or differential, and output data can be accessed via the APB bus or DMA interface.

8.1.2 Key Features

- Input voltage range: 0~1.1V, 10-bitresolution
- Supports single-ended Mode and differential Mode
- Supports 8 single-ended analog inputs or 4 pairs of differential analog inputs
- Supports single measurement mode and continuous measurement mode
- Each measurement can be divided into 8 time slots, with each time slot individually configurable for an analog input channel
- Supports software (register write) and hardware (e.g., timer) trigger methods
- Supports DMA channels
- Sampling frequency is configurable, with a maximum sampling frequency of 3 MHz
- An interrupt is generated upon conversion completion

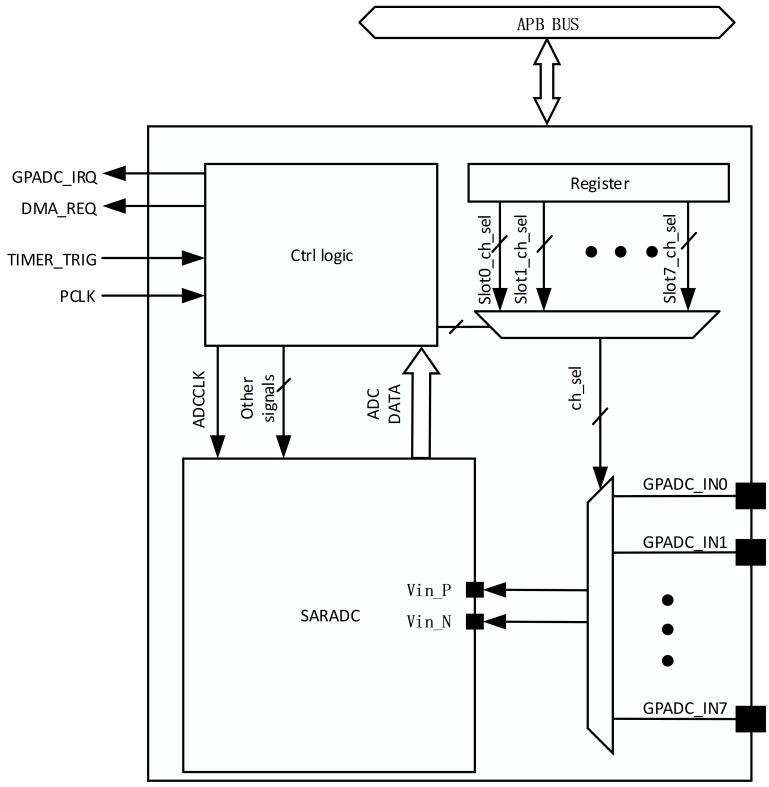


Figure 8-1: GPADC Block Diagram

8.1.3 Function Description

8.1.3.1 GPADC Clock Generation

The GPADC clock is generated by dividing the PCLK frequency. The ADCCLK frequency is set by configuring the ADC_CLK_DIV field in the ADC_CTRL_REG. The calculation formula is:

$$f_{ADCCLK} = f_{PCLK}/(ADC\_CLK\_DIV + 1)$$

Since the ADCCLK is derived by dividing the PCLK frequency, it is necessary to verify the PCLK frequency when setting the ADCCLK frequency.

8.1.3.2 Slot Configuration

Each round of GPADC sampling consists of 8 slots operating sequentially. Each slot can be independently enabled or disabled by setting the SLOT_EN bit in the ADC_SLOT*_REG register. The input channels for each slot can be independently configured by setting the PCHNL_SEL and NCHNL_SEL fields in the ADC_SLOT*_REG register.

8.1.3.3 Single-Ended/Differential Mode

Setting the ANAU_GPADC_SE bit to 1 in the ADC_CFG_REG register configures the GPADC for single-ended input mode, where only the PCHNL_SEL field in each slot's configuration register needs to be set to select the input channel.

Setting the ANAU_GPADC_SE bit in the ADC_CFG_REG register to 0 configures the GPADC for differential input mode.

The input channels corresponding to Vin_P and Vin_N are selected by configuring PCHNL_SEL and NCHNL_SEL in the respective slot configuration registers.

8.1.3.4 Input Channel Selection

The GPADC supports 8 input channels. By configuring PCHNL_SEL and NCHNL_SEL in the ADC_SLOT*_REG registers, the input channels to be sampled in each slot can be specifed.

In single-ended mode, only PCHNL_SEL needs to be configured to select the input channel.

8.1.3.5 Sampling Mode

If ADC_OP_MODE in the ADC_CTRL_REG register is set to 0, the GPADC operates in single-sample mode. In this mode, each time the GPADC is triggered, GPADC will complete one round of sampling according to the configuration of each time slot, then return to the waiting trigger state.

If ADC_OP_MODE in ADC_CTRL_REG is set to 1, GPADC operates in continuous sampling mode. In this mode, after each GPADC start, GPADC will cyclically sample according to the configuration of each time slot. Setting ADC_STOP to 1 in ADC_CTRL_REG causes GPADC to return to the waiting trigger state.

8.1.3.6 Start GPADC

- Start by writing to the register
  Setting ADC_START to 1 in the ADC_CTRL_REG register starts the GPADC.
  After triggering GPADC, if it is in single sampling mode, it returns to the waiting trigger state after completing one round of sampling. If the GPADC is in continuous sampling mode, the ADC_STOP bit in the ADC_CTRL_REG must be set to 1 to return the GPADC to the waiting trigger state
- Timer Trigger
  GPADC supports TIMER trigger. To enable the TIMER trigger function, the TIMER_TRIG_EN bit in the ADC_CTRL_REG must be set to 1. There are a total of 8 trigger sources, which can be selected via the TIMER_TRIG_SRC_SEL bit in the ADC_CTRL_REG. The correspondence is shown in the following table:

TIMER_TRIG_SRC_SEL	TRIG_SRC
0	GPTIM3 TRGO
1	GPTIM4 TRGO
2	GPTIM5 TRGO
3	BTIM3 TRGO
4	BTIM4 TRGO
5	GPTIM3 CH0 Output
6	GPTIM3 CH1 Output
7	GPTIM3 CH2 Output

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

8.1.3.7 Data Access

The data obtained from GPADC conversion can be accessed by the following two methods:

- Register Read

  Software can directly read the GPADC conversion results by accessing the registers. Data corresponding to each time slot is stored in the registers ADC_RDATA*, with the correspondence between registers and time slot data 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 for each time slot is right-aligned to bit 0 or bit 16 of the register. Using ADC_RDATA0 as an example to illustrate the alignment of time slot data within the register, the alignment method is shown in the table below:

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

- DMA Mode

  The GPADC conversion results are read via DMAC2 within LPSYS. The usage procedure is as follows:

  Set the DMA_EN bit of the register ADC_CTRL_REG to 1.

  Set the source address of the DMA is set to 0x40056034. The alignment mode of the ADC data at this address is:

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

  For other DMA settings, refer to the DMAC section.

8.1.3.8 Notifcation Mechanism

GPADC generates an interrupt upon completion of sampling conversion, which is reported to the CPU.

This interrupt can be masked by setting the GPADC_IMR bit to 1 in the GPADC_IRQregister.

This interrupt can be cleared by setting the GPADC_ICR bit to 1 in the GPADC_IRQregister.

8.1.3.9 configuration Startup Procedure

The typical procedure for using the GPADC is as follows:

- Configure PINMUX.
- Configure the GPADC clock frequency, input channel selection, and related parameters.
- Set Set the EN bit to 1 in the BGR register of the TSEN module to enable the Bandgap.
- Set Set the ANAU_IARY_EN bit to 1 in the ANAU_ANA_TP register of the TSEN module to enable the Iarray.
- Set both ANAU_GPADC_LDOREF_EN and ANAU_GPADC_LDOCORE_EN bits to 1 in the ADC_CFG_REG1 register to enable the LDO supplying voltage to the GPADC.

- Set the FRC_ADC_EN bit to 1 in the ADC_CFG_REG1 register to enable the GPADC trigger, start sampling, and read data.
- After sampling is complete, clear the FRC_EN_ADC bit to 0 in the ADC_CTRL_REG register to disable the GPADC module. In continuous sampling mode, it is necessary to first set the ADC_STOP bit in the register ADC_CTRL_REG to 1 and then to 0 to interrupt the sampling process.
- Set both ANAU_GPADC_LDOREF_EN and ANAU_GPADC_LDOCORE_EN bits to 1 in the ADC_CFG_REG1 register to enable the LDO supplying voltage to the GPADC.
- Bandgap and Iarrayare shared with TSEN; it is recommended not to disable them. If disabled, the operational status of TSEN must be carefully monitored.

The process requires meeting certain circuit stabilization times.

- After Pinmux configuration, the voltage at the PAD connected to the input channel requires a stabilization time determined by the RC value connected to the PAD.
- After enabling the bandgap,wait at least 50us.
- After enabling the reference voltage provided by the LDO,the LDO requires at least 200us to stabilize.
- Enable GPADC; a startup time of at least 50us is required after enabling.
- Finally, the GPADC is started by writing to the register trigger or by Timer trigger.

8.1.4 GPADC Registers

Table 8-1: GPADC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ADC_CFG_REG1**	ADC Analog Config Register 1
[31:28]			RSVD	
[27:24]	rw	4'h4	ANAU_GPADC_CMM	Tune CDAC CM voltage 375mV range (increasing) / 25mV step, 8: for 0.5V Vcm,in
[23:21]	rw	3'h3	ANAU_GPADC_CMPCL	Tune ADC comparator CL= 3: 40f, range: 10fF (0) ~ 80fF (7) / 10fF step
[20:19]	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)
[18]	rw	1'h0	ANAU_GPADC_LDOCORE_EN	Enable LDORF for ADC
[17]	rw	1'h0	ANAU_GPADC_LDOREF_EN	Enable LDORF for ADC VREF
[16:13]	rw	4'hA	ANAU_GPADC_LDOCORE_SEL	Set reference voltage for LDOCORE, range = 0.35V(0) ~ 0.65V(15), step = 20mV
[12:9]	rw	4'h5	ANAU_GPADC_LDOVREF_SEL	Set reference voltage for LDOREF, range = 0.35V(0) ~ 0.65V(15), step = 20mV
[8:6]	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
[5:3]	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
[2]	rw	1'h1	ANAU_GPADC_ATTN3X	Attenuate input voltage by 3 for GPADC
[1]	rw	1'h0	ANAU_GPADC_MUTE	Short GPADC input to CMREF, i.e., VREF/2
[0]	rw	1'h0	ANAU_GPADC_SE	Set GPADC in single-ended mode, signal range at P-input: 0 ~ VREF
0x04			**ADC_Slot0_REG**	ADC Slot0 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	
0x08			**ADC_Slot1_REG**	ADC Slot1 Config Register

Continued on next page...

Table 8-1: GPADC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[31:26]			RSVD	
[25:16]	r	10'h0	SLOT1_RDATA	
[15:10]			RSVD	
[9:0]	r	10'h0	SLOT0_RDATA	
0x28			**ADC_RDATA1**	ADC Read Data1
[31:26]			RSVD	
[25:16]	r	10'h0	SLOT3_RDATA	
[15:10]			RSVD	
[9:0]	r	10'h0	SLOT2_RDATA	
0x2C			**ADC_RDATA2**	ADC Read Data2

Continued on next page...

Table 8-1: GPADC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:26]			RSVD	
[25:16]	r	10'h0	SLOT5_RDATA	
[15:10]			RSVD	
[9:0]	r	10'h0	SLOT4_RDATA	
0x30			**ADC_RDATA3**	ADC Read Data3
[31:26]			RSVD	
[25:16]	r	10'h0	SLOT7_RDATA	
[15:10]			RSVD	
[9:0]	r	10'h0	SLOT6_RDATA	
0x34			**ADC_DMA_RDATA**	ADC Read Data For DMA
[31:27]			RSVD	
[26:16]	r	11'h0	DMA_RDATA_RAW	
[15:11]			RSVD	
[10:0]	r	11'h0	DMA_RDATA	
0x38			**ADC_CTRL_REG**	ADC Control Register
[31:22]			RSVD	
[21]	rw	1'b0	DMA_DATA_SEL	0: combined data 1: raw data
[20]	rw	1'b0	TIMER_TRIG_TYP	0: pulse no edge detect needed 1: level,need edge detect
[19:17]	rw	3'h0	TIMER_TRIG_SRC_SEL	
[16]	rw	1'b0	FRC_EN_ADC	
[15]	rw	1'b0	CHNL_SEL_FRC_EN	
[14]	rw	1'b1	TIMER_TRIG_EN	
[13]			RSVD	
[12]	rw	1'b1	DMA_EN	
[11:7]	rw	5'h2	ADC_CLK_DIV	
[6:3]	rw	4'h6	INIT_TIME	
[2]	rw	1'b0	ADC_STOP	
[1]	w1s	1'b0	ADC_START	
[0]	rw	1'b0	ADC_OP_MODE	0: finite 1: inifinite
0x40			**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 SDADC

SDADC is located in LPSYS.

8.2.1 Introduction

SDADC (sigma-delta ADC) primarily performs signal measurement, supporting continuous and single sampling. The reference voltage is 0~2V, with gain configurable from 0.25 to 4 times. The measurable range for single-ended input is 0V~(0.65* reference voltage / gain) , and for differential input is -(0.65* reference voltage/gain)~(0.65* reference voltage/gain) . However, the measurement range cannot exceed 0~AVDD . The minimum measurement error is ±60uV . Single sampling can achieve a sampling rate of 4kHz , while continuous sampling can achieve a sampling rate of 8kHz.

8.2.2 Main Features

- Supports 16-bitdata precision;
- Supports up to 2 differential or 5 single-ended channel samplings;
- Supports software-configurable multiple channels with multiple samplings;

8.2.3 Operating Modes

The SDADC is generally divided into two operating modes: single sampling and continuous sampling. Single sampling is further encapsulated to support multiple channels with multiple samplings mode as well as true single sampling.

8.2.3.1 Single Sampling

1. configure 0x00 CFG0 with contin_lvset to 0 to select non-continuous mode, and dsample_modeset to 0 to select single sampling.
2. configure 0x0c CFG3 to select the corresponding analog channels using sel_nch_v and sel_pch_v.
3. configure 0x14 TRIG to trigger this sampling with adc_start.
4. Wait for the sdadc interrupt, then read the sampling value of this sdadc from the data at address 0x40 after the interrupt.

8.2.3.2 Multiple Sampling

1. configure 0x00 CFG0 with contin_lv set to 0 to select non-continuous mode, and dsample_mode set to 1 to select multiple sampling.
2. configure registers 0x18~0x28 according to the desired channel and the number of samples per channel . For example: if 3 channels are to be sampled this time, with 8 samples per channel , set en to 1 and shift_num to 3 in 0x18~0x20 , and configure sel_pch and sel_nch in 0x18~0x20 according to the analog channels corresponding to each channel .
3. configure 0x14 TRIG to trigger this sampling with adc_start.
4. Wait for the sdadc interrupt, then read the sampling value of this channel from the data at register offset addresses 0x2c~0x3c after the sdadc interrupt.

8.2.3.3 Continuous Sampling Mode

1. configure the sampling rate in the register SINC_CFG 0x44 according to sinc_rate (the sampling rate is calculated as 1MHz/sinc_rate, where 1MHz is the AD clock frequency).
2. configure the dma registers according to the usage instructions to transfer data from the sdadc ffo to the specifed ram address via dma.
3. configuration 0x00 CFG0 Setting contin_lv to 1 selects non-continuous mode; the sample_mode setting is irrelevant.
4. From Read value from the address specifed by DMA

8.2.4 SDADC Register

Table 8-2: SDADC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CFG0**	
[31:19]			RSVD	
[18]	rw	1'b0	ppu_lv	reference generate pre-power-up
[17]	rw	1'b1	hsup_lv	supply votage 0x0:1.8v 0x1: 3.3v
[16:14]	rw	3'd0	dc_tr_lv	DCTR
[13]	r	1'b0	sdadc_data_rdy	1 indicates sample adc data done once
[12]	rw	1'b0	dsample_mode	1: sample adc data on a maximum of five channels in non-contious mode; 0: sample adc data on only one channel in non-contious mode
[11]	rw	1'b0	clk_inv_lv	inverse clk to folter or not 0x0:no 0x1:yes
[10]	rw	1'b0	contin_lv	continous sample or not 0x0:no 0x1:yes
[9:6]	rw	4'ha	ldo_sel_lv	ldo voltage selection
[5:4]	rw	2'd1	fck_sel_lv	ADC clock dividing selection
[3]	rw	1'b0	run_lv	start sdadc sample in continous mode
[2:1]	rw	2'd1	clrlen_lv	length of clear pulse
[0]	rw	1'b0	pu_lv	power up
0x04			**CFG1**	
[31]			RSVD	
[30:22]	rw	9'h1ff	chop3_num_lv	chopping weight controlling of 3rd stage
[21:13]	rw	9'h81	chop2_num_lv	chopping weight controlling of 2nd stage
[12:4]	rw	9'h6b	chop1_num_lv	chopping weight controlling of 1st stage
[3]	rw	1'b1	chop_stop_en_lv	chopping weight controlling enable
[2:1]	rw	2'd3	fchop_sig_sel_lv	siganal path chopping frequency
[0]	rw	1'b1	chop_en_lv	chopping enable
0x08			**CFG2**	
[31:28]			RSVD	
[27:26]	rw	2'd3	fchop_ref_sel_lv	reference buffer chopping frequency
[25:17]	rw	9'h7d	chop_ref_num_lv	chopping weight controlling of reference buffer
[16]	rw	1'b0	se_diff_sel_lv	single ended or differencial input 0x0: single end 0x1:differencial
[15]	rw	1'b1	dem_en_lv	DEM enable of gain controlling circuits
[14:12]	rw	3'h4	gain_deno_lv	denominator of gain
[11:9]	rw	3'h4	gain_nume_lv	numerator of gain
[8:0]	rw	9'ha0	sample_num_lv	number of sample
0x0c			**CFG3**	
[31:23]			RSVD	
[22]	rw	1'b1	refbuf_chop_mx_lv	reference buffer chopping mixed with dout 0x0: no 0x1:yes
[21]	rw	1'b1	refbuf_az_lv	reference buffer autozero 0x0: no 0x1: yes
[20:18]	rw	3'h3	amp2_bm_lv	bias mode of 2nd and 3rd opamp
[17:15]	rw	3'h3	refbuf_bm_lv	bias mode of reference buffer
[14:12]	rw	3'h3	amp1_bm_lv	bias mode of first opamp
[11]	rw	1'b0	refbuf_bp_lv	reference buffer bypass 0x0: no 0x1:yes
[10:8]	rw	3'd7	int_vref_set_lv	set internal reference voltage
[7:6]	rw	2'h0	vref_sel_lv	reference source selection
[5:3]	rw	3'h0	sel_nch_lv	Select N-side input channel for SDMADC, 0 for channel 0 , 4 for channel 4
[2:0]	rw	3'h1	sel_pch_lv	Select P-side input channel for SDMADC, 0 for channel 0 , 4 for channel 4
0x10			**CAL**	
[31:12]			RSVD	
[11]	r	1'b0	oscal_rdy_lv	1 indicates calibration done once
[10:7]	r	4'h0	os_set2_lv	offset code for rebuf
[6:3]	r	4'h0	os_set1_lv	offset code for 1st opamp
[2]	rw	1'b1	oscal_rst_lv	reset for offset calibration 0x0: reset 0x1: release
[1:0]	rw	2'h0	oscal_en_lv	offset calibration enable 0x0: no 0x1: 1st opamp 0x2: rebuffer 0x3: 2nd opamp 0x4:3rd opamp

Table continued on next page...

Table 8-2: SDADC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x14			**TRIG**	
[31:6]			RSVD	
[5]	w1c	1'b0	adc_start	trigger sdadc sample in non-continous mode once
[4]	rw	1'b0	gpio_trig_en	enable gpio postive edge to trigger sdadc sample in non-continous mode
[3:1]	rw	3'h0	timer_trig_src_sel	timer trigger source select
[0]	rw	1'b0	timer_trig_en	enable timer trigger source select
0x18			**CH0_CFG**	
[31:9]			RSVD	
[8:7]	rw	2'h0	shift_num	channel 0 sample times (= 2^{shift_num})
[6:4]	rw	3'h0	sel_nch	channel 0 selection N-side input
[3:1]	rw	3'h0	sel_pch	channel 0 selection P-side input
[0]	rw	1'b0	en	enable channel 0 sample
0x1c			**CH1_CFG**	
[31:9]			RSVD	
[8:7]	rw	2'h0	shift_num	channel 1 sample times (= 2^{shift_num})
[6:4]	rw	3'h0	sel_nch	channel 1 selection N-side input
[3:1]	rw	3'h0	sel_pch	channel 1 selection P-side input
[0]	rw	1'b0	en	enable channel 1 sample
0x20			**CH2_CFG**	
[31:9]			RSVD	
[8:7]	rw	2'h0	shift_num	channel 2 sample times (= 2^{shift_num})
[6:4]	rw	3'h0	sel_nch	channel 2 selection N-side input
[3:1]	rw	3'h0	sel_pch	channel 2 selection P-side input
[0]	rw	1'b0	en	enable channel 2 sample
0x24			**CH3_CFG**	
[31:9]			RSVD	
[8:7]	rw	2'h0	shift_num	channel 3 sample times (= 2^{shift_num})
[6:4]	rw	3'h0	sel_nch	channel 3 selection N-side input
[3:1]	rw	3'h0	sel_pch	channel 3 selection P-side input
[0]	rw	1'b0	en	enable channel 3 sample
0x28			**CH4_CFG**	
[31:9]			RSVD	
[8:7]	rw	2'h0	shift_num	channel 4 sample times (= 2^{shift_num})
[6:4]	rw	3'h0	sel_nch	channel 4 selection N-side input
[3:1]	rw	3'h0	sel_pch	channel 4 selection P-side input
[0]	rw	1'b0	en	enable channel 4 sample
0x2c			**CH0_DOUT**	
[31:24]			RSVD	
[23:0]	r	24'h0	data	channel 0 average value
0x30			**CH1_DOUT**	
[31:24]			RSVD	
[23:0]	r	24'h0	data	channel 1 average value
0x34			**CH2_DOUT**	
[31:24]			RSVD	
[23:0]	r	24'h0	data	channel 2 average value
0x38			**CH3_DOUT**	
[31:24]			RSVD	
[23:0]	r	24'h0	data	channel 3 average value
0x3c			**CH4_DOUT**	
[31:24]			RSVD	
[23:0]	r	24'h0	data	channel 4 average value
0x40			**SINGLE_DOUT**	

Table continued on next page...

Table 8-2: SDADC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:24]			RSVD	
[23:0]	r	24'h0	data	sample date when contin_lv is 0 and dsample_mode is 0
0x44			**SINC_CFG**	
[31:9]			RSVD	
[8]	rw	1'b0	sinc_order_sel	1:select four differentiators in sinc filter; 0:select three differentiators in sinc filter
[7:0]	rw	8'h64	sinc_rate	dowmsampling rate of sinc filter
0x48			**COMP_CFG0**	
[31:13]			RSVD	
[12:1]	rw	12'h15c	comp_coeff0	coefficient 0 of compensating filter
[0]	rw	1'b0	comp_bypass	1: bypass compensating filter ; 0: enable compensating filter
0x4c			**COMP_CFG1**	
[31:24]			RSVD	
[23:12]	rw	12'h57e	comp_coeff2	coefficient 2 of compensating filter
[11:0]	rw	12'hb31	comp_coeff1	coefficient 1 of compensating filter
0x68			**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	
0x6c			**DMA_CFG**	
[31:1]			RSVD	
[0]	rw	1'b0	rx_dma_msk	1:disable sdadc dma request; 0: enable sdadc dma request
0x74			**FIFO_ST**	
[31:1]			RSVD	
[0]	r	1'b0	empty	1 indicates sdadc fifo is empty
0x78			**INT_ST**	
[31:2]			RSVD	
[1]	r	1'b0	dsample	1 indicates sdadc sample done in non-continous mode and as one of irq source at same time
[0]	r	1'b0	overflow	1 indicates sdadc fifo has already overflowed and as one of irq source at same time
0x7c			**INT_MSK**	
[31:2]			RSVD	
[1]	rw	1'b0	dsample	1:disable sdadc sample done irq to system; 0: enable sdadc sample done irq to system
[0]	rw	1'b0	overflow	1:disable sdadc fifo overflow irq to system; 0: enable sdadc fifo overflow irq to system
0x80			**INT_CLR**	
[31:1]			RSVD	
[0]	w1c	1'b0	int_clr	clear sdadc irq
0x84			**FSM_ST**	
[31:1]			RSVD	
[0]	r	1'b0	active	1 indicates sdadc non-continous mode is running

8.3 TSEN

TSEN is located in LPSYS.

