SF32LB56x

User Manual

V1.0.5

UM5601-SF32LB56x-EN

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

chapter*Revision History addcontentslinetocchapterRevision 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
2026-01-13	1.0.5	Added description of the RTC module
2025-12-18	1.0.4	Added Chapter2 Measurement and Calibraton of Clocks
2025-07-29	1.0.3	Added Bluetooth introduction and the configuration of external LNA and PA
2025-04-08	1.0.2	Added Bluetooth coexistence section and description of the BUSMON module
2025-03-10	1.0.1	Corrected descriptions in the input/output section regarding the number of low-power IOs, USB endpoints, and supported data bit widths of SPI
2025-03-03	1.0	Revised descriptions in the input/output section
2025-01-10	0.9	Added description of package-integrated IOs and overview of each IP
2024-12-20	0.8	Updated modules including LCDC and EFUSE
2024-12-06	0.7	Updated sections on clock and reset, and low power
2023-06-15	0.6	Added timer cascade
2023-06-14	0.5	Added SPIsection
2023-02-23	0.4	Updated IOsection and added AONsection
2023-02-21	0.3	Added GPADCandAESsections
2023-02-03	0.2	Added sections on RCC, IO, I2S, CRC, and others
2023-01-31	0.1	Initial Draft

Product Overview

The SF32LB56x series comprises highly integrated, high-performance MCU chips designed for Ultra-Low Power Artifcial Intelligence Internet of Things (AIoT) applications. The chip employs a big-little core architecture based on the Arm Cortex-M33 STAR-MC1 processor, integrating a high-performance 2D/2.5D graphics engine, an artifcial intelligence neural network accelerator, dual-mode Bluetooth 5.3 , and an audio CODEC . It is broadly applicable to various scenarios, including wristband wearable electronic devices, intelligent mobile terminals, and smart home systems.

The chip's big core performance processor operates at a maximum frequency of 240MHz , deliv ering a single-core performance of 984 CoreMark , thereby providing the high-performance com puting power necessary for rich applications and smooth human-machine interaction. The little core low-power processor operates at a maximum frequency of 96MHz , delivering a perfor mance of 394 CoreMark and a power efciency of 3.8uA/CoreMark . It balances the concurrent control of multiple sensors via the Sensor Hub and the execution of the Bluetooth protocol stack, effectively reconciling the high computational performance required for smooth human-mach ine interaction with the ultra-low power consumption necessary for extended standby time.

The chip integrates a 2D/2.5D GPU with a maximum operating frequency of 240MHz, Supports four-layer overlay,alpha aliasing, hardware-accelerated real-time rotation and scaling, as well as various common graphic format conversions. Supports hard ware-accelerated lossless compression and decompression of graphics, and native animation, signifcantly improving bandwidth utilization and reducing storage costs. The chip features a built-in LCD controller supporting multiple interfaces such as 8080/QSPI/JDI, enabling autonomous full-screen refresh rates up to 60fps without CPU dependency, and supports low-power screen-off always-on display

Integrates a world-class dual-mode Bluetooth 5.3 transceiver, with classic Bluetooth EDR2 mode maximum transmit power of 13dBm, peak receive current as low as 2.2mA@3.3V, low-power Bluetooth re ceive sensitivity of -100dBm (1Mbps), and classic Bluetooth EDR2 mode sensitivity of -95.5dBm. Integrated high-fdelity audio ADC and DAC , supporting Bluetooth calls and headphone MP3 playback.

Functional Block Diagram

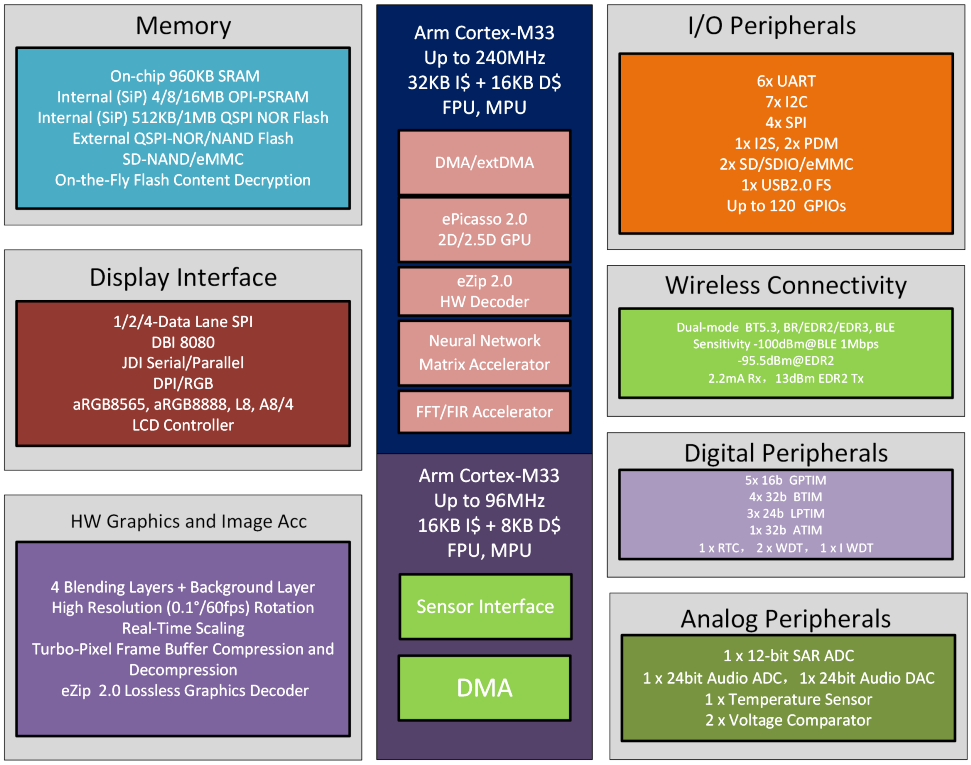


Figure 0-1: Functional Block Diagram

Product features

CPU and memory

- Performance processor/Big core (HCPU)
  - Processor:Arm Cortex-M33 STAR-MC1
  - Operating frequency: up to 240 MHz, adjustable
  - Up to 370 DMIPS,984 EEMBC CoreMark
  - I/D-Cache：32KB(2-way)+16KB(4-way)
  - SRAM : 800KB (of which 128KB is Retention SRAM)
  - CoreMark power efciency: <34uA/MHz @3.3V
  - Single-precision floating-point unit (FPU)
  - Memory Protection Unit (MPU)
- Ultra-low power processor / Little core (LCPU)
  - ⁻Processor:Arm Cortex-M33 STAR-MC1
  - Operating frequency: up to 96MHz, adjustable
  - Up to 148 DMIPS, 394 EEMBC CoreMark
  - I/D-Cache：16KB(2-way)+8KB(4-way)
  - SRAM: 160KB (all Retention SRAM)
  - CoreMark power efciency: <15.5uA/MHz @3.3V
  - Single-precision floating-point unit (FPU)
  - Memory Protection Unit (MPU)