8.3.1 Introduction

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

8.3.2 Key Features

- Resolution of 0.2°C
- Accuracy from -3°C to 3°C
- Supports a temperature range from -40°C to 125°C
- Supports reading via polling or interrupt methods

8.3.3 Function Description

8.3.3.1 Clock Signal

The operating clock of TSEN is derived from the PCLK of LPSYS through clock division. The division ratio is configured by ANAU_TSEN_CLK_DIV in the register TSEN_CTRL_REG. The frequency calculation formula is:

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

8.3.3.2 Reading Procedure

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

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

Polling reading procedure:

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

The process for reading via interrupt:

Enable the TSEN interrupt, set the TSEN_IMR bit in the TSEN_IRQ register to 0, start the TSEN, and upon conversion completion, a TSEN_IRQ interrupt will be sent to the CPU. During interrupt handling, set the TSEN_ICR bit in the TSEN_IRQ register to 1 to clear the interrupt and read the data.

8.3.3.3 Usage Procedure

The operation of the TSEN must follow the procedure below.

1. Configure the division ratio according to the PCLK frequency; the clock frequency of the TSEN should be 1 MHz or 2 MHz.
2. Set the EN bit in the BGR register to 1 to enable the Bandgap.
3. Set ANAU_IARY_EN in register ANAU_ANA_TP to 1 to enable the Iarray.
4. Set ANAU_TSEN_EN in register TSEN_CTRL_REG to 1 to enable the TSEN clock.

5. Set ANAU_TSEN_PU in register TSEN_CTRL_REG to 1 and ANAU_TSEN_RUN to 0.
6. Set ANAU_TSEN_RSTB in register TSEN_CTRL_REG to 0 first, then to 1 with a low-level duration of at least 20us.
7. Set ANAU_TSEN_RUN in register TSEN_CTRL_REG to 1 and read the result via polling or interrupt. Alternatively, wait for an absolute time of 3ms before reading.
8. Disable TSEN: Set ANAU_TSEN_RUN/ANAU_TSEN_PU/ANAU_TSEN_EN bits in the register TSEN_CTRL_REG to 0, and set ANAU_TSEN_RSTB to 1 .
9. Bandgap and Iarray are shared with GPADC and it is recommended not to disable them; if disabling is necessary, ensure it does not interfere with the operation of GPADC.

8.3.4 TSEN Registers

Table 8-3: TSEN Register Map

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 pclk 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	
0x10			**ANAU_ANA_TP**	Tsen IRQ Register
[31:1]			RSVD	
[0]	rw	1'b0	ANAU_IARY_EN	
0x14			**BGR**	Bandgap registers
[31:12]			RSVD	
[11:8]	rw	4'hC	VREF12	select VREF 1.2V
[7:4]	rw	4'hC	VREF06	select VREF 0.6V
[3:1]			RSVD	
[0]	rw	1'h0	EN	Bandgap enable

9 Timer

9.1 BTIM

The chip contains a total of 4 BTIMs, of which BTIM1 and BTIM2 are located in HPSYS and can send requests to DMAC1; BTIM3 and BTIM4 are located in LPSYS and can send requests to DMAC2.

9.1.1 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.

9.1.2 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

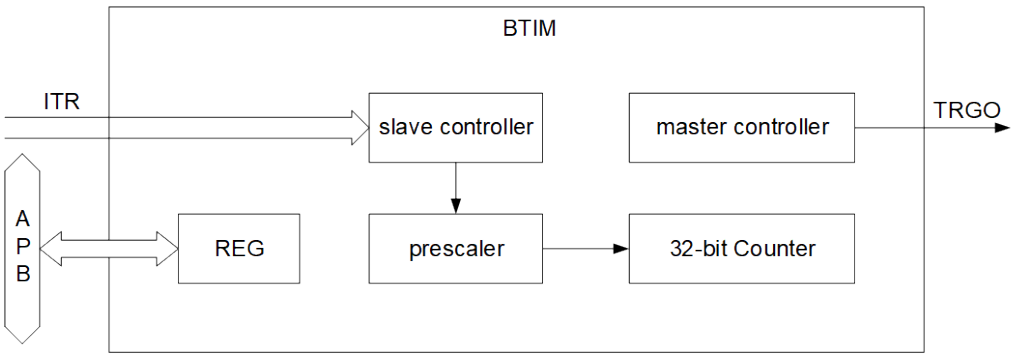


Figure 9-1: **BTIM Block Diagram**

9.1.3 BTIM Function Description

9.1.3.1 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 PCLK clock 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 fixed 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 .

9.1.3.2 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.

9.1.3.3 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.

9.1.3.4 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.

9.1.3.5 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.

9.1.3.6 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 .

9.1.3.7 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.

9.1.4 BTIM Register

Table 9-1: BTIM Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	TIM control register 1
[31:8]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[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)
[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).

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]			RSVD	
0x08			**SMCR**	TIM slave mode control register
[31:24]			RSVD	
[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	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	UIE	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled
0x10			**SR**	TIM status register
[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 (up-counting), 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.

9.2 GPTIM

The chip contains a total of 5 GPTIMs, among which GPTIM 1 and GPTIM 2 are located in HPSYS , with input/output connected to IO(PA) , and can send requests to DMAC1 ; GPTIM3, GPTIM4, and GPTIM5 are located in LPSYS, with input and output connected to IO(PB), and can send requests to DMAC2.

9.2.1 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.

9.2.2 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 65536 for 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 filtering 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

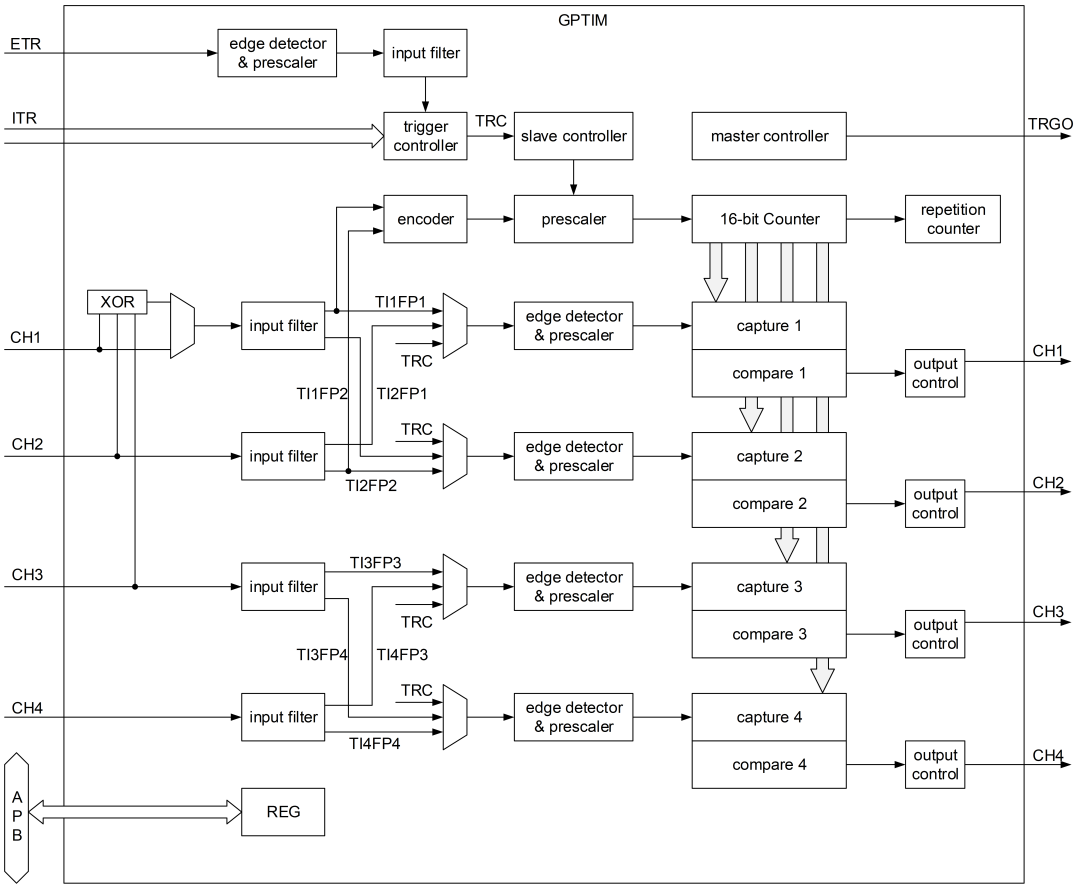


Figure 9-2: GPTIM Block Diagram

9.2.3 GPTIM Function Description

9.2.3.1 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.

9.2.3.2 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.

9.2.3.3 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.

9.2.3.4 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.

9.2.3.5 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, filtering, 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.

9.2.3.6 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, filtering, 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 fixed 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.

9.2.3.7 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 filter 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 filtering 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 confirmed.

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.。

9.2.3.8 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.

9.2.3.9 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.

9.2.3.10 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 Figure 9-3 . When the counter value CNT is less than CCR1/2 , the output is high; otherwise, it is low.

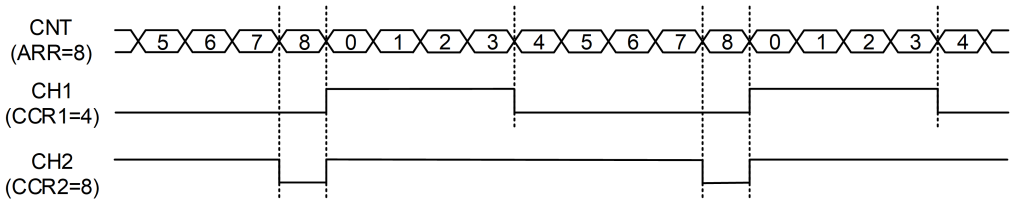


Figure 9-3: PWM output in increment counting mode

In center-aligned counting mode, if you configure CCMR1_OC1M and CCMR1_OC2M to 0110(0x6) , the PWM output is illustrated in Figure 9-4 . 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.

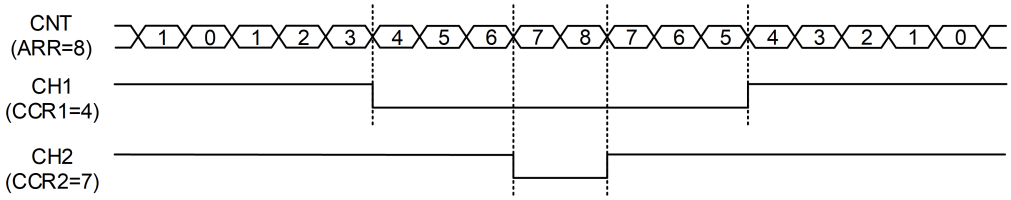


Figure 9-4: PWM output in center-aligned counting mode

9.2.3.11 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 CH3/4 can 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 Figure 9-5. 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.

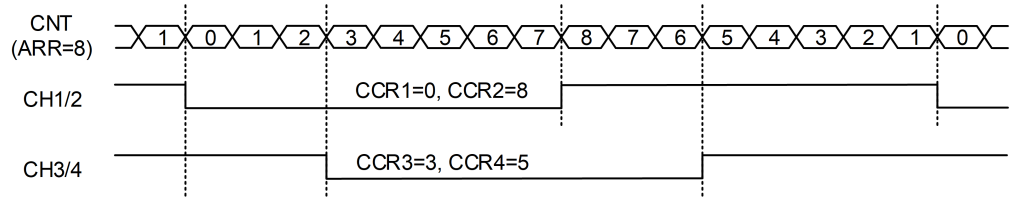


Figure 9-5: Asymmetric PWMO utput

9.2.3.12 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_OC1M is configured to 1101(0xd), CCMR1_OC2M to 0110(0x6), CCMR2_OC3M to 1100(0xc), CCMR2_OC4M is If 0111(0x7), then the PWM output is as shown in Figure 9-6. 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.

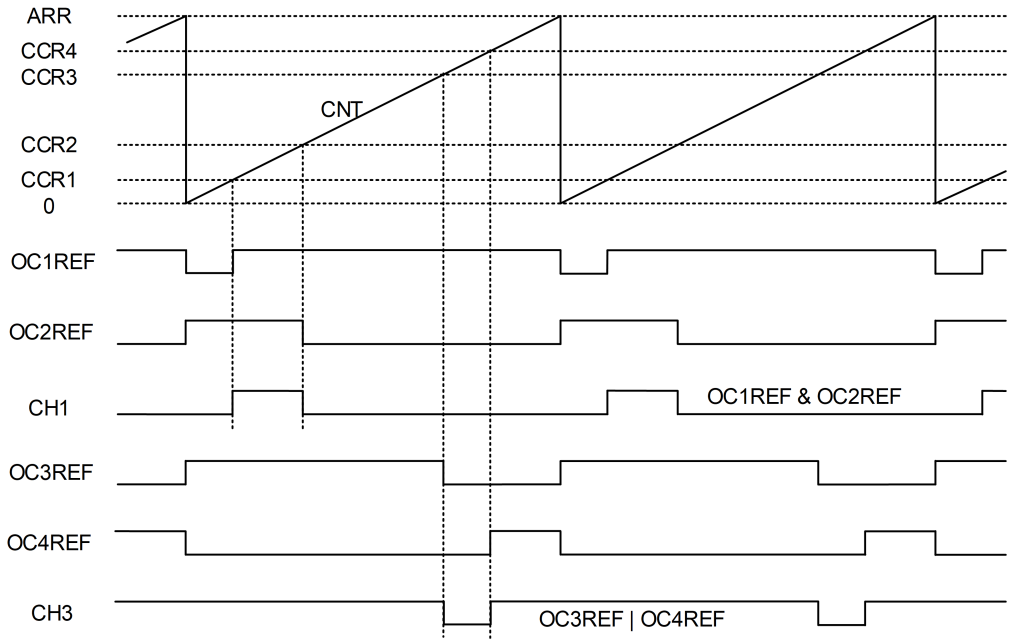


Figure 9-6: Combined PWM Output

9.2.3.13 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 TI2FP2 to 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 defining 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 ARR and 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.

9.2.3.14 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, 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, configure SMCR_SMS to 0011(0x3). CCER_CC1P/CC2P is used to select the polarity of CH1 and CH2. If necessary, the input filter 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.

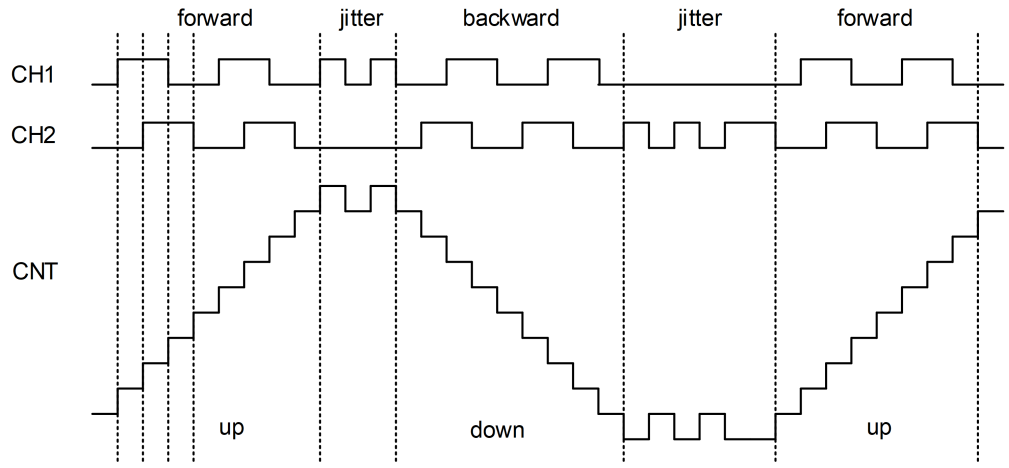
In Encoder Interface Mode, the counting events of the counter are the decoded outputs of the quadrature encoder interface. The counter only performs continuousounting between 0 and ARR (incrementing from 0 to ARR or decrementing from ARR to 0, depending on the specific counting direction). Therefore, 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。



9.2.3.15 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.

9.2.3.16 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.

9.2.4 GPTIM Register

Table 9-2: 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

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[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)
[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 9-2: 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 9-2: 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 9-2: 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 9-2: 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 9-2: 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 (up-counting), 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 9-2: 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 9-2: 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
[19]	rw	1'h0	OC3PE	Output compare 3 preload enable

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[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 9-2: 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 9-2: 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 9-2: 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 LPTIM

The chip contains 3 LPTIMs, among which LPTIM 1 is located in HPSYS_AON and can remain operational when HPSYS enters low-power mode (excluding hibernate), with input and output connected to IO(PA) ; LPTIM2 and LPTIM3 are located in LPSYS_AON and can remain operational when LPSYS enters low-power mode (excluding hibernate), with input and output connected to IO(PB) .

9.3.1 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.

9.3.2 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 filtering
- 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

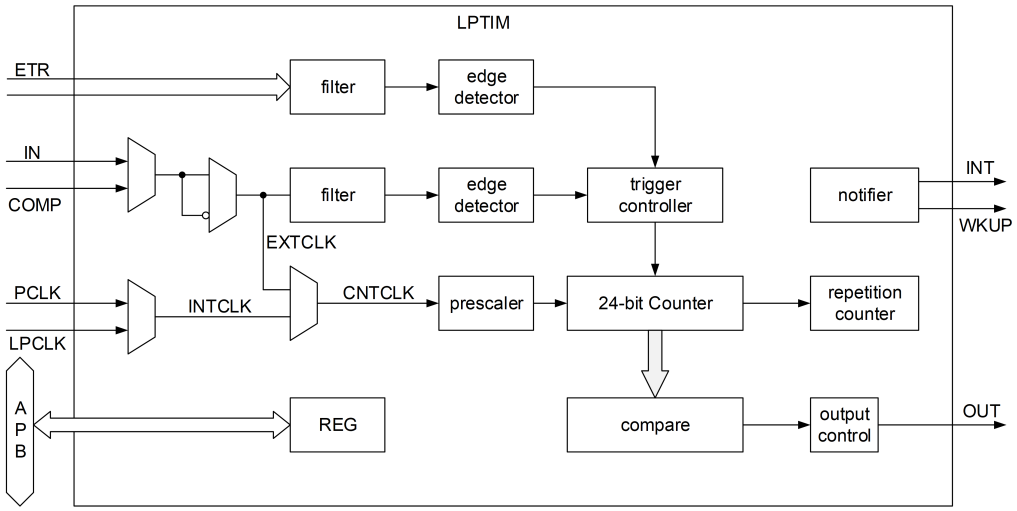


Figure 9-7: LPTIM structure diagram

9.3.3 LPTIM function description

9.3.3.1 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 fixed 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.

9.3.3.2 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-filtering 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.

9.3.3.3 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

9.3.3.4 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.

9.3.3.5 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 filtered ETRoptional edge arrives.

9.3.3.6 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.

9.3.3.7 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.

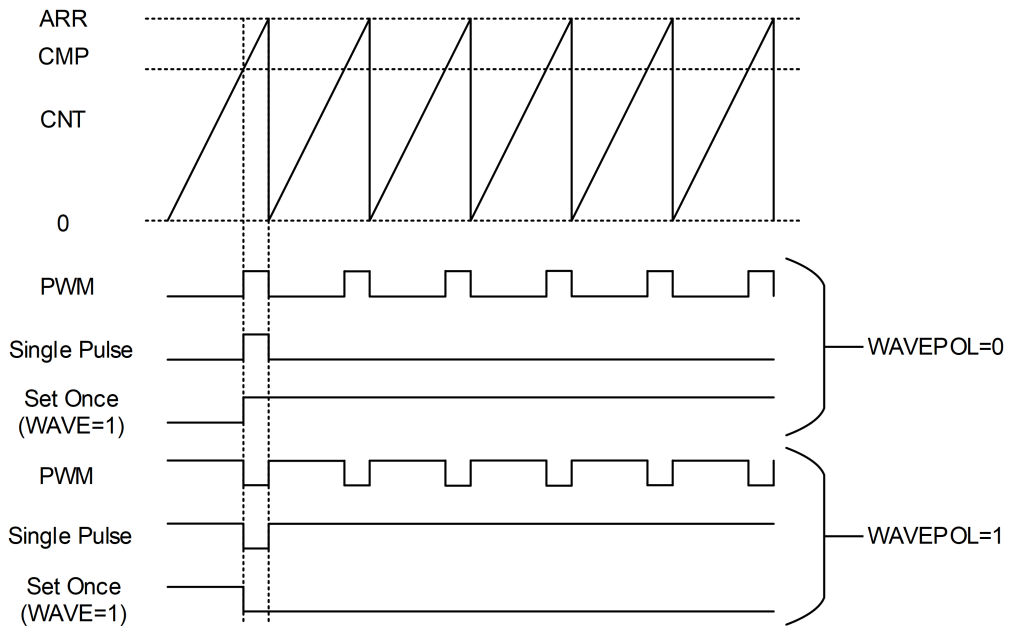


Figure 9-8: PWM Output

9.3.3.8 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.

9.3.4 LPTIM Register

Table 9-3: 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.
[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.

Continued on the next page...

Table 9-3: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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

Continued on the next page...

Table 9-3: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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 9-3: 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 9-3: 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 9-3: 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.

9.4 WDT

The chip contains 3 WDTs, of which WDT1 is located in HPSYS and 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 remains operational after the chip enters low-power mode. The specific reset scope of each WDT can be found in the Clock and Reset chapter.

9.4.1 Introduction

The watchdog timer, as a type of counter, is primarily used to reset the system upon reaching a preset time to prevent software hang.

Basic functions of the watchdog timer:

- Supports two operating modes:
  - mode0
    - wdt does not generate interrupts; it directly resets the system upon reaching the preset time.
    - Supports up to a 24-bit counter
  - mode1
    - Divided into two counting stages: an interrupt is generated upon reaching the first preset time, and the system resets upon reaching the second preset time. Resets the system.
    - Each time stage supports up to a 24-bit counter
- Supports write protection to prevent software misoperation of the wdt

9.4.2 WDT Operating Modes

wdt has two resetgeneration modes depending on requirements:

Mode 1: Counts only one cycle; upon completion, directly generates a resetsignal.

Mode 2: Counts two cycles; an interrupt is generated at the end of the first cycle, and a resetsignal is generated at the end of the second cycle.