Wireless connectivity

- Dual-mode Bluetooth 5.3, supports BLE Audio
- Sensitivity: -100dBm (BLE/1Mbps), -96.2dBm (BR),-95.5dBm（EDR2）, -88.7dBm（EDR3）
- Maximum Transmit Power:13dBm (EDR2/3),17dBm (BR/BLE)
- Receiver Peak Current Consumption (BR): 2.2mA@3.3V

Graphics Display

- 2D/2.5D Graphics Engine—ePicasso™2.0
  - Supports Four Layers with Alpha Aliasing, plus Solid Color Background Layer
  - Supports Hardware-Accelerated Rotation, Scaling, and Mirroring
  - Maximum Resolution 512×512
  - Supports aRGB8565,aRGB8888,L8,A8/4, supports Alpha Aliasing
- Lossless Decompression Accelerator—eZip™2.0
  - Hardware lossless graphics decompression, supporting lossless animation eZip-A
  - SupportsaRGB8565，aRGB8888，L8，A8/4 formats
  - Supports integration with ePicasso™2.0without intermediate buffering
- LCD Controller
  - Supports 8080,SPI,Dual-SPI,Quad-SPI,JDI interfaces
  - Supports two-layer alphaaliasing, plus a solid color background layer
  - TurboPixel™frame buffer compression and decompression

Audio

- 1× High-fdelity 24-bit audio DAC, 108dB SNR
- 1× High-fdelity 24-bit audio ADC, 99dB SNR
- 2×PDM digital microphone input
- 1×I^2S
- Audio Sampling Rate Conversion Accelerator
- Audio EQEqualization Accelerator

Neural Network Matrix Accelerator

- Designed for TinyMLscenarios, effciently performing matrix convolution operations
- Maximum processing capability of 1.92GOPS
- Power efciency exceeding 10TOPS/W

Digital Signal Processing Accelerator

- One FFT accelerator within the Big core
- One FIR filter accelerator within the Big core
- Each processor is equipped with a CORDIC trigonometric function coprocessor

Storage Interface

- 4×MPI，supporting QSPI-NOR、SPI-NAND、QPI/OPI-PSRAM
- 2×SD/SDIO/eMMC，one set each of 4-wire and 8-wire, supporting SD3.0, SDIO3.0 , as well as eMMC 4.51

System Clock

- Oscillator
  - 48MHz Crystal Oscillator
  - Low-Power RCOscillator:1 MHz, 48 MHz
  - Ultra-Low-Power RCOscillator:10 kHz
  - Ultra-Low-Power 32.768 kHzCrystal Oscillator,

optional

- PLL
  - Dedicated Audio PLL
  - 3×PLL，maximum frequency 384 MHz , in increments of 24 MHz

Security

- AES Accelerator
  - Symmetric encryption and decryption support for AES- 128, AES-192, AES-256,SM3 and SM4 algorithms
  - Encryption and decryption modes include ECB, CTR, and CBC
  - AES operates via DMA, processing source data and writing to the destination Label Address
  - Supports external Key and also supports Root Key
  - Supports real-time decryption of encrypted data read from NOR Flash
- SHA1 andSHA256hardware acceleration
- Supports fully programmable polynomial calculations using 7/8/16/32 bits CRC , data input supports 8/16/32 bit widths; CRC calculation can be driven by CPU or DMA
- True Random Number Generator (TRNG)
- Supports Secure Boot
- Built-in 1024-bit eFuse for storing the Root of Trustand unique ID(UID)
- PSA Certified Level 1 certifcation

Others

- DMA
  - General-purpose DMA: for effcient data transfer between peripherals
  - extDMA : Used for high-efciency data transfer with external storage
- Timers
  - 5×16b GPTIM，1×32b ATIM，4×32b BTIM，3×24b LPTIM
  - 1×RTC
  - 2× Watchdog24b WDT，1×Independent Watchdog IWDT
- Analog
  - 1×12-bit General-purposeSAR ADC, total 8channels
  - 1×On-chip temperature sensor
  - 2×Low-power voltage comparator
- Peripheral connectivity
  - 6×UART，7× I^2C，4×SPI，1×ISO7816
  - 1×USB2.0 FS
  - SIM card controller
  - Peripheral Task Controller（PTC）
- Power management
  - Input voltage：1.7-3.6V，-40 to 85°C
  - Built-in high-efciency Buckand low-power LDO
  - Sleep current during RTC operation: 600nA
  - Sleep current during pin wake-up configuration:300nA

Package

- WBBGA175，120 个 GPIO，6.5×6.1×0.94mm
- QFN68L，44 个 GPIO，7×7×0.75mm

Application Scenarios

Smart Wearables

- High-end Smartwatch
- Smart Bracelet
- Wearable Medical Devices
- Fitness Equipment

Industrial

- Cost-effective Display Solutions
- Graphical Human-Machine Interface Devices
- Industrial Sensor Control Center
- Industrial Equipment Monitoring
- Industrial Instruments and Meters

Automotive

- Electric Vehicle Central Control Devices
- Car Key
- Wearable Automotive Remote Control Devices

Home Automation

- Small- and Medium-sized Smart Appliances
- Smart Door Lock

General

- Low-Power Sensor Hub
- Bluetooth mesh

Contents

List of Figures

List of Tables

1 Chip Overview

1.1 System Architecture

SF32LB56x() is a series of highly integrated, high-performance MCU chips designed for ultra-low power Artifcial Intelligence Internet of Things (AIoT) applications. The chip employs a big.LITTLE architecture based on the Arm Cortex-M33 STAR-MC1 processor, comprising:

- Performance processor / Big core (HCPU): maximum operating frequency of 240 MHz, equipped with 32 KB instruction cache (I-Cache) and 16 KB data cache (D-Cache), and 800 KB SRAM (including 128 KB Retention SRAM); As the system master controller, it effciently accesses both on-chip and off-chip memory, primarily serving system control, human-machine interaction, and high-performance computing.
- Ultra-low power processor / Little core (LCPU): maximum operating frequency 96MHz, equipped with 16KB instruction cache (I-Cache) and 8KB data cache (D-Cache), 160KB SRAM (entirely Retention SRAM); Primarily functions as the system's ultra-low power sensor hub (Sensor Hub) and controller for low-power Bluetooth connections, fulflling various data acquisition, processing, transmission, and control requirements in ultra-low power scenarios

1.2 Cortex-M33 STAR-MC1 "Star" Processor

The Cortex-M33 STAR-MC1 processor is the first processor in the "Star" series launched by Arm China (Arm China). This processor inherits the main features of the Cortex-M33 , supports all functions of the existing Armv8-M architecture, features an in-order three-stage pipeline, signifcantly reduces system power consumption, has partial dual-issue 16-bit instruction capability, and further improves the coprocessor interface by adding support for cache (Cache). Support.