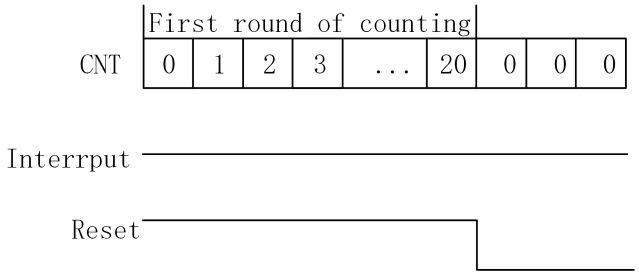


Figure 9-9: The relationship between the interrupt and resetsignal generation in Mode 1 and the counter (assuming a timeout value of 20, withresetactive low)

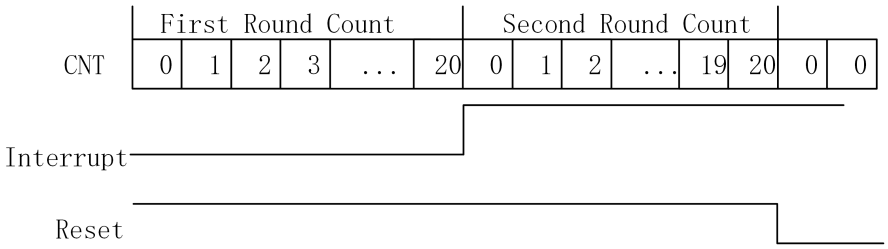


Figure 9-10: The relationship between the interrupt and resetsignal generation in Mode 2 and the counter (assuming a timeout value of 20, withresetactive low).

“Watchdog feeding” behavior under both modes:

In Mode 1,if the watchdog is ’fed’ before the first counting cycle ends, the counter restarts counting from zero.

In Mode 2, if the watchdog is ’fed’ before the first counting cycle ends, the counter restarts counting from zero. If the watchdog is not ’fed’ before the first counting cycle ends, the wdt enters the second counting cycle. At this point, clearing the interrupt or ’feeding’ the watchdog can cause the wdt to re-enter the first counting cycle.

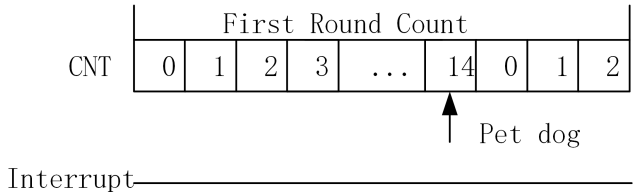


Figure 9-11: Mode 1 The effect of the ’dog feeding’ action on the counter (assuming the timeout value is 20)

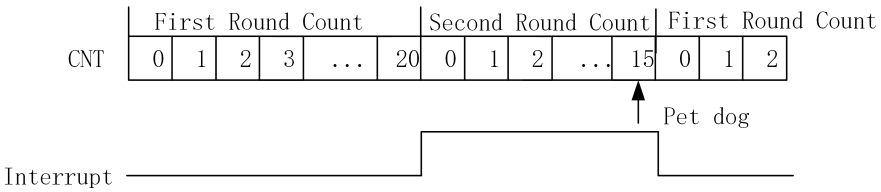


Figure 9-12: Mode 2 The effect of the ’dog feeding’ operation on the counter during the second round of counting is the same as in the first round (assuming the timeout value is 20)

Methods to stop the wdtin two modes:

In mode 1, configuring register0x0C(counter_control)=0x34before the count ends will stop thewdt.

In mode 2 , configuring register 0x0C(counter_control)=0x34 before the first round count ends will stop thewdt . If in the second counting cycle stage, it is necessary to clear the interrupt or perform a 'watchdog feed' operation to reset the wdt to the first counting cycle before configuring register 0x0C (counter_control) = 0x34 to stop the wdt.

Methods to clear the interrupt:

In mode 2, after the first counting cycle of thewdtcompletes, an interrupt is generated. In register0x14 WDT_SR, the int_assertbit is set to 1. At this point, clearing can be performed by configuring 0x10 of WDT_IDR's int_clror by configuring 0x0C WDT_CCR'scounter_controlto 0x76, which corresponds to feeding the watchdog to clear it.

9.4.2.1 WDT Register Configuration Procedure

1. Select the operating mode of the wdt according to requirements. Configure 0x08 in the response_mode register: set 0x0 to select mode 1 and set 0x1 to select mode 2.
2. Based on the wdt trigger reset time configuration, the 0x00 count_value_0 register (timeout value for the first count cycle in both modes) and the 0x04 count_value_1 register (timeout value for the second count cycle in mode 2).
3. Configure the length of thereset length field in 0x08 according to requirements.
4. Configure the counter_control register at 0x0c (=0x76) to trigger the wdt to start operation.
   The sequence of steps 1 to 3 is not mandatory, provided they are completed before step 4.

9.4.2.2 Precautions

1. The wdt provides a write protection function to prevent the configuration within the wdt from being accidentally overwritten. The usage is as follows:
   When the wrpt register in configuration 0x18 is set to 0x58ab99fc and the wrpt_st bit in register 0x18 is 1, write protection is enabled. In this state, all registers within the wdt cannot be overwritten, but read operations remain unaffected. To disable write protection, set the wrpt register in configuration 0x18 to 0x51ff8621.
2. When the sync_fg register at 0x1c within the wdt is 1, it indicates that the operations start, stop, irq clear, and reset flag clear have been synchronized from pclk to wdt clk and have taken effect.
3. rst_fg being 1 indicates that this wdt caused a reset ; this register is only valid for iwdt.
4. If the counter_control at 0xc cannot be written, first check whether the corresponding sys rcc wdt clock enable register is configured correctly to 1。

9.4.3 WDT Register

Table 9-4: 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
[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

Continued on the next page...

Table 9-4: WDT Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
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

9.5 RTC

9.5.1 RTC Register

Table 9-5: 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
[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

Continued on the next page...

Table 9-5: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	
[2:1]			RSVD	
[0]	rw	1'h0	WUCKSEL	
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	
[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

Continued on the next page...

Table 9-5: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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]	w	1'b0	ADD1S	
[30:15]			RSVD	
[14:0]	rw	15'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	
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

Continued on the next page...

Table 9-5: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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			**BKP10R**	Backup 10 Register
[31:0]	rw	32'h0	BKP	
0x5C			**BKP11R**	Backup 11 Register
[31:0]	rw	32'h0	BKP	
0x60			**BKP12R**	Backup 12 Register
[31:0]	rw	32'h0	BKP	
0x64			**BKP13R**	Backup 13 Register
[31:0]	rw	32'h0	BKP	
0x68			**BKP14R**	Backup 14 Register
[31:0]	rw	32'h0	BKP	
0x6C			**BKP15R**	Backup 15 Register
[31:0]	rw	32'h0	BKP	
0x70			**BKP16R**	Backup 16 Register
[31:0]	rw	32'h0	BKP	
0x74			**BKP17R**	Backup 17 Register
[31:0]	rw	32'h0	BKP	
0x78			**BKP18R**	Backup 18 Register
[31:0]	rw	32'h0	BKP	
0x7C			**BKP19R**	Backup 19 Register
[31:0]	rw	32'h0	BKP	
0x80			**BKP20R**	Backup 20 Register
[31:0]	rw	32'h0	BKP	
0x84			**BKP21R**	Backup 21 Register
[31:0]	rw	32'h0	BKP	
0x88			**BKP22R**	Backup 22 Register
[31:0]	rw	32'h0	BKP	
0x8C			**BKP23R**	Backup 23 Register
[31:0]	rw	32'h0	BKP	
0x90			**BKP24R**	Backup 24 Register
[31:0]	rw	32'h0	BKP	
0x94			**BKP25R**	Backup 25 Register
[31:0]	rw	32'h0	BKP	
0x98			**BKP26R**	Backup 26 Register
[31:0]	rw	32'h0	BKP	
0x9C			**BKP27R**	Backup 27 Register
[31:0]	rw	32'h0	BKP	
0xA0			**BKP28R**	Backup 28 Register
[31:0]	rw	32'h0	BKP	
0xA4			**BKP29R**	Backup 29 Register
[31:0]	rw	32'h0	BKP	
0xA8			**BKP30R**	Backup 30 Register
[31:0]	rw	32'h0	BKP	
0xAC			**BKP31R**	Backup 31 Register

Continued on the next page...

Table 9-5: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	BKP	

10 High-Performance Dedicated Computing

10.1 ePicasso™ High-Performance 2.5D Graphics Engine

ePicasso is located in HPSYS.

In 2.5D image processing, many common image operations consume a signifcant amount of CPU computing resources. ePicasso™ is an acceleration engine specifically designed for 2.5D image processing, capable of delivering exponential speed improvements for common functions in 2.5D image operations such as layer blending, scaling, and rotation. Furthermore, ePicasso™ is compatible with various common RGB image formats, thereby simplifying the conversion of different image formats within the system.

10.1.1 Layer Blending

ePicasso™ supports up to four foreground layers and one monochrome background layer for blending. Input and output formats include commonly used RGB565 ,RGB888 ,ARGB8565 , and ARGB8888 . Each foreground layer features an independent blending mode and blending region. Additionally, each layer provides an independent filter configuration option, enabling the layer to flter out a specific color. This function can be utilized for simple image capture.

10.1.2 Graphics Scaling

ePicasso™ features a layer called the functional layer, which, in addition to supporting overlay capabilities, can also perform graphics scaling. The maximum scaling factor can reach up to eight times, with a precision of 1/256 . Scaling factors in the X and Y directions can be configured independently to meet various requirements.

10.1.3 Graphics Rotation

The functional layers of ePicasso TM not only support zooming but also enable high-precision image rotation. Users can customize the sin/cos values of the rotation angle to accommodate arbitrary rotation requirements. Rotation and zoom functions can be enabled simultaneously, performing both image operations in a single step, thereby enhancing image processing performance.

10.2 LCDC

The chip contains 2 LCDC units, where LCDC1 is located in HPSYS and supports a full-featured interface; LCDC2 is located in LPSYS and supports only SPI/DSPI/QSPI interfaces.

10.2.1 Introduction

LCDC, short for LCD Controller, primarily reads image data from memory and transmits it to the corresponding display according to different screen interfaces. LCDC supports a wide range of display interfaces, including:

- DPI/RGB Interface: for high-resolution displays without GRAM
- DBI/8080 Interface: for medium- to low-resolution displays with GRAM
- SPI/DSPI/QSPI Interface: for small-sized IoT medium- to low-resolution displays with GRAM
- JDI Serial/Parallel Interface: for JDI vendor-specific reflective displays

In addition to abundant interfaces, the LCDC supports a wide range of image data formats, including RGB565, RGB888, and ARGB8888. Building on fundamental functions, the LCDC also supports simple layer blending, enabling basic image processing acceleration.

10.2.2 Architecture Introduction

The following fgure illustrates the basic block diagram of the LCDC:

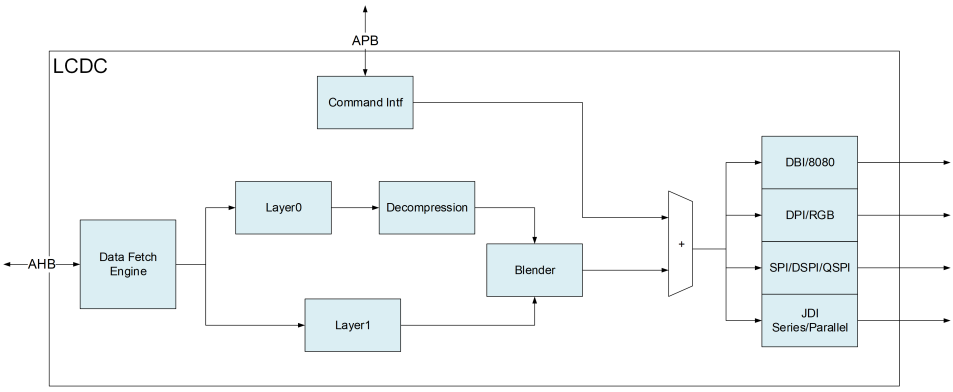


Figure 10-1: LCDC Block Diagram

There are two primary data paths shown in the figure.

1. Configuration path: The upper layer transmits screen configuration commands via the APB bus. Upon receiving these commands, theLCDC sends the corresponding data to the respective screen interface through the Command Intf module. This path operates at a lower speed and is primarily used for screen initialization configuration and basic function verifcation.
2. Image transmission path: The LCDC's Data Fetch Engine retrieves image data from memory via the AHB bus and transmits it to Layer0/Layer1 according to configuration. Layer0 includes a dedicated decompression module that decompresses received compressed data into image data. When the data for two layers is prepared, Blender blends the image data and sends it to the corresponding screen interface. This path is the primary channel through which LCDC delivers image data to the screen. It should also be noted that Layer0 and Layer1 can be enabled independently; even if both are disabled, Blender can be configured to send monochrome data to the screen.

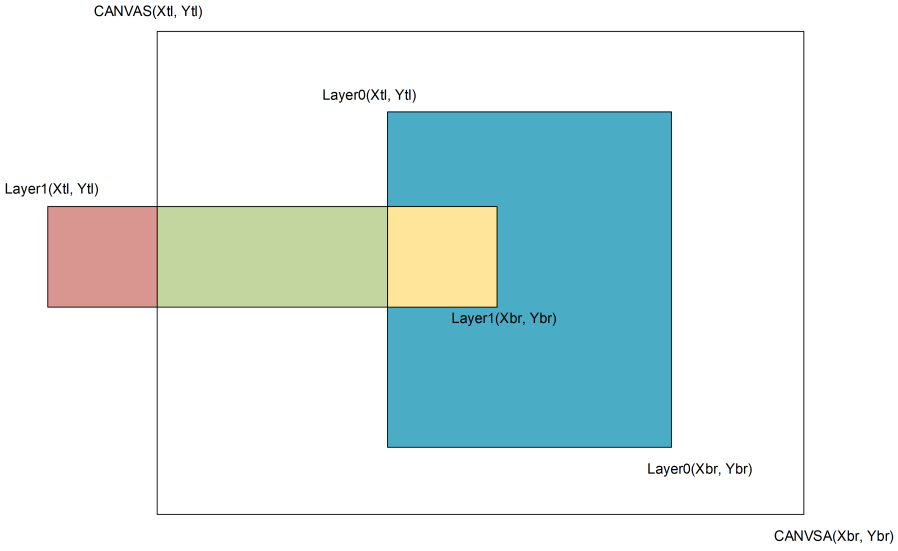
10.2.3 Configuration Procedure

The configuration of LCDC is mainly divided into two parts: the first part involves layer configuration, targeting the two Layers and Blender settings; the second part involves interface configuration, targeting parameter settings for different interfaces. Interface configuration addresses parameter settings corresponding to various interfaces.

10.2.3.1 Layer Configuration

Layer configuration includes the configurations of Layer0/Layer1 and the Blender . Users can configure the Layer0_CONFIG and Layer1_CONFIG registers to enable Layer0 and Layer1 , and set their color formats and blending coeffcients. Since Layer0 also supports compressed mode images, when Layer0 of Decompression module is enabled, its original color format configuration becomes invalid, and the decompressed color format is uniformly RGB888.

When layer data is sent to the Blender , the Blender performs blending operations based on the user-configured layer coordinates and canvas coordinates. Each layer has its own independent rectangular area, and the canvas also has its own rectangular area. The Blender only processes the overlapping portions of the images. Please refer to the following fgure for details:



In the fgure, CANVAS ,Layer0 , andLayer1 each have two sets of coordinates representing their respective rectangular regions. (Xtl, Ytl) denotes the top-left corner coordinates of the rectangle, and (Xbr, Ybr) denotes the bottom-right corner coordinates. It can be observed that Layer0 is entirely within the CANVAS , so its entire area participates in the blending calculation. Layer1 consists of three parts: the red area lies outside the CANVAS and thus does not participate in the calculation; the yellow area overlaps with both CANVAS and Layer0 , so the Blender computes these two layers together with the CANVAS ; fnally, the green area overlaps only with the CANVAS and will therefore be blended with the CANVAS . The remaining white CANVAS area, as it does not overlap with any layer, will directly output the background color of the CANVAS .

In most scenarios, if the LCDC only outputs the image framebuffer to the screen, it is sufcient to configure the layer coordinates to match the CANVAS coordinates.

10.2.3.2 Interface Configuration

Due to signifcant differences in transmission protocols among various interfaces, the corresponding configuration parameters also differ. The following lists the configuration parameters for different interfaces.

DPI/RGB Interface：

DPI/RGB is a synchronous parallel interface driven by the Pixel clock. Its main configuration parameters are detailed in the following table:

Parameter Name	Description	Configuration
Freq(pclk)	Pixel Clock Rate	PCLK_DIV, Pixel Clock division ratio, with the source clock being the System Clock
Hsync Width	Hsync Signal Active Cycle Count	HSW， cycles corresponding to PixelClock
Vsync Height	Vsync Signal Active Line Count	VSH
Horizontal Active Width	Active Line Width	HAW， number of valid Pixels per line
Vertical Active Height	Number of Active Lines	VAH， number of lines containing valid Pixels
Horizontal Back Porch	HBP Parameter	HBP， corresponding to the HBP parameter of the DPI interface
Vertical Back Porch	VBP Parameter	VBP， corresponding to the VBP parameter of the DPI interface
Horizontal Front Porch	HFP Parameter	HFP， corresponding to the HFP parameter of the DPI interface
Vertical Front Porch	VFP Parameter	VFP， corresponding to the VFP parameter of the DPI interface

After configuring these parameters, the user must also configure the phase information of signals such as Hsync , Vsync , DE , and Pclk according to the actual screen characteristics. By default, these signals are active high.

Finally, note that after completing the configuration of the DPI interface, configuring the DPI_EN signal will enable the DPI interface. Once enabled, only by setting DPI_EN to 0 can the data transmission of the LCDC be stopped; otherwise, the LCDC will continuously send the Frame buffer data at the currently configured address to the display.

DBI/8080 interface:

The DBI/8080 interface is used to drive screens with GRAM. According to the MIPI protocol, the DBI parallel interface is further divided into two types: TypeA and TypeB, The signals differ slightly in form but can be logically converted to each other. When selecting the LCDCinterface, the corresponding TypeAand TypeBcan also be selected.

The main parameter configuration of the DBI interface is referenced in the table below:

Parameter Name	Description	configuration
PWH	Number of cycles in the inactive state of the WRX/RDX signa	The cycle corresponds to the system clock
PWL	Number of cycles in the active state of the WRX/RDX signal	The cycle corresponds to the system clock
TAH	Number of delay cycles from WRX/RDX signal inactive to CSX signal inactive	The cycle corresponds to the system clock
TAS	Number of delay cycles from CSX signal active to WRX/RDX signal active	The cycle corresponds to the system clock

The above parameters are the main configuration parameters for the DBI interface, where PWH and PWL determine the rate of the DBI interface; their sum equals the cycle count for a single data transfer.

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, phase inversion can be achieved by configuring the corresponding signal's POL register.

In addition to the image transmission path, theDBI interface also supports configuration paths. After configuring the DBI interface, the user can first set the value to be written into the LCD_WR register, trigger read/write operations via the WR_TRIG and RD_TRIG registers, and then obtain the read value through the LCD_RD register.

SPI/DSPI/QSPI interfaces:

SPI,DSPI, and QSPI all belong to SPI-type screen interfaces, differing in the number of data lines. SPI usually has a single data line,DSPI has two, and QSPI has four. Apart from the difference in the number of data lines, theSPI interface is also categorized into 3-wire SPI and 4-wire SPI based on the transmission method of theD/C (Data/Command selection) signal. In 3-wire SPI, theD/C functions as a data bit transmitted via SDO, whereas in 4-wire SPI, theD/C signal has a dedicated line for identifcation; consequently, the effective bandwidth of 3-wire SPI is slightly lower than that of 4-wire SPI.

The primary parameter configurations of the SPI interface are 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 and the 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, the LCDC does not immediately acquire data to send to the screen, it will automatically control the SPI interface's CS signal, setting it to a disabled state
SPI_CLK_AUTO_DIS	SPI clock automatic stop	During the data transmission phase, if the bus is busy, the LCDC does not immediately acquire data to send to the screen, it will automatically stop the SPI interface's clock signal

In addition to the above main parameters, there are other parameters primarily used for read operations of the SPI interface, as well as for the phase of the SPIsignals. For details, please refer to the descriptions in the registers table.

Similar to the DBI interface, the SPI interface also supports path configuration. After selecting the SPI interface, similar to the DBI, users can trigger read and write operations through the WR_TRIG and RD_TRIG registers. For SPI , the user must also configure WR_LEN and RD_LEN registers, which determine the number of bytes per single read/write operation in the SPI interface configuration path. 0 represents 1 byte, with a maximum of 4 bytes.

JDI Serial/Parallel Interface:

The JDI Serial and JDI Parallel interfaces are dedicated to JDI reflective displays. The JDI Serial is a serial interface supporting relatively lower-resolution screens while the JDI Parallel is a parallel interface supporting higher-resolution screens. The configuration of the JDI Serial interface is relatively straightforward, with the following parameters:

Parameter Name	Description	configuration
CLK_DIV	JDI Serial Clock division ratio	The source clock is the system clock
JDI_SER_FORMAT	JDI Serial Color format	JDI Serial Supports 1bit, 3bit, and 4bit color formats
JDI_WR_CMD	JDI Send command	JDI Serial The interface sends a command sequence before transmitting actual data. The command content can be configured here.

The above are the primary parameter configurations for the JDI Serial interface. The JDI Serial interface also supports path configuration. Users must configure the JDI Serial WR_LEN register to define the bit width of a single transmission on the JDI Serial interface, and then send data via WR_TRIG.

The JDI Parallel interface supports relatively higher resolution displays with richer color depth. It includes more configu-

ration parameters, which must be set according to the JDI Parallel interface protocol. 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 VCKto HSTsignal	Based on system clock cycles
HCK_DLY	Delay from HSTto HCKsignal	Based on system clock cycles
ENB_ST_COL	ENB signal start column number	ENB signal first valid column number
ENB_END_COL	ENB signal end column number	ENB signal last valid column number
ENB_ST_LINE	ENB signal start row number	ENB signal first valid row number
ENB_END_LINE	ENB signal end column number	ENB signal last valid row number
DP_MODE	DP Mode	Supports large and small dual-pixel modes

The JDI Parallel interface has numerous configuration parameters; it is necessary to consult 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 and sends it to the display. The JDI Parallel interface will only stop after the current frame data transmission completes when it is disabled.

10.2.4 LCDC Register

Table 10-1: LCDC Register Map

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**	
[31:22]			RSVD	
[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

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[18]	rw1c	1'h0	DPIL_INTR_RAW_STAT	raw status of dpi line interrupt
[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
[15:6]			RSVD	
[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**	
[31:9]			RSVD	
[8]	rw	1'h1	AUTO_GATE_EN	auto clock gating enable
[7:6]			RSVD	
[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**	
[31:24]			RSVD	
[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**	
[31:29]			RSVD	
[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

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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 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_DECOMP**	
[31:23]			RSVD	
[22:13]	rw	10'h0	col_size	number of colums in a line of original image
[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.
[0]	rw	1'h0	enable	decompression enable
0x34			**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
0x38			**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

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
0x3c			**LAYER0_DECOMP_STAT**	
[31:6]			RSVD	
[5:0]	r	6'h0	buf_max_depth	buf max usage
0x40			**LAYER1_CONFIG**	
[31:29]			RSVD	
[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 others: reserved
0x44			**LAYER1_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
0x48			**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
0x4c			**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
0x50			**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.
0x60			**LCD_CONF**	
[31:17]			RSVD	
[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

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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: DSI interface 3'b011: DPI interface 3'b100: JDI serial interface 3'b101: JDI parallel interface 3'b110: 8080 DBI Type A others: reserved
[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
0x64			**LCD_IF_CONF**	
[31:25]			RSVD	
[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
[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.