The Cortex-M33 STAR-MC1 achieves performance of 1.5 DMIPS/MHz and 4.02 CoreMark/MHz . Compared with the previous generation of Arm processors in the same class, at the same operating frequency, the performance of the Cortex-M33 STAR-MC1 is improved by20%。

The Cortex-M33 STAR-MC1 provides a coprocessor (Coprocessor) interface to further enhance customized computing capabilities based on different scenario requirements. Using the MCR (Move from Coprocessor to Register) and MRC (Move from Register to Coprocessor) instructions, register data and computation results can be transferred between the Cortex-M33 STAR-MC1 and the coprocessor, making it well suited for operations involving small data volumes, complex yet relatively fragmented computations, and low latency. While the coprocessor performs computations, the Cortex-M33 STAR-MC1 processor can concurrently execute other instructions, thereby signifcantly improving execution effciency.

Additionally, the processor supports the Digital Signal Processing (DSP) instruction set and the Floating Point Unit (FPU)。

The Cortex-M33 STAR-MC1 integrates tightly coupled memory (TCM) and cache (Cache) technologies, enhancing the flexibility of utilizing various built-in and external storage systems with differing characteristics, thereby ensuring the processor's real-time responsiveness and computational effciency across diverse scenarios

1.3 Performance Processor (Big Core) System (HPSYS)

1.3.1 Bus Architecture

The HPSYS internally provides a bus matrix based on the AHB protocol, supporting multiple master devices to access multiple slave device address spaces concurrently

As shown in Figure 1-1， the bus master device is located at the top, and the slave device address space is located on the right; the black dot at the intersection indicates bus connectivity.

HCPU can access all address spaces of HPSYS and can access all address spaces of LPSYS across cores via HP2LP.

DMAC1 can access all address spaces of HPSYS and can access all address spaces of LPSYS across domains via HP2LP.

Some master devices ofLPSYScan access the address space of HPSYS across domains via LP2HP.

HPSYS_ITCM can only be accessed by HCPU and DMAC1. DTCM shares a 128KB address space with HPSYS_RAM0 and can be accessed by HCPU and other master devices.

When multiple master devices simultaneously access the same slave device address space, the access order is determined based on polling arbitration.

When multiple master devices not connected by borders simultaneously access different slave device address spaces, they operate independently. When two master devices connected by borders initiate access simultaneously, the access order is determined based on fixed priority or polling arbitration.

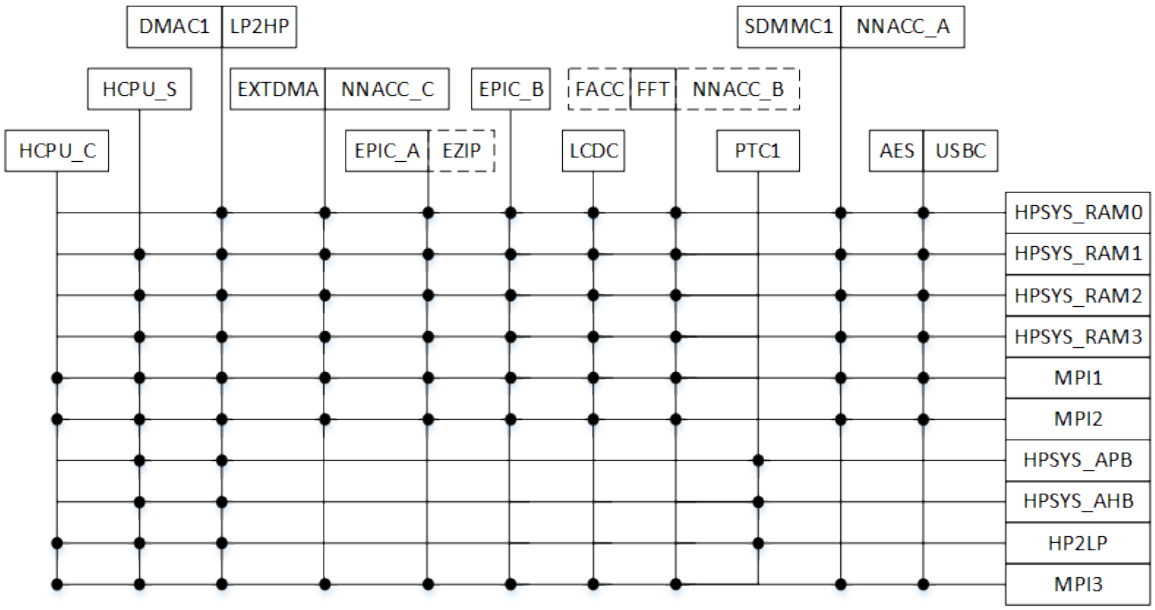


Figure 1-1: Performance Processor (Big Core) System Bus Architecture

1.3.2 Memory Type

1.3.2.1 Cache

HCPU is configured with 32KB 2-way I-Cache (Level 1 instruction cache) and 16KB 4-way D-Cache (Level 1 data cache), signifcantly enhancing CPU execution effciency during XIP. The software must properly configure the MPU (Memory Protection Unit) settings for cache and non-cache address segments, balancing effciency and usability.

1.3.2.2 TCM

The HCPU is configured with 128KB zero-wait-cycle D-TCM , with an address space from 0x2000_0000 to 0x2001_FFFF , which can be used to place code and data with stringent real-time requirements. This TCM memory is also connected to the bus and can be accessed by other AHB masters. When the system enters low-power mode, this memory can retain data.

1.3.2.3 SRAM

The HPSYSbus contains a total of800KB SRAM, including:

- 0x2000_0000-0x2001_FFFF , 128KB zero-wait-cycle SRAM (shared with D-TCM), accessible by all AHB masters, The maximum frequency is 240MHzand includes retentionfunctionality.
- 0x2002_0000-0x200C_7FFF ， 672KB SRAM， wait-cycle is 0or 1, accessible by allAHB masters, with a maximum frequency of 240MHz.

1.3.2.4 External RAM

HPSYS supports external OPI DDR pSRAM with an address space from 0x6000_0000 to 0x63FF_FFFF ; the actual accessible address range depends on the capacity of the external device. The interface maximum frequency is DDR 168MHz with a data width of 8-bit .

1.3.2.5 External Flash

HPSYS supports externalNOR/NAND FLASH, including:

- 0x6000_0000－0x63FF_FFFF Address range can be mapped to embedded FLASH, with a recommended operating frequency of 96MHz
- 0x6400_0000 － 0x9FFF_FFFF Address range can be mapped to external FLASH, with a recommended operating frequency of 72MHz

1.3.3 Address Mapping

Table 1-1: HPSYSAddress Mapping

Category	Memory /IP	Address space	HCPU				LCPU	
			Starting Address		Ending Address		Starting Address	Ending Address
HPSYS_ITCM		**64KB**	**0x0000_0000**		**0x0000_FFFF**		**NA**	**NA**
	ROM	64KB	0x0000_0000		0x0000_FFFF		-	-
	Reserved	-	-		-		-	-
External Memory		**1024MB**	**0x1000_0000**	**0x6000_0000**	**0x1BFF_FFFF**	**0x9FFF_FFFF**	**0x6000_0000**	**0x9FFF_FFFF**
	MPI1 Memory	4MB	0x1000_0000	0x6000_0000	0x103F_FFFF	0x603F_FFFF	0x6000_0000	0x603F_FFFF
	MPI2 Memory	60MB	0x1040_0000	0x6040_0000	0x13FF_FFFF	0x63FF_FFFF	0x6040_0000	0x63FF_FFFF
	MPI3 Memory	128MB/960MB	0x1400_0000	0x6400_0000	0x1BFF_FFFF	0x9FFF_FFFF	0x6400_0000	0x9FFF_FFFF
HPSYS_RAM		**800KB**	**0x2000_0000**		**0x200C_7FFF**		**0x2A00_0000**	**0x2A0C_7FFF**
	RAM0 (Retention)	128KB	0x2000_0000		0x2001_FFFF		0x2A00_0000	0x2A01_FFFF
	RAM1	128KB	0x2002_0000		0x2003_FFFF		0x2A02_0000	0x2A03_FFFF
	RAM2	256KB	0x2004_0000		0x2007_FFFF		0x2A04_0000	0x2A07_FFFF
	RAM3	288KB	0x2008_0000		0x200C_7FFF		0x2A08_0000	0x2A0C_7FFF

Continued on next page...

Table 1-1 Address Mapping (continued)

Category	Memory /IP	Address space	HCPU		LCPU	
			Starting Address	Ending Address	Starting Address	Ending Address
HPSYS_APB1		**256KB**	**0x4000_0000**	**0x4003_FFFF**	**0x4000_0000**	**0x4003_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
	USART1	4KB	0x4000_4000	0x4000_4FFF	0x4000_4000	0x4000_4FFF
	USART2	4KB	0x4000_5000	0x4000_5FFF	0x4000_5000	0x4000_5FFF
	EZIP1	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
	Reserved	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
	Reserved	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
	Reserved	4KB	0x4001_8000	0x4001_8FFF	0x4001_8000	0x4001_8FFF
	NNACC1	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
	Reserved	4KB	0x4001_E000	0x4001_EFFF	0x4001_E000	0x4001_EFFF
	Reserved	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
	ATIM1	4KB	0x4002_3000	0x4002_3FFF	0x4002_3000	0x4002_3FFF
	Reserved	4KB	0x4002_4000	0x4002_4FFF	0x4002_4000	0x4002_4FFF
	AUDPRC	4KB	0x4002_5000	0x4002_5FFF	0x4002_5000	0x4002_5FFF
	AUDCODEC	4KB	0x4002_6000	0x4002_6FFF	0x4002_6000	0x4002_6FFF
	FFT1	4KB	0x4002_7000	0x4002_7FFF	0x4002_7000	0x4002_7FFF
	FACC1	4KB	0x4002_8000	0x4002_8FFF	0x4002_8000	0x4002_8FFF
	USART3	4KB	0x4002_9000	0x4002_9FFF	0x4002_9000	0x4002_9FFF
	Reserved	4KB	0x4002_A000	0x4002_AFFF	0x4002_A000	0x4002_AFFF
	CAN1	4KB	0x4002_B000	0x4002_BFFF	0x4002_B000	0x4002_BFFF
	Reserved	4KB	0x4002_C000	0x4002_CFFF	0x4002_C000	0x4002_CFFF
	SCI	4KB	0x4002_D000	0x4002_DFFF	0x4002_D000	0x4002_DFFF
	Reserved	4KB	0x4002_E000	0x4002_EFFF	0x4002_E000	0x4002_EFFF
	I2C4	4KB	0x4002_F000	0x4002_FFFF	0x4002_E000	0x4002_FFFF
	Reserved	64KB	0x4003_0000	0x4003_FFFF	0x4003_0000	0x4003_FFFF
HPSYS_APB2		**256KB**	**0x4004_0000**	**0x4007_FFFF**	**0x4004_0000**	**0x4007_FFFF**
	HPSYS_AON	4KB	0x4004_0000	0x4004_0FFF	0x4004_0000	0x4004_0FFF
	LPTIM1	4KB	0x4004_1000	0x4004_1FFF	0x4004_1000	0x4004_1FFF
	Reserved	4KB	0x4004_2000	0x4004_2FFF	0x4004_2000	0x4004_2FFF
	Reserved	52KB	0x4004_3000	0x4004_FFFF	0x4004_3000	0x4004_FFFF
	Reserved	64KB	0x4005_0000	0x4005_FFFF	0x4005_0000	0x4005_FFFF
	Reserved	64KB	0x4006_0000	0x4006_FFFF	0x4006_0000	0x4006_FFFF
	Reserved	64KB	0x4007_0000	0x4007_FFFF	0x4007_0000	0x4007_FFFF
HPSYS_AHB		**256KB**	**0x4008_0000**	**0x400B_FFFF**	**0x4008_0000**	**0x400B_FFFF**
	GPIO1	4KB	0x4008_0000	0x4008_0FFF	0x4008_0000	0x4008_0FFF
	MPI1	4KB	0x4008_1000	0x4008_1FFF	0x4008_1000	0x4008_1FFF
	MPI2	4KB	0x4008_2000	0x4008_2FFF	0x4008_2000	0x4008_2FFF
	MPI3	4KB	0x4008_3000	0x4008_3FFF	0x4008_3000	0x4008_3FFF
	Reserved	4KB	0x4008_4000	0x4008_4FFF	0x4008_4000	0x4008_4FFF
	SDMMC1	4KB	0x4008_5000	0x4008_5FFF	0x4008_5000	0x4008_5FFF
	SDMMC2	4KB	0x4008_6000	0x4008_6FFF	0x4008_6000	0x4008_6FFF
	USBC	4KB	0x4008_7000	0x4008_7FFF	0x4008_7000	0x4008_7FFF
	CRC1	4KB	0x4008_8000	0x4008_8FFF	0x4008_8000	0x4008_8FFF
	Reserved	28KB	0x4008_9000	0x4008_FFFF	0x4008_9000	0x4008_FFFF
	GFX_RAM	64KB	0x4009_0000	0x4009_FFFF	0x4009_0000	0x4009_FFFF
	Reserved	128KB	0x400A_0000	0x400B_FFFF	0x400A_0000	0x400B_FFFF