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
0x68			**LCD_MEM**	
[31:0]	rw	32'h0	ADDR	address for AHB LCD/AHB RAM
0x6c			**LCD_O_WIDTH**	
[31:10]			RSVD	
[9:0]	rw	10'h0	OFFSET	AHB RAM address offset for each line
0x70			**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
0x74			**LCD_WR**	
[31:0]	rw	32'h0	DATA	LCD write data
0x78			**LCD_RD**	
[31:0]	r	32'h0	DATA	LCD read data
0x7c			**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)
[21:20]	rw	2'h0	RD_LEN	SPI read data length(single access)

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	DEV_ID	SPI device ID
0x80			**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
0x84			**TE_CONF2**	
[31:0]	rw	32'h0	DLY_CNT	TE delay counter
0x88			**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
0x8c			**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
0x90			**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
0x94			**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
0x98			**DPI_IF_CONF5**	
[31:23]			RSVD	
[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

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
0x9c			**DPI_CTRL**	
[31:3]			RSVD	
[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
0xa0			**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
0xa4			**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
0xa8			**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
0xac			**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
0xb0			**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
0xb4			**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
0xb8			**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
0xbc			**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
0xc0			**JDI_PAR_CONF5**	
[31:16]	rw	16'h0	VCK_WIDTH	jdi parallel interface VCK width, VCK width = lcd_ck_cycle * VCK_WIDTH
[15:0]	rw	16'h0	VST_WIDTH	jdi parallel interface VST width, VST width = lcd_ck_cycle * VST_WIDTH
0xc4			**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
0xc8			**JDI_PAR_CONF7**	
[31:17]			RSVD	

Table continued on next page...

Table 10-1: LCDC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
0xcc			**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
0xd0			**JDI_PAR_STAT**	
[31:16]	r	16'h0	VPOS	jdi parallel vertical position
[15:0]	r	16'h0	HPOS	jdi parallel horizontal position
0xd4			**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
0xd8			**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
0xdc			**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

10.3 eZip™ Lossless Compression Decoder

eZip is located in HPSYS.

The eZip™ decoder is a real-time lossless decompression module based on a proprietary algorithm, with a compression ratio comparable to the Zip format. It can be used to decode and store general data, thereby accelerating real-time data loading capabilities. If data is transmitted from outside the chip, compressed transmission helps shorten transmission time and reduce transmission power consumption.

In addition, the eZip™ also supports proprietary format image compression, with a compression ratio comparable to the PNG format, and supports independent DMA operations or coordinated reading with ePicasso™. When operating independently, the eZip™ module can flexibly decompress and transfer compressed images stored in Flash or RAM to the target buffer via the DMA mechanism. In linked mode, the ePicasso™ utilizes the eZip™ module to read images from storage and decompress them in real time, then performs the required 2.5D calculations according to the standard graphics pipeline, thereby eliminating the need for a temporary buffer to store decompressed images.

Through the above mechanism, the eZip™ effectively reduces storage capacity requirements for image assets, maximizes asset richness within limited storage, and decreases bandwidth demands on external memory, thereby signifcantly im-

proving overall system operational effciency.

The eZip™ module is designed to decode and output eZip™ compressed images. This module reads compressed data via the AHB bus; the decoded image data can be configured to output through the AHB bus or directly sent to the epic module for further processing.

This module features the following characteristics:

- The input and output data addresses via the AHB bus are configurable.
- Output image data can be directly sent to the epic module.
- Capable of outputting image data from a specifed region.
- Supports decoding parameter cache functionality, which reduces decoding time upon a cache hit.

11 Audio

11.1 PDM

The chip includes 2 PDMs, both located within HPSYS , with input and output connected to IO(PA) , and capable of sending requests to DMAC1.

11.1.1 Introduction

PDM (Pulse Density Modulation) The pulse density modulation interface is primarily used to convert PDM audio signals collected by the PDM microphone into PCM (Pulse Code Modulation) signals for subsequent audio processing.

Main functions:

- Supports simultaneous stereo signals for left and right channels, and can also independently capture mono signals.
- Available PDM microphone clock rate:3.072 MHz 、1.536MHz、0.768MHz、1.024MHz、2.4MHz、1.6MHz、0.8MHz etc.
- Supported PCM data rate: 48 kHz 、 32kHz、 24kHz、 16kHz、 12kHz、 8kHz etc.
- Supports 32-bit、 24bit、 16bit、 8-bit PCM signals
- Supports gain adjustment with 0.5 dB resolution ranging from -15 dB to 45 dB

11.1.2 User Instructions

The PDM (Pulse Density Modulation) module is designed to support digital microphones.

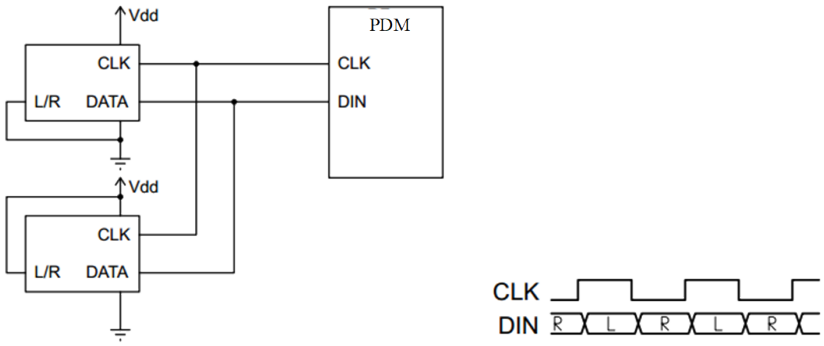


Figure 11-1: Typical connection of digital microphones via the PDM module

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 pin (L/R) , one microphone provides valid data on the rising edge of the CLK , while the other provides valid data on the falling edge of the CLK .

11.1.2.1 Overall Structure of the PDM Module

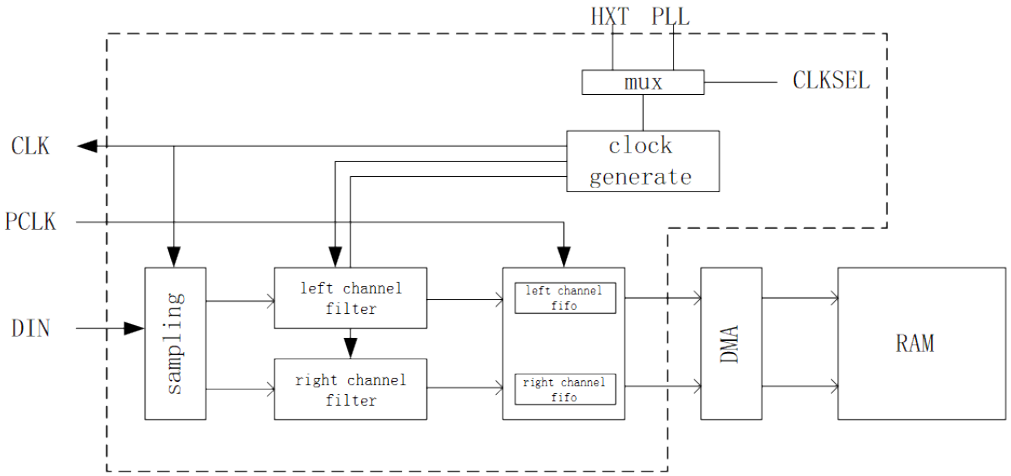


Figure 11-2: Overall Structure of the PDM Module

11.1.2.2 Clock Structure of the PDM Module

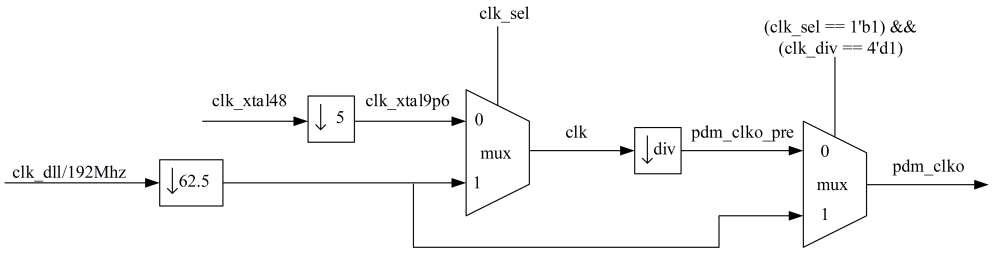


Figure 11-3: Clock Structure of the PDM Module

The PDM Module has two clock sources: one is the system's 48MHz Crystal Oscillator, and the other is the system's clk_dll clock after a 62.5 fold clock division. The 48MHz Crystal Oscillator first passes through a 5 fold clock divider inside the PDM Module to obtain a 9.6MHz clock. This clock is then selected between itself and the audio PLL clock, and after passing through a configurable clock divider, the final clock pdm_clko is provided to the digital microphone. The final output data rate of the PDM Module is pdm_clko / (sinc_rate×lpf_downsample) . Among them, sinc_rate and lpf_downsample configure the registers sinc_rate and lpf_ds according to the following table to obtain the final data.

Table 11-1: configuration Table of PDM Microphone Clock Source and Corresponding Output Data Rate

PDM_CLK(MHz)	Fs (PCM OutputRate, KHz)	OSR (Oversampling Rate)	SINC RATE (CIC Downsampling Rate)	LPF Downsampling 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

Table continued on next page...

Table 11-1: configuration Table of PDM Microphone Clock Source and Corresponding Output Data Rate (Continued)

PDM_CLK(MHz)	Fs (PCM OutputRate, KHz)	OSR (Oversampling Rate)	SINC RATE (CIC Downsampling Rate)	LPF Downsampling 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	

Taking the configuration in the first row of the table as an example, the register configuration of the PDM module is explained. Output clock 3.072MHz , output data rate 48 kHz . Output data bit width 16 bits, stereo enabled.

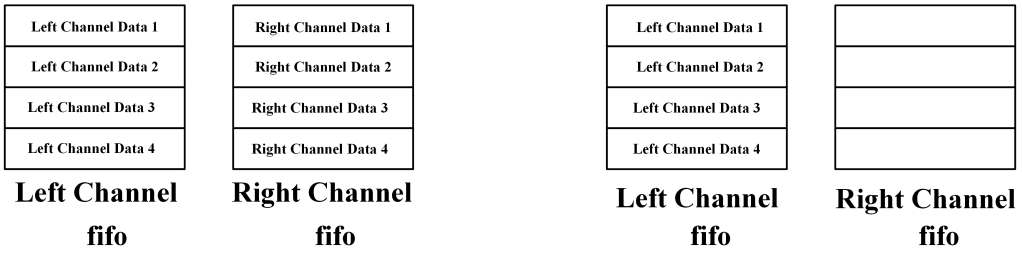
PDM Registers Configuration Procedure:

1. Select the output rate according to the table. Select the clock source for PDM_CLK according to the table and configure the clk_sel register in 0x00, where 0x0 indicates the input clock of the pdm module is 9.6MHz, and 0x1 indicates the input clock of the pdm module is the 192 MHz dll clock divided by 62.5. In this example, the clk_sel is configured as 1. (The configuration of the dll clock is not covered here.)
2. Configure the sinc_rate and sinc_order_sel registers in 0x08 according to the SINC RATE and SINC ORDER values in the table, where s inc_order_sel set to 1 corresponds to a SINC ORDER of 4 in the table, and 0 corresponds to 3. According to the LPF post-downsampling rate configuration in register 0x34, specifically the lpf_ds register and the lpf_bypass register, setting the lpf_ds register to 1 corresponds to an LPF downsampling factor of 4 as indicated in the Excel sheet; setting the lpf_ds register to 0 corresponds to an LPF downsampling factor of 2 in the Excel sheet; setting lpf_ bypass to 1 corresponds to LPF BYPASS as shown in the table. In this example, 0x08 sinc_rate=64, sinc_order_sel=0, 0x34 lpf_ds=0.
3. Enable the corresponding signals according to the channel configuration:

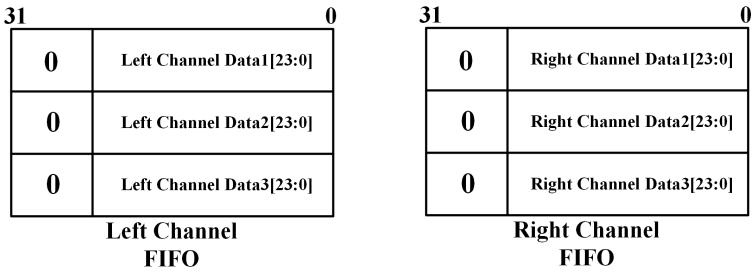
		Stereo		Left channel	Right channel
Register 0x0 CFG0	left_en	1	0	1	0
	right_en	1	0	0	1
	stereo_en	0	1	0	0

When the swap_en register at 0x0 is set to 0, the PDM module interprets the right channel as collecting data output on the rising edge of the PDM mic, and the left channel as collecting data output on the falling edge of the PDM mic; When the swap_en register at 0x0 is set to 1, the behavior is reversed: the pdm module treats the data output on the falling edge of the pdm mic as the right channel acquisition, and the data output on the rising edge of the pdm mic as the left channel acquisition. Configure according to requirements. In this example, 0x0 stereo_en=1 left_en=0 rignt_en=0 or stereo_en=0 left_en=1 rignt_en=1.

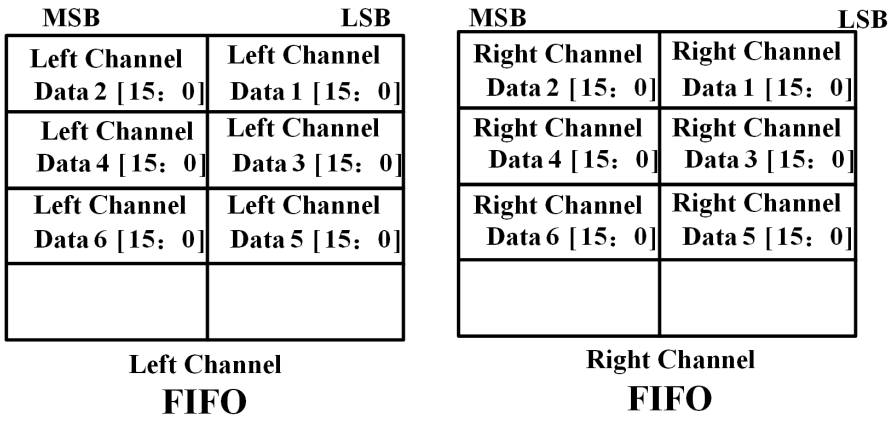
4. Configure according to the output data format of the pdm module. For different output data bit widths, configure the byte_trunc register at 0x38 accordingly. Setting 0 corresponds to 24-bit output, and setting 1 corresponds to 16-bit output.
5. Configure the byte_con register at 0x38 based on whether the left and right channel data need to be mixed. Setting 0 stores the left and right channel data separately in their respective ffos; setting 1 stores both channels' data in the left channel fifo. Diagram illustrating the differences in output data storage corresponding to different byte_con register configurations.



Data less than 32 bits will be automatically zero-padded at the high bits to form 32 bit data. For example, 24 bit stereo data with byte_con set to 0 is shown in the fgure below.



For 16 bit stereo data, byte_con set to 0 is shown in the fgure below.



In this example, there is no requirement for the data storage format; users should select the configuration according to their needs.

6. Configure the dma registers according to the dma usage instructions to transfer data from the pdm fifo to the specifed ram address one 32 bit data at a time via dma .
7. Configure 0x0 Medium pdmcoreen Register enable for the pdm Module.

The sequence of 1~6 is not fixed, provided it is completed before 7.

11.1.2.3 Precautions

1. Interrupts generated by the PDM Module are all triggered by errors caused by FIFO overflow.
2. In 0x18 PGA_CFG, pga_gain_l and pga_gain_r configure the gain of the left and right channels, with a range of 0~120. The range is -15dBm~45dBm; 0 represents -15dBm gain, 120 represents 45dBm gain, with each bit representing 0.5dBm.
3. 24-bitoutput data does not support combining left and right channels

11.1.3 PDM Registers

Table 11-2: PDM Register Map

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**	

Continued on next page...

Table 11-2: PDM Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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**	
[31:8]			RSVD	
[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: trunc 24bits to 16bit ; 0: maintain 24bits ;
[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
[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	

Continued on next page...

Table 11-2: PDM Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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

The chip contains 2 I2S interfaces, both located in HPSYS , with input and output connected to IO(PA) , capable of sending requests to DMAC1.

I2S1 supports only master mode reception and can be used to connect microphones.

I2S2 supports master mode reception and transmission, suitable for connecting microphones and amplifers.

11.2.1 Introduction

I2S (also called IIS , i.e., Inter IC Sound) bus , also known as the integrated circuit internal audio bus, is a bus standard developed by Philips for audio data transmission between digital audio devices. This bus adopts a master/slave mode, dedicated to data transmission between audio devices, and is widely used in various multimedia systems.

Currently, the I2S interface supports MSB alignment (left alignment),LSB alignment (right alignment), and the I2S standard mode.

The I2S standard mode is illustrated in the fgure below:

Data transmission begins with the MSB during the second rising edge of the BCLK following the LRCLK signal; subsequent bits are transmitted sequentially down to the LSB. Transmission depends on word length, BCLK frequency, and sampling rate (BCLK = Fs×number of channels×sampling bit depth). There must be an unused BCLK cycle between the LSB of one sample and the MSB of the next. When LRCLK is 0, left channel data is transmitted; when LRCLK is 1, right channel data is transmitted.

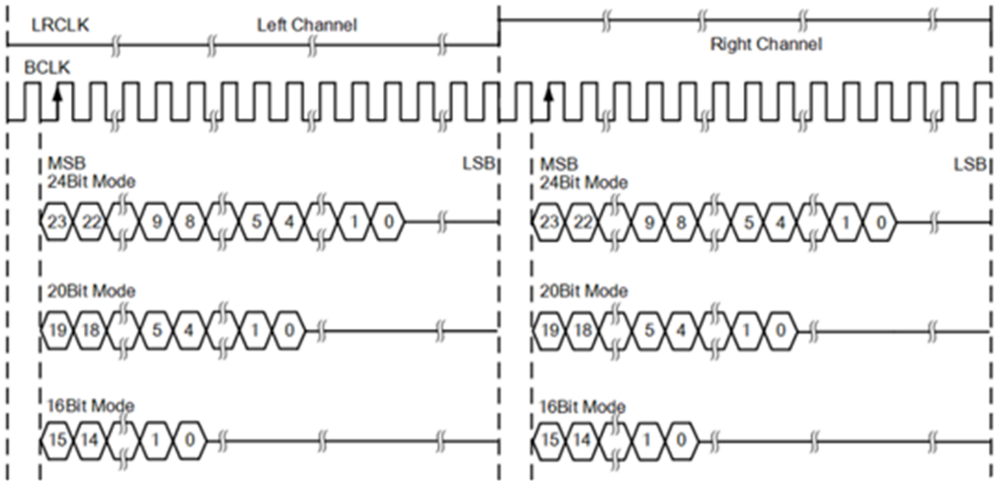


Figure 11-4: I2S Standard Mode

I2S Left-Justifed Mode as shown in the figure below:

The MSB of the standard left-justifed format data is not delayed by one clock cycle relative to BCLK. The MSB of the left-justifed format left and right channel data is valid at the first rising edge of BCLK following the edge transition of LRCLK. LRCLK is 1 for transmitting left channel data, and LRCLK is 0 for transmitting right channel data.

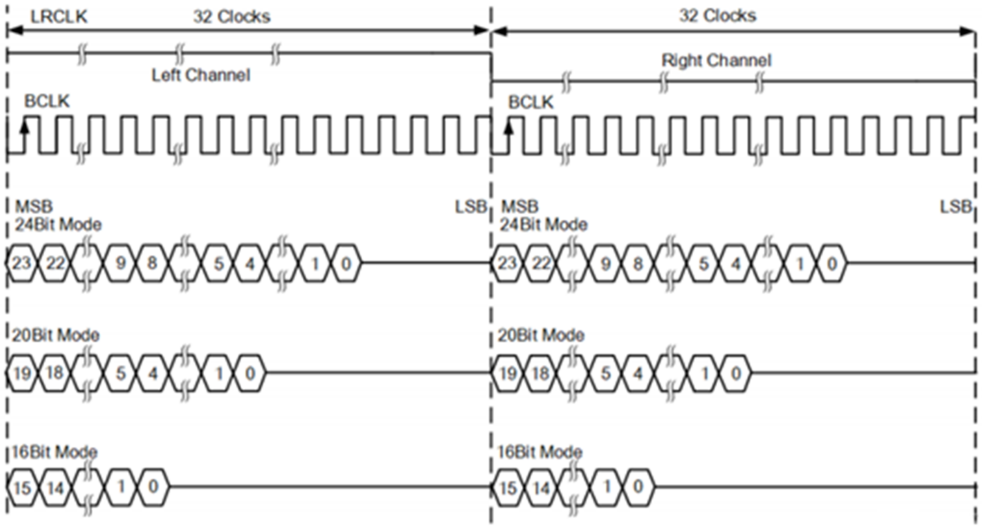


Figure 11-5: I2S Left-Justifed Mode

I2S Right-Justifed Mode as shown in the figure below:

Upon completion of the LSB transmission of the audio data, LRCLK performs its second toggle, LRCLK is 1 for transmitting left channel data, and LRCLK is 0 for transmitting right channel data.

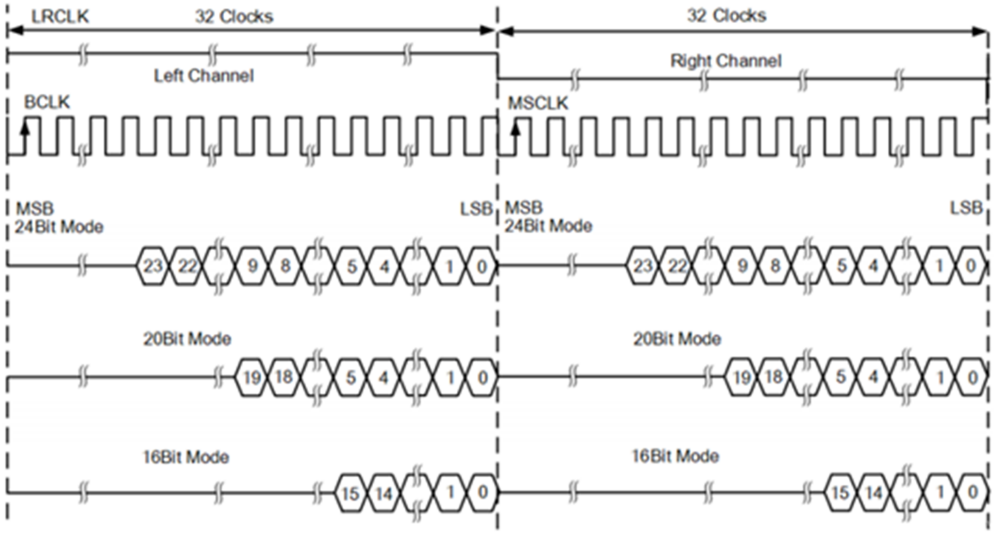


Figure 11-6: I2S Right-aligned

11.2.2 I2S Function Description

Currently, the I2S module only supports master mode. In master mode, the MCU can provide the sampling clock LRCK and the bit clock BCLK .

The I2S module supports two data formats: left-aligned and right-aligned. In master mode, users can define the ratio between BCLK and LRCK according to requirements.

11.2.3 I2S Registers

Table 11-3: I2S Register Map

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
[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

Continued on next page...

Table 11-3: I2S Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	
[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 next page...

Table 11-3: I2S Register Map (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**	
[31:1]			RSVD	
[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 next page...

Table 11-3: I2S Register Map (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**	
[31:1]			RSVD	
[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 next page...

Table 11-3: I2S Register Map (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 next page...

Table 11-3: I2S Register Map (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 next page...

Table 11-3: I2S Register Map (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**	
[31:2]			RSVD	
[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
[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

Continued on next page...

Table 11-3: I2S Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
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	

12 Accelerator

12.1 Neural Network Accelerator

12.1.1 Neural Network Matrix Convolution Accelerator (NNACC)

The matrix convolution accelerator is designed to meet the demand for fundamental matrix computing power in machine learning operations and can be broadly applied across various neural network frameworks. The accelerator offers a comprehensive memory access interface and provides flexible data address configuration. Supports a maximum input matrix size of 255×255 and up to 128 input/output channels. Supports 8-bit integer operations, fulflling the majority of edge AI computing requirements, including voice command recognition, heart rate monitoring, step counting, electrocardiogram, and other sensor computation scenarios.