1.3.4 Interrupt Vector Table

Table 1-2: HCPU Interrupt Vector Table

IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source
NMI	**WDT1**	IRQ[25]	**LCPU_IRQ[25]**	IRQ[51]	**DMAC1_CH2**	IRQ[77]	**EXTDMA**
IRQ[0]	**AON**	IRQ[26]	**LCPU_IRQ[26]**	IRQ[52]	**DMAC1_CH3**	IRQ[78]	**I2C4**
IRQ[1]	**LCPU_IRQ[1]**	IRQ[27]	**LCPU_IRQ[27]**	IRQ[53]	**DMAC1_CH4**	IRQ[79]	**SDMMC1**
IRQ[2]	**LCPU_IRQ[2]**	IRQ[28]	**LCPU_IRQ[28]**	IRQ[54]	**DMAC1_CH5**	IRQ[80]	**SDMMC2**
IRQ[3]	**LCPU_IRQ[3]**	IRQ[29]	**LCPU_IRQ[29]**	IRQ[55]	**DMAC1_CH6**	IRQ[81]	**NNACC1**
IRQ[4]	**LCPU_IRQ[4]**	IRQ[30]	**LCPU_IRQ[30]**	IRQ[56]	**DMAC1_CH7**	IRQ[82]	**PDM1**
IRQ[5]	**LCPU_IRQ[5]**	IRQ[31]	**LCPU_IRQ[31]**	IRQ[57]	**DMAC1_CH8**	IRQ[83]	**CAN1**
IRQ[6]	**LCPU_IRQ[6]**	IRQ[32]	**LCPU_IRQ[32]**	IRQ[58]	**LCPU2HCPU**	IRQ[84]	**GPIO1**
IRQ[7]	**LCPU_IRQ[7]**	IRQ[33]	**LCPU_IRQ[33]**	IRQ[59]	**UART1**	IRQ[85]	**MPI1**
IRQ[8]	**LCPU_IRQ[8]**	IRQ[34]	**LCPU_IRQ[34]**	IRQ[60]	**SPI1**	IRQ[86]	**MPI2**
IRQ[9]	**LCPU_IRQ[9]**	IRQ[35]	**LCPU_IRQ[35]**	IRQ[61]	**I2C1**	IRQ[87]	**MPI3**
IRQ[10]	**LCPU_IRQ[10]**	IRQ[36]	**LCPU_IRQ[36]**	IRQ[62]	**EPIC**	IRQ[88]	**FFT1**
IRQ[11]	**LCPU_IRQ[11]**	IRQ[37]	**LCPU_IRQ[37]**	IRQ[63]	**LCDC1**	IRQ[89]	**EZIP1**
IRQ[12]	**LCPU_IRQ[12]**	IRQ[38]	**LCPU_IRQ[38]**	IRQ[64]	**I2S1**	IRQ[90]	**AUDPRC**
IRQ[13]	**LCPU_IRQ[13]**	IRQ[39]	**LCPU_IRQ[39]**	IRQ[65]	**rsvd**	IRQ[91]	**PDM2**
IRQ[14]	**LCPU_IRQ[14]**	IRQ[40]	**LCPU_IRQ[40]**	IRQ[66]	**EFUSEC**	IRQ[92]	**USBC**
IRQ[15]	**LCPU_IRQ[15]**	IRQ[41]	**LCPU_IRQ[41]**	IRQ[67]	**AES**	IRQ[93]	**I2C3**
IRQ[16]	**LCPU_IRQ[16]**	IRQ[42]	**LCPU_IRQ[42]**	IRQ[68]	**PTC1**	IRQ[94]	**ATIM1**
IRQ[17]	**LCPU_IRQ[17]**	IRQ[43]	**LCPU_IRQ[43]**	IRQ[69]	**TRNG**	IRQ[95]	**UART3**
IRQ[18]	**LCPU_IRQ[18]**	IRQ[44]	**LCPU_IRQ[44]**	IRQ[70]	**GPTIM1**	IRQ[96]	**AUD_HP**
IRQ[19]	**LCPU_IRQ[19]**	IRQ[45]	**LCPU_IRQ[45]**	IRQ[71]	**GPTIM2**	IRQ[97]	**SCI**
IRQ[20]	**LCPU_IRQ[20]**	IRQ[46]	**LPTIM1**	IRQ[72]	**BTIM1**	IRQ[98]	**FACC1**
IRQ[21]	**LCPU_IRQ[21]**	IRQ[47]	**rsvd**	IRQ[73]	**BTIM2**	IRQ[99]	**rsvd**
IRQ[22]	**LCPU_IRQ[22]**	IRQ[48]	**rsvd**	IRQ[74]	**UART2**	\	\
IRQ[23]	**LCPU_IRQ[23]**	IRQ[49]	**RTC**	IRQ[75]	**SPI2**	\	\
IRQ[24]	**LCPU_IRQ[24]**	IRQ[50]	**DMAC1_CH1**	IRQ[76]	**I2C2**	\	\

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

1.4.1 Bus Architecture

LLPSYS internally provides a bus matrix based on the AHBprotocol, supporting multiple master devices to access multiple slave device address spaces concurrently.