12.1.2 Neural Network Co-Processor (NN Co-Processor)

The neural network co-processor is connected to the hpcpu/lpcpu via the co-processor interface. Software invokes this processor through dedicated co-processor instructions. The co-processor features include:

- Data bus width is 64Bit
- Supports 8Bit width MAC operations
- Supports single instruction 4 independent MAC operations

12.2 CRC

The chip contains 2 CRC units, with CRC1 located in HPSYS and CRC2 located in LPSYS.

12.2.1 Introduction

CRC (Cyclic Redundancy Check) supports CRC calculations with specifed bit widths, arbitrary generating polynomials, and arbitrary initial values. Data can be input via CPU or DMA, with the minimum input unit being a single byte and no maximum byte length limitation. A single PCLK cycle is sufcient to complete the computation for a single-byte input. The verifcation result is obtained immediately after all data input is completed. Supports bit order reversal for both input and output data. Supports input data with various valid bit widths.

12.2.2 Main Features

- 7/8/16/32-bit CRC calculation
- Arbitrary custom polynomial
- Arbitrary initial value
- Input data supports valid bit widths of single byte/double byte/three bytes/four bytes
- Input data supports bit order reversal for byte/double byte/four byte
- Output data supports bit reversal of high and low bits.

- Calculation speed is 1 byte per PCLK cycle.

12.2.3 CRC Configuration Method

Before starting CRC calculation, the corresponding registers must be pre-configured, including polynomial width, valid data bit width, input/output reversal mode, polynomial, and initial value. The mainstream CRC format configuration methods are shown in the table below.

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 result XOR operation; post-processing must be performed by software after reading the result.

12.2.4 Data Format

The basic data unit for CRC calculation is the byte. The data written to the DR register is fixed at 4 bytes. Which bytes participate in the calculation is specifed by CR_DATASIZE. The gray cells in the table below indicate the bytes participating in the calculation.

Table 12-2: Data involved in computation

CR_DATASIZE	DR			
0	BYTE3	BYTE2	BYTE1	BYTE0
1	BYTE3	BYTE2	BYTE1	BYTE0
2	BYTE3	BYTE2	BYTE1	BYTE0
3	BYTE3	BYTE2	BYTE1	BYTE0

Each data byte participating in the computation can be specifed individually; however, it should be noted that changing CR_DATASIZE must be done when SR_DONE is 1 ; otherwise, it may affect the current data calculation result.

The computation sequence proceeds in the order of BYTE0, BYTE1, BYTE2, BYTE3. When calculating each byte, the default order is from the most significant bit to the least signifcant bit. If the input inversion mode is configured, the order follows the inverted sequence from the most signifcant bit to the least signifcant bit. The table below provides configuration examples; the input format can be flexibly adjusted according to the data format in memory.

Table 12-3: Computation sequence

DATASIZE	REV_IN	DR	Input after reversal	First cycle Calculate bytes	Second beat Calculate bytes	Third beat Calculate bytes	Fourth beat Calculate bytes
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 PCLK cycle. When data is input continuously, the calculation time is approximately the number of bytes×PCLK cycles. However, the data input rate through APB may be lower than the calculation rate, so the total time consumed by CRC is determined by both data write time and computation time.

12.2.6 CRC Configuration procedure

1. Configure the CRC format and set POL as required., INIT, CR_POLYSIZE, CR_REV_IN, CR_REV_OUT, CR_DATASIZE。
2. Set CR_RESET to 1 to initialize the CRC.
3. The required data is continuously transferred to the DR register by the CPU or DMA.
4. If the remaining data byte count does not match CR_DATASIZE, it is necessary to first query SR_DONE. When it equals 1, modify CR_DATASIZE and write the remaining data into the DR register.
5. Read the DR register to obtain the calculation result, and perform a software XOR operation as required to obtain the final CRC value.

12.2.7 CRC Registers

Table 12-4: CRC Register Map

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

Continued on next page...

Table 12-4: CRC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
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

AES is located within HPSYS.

13.1.1 Introduction

The AES_ACC module of the SF32LB58x primarily accelerates encryption and decryption computations for dedicated algorithms in the security domain . Symmetric encryption algorithms include AES128, AES192, AES256, and SM4. Supported modes include ECB, CTR, and CBC. Hash algorithms include SHA1, SHA224, SHA256, and SM3. Upon activation, the AES_ACC module uses the internal DMA to read the input data, processes it according to the algorithm, and writes the resulting output to the target address via the internal DMA or stores it in the module's internal registers.

13.1.2 AES Function Description

13.1.2.1 Symmetric Encryption Algorithms

Symmetric encryption algorithms mainly include AES and SM4 , where AES is further divided into AES128 , AES192 , and AES256 based on the length of the Key , while SM4 is the national standard symmetric encryption algorithm. Regarding strength, the longer the key length, the higher the security, but the encryption and decryption time also increases; the SM4 algorithm requires the longest processing time.

13.1.2.2 Symmetric Encryption Modes

Symmetric encryption modes mainly include ECB, CTR, and CBC.

ECB Mode: TheECB mode directly encrypts and decrypts plaintext data using the KEY . Data is processed in blocks of 16 bytes , and each encryption or decryption operation is performed on the entire 16-byte block. The advantage is that each data group is independent, enabling parallel computation and supporting random read and write operations by group. The disadvantage is that if different data groups have identical plaintext, their ciphertext will also be identical, making it susceptible to cryptanalysis.

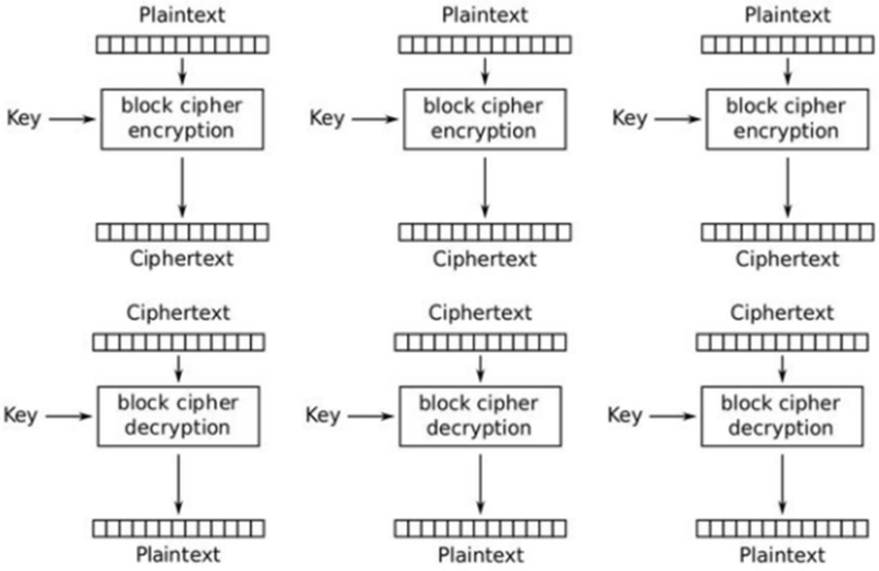


Figure 13-1: ECB Mode Decryption

CTR Mode: The CTR mode encrypts a vector composed of a NONCE and a COUNTER using the KEY, then XORs the encrypted result with the plaintext data to produce the ciphertext, thereby performing encryption. During decryption, the same vector is encrypted and XORed with the ciphertext data to recover the plaintext, thereby performing decryption. In practical applications, the NONCE is typically represented by a constant, while the COUNTER uses the data address, ensuring that each data group's vector is unique, so the XOR data used during encryption and decryption also differs. With an understanding of the vector composition method used by CTR, this mode encrypts and decrypts each data block in dependently, enabling parallel computation, similar to the ECB mode. Additionally, this mode involves only encryption and XOR operations, resulting in a relatively simple structure. The vector operations during encryption and decryption are independent of the data, making this mode commonly used for encrypting and decrypting externally stored data.

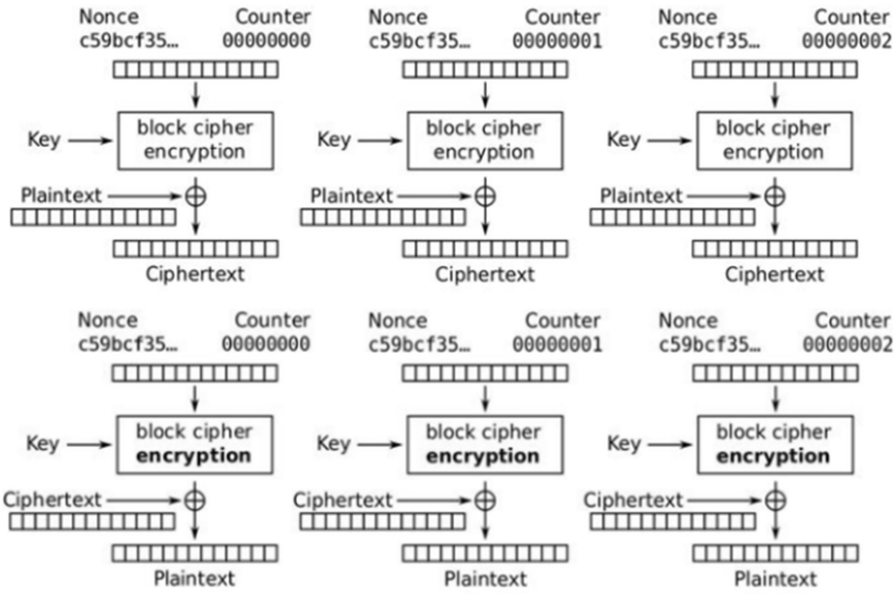


Figure 13-2: CTR Mode Decryption

CBC Mode: CBC mode uses the ciphertext of the previous data block as the initialization vector, XORs it with the current block, and then encrypts the result with the KEY to generate the ciphertext. During decryption, the ciphertext is decrypted with the KEY and then XORed with the ciphertext of the previous block to produce the plaintext. This mode supports both encryption and decryption of data and can also use the last ciphertext block as the MAC value for data integrity verifcation. Because each data block is interrelated during encryption and decryption, it offers higher security. However, it does not support parallel computation or random read/write access to individual data blocks.

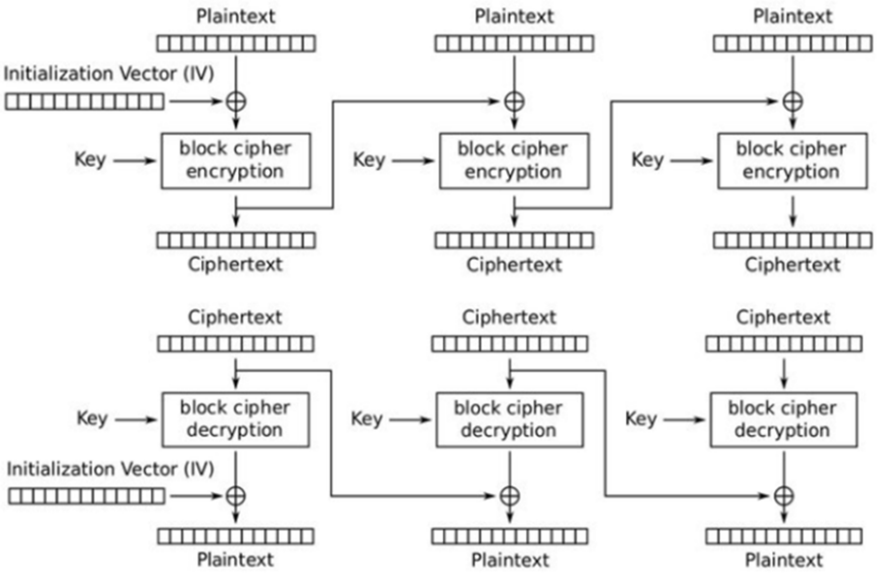


Figure 13-3: CBC Mode Decryption

13.1.2.3 Multiple Invocations of Symmetric Encryption and Decryption

Sometimes, it is necessary to perform encryption and decryption operations on large volumes of data. However, due to limitations in storage capacity or transmission rate, the data is divided into multiple batches for processing. When multiple invocations of the symmetric encryption and decryption module are required to continuously process a large data segment, it is essential to consider the preservation and maintenance of data context between calls.

ECB Mode:

ECB Mode performs encryption and decryption operations independently on each 16-byte data block. For a single encryption or decryption operation , the data size must be an integer multiple of 16 bytes. Excess data can be combined with the next batch of data for processing.

CTR Mode:

In CTR Mode, each group of 16 bytes of data corresponds to a specific NONCE and COUNTER value. Typically, the NONCE is constant, and the COUNTER represents the current data group number, incrementing by 1 for each group. For single encryption or decryption operations, the upper-layer software must ensure the data size to is an integer multiple of 16 bytes and record the COUNTER value based on the data size. In the subsequent invocation, the accumulated COUNTER value is used as the initial vector input to the corresponding IVregister. Excess data can be combined and processed when the next batch of data is ready.

CBC Mode:

In CBC Mode, the initial vector for each data group is derived from the ciphertext of the previous group. For single encryption or decryption operations, the upper-layer software must ensure the data size is an integer multiple of 16 bytes and record the ciphertext of the last processed data group. In the subsequent operation, this ci phertext is used as the initial vector input to the corresponding IV register. Excess data will be merged and processed once the next batch of data is ready.

13.1.3 AES Register

Table 13-1: AES Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**COMMAND**	
[31:3]			RSVD	
[2]	rw	1'h0	AUTO_GATE	auto clock gating
[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**	
[31:2]			RSVD	
[1]	r	1'h0	FLASH_KEY_VALID	flash key valid indicator
[0]	r	1'h0	BUSY	AES_ACC block is busy
0x08			**IRQ**	
[31:19]			RSVD	
[18]	rw1c	1'h0	SETUP_ERR_RAW_STAT	setup error raw status
[17]	rw1c	1'h0	BUS_ERR_RAW_STAT	bus error raw status
[16]	rw1c	1'h0	DONE_RAW_STAT	done raw status
[15:3]			RSVD	
[2]	rw1c	1'h0	SETUP_ERR_STAT	setup error status
[1]	rw1c	1'h0	BUS_ERR_STAT	bus error status
[0]	rw1c	1'h0	DONE_STAT	done status
0x0C			**SETTING**	
[31:3]			RSVD	
[2]	rw	1'h0	SETUP_ERR_IRQ_MASK	setup error interrupt mask, 0: mask the interrupt
[1]	rw	1'h0	BUS_ERR_IRQ_MASK	bus error interrupt mask, 0: mask the interrupt
[0]	rw	1'h0	DONE_IRQ_MASK	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

Table continued on next page...

Table 13-1: AES Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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	input data address
0x18			**DMA_OUT**	
[31:0]	rw	32'h0	ADDR	output data address
0x1C			**DMA_DATA**	
[31:28]			RSVD	
[27:0]	rw	28'h0	SIZE	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**	
[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

13.2 TRNG

TRNG is located in HPSYS.

13.2.1 Introduction

TRNG, the True Random Number Generator, utilizes an oscillator circuit composed of digital logic to generate random entropy sources, which, after a series of validations, produce a random seed. This seed is then used by a pseudo-random number generator to generate random numbers for system use.

13.2.2 Module Architecture

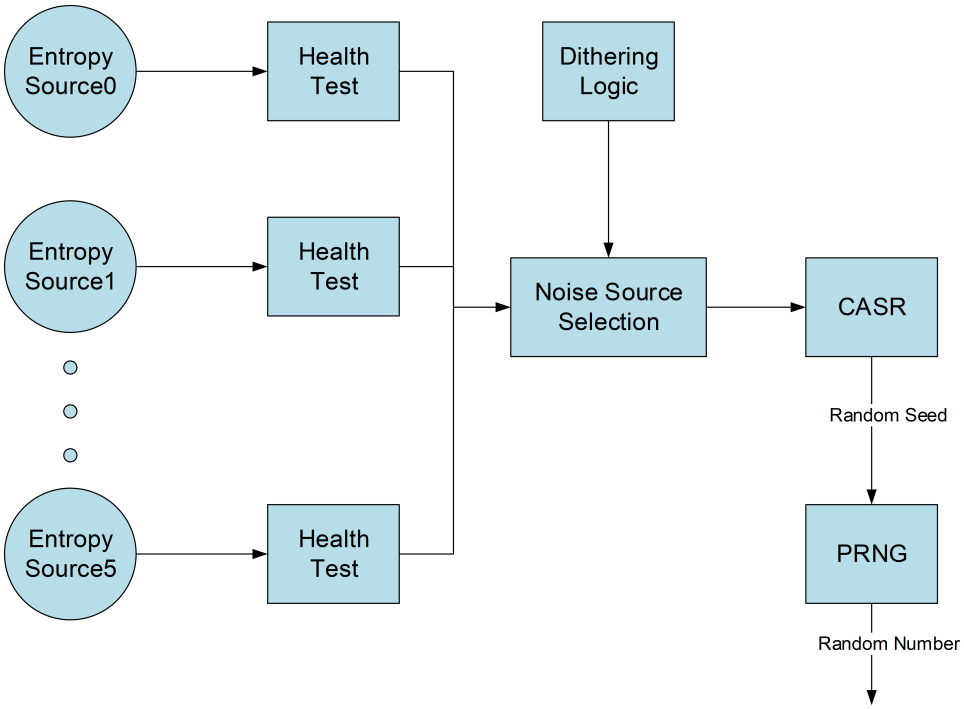


Figure 13-4: TRNG Block Diagram

The True Random Number Generator primarily consists of 3components: entropy source, random seed generator, and random number generator.

13.2.3 Function Description

13.2.3.1 Entropy Source

The entropy source comprises six inverter chains. Each time a new random seed is generated, all inverter chains are activated, and a specific verifcation algorithm is employed to produce the random seed. The generation time for the random seed is relatively long, and power consumption is correspondingly higher. It is generally recommended to regenerate a new seed under two circumstances: first, when the original seed' s lifecycle has expired and a new seed must be periodically generated to replace it; second, when the software manually initiates seed generation due to special requirements.

The entropy source activates each time a seed is generated. TheVN validator threshold is used to flter the entropy source; a higher threshold results in a more lenient flter, enabling faster seed generation but with reduced randomness.

Users can directly trigger gen_seed_start to activate the entropy source module and generate a new random seed.

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, providing them to the subsequent PRNG module. Users can also obtain random seeds

directly via registers. When the user configures use_ext_seed to 1 , the subsequent PRNG module will not use 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 an existing random seed, featuring a relatively short and fixed period. It should be noted that the pseudo-random number generator may enter a self-lock state due to certain internal data sequences causing a deadlock in the linear feedback register. Upon detecting the prng_lockup state, it is recommended to regenerate a 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 generate a random seed before producing random numbers.

13.2.4 TRNG Registers

Table 13-2: TRNG Register Map

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	
[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**	

Continued on next page...

Table 13-2: TRNG Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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**	
[31:0]	r	32'h0	val	random number value7

13.3 efusec

efusec is located in HPSYS.

13.3.1 Introduction

efusec, i.e., efuse control, manages the blowing of corresponding efuse bits via write operations. Blown bits read back as 1. The stored information can be retrieved through read operations.

13.3.2 Key Features

- Controls 4 blocks of 256-bit efuse units
- Supports reading and writing of 1024-bit information
- Each operation controls the read/write of a single efuse unit
- Read/write operations are single bitserial
- Read/write completion can generate an interrupt signal
- Read and write masking functionality can be implemented
- Partial stored information is output directly through the module interface
- The default value of efuseis all 0

13.3.3 Write Operation

The write operation of efusec is a programming operation on the efuse , where bits written as 1 are set to open circuit by fusing, causing the read-back data value to be 1. The information to be written is configured via 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 MODE Register selection for write operation
- Configure IE Register to enable interrupt status
- Configure PGM_DATA*Register for writing programming data
- Configure TIMER Register to set critical duration
- Configure EN Register to start write operation
- Wait for interrupt arrival or query SR Register to obtain write completion status

A fused bit cannot be restored to the 0 state; an unfused bitcan be fused again through a write operation.

13.3.4 Read Operation

The read operation of eFusec involves reading the stored information of the efuse. The read data can be queried via the BANK*_DATA* registers, with each unit having an independent register for data retrieval. The control flow of the read operation is as follows:

- Configure the BANKSEL register to select the efuse cell for reading
- Configure the MODEregister to select the read operation
- Configure IE Register to enable interrupt status
- Configure TIMER Register to set critical duration
- Configure EN Register to start write operation
- Wait for an interrupt or poll the SR register to obtain the read completion status
- Read the BANK*_DATA* register to acquire the read value
- Partial read values are directly output via interface signals

13.3.4.1 Read/Write Timing Control

The read/write timing of the efuse has specific requirements, particularly the blow time. When the operating clock of the efusec module changes, the clock count value must be adjusted to satisfy the efuse read/write timing requirements. The timing control points are as follows

- TCKHP : Single-bit blow time, 10us
- THPCK : Hold time for write enable start,>20ns
- THRCK : Hold time for read enable start,>500ns

These three timing parameters can be configured via the TIME register based on the working clock count value. The register reset value corresponds to the count value at a 48MHz working clock.

13.3.4.2 Read-Write Mask Function

To prevent unauthorized read and write operations, when data is configured and no further updates are intended, a fixed bit in bank0 can be controlled to mask the read and write functions of the efuse modules in banks 1 to 3. Once masked, the corresponding efuse cannot be written to, nor can the read values be updated.

- bank0[255:254]=2’b11,Mask bank3 write function;
- bank0[253:252]=2’b11,Mask bank3 read function;
- bank0[251:250]=2’b11,Mask bank2 write function;
- bank0[249:248]=2’b11,Mask bank2 read function;
- bank0[247:246]=2’b11,Mask bank1 write function;
- bank0[245:244]=2’b11,Mask bank1 read function;

13.3.4.3 Module Interface Output Signals

Certain efusevalues are output via module interface signals, as follows:

Table 13-3: efuse Interface Signals

Index	Signal Name
1	idsel
2	swddis
3	pkgid[1:0]
4	uid[127:0]
5	rootkey[255:0]

13.3.5 efuse Registers

Table 13-4: efusec Register Map

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

Continued on next page...

Table 13-4: efusec Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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

Continued on next page...

Table 13-4: efusec Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[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	

13.4 BUSMON

13.4.1 Introduction

BUSMON (Bus Monitor) can monitor the transmission behavior of each master and slave device on the system AHB bus.BUSMON has a total of 8 channels; each channel can select one master or slave device for monitoring, count the number of AHB read and write transmissions within a specifed address range, and trigger a PTC event upon reaching a

defned count.

BUSMON can be used to monitor whether a specific address has been tampered with, as well as to measure the bus load of devices.

The chip contains two BUSMON units. BUSMON1 can monitor the AHB devices of the HPSYS. BUSMON2 can monitor the AHB devices of the LPSYS.

13.4.2 Main Features

- 8 independently configurable channels can operate simultaneously
- Capable of monitoring all master and slave devices within the subsystem
- Configurable address range for monitoring
- Selectable statistics for read/write/read-write transfers
- 31-bit counter, 24-bit comparator
- Counter overflow can automatically reset and restart; overflow status is queryable
- 8 channels, each generating PTC triggers

13.4.3 Monitoring Device Selection

Each channel selects the monitored master or slave device via CRx.SEL. When a master device is selected, the channel monitors AHB transfer requests issued by the master device. When a slave device is selected, the channel monitors AHB transfer requests received by the slave device.