As shown in Figure1-2,the bus master device is located at the top, and the slave device address space is located on the right; the black dot at the intersection indicates bus connectivity.

LCPU and DMAC2 can access all address spaces of LPSYS and can perform cross-domain access to all address spaces of HPSYS except HPSYS_ITCM via LP2HP.

Certain master devices ofHPSYScan perform cross-domain access to all address spaces of LPSYSvia HP2LP.

LPSYS_ITCM and LPSYS_DTCM can be accessed by LCPU and DMAC2, and can also be accessed cross-domain by certain master devices of HPSYS

When multiple master devices simultaneously access the same slave device address space, the access order is determined based on polling arbitration.

When multiple master devices not connected by borders simultaneously access different slave device address spaces, they operate independently. When multiple master devices connected via the bus frame initiate access simultaneously, the access order is determined based on fixed priority or round-robin arbitration principles.

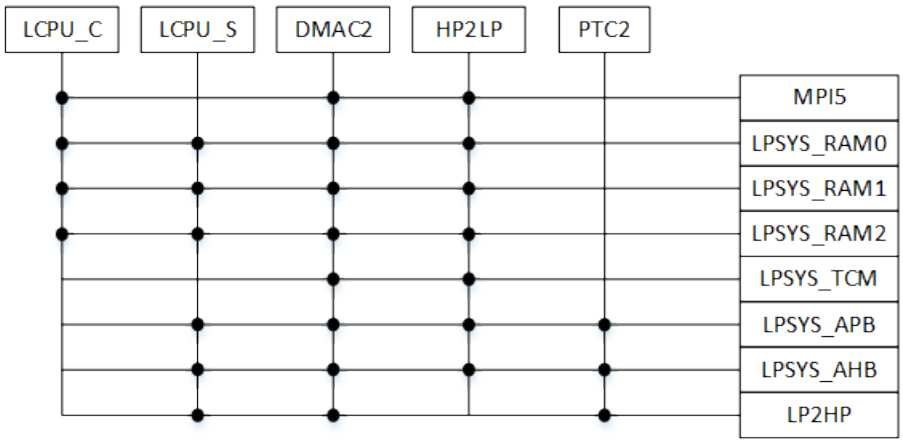


Figure 1-2: Low-power processor (Little core) System Bus Architecture

1.4.2 Memory type

1.4.2.1 Cache

LCPU is configured with16KB 2-way I-Cache(Level 1 instruction cache) and8KB 4-way D-Cache(Level 1 data cache).

1.4.2.2 TCM

LCPU is configured with 16KB zero-wait-cycle I-TCM , with an address space from 0x003F_C000 to 0x003F_FFFF. This TCM memory is dedicated to the LCPU, while the HCPU can initialize it via the address range 0x20BF_C000 to 0x20BF_FFFF. It is recommended to place code and data with stringent real-time (or latency determinism) requirements.

The LCPU is configured with a 16KB zero-wait-cycle D-TCM , with an address space from 0x203F_C000 to 0x203F_FFFF. This TCM memory is dedicated to the LCPU , and the HCPU can initialize it via the same address space.

1.4.2.3 SRAM

The LPSYS bus contains a total of 128KB SRAM with a wait-cycle of 0 or 1 , and an address space from 0x2040_0000 to 0x2041_FFFF . The maximum frequency is 48MHz at 0.9V and 96MHz at 1.1V .

1.4.2.4 External Flash

LPSYS Default externalNOR FLASHused for system boot, with address space from0x1C00_0000to0x1FFF_FFFF.

1.4.3 Address Mapping

Table 1-3: LPSYS Address Mapping

Category	Memory /IP	Address space	HCPU		LCPU	
			Starting Address	Ending Address	Starting Address	Ending Address
LPSYS_ITCM		4MB	0x2080_0000	0x20BF_FFFF	0x0000_0000	0x003F_FFFF

Continued on next page...

Table 1-3 Address Mapping (continued)