Table 13-5: BUSMON1 Select Monitor HPSYS Device

CRx.SEL	CH1	CH2	CH3	CH4	CH5	CH6	CH7	CH8
0	HCPU_C	HCPU_C	HCPU_C	HCPU_C	HCPU_C	HCPU_C	HCPU_C	HCPU_C
1	HCPU_S	HCPU_S	HCPU_S	HCPU_S	HCPU_S	HCPU_S	HCPU_S	HCPU_S
2	DMAC1	EXTDMA	EPIC_A	EPIC_B	LCDC1	EZIP	USBC	SDMMC2
3	SDMMC1	LP2HP	PTC1	LCDC1	DMAC1	EXTDMA	EPIC_A	EPIC_B
4	/	/	/	/	/	/	/	/
5	/	/	/	/	/	/	/	/
6	RAM0	RAM1	RAM2	RAM3	RAM4	RAM5	HP_APB	HP_AHB
7	QSPI1	QSPI2	QSPI3	HP2LP	PSRAM	QSPI1	QSPI2	/

Table 13-6: BUSMON2 Select Monitor HPSYS Device

CRx.SEL	CH1	CH2	CH3	CH4	CH5	CH6	CH7	CH8
0	LCPU_C	LCPU_C	LCPU_C	LCPU_C	LCPU_S	LCPU_S	LCPU_S	LCPU_S
1	DMAC2	LCDC2	HP2LP	PTC2	DMAC2	LCDC2	HP2LP	PTC2
2	RAM0	RAM1	RAM2	RAM3	RAM4	RAM5	LP_APB	LP_AHB
3	QSPI4	PHY_DUMP	LP2HP	TCM	/	/	/	/

13.4.4 Monitored Address Range

Each channel can individually configure the monitored address range via registers. The monitored address range for channel x is [MINx, MAXx), where the MIN address is included and theMAX address is excluded. For example, when

MIN1=0x60010000 and MAX1=0x60020000, channel 1 will count AHB transfers for addresses from 0x60010000 to 0x6001FFFF. of AHB transfers at address 0x60020000 will not be counted.

13.4.5 Monitored Transfer Types

Each channel can configure the monitored transfer types via the CRx.OP register. When CRx.OP is 0, the channel monitors only read transfers within the address range. When CRx.OP is 1, the channel monitors only write transfers within the address range. When CRx.OP is 2 or 3, the channel monitors all transfers within the address range.

13.4.6 Monitoring count

Once the channel is enabled, each time the monitored master device initiates an AHB transfer request matching the monitored address range and transfer type, or the monitored slave device receives an AHB transfer request matching the monitored address range and transfer type, the channel counter CNTx.CNT increments by 1. If the channel compare enable CRx.CMPEN is set to 1 , when the channel counter value equals the compare value CRx.CMP , the overflow flag CNTx.OF is set to 1 and a PTC event trigger is generated. If the channel compare enable CRx. CMPEN is set to 0 , when the channel counter value equals 0x7fffffff , the overflow flag CNTx.OF is set to 1 and a PTC event trigger is generated. When an overflow occurs, if the auto-clear enable CRx.AUTOCLR is set to 1 , the counter will automatically clear to 0 and restart counting.

13.4.7 Notifcation Mechanism

BUSMON does not directly generate interrupts. To obtain monitoring results, the real-time count value of the channel monitoring can be directly queried from CNTx , or the PTC event trigger generated upon count overflow can be used to produce a PTC interrupt or perform other operations.

13.4.8 Configuration Procedure

The configuration procedure to start BUSMON channel x is as follows:

1. Enable the BUSMON clock in the RCC module corresponding to the subsystem;
2. Write 1 to CCR.CLRx to reset channel x;
3. Configure CRx.SEL to select the master or slave device monitored by the channel;
4. Configure MINx and MAXx to select the monitored address range;
5. Configure CRx.OP to select the monitored transfer type;
6. If an interrupt is required, configure the comparison value CRx.CMP and configure the PTC module interrupt, selecting BUSMON channel x as the trigger source;
7. Write 1 to CER.ENx to enable channel x.

13.4.9 BUSMON Registers

Table 13-7: BUSMON Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR1**	control register for channel 1
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow

Continued on next page...

Table 13-7: BUSMON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x04			**CNT1**	counter for channel 1
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x08			**MIN1**	minimum address for channel 1
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x0C			**MAX1**	maximum address for channel 1
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x10			**CR2**	control register for channel 2
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x14			**CNT2**	counter for channel 2
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x18			**MIN2**	minimum address for channel 2
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x1C			**MAX2**	maximum address for channel 2
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x20			**CR3**	control register for channel 3
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table

Continued on next page...

Table 13-7: **BUSMON Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0x24			**CNT3**	counter for channel 3
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x28			**MIN3**	minimum address for channel 3
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x2C			**MAX3**	maximum address for channel 3
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x30			**CR4**	control register for channel 4
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x34			**CNT4**	counter for channel 4
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x38			**MIN4**	minimum address for channel 4
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x3C			**MAX4**	maximum address for channel 4
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x40			**CR5**	control register for channel 5
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x44			**CNT5**	counter for channel 5
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x48			**MIN5**	minimum address for channel 5
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x4C			**MAX5**	maximum address for channel 5
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x50			**CR6**	control register for channel 6
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow

Continued on next page...

Table 13-7: BUSMON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x54			**CNT6**	counter for channel 6
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x58			**MIN6**	minimum address for channel 6
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x5C			**MAX6**	maximum address for channel 6
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x60			**CR7**	control register for channel 7
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table
0x64			**CNT7**	counter for channel 7
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x68			**MIN7**	minimum address for channel 7
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x6C			**MAX7**	maximum address for channel 7
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x70			**CR8**	control register for channel 8
[31:8]	rw	24'hffffff	CMP	compare value to generate overflow
[7]	rw	1'h0	CMPEN	compare enable 1: overfow flag is generated when CNT reaches CMP 0: overflow flag is generated when CNT reaches maximum (all bit 1)
[6]	rw	1'h0	AUTOCLR	counter will be cleared automatically each time overflow happens
[5:4]	rw	2'h0	OP	select operation to be monitored 2'b00: read 2'b01: write 2'b10, 2'b11: read or write
[3]			RSVD	
[2:0]	rw	3'h0	SEL	select monitor source. Refer to source table

Continued on next page...

Table 13-7: **BUSMON Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
0x74			**CNT8**	counter for channel 8
[31]	r	1'h0	OF	overflow flag, happens when CNT reaches CMP or CNT reaches maximum (all bit 1)
[30:0]	r	31'h0	CNT	counter value
0x78			**MIN8**	minimum address for channel 8
[31:0]	rw	32'h0	ADDR	lower boundary of memory moniter range, included
0x7C			**MAX8**	maximum address for channel 8
[31:0]	rw	32'h0	ADDR	upper boundary of memory moniter range, excluded
0x80			**CER**	channel enable register
[31:8]			RSVD	
[7]	rw	1'h0	EN8	counter enable for channel 8
[6]	rw	1'h0	EN7	counter enable for channel 7
[5]	rw	1'h0	EN6	counter enable for channel 6
[4]	rw	1'h0	EN5	counter enable for channel 5
[3]	rw	1'h0	EN4	counter enable for channel 4
[2]	rw	1'h0	EN3	counter enable for channel 3
[1]	rw	1'h0	EN2	counter enable for channel 2
[0]	rw	1'h0	EN1	counter enable for channel 1
0x84			**CCR**	channel clear register
[31:8]			RSVD	
[7]	w1s	1'h0	CLR8	write 1 to clear counter and overflow flag for channel 8. No need to write 0
[6]	w1s	1'h0	CLR7	write 1 to clear counter and overflow flag for channel 7. No need to write 0
[5]	w1s	1'h0	CLR6	write 1 to clear counter and overflow flag for channel 6. No need to write 0
[4]	w1s	1'h0	CLR5	write 1 to clear counter and overflow flag for channel 5. No need to write 0
[3]	w1s	1'h0	CLR4	write 1 to clear counter and overflow flag for channel 4. No need to write 0
[2]	w1s	1'h0	CLR3	write 1 to clear counter and overflow flag for channel 3. No need to write 0
[1]	w1s	1'h0	CLR2	write 1 to clear counter and overflow flag for channel 2. No need to write 0
[0]	w1s	1'h0	CLR1	write 1 to clear counter and overflow flag for channel 1. No need to write 0

14 Storage Interface

14.1 SD/SDIO/eMMC

The chip contains 2 SDMMC units, both located in HPSYS.

14.1.1 Introduction

SDMMC supports SD protocol 3.0 and eMMC standard 4.51, serving as a HOST controller to interface with SD/SDIO/eMMC devices. SDMMC features an integrated chained DMA controller and a 1K-byte FIFO, enabling autonomous data read/write and supporting block data transfer. SDMMC supports SDR single-line, 4 four-line, and 8-line modes, and also supports DDR 4-line mode.

14.1.2 Key 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 withJ EDEC JESD84-B451 eMMC 4.51 Specification
- Supports SDSC/SDHC/SDXC/SDHScards
- Supports UHS-1:SDR12/SDR25/DDR50
- Supports SDR 1-line, 4-line, and 8-line modes
- Supports DDR 4-line mode
- Built-in 1K byte FIFO,with maximum support for single block of 512 bytes
- Configurable clock
- Built-in chained DMA

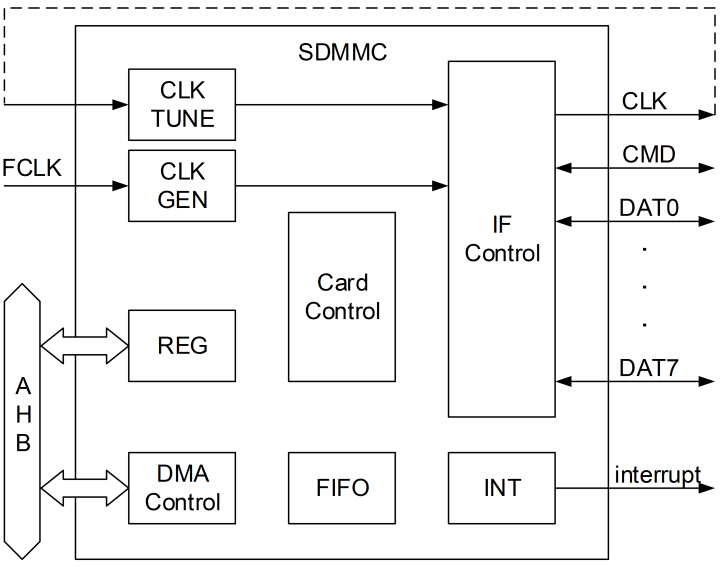


Figure 14-1: **SDMMC Block Diagram**

14.1.3 Function description

14.1.3.1 SD/eMMC Interface

The controller supports the SD/eMMC interface, which comprises one CLK line, one CMD line, and eight DAT lines (DAT0~DAT7). The CLK line is used to output the clock signal. The CMD line transmits CMD commands and response signals (RSP) according to the protocol. The DAT lines transmit data streams in compliance with the protocol. The CMD line supports only single data rate (SDR) transmission on the rising edge, i.e., CMD is valid solely on the rising edge of CLK. The DAT lines support single data rate (SDR) transmission, i.e., DAT is valid only on the rising edge of CLK; They also support double data rate (DDR) transmission, i.e., DAT is valid on both the rising and falling edges of CLK.

14.1.3.2 Clock Configuration

The functional clock FCLK of SDMMC equals the system clock HCLK frequency. When SDMMC is operating, CR3_INTCLKEN must be set to 1.

The CLK output to the SD/eMMC interface is generated by dividing FCLK. The division ratio CLKDIV is 10 bits in total, where the lower 8 bits are specifed by CR3_CLKDIVL and the upper 2 bits are specifed by CR3_CLKDIVU. When CLKDIV equals 0, the CLK frequency equals the FCLK frequency; otherwise, CLK frequency = FCLK frequency / (2×CLKDIV). For example, when FCLK is 192 MHz, to output a 400 kHz CLK, CLKCR_DIV must be configured to 240. When performing DDR transmission, the CLKDIV must be configured to an even number. The CLK frequency supports up to 48MHz.

CLK output is controlled by CR3_CLKEN; setting it to 1 enables continuous clock output, while setting it to 0 disables the output. When adjusting the clock frequency, the CLK output must be disabled and re-enabled after the configuration is complete.

14.1.3.3 Send Command

A command is a fixed-format packet sent by the controller via the CMD line and received by the SD/eMMC/SDIO device, used to configure the status of the SD/eMMC/SDIO device and control data transmission. The command packet length is fixed at 48bit.

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 identifed by a 6bit command index that distinguishes different functions and is accompanied by a 32bit parameter. The device verifes the received command using a 7bit CRC and transmits a response when a reply is required.

According to the 6bit command index (command index) , commands are generally named CMDn , where n represents the command index configured by the CR1_CMDIDX register.

The 32bit command parameter (argument)is configured via the ARG1 register.

Certain commands, such as CMD0 and CMD4 , do not require a response; Certain commands, such as CMD8 and CMD11 , require receiving a 48bit response; Certain commands, such as CMD2, require receiving a 136-bit response. Before sending a command, the expected response must be selected by configuring the CR1_RSPTYPE register

Some responses contain the command index and the CRC checksum. To verify the correctness of the response content, configure CR1_CHECKIDX and CR1_CHECKCRC as required.

Certain commands, such as CMD18andCMD24,require data read/write after transmission; therefore, the CR1_DATAPRESENT bit must be set to 1.

Command transmission is triggered by writing to CR1_CMDIDX; thus, it is recommended to prepare all command parameters and write them to the CR1 register in a single operation. In progress, initiate the command transmission process.

The status of command transmission can be queried via SR_CMDBUSY. When SR_CMDBUSY is 1, it indicates that the command transmission process (including response reception) is still ongoing.

Upon completion of the command transmission process, the command completion flag ISR_CC is set to 1 (when ISER_CCEN is 1), and an interrupt can be generated if IER_CCIE is 1. The status of response reception can be queried in the ISR register, including response timeout error ISR_CTOERR (no response start bit detected within 64 CLK), response CRC error ISR_CCRCERR, response end bit error ISR_CEBERR, and response index error ISR_IDXERR.

The received response argument is stored in RSP1 RSP4 , with the 48-bit response argument stored in RSP1 .

If the response includes a busy signal (e.g., R1b), after the busy signal ends, the transfer complete flag ISR_TC will be set to 1 (when ISER_TCEN is 1), and when IER_TCIE is 1, an interrupt can be generated.

Recommended software procedure for command transmission:

1. Configure the index, argument, and expected response type in a single step, then initiate command transmission.
2. Wait for the command complete interrupt and the transfer complete interrupt (if the response includes a busy signal) .
3. Verify the returned response and retrieve the argument.

14.1.3.4 Data Transfer

Data transfer includes reading data from the slave device and writing data to the device, initiated by the controller sending specific commands (such as CMD17 ,CMD24 , etc.); data transfer concludes after the preset length or is terminated by the controller sending a specific command (such as CMD12).

Data transfer defined by the SD and SDIO protocols may utilize single-line or 4-line modes. Data transfer defined by the eMMC protocol may utilize single-line, 4-line, or 8-line modes. The controller supports single-line,4-line, or 8-line transfers, configured via the CR2_DATWIDTH and CR2_EXTWIDTH registers. In single-line mode, data is transmitted via DAT0, which is also used to indicate the busy status of data transmission. In 4-line mode, data is transmitted via DAT0~DAT3, with DAT0 also indicating the busy status of data transmission. In 8-line mode, data is transmitted via DAT0~DAT7, with DAT0 also indicating the busy status of data transmission.

The controller supports 4-line DDR transmission. Configure the CR4_UHSMODE register to 4 for DDR transmission; otherwise, transmission operates in SDR mode.

Data Transmission Mode Configuration				
Transmission Mode	Line Width	CR2_DATWIDTH	CR2_EXTWIDTH	CR4_UHSMODE
SDR	Single Line	0	0	0
	4 Line	1	0	0
	8 Line	x	1	0
DDR	4 Line	1	0	4

Data is transmitted in blocks, with the size of each block varying according to the device type. For SD and eMMC devices, the single block data size is typically 512 bytes, but exceptions exist. Certain devices permit changing the single block data size via commands (such as CMD16). Before the controller initiates data transfer, the BSR_BSIZE register shall be configured according to the device's single block data byte size. The controller must also configure the CR1_MULTIBLK and BSR_BCNT registers to specify the total number of blocks for a single data transfer. For single block transfer, CR1_MULTIBLK shall be set to 0 , in which case BSR_BCNT is disregarded. For multi-block transfer, CR1_MULTIBLK and CR1_BCNTEN shall be set to 1 , and BSR_BCNT shall be configured to the total number of blocks to be transferred.

The direction of data transfer is configured by CR1_DIR, 0 indicates read, and 1 indicates write.

After the data transfer configuration is completed, when the controller sends a read/write command (command configured by CR1_DATAPRESENT set to 1),automatic data transfer will commence.

Data transfer must pass through the controller's internal FIFO buffer. Data to be sent is written into the FIFO by the CPU or the controller's built-in DMA, after which the controller writes the data from the FIFO to the device. Data read from the slave device is also temporarily stored in the FIFO, then retrieved by the CPU or the built-in DMA. The capacity of the FIFO is 1024 bytes, with the CPU access interface being the BUF register.

During multi-block transmission, the device will only stop transmission upon receiving the transmission termination command (such as CMD12). The controller supports automatic transmission of the termination command after all data blocks have been transmitted. This is enabled by setting CR1_AUTOCMD to 1.

Data transmission completion status can be queried via SR_DATBUSY.

There are two methods to terminate data transmission:

1. Normal transmission completion. When the specifed number of data blocks have been fully transmitted without errors, the transmission complete flag ISR_TC is set to 1 (when ISER_TCEN is 1), and an interrupt can be generated if IER_TCIE is 1
2. Transmission errors include data timeout (ISR_DTOERR) , data CRC error (ISR_DCRCERR) , and data end bit error (DEBERR) —three types in total. Each error can generate a corresponding interrupt. The data timeout duration is configured by CR3_DTOCNT, with the calculation formula: Timeout duration = FCLK period*4096*2^{CR3_DTOCNT}, where CR3_DTOCNT is less than 15. For example, when FCLK is48 MHz and CR3_DTOCNT is13, the timeout duration is approximately 699 ms. When a transmission error occurs, a normal completion interrupt ISR_TC may not be generated; therefore, both normal completion and error interrupts must be monitored simultaneously.

Recommended software procedure for data transmission:

1. Configure the transfer direction, transfer line width, timeout duration, automatic command mode, single block data size, and total block count.
2. Configure the built-in DMA (SDMA or ADMA).
3. Send read command (such as CMD17)or write command (such as CMD24)
4. Wait for command interrupt and process device response (such as R1)
5. Wait for data transfer completion interrupt or error interrupt

14.1.3.5 Interrupt Generation

SDMMC can generate interrupts to notify the CPU of specific events. These events include command transmission completion, data completion, and various errors. The occurred events can be queried via the SR register. Note that only

events enabled by the ISER can be recorded, and whether the recorded events generate interrupts is configured via the IER register. Therefore, when enabling interrupts, the corresponding events' ISER and IER must both be set to 1 .

14.1.3.6 FIFO Management

The SDMMC integrates a 1KB FIFO for buffering read and write data. The FIFO works in conjunction with the built-in DMA to complete data transfer. When writing data to the device, the controller initiates a single data block write operation only when the amount of data stored in the FIFO reaches or exceeds the size of one data block; otherwise, it waits for the CPU or DMA to transfer data into the FIFO. During the waiting period, the interface CLK will not pause output. During the process of writing multiple data blocks, each data block write operation follows the aforementioned mechanism.

When reading data from the slave device, the controller initiates a single data block read operation only when the remaining space in the FIFO is greater than or equal to the size of one data block; otherwise, it waits for the CPU or DMA to transfer data out of the FIFO. During the waiting period, the interface CLK halts output to prevent the device from transmitting data. When reading multiple data blocks, each block read adheres to the aforementioned mechanism.

14.1.3.7 DMA Transfer

During data transfer, the contents of the FIFO can be moved directly by the CPU via the FIFO input, or automatically transferred through the built-in DMA.

The built-in DMA is divided into two modes: SDMA and ADMA. SDMA configuration is straightforward and supports basic DMA functions. ADMA supports chained transfers with enhanced functionality and requires the creation of a chained descriptor table prior to use.

DMA functionality is enabled via the CR1_DMAEN register. The DMA Mode is selected via CR2_DMAMODE, 0 indicates SDMA, and 2 indicates ADMA.

In SDMA Mode, the start address for data transfer SAR must be configured, typically pointing to memory such as SRAM or PSRAM. After data transfer initiation, SDMA automatically transfers data to be written from memory into the FIFO, or transfers read data from the FIFO to memory. During multi-block transfers, SDMA automatically transfers multiple blocks of consecutive address data and automatically pauses at specific memory address boundaries.

To facilitate fle system management, SDMA Mode pauses data transfer when crossing specific memory address boundaries. The memory address boundary is configured by BSR_SDMABDY, selectable from 4KB to 512KB. When the memory address crosses a specific address boundary, SDMA automatically pauses, setting the SR_DMA flag (when ISER_DMAEN is 1), and can generate an interrupt (when IER_DMAIE is 1). When paused, the memory address of the next data to be transferred can be read from the NSAR register (at this time, the SAR register still holds the previously configured value). After the CPU detects the pause, it must write the memory address of the subsequent data to be transferred into the SAR register; SDMA will then automatically resume the previous transfer from the newly configured memory address. For example, if BSR_SDMABDY is configured to 0 (4KB boundary), and the transfer starts from 0x20019c00 with 2KB of data, then after the 4 bytes of data at 0x20019ffc are transferred, SDMA will automatically pause. At this point, the value of the NSAR register is 0x2001a000. If a new address 0x2001b000 is written to the SAR register at this time, SDMA will automatically resume transferring the remaining 1KB of data starting from 0x2001b0

ADMA mode requires the creation of a linked descriptor table prior to use. The starting address of the linked descriptor table must be 4-byte aligned and written to the ASAR register.

Each entry in the chained descriptor table occupies 8 bytes of space, and the starting address shall be aligned to 4 bytes. Entries are classifed into transfer entries and jump entries. The next entry of a transfer entry must be stored sequentially;

that is, the starting address of the next entry equals the starting address of the current transfer entry plus 8 bytes. The entry pointed to by a jump entry can be stored at any 4 -byte aligned address, regardless of the storage location of the jump entry itself. The entry format is as follows.

Chained Descriptor Table Entry Format								
	bit field							
baseaddr	31~16	15~6	5	4	3	2	1	0
	LENGTH	reserved(0)	ACT2	ACT1	0	INT	END	VALID
baseaddr+4	63~32							
	ADDR							

The meanings of the entry bit fields are as follows:

VALID	A value of 1 indicates the current entry is valid; a value of 0 will generate the error flag ISR_ADMAERR and may trigger an Interrupt.
END	A value of 1 indicates that the current table entry is the fnal entry, with no subsequent entries. Completion of the fnal table entry execution generates the flag ISR_TC and may trigger an interrupt.
ACT1/ACT2	Used to identify the type of table entry. set to 0indicates that the current table entry does not require execution and proceeds directly to the next entry. set to 1andACT1set to 0indicates that the current table entry is a transfer entry, and ADMA performs data transfer. ADDR indicates the starting address of the data. ACT2 set to 1 and ACT1 set to 1 indicates that the current table entry is a jump entry, where ADMA does not perform data transfer, and ADDR indicates the storage address of the next table entry.
LENGTH	When the current table entry is a transfer entry, this indicates the number of bytes to transfer, where 0 represents 65536 bytes. The transfer byte count must be an integer multiple of the single block data byte count.
ADDR	When the current entry is a transfer descriptor, it indicates the starting address of the data. When the current entry is a jump descriptor, it indicates the storage address of the next entry

An example of an ADMA linked descriptor table is shown in the fgure below. Entries ABCD are sequentially stored in the address space starting at address A, while entries EF are sequentially stored in the address space starting at address E. Entries ABCD are executed sequentially; upon reaching entry D, execution jumps to entry E; Entries EF are executed sequentially; upon completion of entry F, the ADMA transfer terminates.

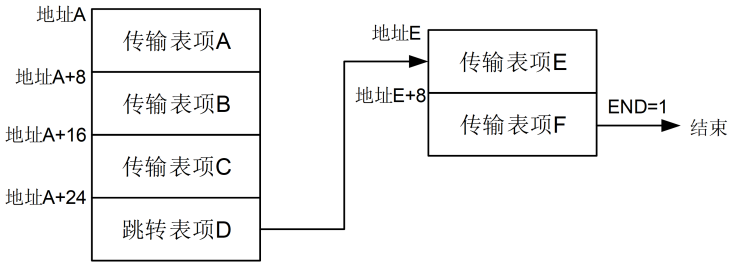


Figure 14-2: ADMA Example of an ADMAlinked descriptor table

14.1.3.8 eMMC Open-Drain Mode

When the eMMC device is in certain states (such as inactive, idle, identifcation states; refer to the eMMC protocol for details), the CMD line must be configured as Open-drain mode is implemented by configuring CR2_PPMODE to 0.

When accessing SD/SDIOdevices, and when eMMC enters other states, the CMD line must be configured as push-pull mode by setting CR2_PPMODE to 1.

14.1.3.9 SDIO Interrupt

The SDIO device can generate an SDIO interrupt to notify the controller by pulling DAT1 low. If the controller needs to respond to the SDIO interrupt, it must enable ISER_CARDEN and IER_CARDIE. Upon detecting the interrupt, ISR_CARD will be set to 1.

14.1.3.10 SD Card Detection

The SD card supports hot swapping. Card insertion and removal detection is typically implemented using the following 3 methods.

1. Detection via dedicated CD (card detect) pin. SD card slots that support insertion detection usually provide a dedicated card detect pin (CD) with a pull-up resistor to the power supply. The chip must allocate an independent IO to monitor this pin level. When no SD card is inserted, the pin level is high. When the SD card is inserted into the slot, this pin is connected to ground, resulting in a low level. Insertion and removal detection of the SD card can be achieved by monitoring the GPIO input level of this IO.

2.Detection via the DAT3 pin. When the SD card is powered on, a 50K ohm pull-up resistor exists internally from the DAT3 pin to the power supply. The chip's external circuitry requires a weak pull-down resistor to ground on the DAT3 pin (typically greater than 470K ohms). The chip determines card insertion by detecting the level of the IO connected to DAT3. When no SD card is inserted, this pin level is low. When the SD card is inserted into the slot, this pin is pulled up to a high level. By determining the GPIO input level of this IO, SD card insertion and removal detection can be realized. When the SD card initiates data transmission, this detection must be stopped, and DAT3 should be used as a normal data line.

3.Detection via command polling. The chip periodically initiates command interactions with the SDcard and determines the card's presence based on whether a response is received.

14.1.3.11 Sampling Clock Adjustment

When SDMMC communicates with the device, the CMD and DAT signals from the slave device must be sampled at the edge of the CLK. Due to signal transmission delay, it may be necessary to adjust the sampling clock edge to ensure accurate sampling.

The SCR_DLY register adjusts the sampling clock edge, with a configurable range of 0~40. A larger configured value results in greater delay.

14.1.4 SDMMC Registers

Table 14-1: SDMMC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**SAR**	System Address/Argument2 Register

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	ADDR	This register contains the physical system memory address used for DMA transfers or the second argument for the Auto CMD23. (1) SDMA System Address This register contains the system memory address for a SDMA transfer. The SDMA transfer waits at the every boundary specified by BSR_SDMABDY. After SDMA has stopped, the next system address of the next contiguous data position cannot be read from this register but from NSAR. The Host Controller generates ISR_DMA Interrupt to request the Host Driver to update this register. The Host Driver sets the next system address of the next data position to this register. When the most upper byte of this register is written, the Host Controller restarts the SDMA transfer. (2) Argument 2 This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without AMDA, the available block count value is limited by BSR_BCNT. 65535 blocks is the maximum value in this case.
0x04			**BSR**	Block Size Register
[31:16]	rw	16'h0	BCNT	Blocks Count For Current Transfer. This register is enabled when MULTIBLK is set to 1 and is valid only for multiple block transfers. Controller decrements the block count after each block transfer. Setting the block count to 0 results in no data blocks is transferred. This register should be accessed only when no transaction is executing (i.e., after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored. When a suspend command is completed, the number of blocks yet to be transferred can be determined by reading this register. Before issuing a resume command, the Host Driver shall restore the previously saved block count. FFFFh 65535 blocks ... ... 0002h 2 blocks 0001h 1 block 0000h Stop Count
[15]			RSVD	

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14:12]	r	3'h0	SDMABDY	Host SDMA Buffer Boundary. The large contiguous memory space may not be available in the virtual memory system. To perform long SDMA transfer, SDMA System Address register shall be updated at every system memory boundary during SDMA transfer. These bits specify the size of contiguous buffer in the system memory. The SDMA transfer shall wait at the every boundary specified by these fields and the Controller generates the ISR_DMA Interrupt to request the Host Driver to update the SDMA System Address register. At the end of transfer, the Controller may issue or may not issue ISR_DMA Interrupt. In particular, ISR_DMA Interrupt shall not be issued after ISR_TC Interrupt is issued. In case of this register is set to 0 (buffer size = 4K bytes), lower 12-bit of byte address points data in the contiguous buffer and the upper 20-bit points the location of the buffer in the system memory. The SDMA transfer stops when the Controller detects carry out of the address from bit 11 to 12. ADMA does not use this register. 000b 4K bytes (Detects A11 carry out) 001b 8K bytes (Detects A12 carry out) 010b 16K Bytes (Detects A13 carry out) 011b 32K Bytes (Detects A14 carry out) 100b 64K bytes (Detects A15 carry out) 101b 128K Bytes (Detects A16 carry out) 110b 256K Bytes (Detects A17 carry out) 111b 512K Bytes (Detects A18 carry out)
[11:0]	rw	12'h0	BSIZE	Transfer Block Size This register specifies the block size of data transfers such as CMD17, CMD18, CMD24, CMD25, and CMD53. Usually set to 512 for SD and eMMC access.
0x08			**ARG1**	Argument 1
[31:0]	rw	32'h0	ARG	Command Argument 1 The SD command argument is specified as bit39-8 of Command-Format in the Physical Layer Specification.
0x0C			**CR1**	Control and Command Register 1
[31:30]			RSVD	
[29:24]	rw	6'h0	CMDIDX	Command Index These bits shall be set to the command number (CMD0-63, ACMD0-63) that is specified in bits 45-40 of the Command-Format in the Physical Layer Specification and SDIO Card Specification.

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:22]	rw	2'h0	CMDTYPE	Command Type There are three types of special commands: Suspend, Resume and Abort. These bits shall be set to 00b for all other commands. (1) Suspend Command If the Suspend command succeeds, the Controller shall assume the SD Bus has been released and that it is possible to issue the next command, which uses the DAT line. The Controller shall de-assert Read Wait for read transactions and stop checking busy for write transactions. The interrupt cycle shall start, in 4-bit mode. If the Suspend command fails, the Controller shall maintain its current state, and the Host Driver shall restart the transfer by setting CR2_CONTREQ. (2) Resume Command The Host Driver re-starts the data transfer. The Controller shall check for busy before starting write transfers. (3) Abort Command If this command is set when executing a read transfer, the Controller shall stop reads to the buffer. If this command is set when executing a write transfer, the Controller shall stop driving the DAT line. After issuing the Abort command, the Host Driver should issue a software reset. 11b Abort CMD12, CMD52 for writing "I/O Abort" in CCCR 10b Resume CMD52 for writing "Function Select" in CCCR 01b Suspend CMD52 for writing "Bus Suspend" in CCCR 00b Normal Other commands
[21]	rw	1'h0	DATAPRESENT	Data Present Select This bit is set to 1 to indicate that data is present and shall be transferred using the DAT line. It is set to 0 for the following: (1) Commands using only CMD line (ex. CMD52). (2) Commands with no data transfer but using busy signal on DAT[0] line (R1b or R5b ex. CMD38) (3) Resume command
[20]	rw	1'h0	CHECKIDX	Command Index Check Enable If this bit is set to 1, the Controller shall check the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this bit is set to 0, the Index field is not checked.
[19]	rw	1'h0	CHECKCRC	Command CRC Check Enable If this bit is set to 1, the Controller shall check the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this bit is set to 0, the CRC field is not checked. The position of CRC field is determined according to the length of the response.
[18]			RSVD	
[17:16]	rw	2'h0	RSPTYPE	Response Type Select 00 No Response 01 Response Length 136 10 Response Length 48 11 Response Length 48 check Busy after
[15:12]			RSVD	
[11]	rw	1'h0	STREAMMODE	Stream Mode Enable The Host driver has to set this bit for MMC CMD11 / CMD20 Stream Read/Write Operations.
[10:6]			RSVD	
[5]	rw	1'h0	MULTIBLK	Multi / Single Block Select This bit is set when issuing multiple-block transfer commands using DAT line. For any other commands, this bit shall be set to 0. If this bit is 0, it is not necessary to set BSR_BCNT

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h0	DIR	Data Transfer Direction Select This bit defines the direction of DAT line data transfers. The bit is set to 1 to read data from the SD card to the Controller and it is set to 0 for all other commands. 1 Read (Card to Host) 0 Write (Host to Card)
[3:2]	rw	2'h0	AUTOCMD	Auto CMD Enable This field determines use of auto command functions. 00b Auto Command Disabled 01b Auto CMD12 Enabled 10b Auto CMD23 Enabled 11b Reserved There are two methods to stop Multiple-block read and write operation. (1) Auto CMD12 Enable When this field is set to 01b, the Controller issues CMD12 automatically when last block transfer is completed. Auto CMD12 error is indicated to CR4. The Host Driver shall not set this bit if the command does not require CMD12. (2) Auto CMD23 Enable When this bit field is set to 10b, the Controller issues a CMD23 automatically before issuing a command specified. The following conditions are required to use the Auto CMD23. A memory card that supports CMD23 (SCR[33]=1) If DMA is used, it shall be ADMA. Only when CMD18 or CMD25 is issued (Note, the Controller does not check command index.) Auto CMD23 can be used with or without ADMA. The Controller issues a CMD23 first and then issues a command specified by CR1. If response errors of CMD23 are detected, the second command is not issued. A CMD23 error is indicated in CR4. 32-bit block count value for CMD23 is set to SAR.
[1]	rw	1'h0	BCNTEN	Block Count Enable This bit is used to enable the Block Count register, which is only relevant for multiple block transfers. When this bit is 0, the Block Count register is disabled, which is useful in executing an infinite transfer. If ADMA data transfer is more than 65535 blocks, this bit shall be set to 0. In this case, data transfer length is designated by Descriptor Table.
[0]	rw	1'h0	DMAEN	DMA Enable This bit enables DMA functionality. SDMA or ADMA mode can be selected by CR2_DMAMODE.
0x10			**RSP1**	Command Response 31 0

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	r	32'h0	RSP	Command Response [31:0] The Response Field indicates bit positions of "Responses" defined in the Physical Layer Spec. Most responses with a length of 48 (R[47:0]) have 32 bits of the response data (R[39:8]) stored in RSP1. Responses of type R1b (Auto CMD12 responses) and R1 (Auto CMD23 response) have response data bits R[39:8] stored in RSP4. Responses with length 136 (R[135:0]) have 120 bits of the response data (R[127:8]) stored in RSP1~RSP4. To be able to read the response status efficiently, the Controller only stores part of the response data. This enables the Host Driver to read 32 bits of response data efficiently in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by the Controller (as specified by CR1_CHECKIDX and the CR1_CHECKCRC) and generate an error interrupt if an error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, the Controller shall check R[47:1], and if the response length is 136 the Controller shall check R[119:1]. Since the Controller may have a multiple block data DAT line transfer executing concurrently with a CMD_wo_DAT command, the Controller stores the Auto CMD12 response in RSP4. The CMD_wo_DAT response is stored in RSP1. This allows the Controller to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. While executing Auto CMD23, the response of CMD23 is saved to RSP4 and the response of multiple-block read and write command is save to RSP1. The response error of CMD23 is indicated in CR4.
0x14			**RSP2**	Command Response 63 32
[31:0]	r	32'h0	RSP	Command Response [63:32]
0x18			**RSP3**	Command Response 95 64
[31:0]	r	32'h0	RSP	Command Response [95:64]
0x1C			**RSP4**	Command Response 127 96
[31:0]	r	32'h0	RSP	Command Response [127:96]
0x20			**BUF**	Buffer Data Port Register
[31:0]	rw	32'h0	DATA	Buffer Data The Controller buffer can be accessed through this 32-bit Data Port register.
0x24			**SR**	Present State Register
[31:25]			RSVD	
[24]	r	1'h0	CMDLVL	CMD Line Signal Level This status is used to check the CMD line level to recover from errors, and for debugging.
[23:20]	r	4'h0	DATLVL	DAT-DAT3 Line Signal Level This status is used to check the DAT line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DAT [0].
[19:12]			RSVD	
[11]	r	1'h0	BUFREN	Buffer Read Enable This status is used for non-DMA read transfers. The Controller may implement multiple buffers to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this bit is 1, readable data exists in the buffer. A change of this bit from 1 to 0 occurs when all the block data is read from the buffer. A change of this bit from 0 to 1 occurs when block data is ready in the buffer and generates ISR_BUFRRDY.

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	r	1'h0	BUFWEN	Buffer Write Enable This status is used for non-DMA write transfers. The Controller can implement multiple buffers to transfer data efficiently. This read only flag indicates if space is available for write data. If this bit is 1, data can be written to the buffer. A change of this bit from 1 to 0 occurs when all the block data is written to the buffer. A change of this bit from 0 to 1 occurs when top of block data can be written to the buffer and generates ISR_BUFWRDY. The Controller should neither set Buffer Write Enable nor generate ISR_BUFWRDY after the last block data is written to the Buffer Data Port Register.
[9]	r	1'h0	RACTIVE	Read Transfer Active This status is used for detecting completion of a read transfer. This bit is set to 1 for either of the following conditions: (1) After the end bit of the readcommand. (2) When read operation is restarted by writing 1 to CR2_CONTREQ. This bit is cleared to 0 for either of the following conditions: (1) When the last data block as specified by block length is transferred to the System. (2) In case of ADMA, end of read operation is designated by Descriptor Table. (3) When all valid data blocks in the Controller have been transferred to the System and no current block transfers are being sent as a result of CR2_BLKGAPSTOP being set to 1. A Transfer Complete interrupt is generated when this bit changes to 0.
[8]	r	1'h0	WACTIVE	Write Transfer Active This status indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the Controller. This bit is set in either of the following cases: After the end bit of the write command. When write operation is restarted by writing 1 to CR2_CONTREQ. This bit is cleared in either of the following cases: After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) In case of ADMA, transfer count is designated by Descriptor Table. After getting the CRC status of any block where data transmission is about to be stopped by CR2_BLKGAPSTOP. During a write transaction, ISR_BLKGAP is generated when this bit is changed to 0, as the result of CR2_BLKGAPSTOP begin set. This status is useful for the Host Driver in determining non DAT line commands can be issued during write busy.
[7:3]			RSVD	

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	r	1'h0	DATACTIVE	DAT Line Active This bit indicates whether one of the DAT line on SD Bus is in use. (a) In the case of read transactions This status indicates whether a read transfer is executing on the SD Bus. Changing this value from 1 to 0 generates ISR_BLKGAP, as the result of CR2_BLKGAPSTOP being set. This bit shall be set in either of the following cases: (1) After the end bit of the read command. (2) When writing 1 to CR2_CONTREQ to restart a read transfer. This bit shall be cleared in either of the following cases (1) When the end bit of the last data block is sent from the SD Bus to the Controller. In case of ADMA, the last block is designated by the last transfer of Descriptor Table. (2) When a read transfer is stopped at the block gap initiated by CR2_BLKGAPSTOP. The Controller shall stop read operation at the start of the interrupt cycle of the next block gap by driving Read Wait or stopping SD clock. If the Read Wait signal is already driven (due to data buffer cannot receive data), the Controller can continue to stop read operation by driving the Read Wait signal. It is necessary to support Read Wait in order to use suspend / resume function. (b) In the case of write transactions This status indicates that a write transfer is executing on the SD Bus. Changing this value from 1 to 0 generate ISR_TC. This bit shall be set in either of the following cases: (1) After the end bit of the write command. (2) When writing 1 to CR2_CONTREQ to continue a write transfer. This bit shall be cleared in either of the following cases: (1) When the SD card releases write busy of the last data block. If SD card does not drive busy signal for 8 SD Clocks, the Controller shall consider the card drive "Not Busy". In case of ADMA, the last block is designated by the last transfer of Descriptor Table. (2) When the SD card releases write busy prior to waiting for write transfer as a result of CR2_BLKGAPSTOP. (c) Command with busy This status indicates whether a command indicates busy (ex. erase command for memory) is executing on the SD Bus. This bit is set after the end bit of the command with busy and cleared when busy is de-asserted. Changing this bit from 1 to 0 generate ISR_TC.
[1]	r	1'h0	DATBUSY	Command Inhibit (DAT) This status bit is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this bit is 0, it indicates the Controller can issue the next SD Command. Commands with busy signal belong to Command Inhibit (DAT) (ex. R1b, R5b type). Changing from 1 to 0 generates ISR_TC interrupt. 1 Cannot issue command which uses the DAT line 0 Can issue command which uses the DAT line

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	r	1'h0	CMDBUSY	Command Inhibit (CMD) If this bit is 0, it indicates the CMD line is not in use and the Controller can issue a SD Command using the CMD line. This bit is set immediately after CR1 is written. This bit is cleared when the command response is received. Auto CMD12 and Auto CMD23 consist of two responses. In this case, this bit is not cleared by the response of CMD12 or CMD23 but cleared by the response of a read/write command. Status issuing Auto CMD12 is not read from this bit. So if a command is issued during Auto CMD12 operation, Controller shall manage to issue two commands: CMD12 and a command set by CR1. Even if the DATBUSY is set to 1, commands using only the CMD line can be issued if this bit is 0. Changing from 1 to 0 generates ISR_CC Interrupt. If the Controller cannot issue the command because of a command conflict error or because of Command Not Issued By Auto CMD12 Error, this bit shall remain 1 and the Command Complete is not set. 1 Cannot issue command 0 Can issue command using only CMD line
0x28			**CR2**	Control Register 2
[31:20]			RSVD	
[19]	rw	1'h0	BLKGAPIE	Interrupt At Block Gap This bit is valid only in 4-bit mode of the SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SD card cannot signal an interrupt during a multiple block transfer, this bit should be set to 0. The Host Driver shall set this bit according to the CCCR of the SDIO card.
[18]	rw	1'h0	RWAITEN	Read Wait Control The read wait function is optional for SDIO cards. If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using the DAT2 line. Otherwise, the Controller has to stop the SD Clock to hold read data, which restricts commands generation. The Host Driver shall set this bit according to the CCCR of the SDIO card. If the card does not support read wait, this bit shall never be set to 1 otherwise DAT line conflict may occur. If this bit is set to 0, Suspend/Resume cannot be supported.
[17]	rw	1'h0	CONTREQ	Continue Request
[16]	rw	1'h0	BLKGAPSTOP	Stop At Block Gap request This bit is used to stop executing read and write transaction at the next block gap for non-DMA, SDMA and ADMA transfers. The Host Driver shall leave this bit set to 1 until the Transfer Complete is set to 1. Clearing both Stop At Block Gap Request and Continue Request shall not cause the transaction to restart. The Controller shall stop read transfer by using Read Wait or stopping SD clock. In case of write transfers in which the Host Driver writes data to the Buffer Data Port register, the Host Driver shall set this bit after all block data is written. If this bit is set to 1, the Host Driver shall not write data to BUF. 1 Stop 0 Transfer
[15:13]			RSVD	
[12]	rw	1'h1	PPMODE	CMD line output mode. Open drain mode should only be used when accessing eMMC device. 1 Push Pull Mode 0 Open Drain Mode
[11:6]			RSVD	

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	rw	1'h0	EXTWIDTH	Extended Data Transfer Width 1 8-bit Bus Width 0 Bus Width is Selected by DATAWIDTH
[4:3]	rw	2'h0	DMAMODE	DMA Select One of supported DMA modes can be selected. DMA mode is enabled by CR1_DMAEN. 00b SDMA 01b Reserved 10b ADMA 11b Reserved
[2]	rw	1'h0	HSMODE	Reserved for debug. Adjust output timing of CMD line and DAT lines.
[1]	rw	1'h0	DATWIDTH	Data Transfer Width This bit selects the data width of the Controller when EXTWIDTH is 0. The Host Driver shall set it to match the data width of the SD card. 1 4-bit mode 0 1-bit mode
[0]			RSVD	
0x2C			**CR3**	Control Register 3
[31:27]			RSVD	
[26]	rw	1'h0	RSTDAT	Software Reset For DAT Line Only part of data circuit is reset. DMA circuit is also reset. Will be cleared automatically. Because it takes some time to complete software reset, the SD Host Driver shall confirm that these bits are 0.
[25]	rw	1'h0	RSTCMD	Software Reset For CMD Line Only part of command circuit is reset. Will be cleared automatically. Because it takes some time to complete software reset, the SD Host Driver shall confirm that these bits are 0.
[24]	rw	1'h0	RST	This reset affects the entire Controller except for the card detection circuit. Register bits (except read-only bits) are cleared to 0. During its initialization, the Host Driver shall set this bit to 1 to reset the Controller. If this bit is set to 1, the host driver should issue reset command and reinitialize the SD card. Will be cleared automatically. Because it takes some time to complete software reset, the SD Host Driver shall confirm that these bits are 0.
[23:20]			RSVD	
[19:16]	rw	4'he	DTOCNT	Data Timeout Counter Value This value determines the interval by which DAT line timeouts are detected. Timeout period is defined as: 1111b Reserved 1110b 4096*214 FCLK periods .... 0001b 4096*21 FCLK periods 0000b 4096*2 FCLK periods
[15:8]	rw	8'h0	CLKDIVL	CLK divider lower 8 bits 10-bit CLKDIV is CLKDIVU,CLKDIVL. If CLKDIV=0, CLK = FCLK. If CLKDIV!=0, CLK = FCLK/(2*CLKDIV). e.g. FCLK=192MHz, CLKDIV=240, then CLK frequency is 400KHz.
[7:6]	rw	2'h0	CLKDIVU	CLK divider upper 2 bits These bits expand clk divider to 10-bit.
[5:3]			RSVD	
[2]	rw	1'h0	CLKEN	SD Clock Enable 0: SD clock disabled 1: SD clock enabled

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]			RSVD	
[0]	rw	1'h0	INTCLKEN	Internal Clock Enable 0: Internal clock disabled 1: Internal clock enabled
0x30			**ISR**	Interrupt State Register
[31:26]			RSVD	
[25]	rw1c	1'h0	ADMAERR	ADMA Error This bit is set when the Controller detects errors during ADMA based data transfer. The state of the ADMA at an error occurrence is saved in the AESR, In addition, the Controller generates this Interrupt when it detects invalid descriptor data (Valid=0). ADMA Error State in the ADMA Error Status indicates that an error occurs. The Host Driver may find that Valid bit is not set at the error descriptor.
[24]	rw1c	1'h0	ACERR	Auto CMD Error Auto CMD12 and Auto CMD23 use this error status. In case of Auto CMD12, this bit is set to 1, not only when the errors in Auto CMD12 occur but also when Auto CMD12 is not executed due to the previous command error.
[23]			RSVD	
[22]	rw1c	1'h0	DEBERR	Data End Bit Error Occurs either when detecting 0 at the end bit position of read data which uses the DAT line or at the end bit position of the CRC Status.
[21]	rw1c	1'h0	DCRCERR	Data CRC Error Occurs when detecting CRC error when transferring read data which uses the DAT line or when detecting the Write CRC status having a value of other than "010".
[20]	rw1c	1'h0	DTOERR	Data Timeout Error This bit is set when detecting one of following timeout conditions. Busy timeout for R1b,R5b type Busy timeout after Write CRC status Write CRC Status timeout Read Data timeout.
[19]	rw1c	1'h0	IDXERR	Command Index Error This bit is set if a Command Index error occurs in the command response.
[18]	rw1c	1'h0	CEBERR	Command End Bit Error This bit is set when detecting that the end bit of a command response is 0.
[17]	rw1c	1'h0	CCRCERR	Command CRC Error Command CRC Error is generated in two cases. If a response is returned and the Command Timeout Error is set to 0 (indicating no timeout), this bit is set to 1 when detecting a CRC error in the command response. The Controller detects a CMD line conflict by monitoring the CMD line when a command is issued. If the Controller drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SD clock edge, then the Controller shall abort the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error shall also be set to 1 to distinguish CMD line conflict
[16]	rw1c	1'h0	CTOERR	Command Timeout Error This bit is set only if no response is returned within 64 SD clock cycles from the end bit of the command. If the Controller detects a CMD line conflict, in which case Command CRC Error shall also be set, this bit shall be set without waiting for 64 SD clock cycles because the command will be aborted by the Controller.