Category	Memory /IP	Address space	HCPU				LCPU			
			Starting Address		Ending Address		Starting Address		Ending Address	
	ROM	384KB	0x2080_0000		0x2085_FFFF		0x0000_0000		0x0005_FFFF	
	Reserved	-	-		-		-		-	
	RAM	16KB	0x20BF_C000		0x20BF_FFFF		0x003F_C000		0x003F_FFFF	
LPSYS_DTCM		**4MB**	**0x2000_0000**		**0x203F_FFFF**		**0x2000_0000**		**0x203F_FFFF**	
	Reserved	-	-		-		-		-	
	RAM	16KB	0x203F_C000		0x203F_FFFF		0x203F_C000		0x203F_FFFF	
External Memory		**64MB**	**0x1C00_0000**		**0x1FFF_FFFF**		**0x1C00_0000**		**0x1FFF_FFFF**	
	MPI5 Memory	64MB	0x1C00_0000		0x1FFF_FFFF		0x1C00_0000		0x1FFF_FFFF	
LPSYS_RAM		**128KB**	**0x20C0_0000**	**0x2040_0000**	**0x20C1_FFFF**	**0x2041_FFFF**	**0x0040_0000**	**0x2040_0000**	**0x0041_FFFF**	**0x2041_FFFF**
	RAM0	64KB	0x20C0_0000	0x2040_0000	0x20C0_FFFF	0x2040_FFFF	0x0040_0000	0x2040_0000	0x0040_FFFF	0x2040_FFFF
	RAM1	32KB	0x20C1_0000	0x2041_0000	0x20C1_7FFF	0x2041_7FFF	0x0041_0000	0x2041_0000	0x0041_7FFF	0x2041_7FFF
	RAM2 (EM)	32KB	0x20C1_8000	0x2041_8000	0x20C1_FFFF	0x2041_FFFF	0x0041_8000	0x2041_8000	0x0041_FFFF	0x2041_FFFF
LPSYS_APB1		**192KB**	**0x5000_0000**		**0x5003_FFFF**		**0x5000_0000**		**0x5003_FFFF**	
	RCC2	4KB	0x5000_0000		0x5000_0FFF		0x5000_0000		0x5000_0FFF	
	DMAC2	4KB	0x5000_1000		0x5000_1FFF		0x5000_1000		0x5000_1FFF	
	MAILBOX2	4KB	0x5000_2000		0x5000_2FFF		0x5000_2000		0x5000_2FFF	
	PINMUX2	4KB	0x5000_3000		0x5000_3FFF		0x5000_3000		0x5000_3FFF	
	PATCH	4KB	0x5000_4000		0x5000_4FFF		0x5000_4000		0x5000_4FFF	
	USART4	4KB	0x5000_5000		0x5000_5FFF		0x5000_5000		0x5000_5FFF	
	USART5	4KB	0x5000_6000		0x5000_6FFF		0x5000_6000		0x5000_6FFF	
	USART6	4KB	0x5000_7000		0x5000_7FFF		0x5000_7000		0x5000_7FFF	
	Reserved	4KB	0x5000_8000		0x5000_8FFF		0x5000_8000		0x5000_8FFF	
	SPI3	4KB	0x5000_9000		0x5000_9FFF		0x5000_9000		0x5000_9FFF	
	SPI4	4KB	0x5000_A000		0x5000_AFFF		0x5000_A000		0x5000_AFFF	
	WDT2	4KB	0x5000_B000		0x5000_BFFF		0x5000_B000		0x5000_BFFF	
	I2C5	4KB	0x5000_C000		0x5000_CFFF		0x5000_C000		0x5000_CFFF	
	I2C6	4KB	0x5000_D000		0x5000_DFFF		0x5000_D000		0x5000_DFFF	
	I2C7	4KB	0x5000_E000		0x5000_EFFF		0x5000_E000		0x5000_EFFF	
	SYSCFG2	4KB	0x5000_F000		0x5000_FFFF		0x5000_F000		0x5000_FFFF	
	GPTIM3	4KB	0x5001_0000		0x5001_0FFF		0x5001_0000		0x5001_0FFF	
	GPTIM4	4KB	0x5001_1000		0x5001_1FFF		0x5001_1000		0x5001_1FFF	
	GPTIM5	4KB	0x5001_2000		0x5001_2FFF		0x5001_2000		0x5001_2FFF	
	BTIM3	4KB	0x5001_3000		0x5001_3FFF		0x5001_3000		0x5001_3FFF	
	BTIM4	4KB	0x5001_4000		0x5001_4FFF		0x5001_4000		0x5001_4FFF	
	Reserved	4KB	0x5001_5000		0x5001_5FFF		0x5001_5000		0x5001_5FFF	
	GPADC	4KB	0x5001_6000		0x5001_6FFF		0x5001_6000		0x5001_6FFF	
	Reserved	4KB	0x5001_7000		0x5001_7FFF		0x5001_7000		0x5001_7FFF	
	AUDADC	4KB	0x5001_8000		0x5001_8FFF		0x5001_8000		0x5001_8FFF	
	LPCOMP	4KB	0x5001_9000		0x5001_9FFF		0x5001_9000		0x5001_9FFF	
	TSEN	4KB	0x5001_A000		0x5001_AFFF		0x5001_A000		0x5001_AFFF	
	PTC2	4KB	0x5001_B000		0x5001_BFFF		0x5001_B000		0x5001_BFFF	
	Reserved	4KB	0x5001_C000		0x5001_CFFF		0x5001_C000		0x5001_CFFF	
	BUSMON2	4KB	0x5001_D000		0x5001_DFFF		0x5001_D000		0x5001_DFFF	
	Reserved	4KB	0x5001_E000		0x5001_EFFF		0x5001_E000		0x5001_EFFF	
	Reserved	4KB	0x5001_F000		0x5001_FFFF		0x5001_F000		0x5001_FFFF	
	Reserved	64KB	0x5002_0000		0x5002_FFFF		0x5002_0000		0x5002_FFFF	
	Reserved	64KB	0x5003_0000		0x5003_FFFF		0x5003_0000		0x5003_FFFF	
LPSYS_APB2		**64KB**	**0x5004_0000**		**0x5007_FFFF**		**0x5004_0000**		**0x5007_FFFF**	
	LPSYS_AON	4KB	0x5004_0000		0x5004_0FFF		0x5004_0000		0x5004_0FFF	
	LPTIM2	4KB	0x5004_1000		0x5004_1FFF		0x5004_1000		0x5004_1FFF	
	LPTIM3	4KB	0x5004_2000		0x5004_2FFF		0x5004_2000		0x5004_2FFF	
	Reserved	4KB	0x5004_3000		0x5004_3FFF		0x5004_3000		0x5004_3FFF	
	Reserved	24KB	0x5004_4000		0x5004_9FFF		0x5004_4000		0x5004_9FFF	
	PMUC	4KB	0x5004_A000		0x5004_AFFF		0x5004_A000		0x5004_AFFF	
	RTC	4KB	0x5004_B000		0x5004_BFFF		0x5004_B000		0x5004_BFFF	
	IWDT	4KB	0x5004_C000		0x5004_CFFF		0x5004_C000		0x5004_CFFF	
	Reserved	12KB	0x5004_D000		0x5004_FFFF		0x5004_D000		0x5004_FFFF	
	Reserved	64KB	0x5005_0000		0x5005_FFFF		0x5005_0000		0x5005_FFFF	
	Reserved	64KB	0x5006_0000		0x5006_FFFF		0x5006_0000		0x5006_FFFF	
	EUROPA	4KB	0x5007_0000		0x5007_0FFF		0x5007_0000		0x5007_0FFF	
	Reserved	60KB	0x5007_1000		0x5007_FFFF		0x5007_1000		0x5007_FFFF	
LPSYS_AHB		**256KB**	**0x5008_0000**		**0x500B_FFFF**		**0x5008_0000**		**0x500B_FFFF**	
	GPIO2	4KB	0x5008_0000		0x5008_0FFF		0x5008_0000		0x5008_0FFF	
	MPI5	4KB	0x5008_1000		0x5008_1FFF		0x5008_1000		0x5008_1FFF	
	RFC	8KB	0x5008_2000		0x5008_3FFF		0x5008_2000		0x5008_3FFF	
	PHY	4KB	0x5008_4000		0x5008_4FFF		0x5008_4000		0x5008_4FFF	
	CRC2	4KB	0x5008_5000		0x5008_5FFF		0x5008_5000		0x5008_5FFF	
	Reserved	40KB	0x5008_6000		0x5008_FFFF		0x5008_6000		0x5008_FFFF	
	MAC	64KB	0x5009_0000		0x5009_FFFF		0x5009_0000		0x5009_FFFF	
	Reserved	128KB	0x500A_0000		0x500B_FFFF		0x500A_0000		0x500B_FFFF	
PHY_DUMP		**64KB**	**0x500C_0000**		**0x500C_FFFF**		**0x500C_0000**		**0x500C_FFFF**	
	PHY_DUMP	64KB	0x500C_0000		0x500C_FFFF		0x500C_0000		0x500C_FFFF	