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15]	r	1'h0	ERR	Error Interrupt If any error bit (bit16~bit31) of ISR are set, then this bit is set. Therefore the Host Driver can efficiently test for an error by checking this bit first. This bit is read only.
[14:9]			RSVD	
[8]	r	1'h0	CARD	Card Interrupt Writing this bit to 1 does not clear this bit. It is cleared by resetting the SD card interrupt factor. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the SD card and the interrupt to the Host System. When this status has been set and the Host Driver needs to start this interrupt service, ISER_CARDEN may be set to 0 in order to clear the card interrupt statuses latched in the Controller and to stop driving the interrupt signal to the Host System. After completion of the card interrupt service (It should reset interrupt factors in the SD card and the interrupt signal may not be asserted), set ISER_CARDEN to 1 and start sampling the interrupt signal again.
[7:6]			RSVD	
[5]	rw1c	1'h0	BUFRRDY	Buffer Read Ready This status is set if the Buffer Read Enable changes from 0 to 1. Refer to SR_BUFREN register. 1 Ready to read buffer 0 Not ready to read buffer
[4]	rw1c	1'h0	BUFWRDY	Buffer Write Ready This status is set if the Buffer Write Enable changes from 0 to 1. Refer to SR_BUFWEN register. 1 Ready to write buffer 0 Not ready to write buffer
[3]	rw1c	1'h0	DMA	DMA Interrupt This status is set if the Controller detects the Host SDMA Buffer boundary during transfer. Refer to BSR_SDMABDY. In case of ADMA, by setting INT field in the descriptor table, Controller generates this interrupt. This interrupt shall not be generated after the Transfer Complete.
[2]	rw1c	1'h0	BLKGAP	Block Gap Event If CR2_BLKGAPSTOP is set, this bit is set when both a read / write transaction is stopped at a block gap. If CR2_BLKGAPSTOP is not set to 1, this bit is not set to 1. (1) In the case of a Read Transaction This bit is set at the falling edge of the DAT Line Active Status (When the transaction is stopped at SD Bus timing. The Read Wait shall be supported in order to use this function. (2) Case of Write Transaction This bit is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 1 Transaction stopped at block 0 No Block Gap Event

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw1c	1'h0	TC	Transfer Complete This bit is set when a read / write transfer and a command with busy is completed. (1) In the case of a Read Transaction This bit is set at the falling edge of Read Transfer Active Status. This interrupt is generated in two cases. The first is when a data transfer is completed as specified by data length (After the last data has been read to the Host System). The second is when data has stopped at the block gap and completed the data transfer by setting CR2_BLKGAPSTOP (After valid data has been read to the Host System). (2) In the case of a Write Transaction This bit is set at the falling edge of the DAT Line Active Status. This interrupt is generated in two cases. The first is when the last data is written to the SD card as specified by data length and the busy signal released. The second is when data transfers are stopped at the block gap by setting CR2_BLKGAPSTOP and data transfers completed. (After valid data is written to the SD card and the busy signal released). (3) In the case of a command with busy This bit is set when busy is de-asserted. Refer to SR_DATACTIVE and SR_CMDBUSY. Transfer Complete has higher priority than Data Timeout Error. If both bits are set to 1, execution of a command can be considered to be completed.
[0]	rw1c	1'h0	CC	Command Complete This bit is set when get the end bit of the command response. Auto CMD12 and Auto CMD23 consist of two responses. Command Complete is not generated by the response of CMD12 or CMD23 but generated by the response of a read/write command. Command Timeout Error has higher priority than Command Complete. If both bits are set to 1, it can be considered that the response was not received correctly.
0x34			**ISER**	Interrupt Status Enable Register
[31:26]			RSVD	
[25]	rw	1'h0	ADMAERREN	ADMA Error Status Enable
[24]	rw	1'h0	ACERREN	Auto CMD Error Status Enable
[23]			RSVD	
[22]	rw	1'h0	DEBERREN	Data End Bit Error Status Enable
[21]	rw	1'h0	DCRCERREN	Data CRC Error Status Enable
[20]	rw	1'h0	DTOERREN	Data Timeout Error Status Enable
[19]	rw	1'h0	IDXERREN	Command Index Error Status Enable
[18]	rw	1'h0	CEBERREN	Command End Bit Error Status Enable
[17]	rw	1'h0	CCRCERREN	Command CRC Error Status Enable
[16]	rw	1'h0	CTOERREN	Command Timeout Error Status Enable
[15:9]			RSVD	
[8]	rw	1'h0	CARDEN	Card Interrupt Status Enable If this bit is set to 0, the Controller shall clear interrupt request to the System. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The Host Driver may clear the Card Interrupt Status Enable before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.
[7:6]			RSVD	
[5]	rw	1'h0	BUFRRDYEN	Buffer Read Ready Status Enable
[4]	rw	1'h0	BUFWRDYEN	Buffer Write Ready Status Enable
[3]	rw	1'h0	DMAEN	DMA Interrupt Status Enable

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'h0	BLKGAPEN	Block Gap Event Status Enable
[1]	rw	1'h0	TCEN	Transfer Complete Status Enable
[0]	rw	1'h0	CCEN	Command Complete Status Enable
0x38			**IER**	Interrupt Enable Register
[31:26]			RSVD	
[25]	rw	1'h0	ADMAERRIE	ADMA Error Interrupt Enable
[24]	rw	1'h0	ACERRIE	Auto CMD Error Interrupt Enable
[23]			RSVD	
[22]	rw	1'h0	DEBERRIE	Data End Bit Error Interrupt Enable
[21]	rw	1'h0	DCRCERRIE	Data CRC Error Interrupt Enable
[20]	rw	1'h0	DTOERRIE	Data Timeout Error Interrupt Enable
[19]	rw	1'h0	IDXERRIE	Command Index Error Interrupt Enable
[18]	rw	1'h0	CEBERRIE	Command End Bit Error Interrupt Enable
[17]	rw	1'h0	CCRCERRIE	Command CRC Error Interrupt Enable
[16]	rw	1'h0	CTOERRIE	Command Timeout Error Interrupt Enable
[15:9]			RSVD	
[8]	rw	1'h0	CARDIE	Card Interrupt Interrupt Enable If this bit is set to 0, the Controller shall clear interrupt request to the System. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The Host Driver may clear the Card Interrupt Status Enable before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.
[7:6]			RSVD	
[5]	rw	1'h0	BUFRRDYIE	Buffer Read Ready Interrupt Enable
[4]	rw	1'h0	BUFWRDYIE	Buffer Write Ready Interrupt Enable
[3]	rw	1'h0	DMAIE	DMA Interrupt Interrupt Enable
[2]	rw	1'h0	BLKGAPIE	Block Gap Event Interrupt Enable
[1]	rw	1'h0	TCIE	Transfer Complete Interrupt Enable
[0]	rw	1'h0	CCIE	Command Complete Interrupt Enable
0x3C			**CR4**	Control Register 4
[31:19]			RSVD	
[18:16]	rw	3'h0	UHSMODE	UHS Mode Select 100b DDR others SDR
[15:8]			RSVD	
[7]	r	1'h0	CNIERR	Command Not Issued By Auto CMD12 Error Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 Error in this register. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 1 Not Issued 0 No Error
[6:5]			RSVD	
[4]	r	1'h0	ACIDXERR	Auto CMD Index Error This bit is set if the Command Index error occurs in response to a command
[3]	r	1'h0	ACEBERR	Auto CMD End Bit Error This bit is set when detecting that the end bit of command response is 0.
[2]	r	1'h0	ACCRCERR	Auto CMD CRC Error This bit is set when detecting a CRC error in the command response.
[1]	r	1'h0	ACTOERR	Auto CMD Timeout Error This bit is set if no response is returned within 64 SDCLK cycles from the end bit of command. If this bit is set to1, the other error status bits are meaningless.

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	r	1'h0	ACNE	Auto CMD12 Not Executed If memory multiple block data transfer is not started due to command error, this bit is not set because it is not necessary to issue Auto CMD12. Setting this bit to 1 means the Controller cannot issue Auto CMD12 to stop memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits are meaningless. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 1 Not Executed 0 Executed
0x54			**AESR**	ADMA Error Status Register
[31:3]			RSVD	
[2]	r	1'h0	LENERR	ADMA Length Mismatch Error This error occurs in the following 2 cases. (1) While CR1_BCNTEN being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length. (2) Total data length cannot be divided by the block length.
[1:0]	r	2'h0	ERRSTATE	ADMA Error State This field indicates the state of ADMA when error is occurred during ADMA data transfer. This field never indicates "10" because ADMA never stops in this state. 00 ST_STOP (Stop DMA) Points next of the error descriptor 01 ST_FDS (Fetch Descriptor) Points the error descriptor 10 Never set this state (Not used) 11 ST_TFR (Transfer Data) Points the next of the error descriptor
0x58			**ASAR**	ADMA System Address Register
[31:0]	rw	32'h0	ADDR	ADMA System Address This register holds byte address of executing command of the Descriptor table. At the start of ADMA, the Host Driver shall set start address of the Descriptor table. The ADMA increments this register address, which points to next line, when every fetching a Descriptor line. When the ADMA Error Interrupt is generated, this register shall hold valid Descriptor address depending on the ADMA state. The Host Driver shall program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. Lower 2-bit of this register is ignored and assumes it to be 00b.
0x5C			**NSAR**	Next System Address Register
[31:0]	r	32'h0	ADDR	This register contains next physical system memory address used for SDMA transfer. When the Host Controller stops a SDMA transfer, this register shall point to the system address of the next contiguous data position. It can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value. After SDMA has stopped, the next system address of the next contiguous data position can be read from this register.
0xEC			**ABSR**	AHB Burst Size Register
[31:7]			RSVD	

Continued on next page...

Table 14-1: SDMMC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6:0]	rw	7'h7	BSIZE	AHB Master Burst Size Register AHB Master performs Burst operations as per this register. 0:Disabled 1:Enabled Bit 00 INCR4 Bit 01 INCR8 Bit 02 INCR16 Bit 03 INCR Bit 04 WRAP4 Bit 05 WRAP8 Bit 06 WRAP16
0xF4			**SCR**	Sampling Clock Register
[31:8]			RSVD	
[7]	rw	1'h0	SEL	select sampling clock source (delay line input) 0: use internal clock 1: use loopback clock
[6]			RSVD	
[5:0]	rw	6'h0	DLY	Stages of delay line for sampling clock

14.2 QSPI Interface

The chip contains a total of 4 QSPI modules, of which QSPI1, QSPI2, and QSPI3 are located in HPSYS and can send requests to DMAC1; QSPI4 is located in LPSYS and can send requests to DMAC2.

The input/output of QSPI1 is connected to IO(SIP) , used to access the single or dual NOR Flash within the chip's integrated package (SiP) .

The inputs and outputs of QSPI2 and QSPI3 are connected to IO(PA)for accessing external NOR/NAND Flash chips.

The inputs and outputs of QSPI4 are connected to IO(PB) for accessing the chip-integrated 4-wire pSRAM within the SiP package or external NOR/NAND Flash chips.

The QSPI controller is a dedicated off-chip memory communication interface, suitable for single-line, dual-line, and quad-line SPI NOR/NAND Flash and pSRAM memory devices. The CPU can flexibly perform various operations via registers or AHB address mapping. The controller also supports Dual Flash mode to further improve bandwidth and capacity.

The QSPI controller supports two operation modes: (1) Register Mode and (2) Address Mapping Mode. Switching between these modes is automatically managed by hardware, enabling dynamic interleaved execution. In either mode, highly customizable SPI command timing is supported to ensure compatibility with various memory devices.

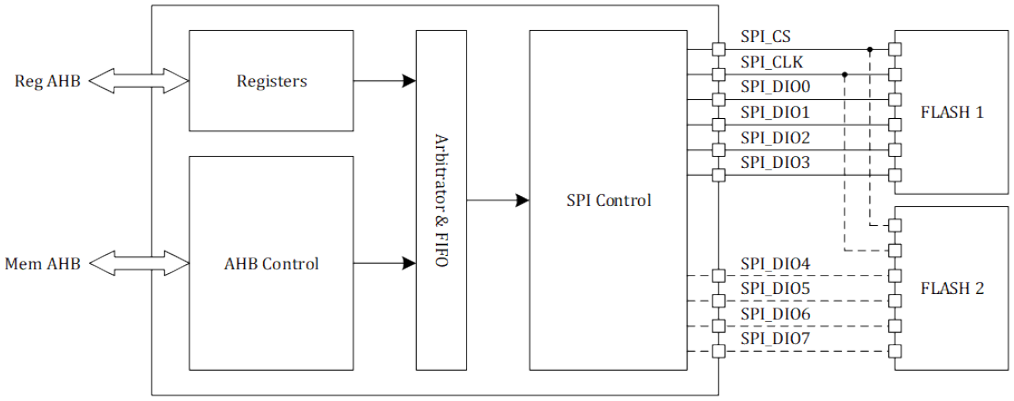


Figure 14-3: QSPI Controller Block Diagram

Register Mode:

- Transmit an SPI command timing sequence via register operations. The command can also be configured as a status polling command that is repeatedly transmitted until the read data satisfes a predefined status.
- Supports transmitting a sequence containing two SPI command timings, where the second command can be configured as a status polling command that is repeatedly transmitted until the read data satisfes a predefined status.
- Supports DMAchannels to perform FIFO data transfers

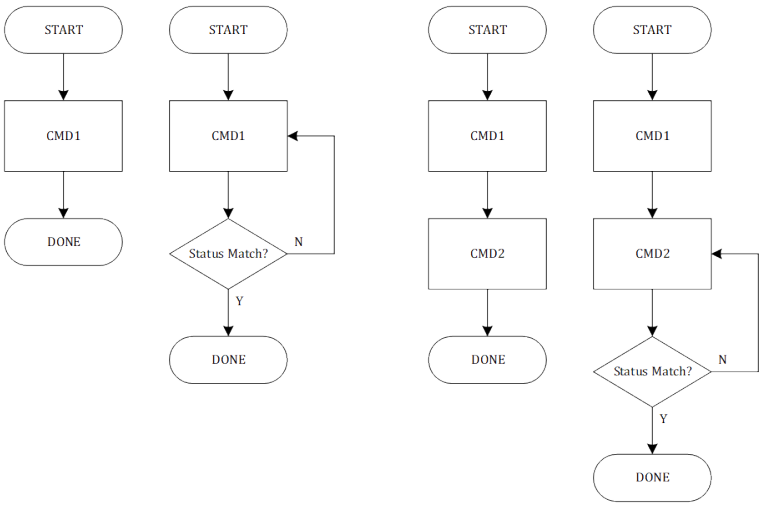


Figure 14-4: Sequence timing for single and multiple commands in register mode

Address mapping mode:

- Maps external memoryinto the AHBaddress space and translates bus read/write operations into predefined SPI-command timings
- Supports Byte (8-bit), Half-word (16-bit), and Word (32-bit) AHB accesses
- Effciently converts AHB Wrap operations to satisfy XIP real-time requirements
- Supports XIP real-time (On-The-Fly) decryption, with modes AES128-CTR or AES256-CTR
- Supports Resume function; after a single access, the CS can be selectively pulled low, and if the subsequent read address is continuous with the previous one, reading commences immediately Data, omitting command and address fields

14.3 OPI-PSRAM Interface

OPI-PSRAM is located in HPSYS . The input/output of OPI-PSRAM is connected to IO(SIP) for accessing the chip-integrated (SiP) 8-line pSRAM .

The OPI-PSRAM interface enables access to external PSRAM via address mapping, supporting both CPU and DMA access. The main features are as follows:

- Supports OPI DDR PSRAM with a maximum clock frequency of 120 MHz
- Maximum addressing range of 256 Mb
- Supports 8-bit or 16-bit bus bandwidth
- Built-in transmit and receive FIFO
- Supports access widths of 8/16/32 bits
- Supports DMA access
- Timeout mechanism

15 Bluetooth

15.1 Bluetooth Overview

Supports BLE 5.2 protocol, with the main features as follows:

- Supports rates (1M/2M/Coded PHY);
- Supports all packet formats (broadcast packet/exttended broadcast packet/data packet, etc.)
- Supports data encryption and decryption
- Supports data stream processing (CRC and whitening)
- Supports two frequency hopping modes
- Supports low-power mode
- Supports maximum 10 dBmtransmit power
- Supports up to 4 BLE links

15.2 Bluetooth coexistence interface

15.2.1 Overview

The Bluetooth coexistence interface enables coexistence between Bluetooth and WIFI. The coexistence interface can output Bluetooth transmission status, reception status, and Bluetooth event status via IO(PB) , which can be used to directly control the antenna switch or provide arbitration information required by external WIFI chips.

The coexistence interface includes 3 output signals: BT_ACTIVE, TX_ACTIVE, and RX_ACTIVE, supporting both external and internal arbitration schemes. The output signals can be directly connected to the chip IO(PB) or can trigger PTC events.

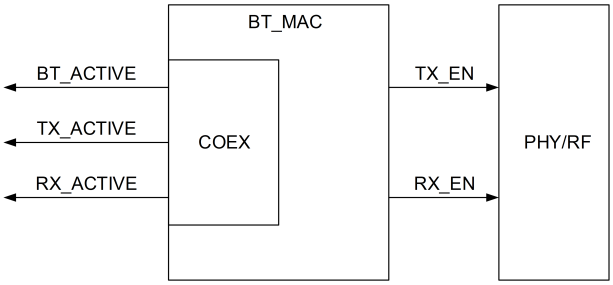


Figure 15-1: Bluetooth Coexistence Diagram

15.2.2 Output Signal BT_ACTIVE

BT_ACTIVE indicates a complete Bluetooth event unit, such as the polling of 3 channels during a BLE broadcast or an interaction during a BLE connection event. It is active high and output via PB46.

Configuration procedure:

1. Set DBGR.SEL_H of LPSYS_CFG to 3, and set bit 7 of DBGR.BITEN_H to 1.
2. Set address 0x41020050 (located within BLE_MAC) to 0x8000.
3. Set the function selection of PB46 to function 9

15.2.3 Output Signal TX_ACTIVE

Indicates that Bluetooth is in the transmit state, active high, output via PB19.

Configuration procedure:

1. Set DBGR.SEL_L of LPSYS_CFG to 7, and set bit 7 of DBGR.BITEN_L to 1.
2. Set the function selection of PB19 to function 9.

15.2.4 Output Signal RX_ACTIVE

Indicates that Bluetooth is in the receive state, active high, output via PB16.

Configuration procedure:

1. Set DBGR.SEL_L of LPSYS_CFG to 7, and set bit 6 of DBGR.BITEN_L to 1.
2. Set the function selection of PB16 to function 9.

15.2.5 Example of a 2-wire external arbitration scheme

In this example, Bluetooth and WIFI share a single antenna, switching via an RF switch controlled by the WIFI chip. Arbitration is performed by the WIFI chip, which determines whether to grant antenna usage rights to Bluetooth based on the TX_ACTIVE and RX_ACTIVE signals output by Bluetooth.

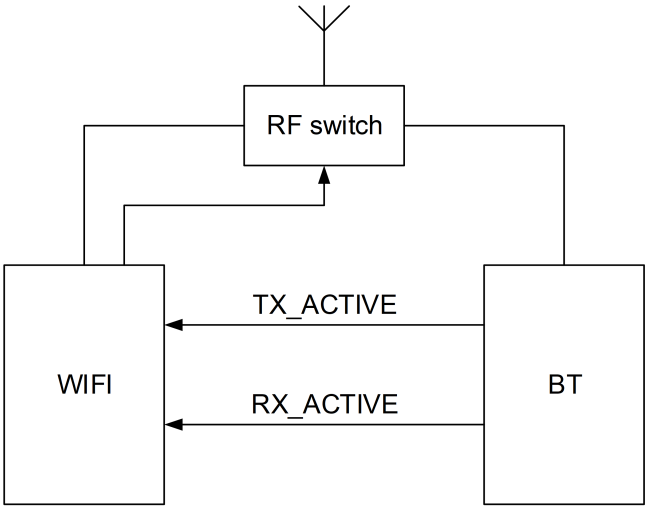


Figure 15-2: Example of a 2-wire external arbitration scheme

15.2.6 Example of a single-wire internal arbitration scheme

In this example, Bluetooth and WIFI share a single antenna, switched via an RF switch, with control signals originating from Bluetooth. Arbitration is performed by Bluetooth, controlling antenna usage rights with the BT_ACTIVE signal and

notifying the WIFI chip. In this mode, Bluetooth priority is fixed higher than WIFI; that is, whenever BLE is transmitting or receiving, the antenna switches to Bluetooth, and when BLE is idle, the antenna switches to WIFI.

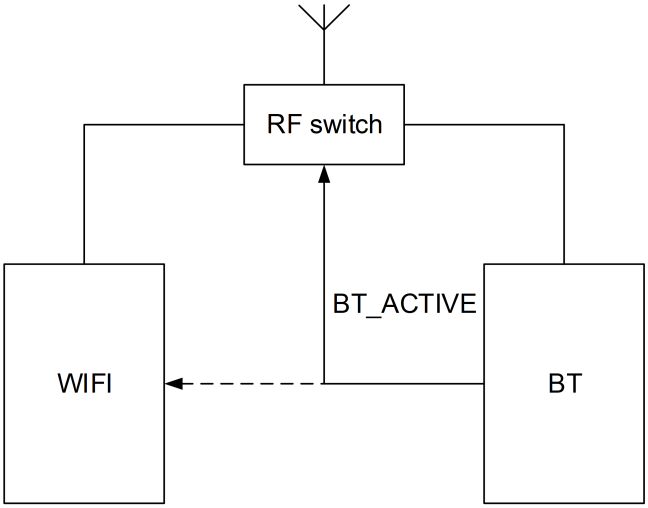


Figure 15-3: Example of a single-wire internal arbitration scheme

15.3 Using external PA and LNA

Using external PA and LNA modules to enhance Bluetooth performance, connecting these modules externally to the chip can further increase Bluetooth transmit power and sensitivity, enabling Bluetooth transmission over distances ranging from hundreds of meters to several kilometers.

By utilizing the coexistence interface output signals TX_ACTIVE and RX_ACTIVE, the operation of external PA and LNA modules can be controlled. The specific configuration method is detailed in the coexistence interface configuration procedure.

Disclaimer and Copyright Notice

Retail samples and small batches can be purchased directly from the Taobao store (sifli.taobao.com).
For large-volume corporate purchases, please send an email to sales@sifli.com for inquiries.
More products and documents are available on wiki.sifli.com.
For problem discussions, you can visit bbs.sifli.com to join the communication.
For open-source projects applying for free samples, you can join QQ Group 674699679 for communication.
To get the latest product information, please follow our official WeChat account.

Address: 419-13, Block B, Science and Technology Headquarters Building, No.320 Pubin Road, Nanjing Area, Jiangsu Free Trade Zone, P. R. China 200131
Email: sales@sifli.com