1.4.4 Interrupt Vector Table

Table 1-4: LCPU Interrupt Vector Table

IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source	IRQ #	IRQ Source
NMI	**WDT2**	IRQ[12]	**UART4**	IRQ[25]	**BTIM3**	IRQ[38]	**FFT2**
IRQ[0]	**AON**	IRQ[13]	**UART5**	IRQ[26]	**BTIM4**	IRQ[39]	**rsvd**
IRQ[1]	**BLE**	IRQ[14]	**UART6**	IRQ[27]	**AUD_LP**	IRQ[40]	**rsvd**
IRQ[2]	**DMAC2_CH1**	IRQ[15]	**BT**	IRQ[28]	**GPADC**	IRQ[41]	**LPCOMP**
IRQ[3]	**DMAC2_CH2**	IRQ[16]	**SPI3**	IRQ[29]	**rsvd**	IRQ[42]	**LPTIM2**
IRQ[4]	**DMAC2_CH3**	IRQ[17]	**SPI4**	IRQ[30]	**HPSYS0**	IRQ[43]	**LPTIM3**
IRQ[5]	**DMAC2_CH4**	IRQ[18]	**I2S3**	IRQ[31]	**HPSYS1**	IRQ[44]	**HPSYS2**
IRQ[6]	**DMAC2_CH5**	IRQ[19]	**I2C5**	IRQ[32]	**TSEN**	IRQ[45]	**HPSYS3**
IRQ[7]	**DMAC2_CH6**	IRQ[20]	**I2C6**	IRQ[33]	**PTC2**	IRQ[46]	**HCPU2LCPU**
IRQ[8]	**DMAC2_CH7**	IRQ[21]	**I2C7**	IRQ[34]	**rsvd**	IRQ[47]	**RTC**
IRQ[9]	**DMAC2_CH8**	IRQ[22]	**GPTIM3**	IRQ[35]	**GPIO2**	\	\
IRQ[10]	**PATCH**	IRQ[23]	**GPTIM4**	IRQ[36]	**MPI5**	\	\
IRQ[11]	**DM**	IRQ[24]	**GPTIM5**	IRQ[37]	**rsvd**	\	\

1.5 Bus Access Permissions

Table 1-5: Bus Access Permissions

	AHB Slave Device							
AHB Master Controller	HP_ITCM	HP_RAM 03	MPI13	HP_AHB HP_APB	LP_DTCM LP_ITCM	LP_RAM 02	MPI5	LP_AHB LP_APB
HCPU	√	√	√(1)	√	√(2)	√(3)	√	√
DMAC1	√(6)	√	√	√	√(2)	√(3)	√	√
EXTDMA	x	√	√	x	x	x	x	x
AES	x	√	√	x	x	x	x	x
LCDC1	x	√	√	x	x	x	x	x
EZIP	x	√	√	x	x	x	x	x
EPIC	x	√	√	x	x	x	x	x
USBC	x	√	√	x	x	x	x	x
NNACC1	x	√	√	x	x	x	x	x
SDMMC1	x	√	√	x	x	x	x	x
FACC1	x	√	√	x	x	x	x	x
FFT1	x	√	√	x	x	x	x	x
PTC1	x	x	x	√	x	x	x	√
LCPU	√(6)	√(4)	√	√	√	√(5)	√	√
DMAC2	√(6)	√(4)	√	√	√	√	√	√
PTC2	x	√(4)	x	√	x	x	x	√

* (1)HCPU can access the 0x10000000starting address to access MPIcontent, and can also access the 0x60000000starting address to access MPIcontent.
(2)HPSYS as the master controller accesses LPSYS's ITCM, the starting address is 0x20800000, which differs from the address accessed by LCPU.
(3)The main controller of HPSYS accesses the SRAM of LPSYS, with the starting address at either 0x20400000 or 0x20C00000.
(4)The main controller of LPSYS accesses the SRAM of HPSYS, with the starting address at 0x2A000000, representing an offset increase of 0x0A000000
(5)LCPU accesses the SRAM of LPSYS, with the starting address at either 0x00400000 or 0x20400000
(6)Other main controllers access the ROM of HCPU, with the starting address at 0xA0000000 , representing an offset increase of 0xA0000000

2 Clock and Reset

2.1 Introduction

The clock and reset module controls the chip clock and reset, providing functions such as clock selection, clock division, module enable, and module reset.

2.2 Reset Sources

The chip's reset sources are primarily categorized into four types: board-level reset, watchdog reset, software reset, and wake-up reset. Each type includes 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.

Power key (PWRKEY) reset. If the chip's power key PB32 remains at a high level for more than 10 seconds, a PWRKEY reset will occur, resetting all modules except RTC and IWDT. The occurrence of a PWRKEY reset can be queried via the WSR_PWRKEY flag in the PMUC, and this flag can be cleared through the WCR_PWRKEY register in the PMUC.

2.2.2 Watchdog Reset Sources

Watchdog reset sources mainly include: Global Watchdog IWDT. If this watchdog times out, it can generate an IWDT reset, resetting all modules except RTC and IWDT. The occurrence of an IWDT reset can be queried via the WSR_IWDT flag in the PMUC; this flag can be cleared through the WDT_ICR register of the IWDT.

HPSYS Watchdog WDT1. If this watchdog times out, it can generate a WDT1 reset, resetting the HCPU and all peripherals in HPSYS except HPSYS_AON. When the WER_WDT1 register in the PMUC is set to 1, the reset scope can be expanded to reset all modules except PMUC, RTC, and IWDT. After the reset scope is expanded, the occurrence of a WDT1 reset can be queried via the WSR_WDT1 flag in the PMUC and cleared through the WCR_WDT1 register in the PMUC.

LPSYS Watchdog WDT2. If this watchdog times out, it can trigger a WDT2 reset, resetting the LCPU and all peripherals within LPSYS except for LPSYS_AON. When the WER_WDT2 register in the PMUC is set to 1, the reset scope can also be expanded to reset all modules except PMUC, RTC, and IWDT. After the reset scope is expanded, the occurrence of a WDT2 reset can be queried via the WSR_WDT2 flag in the PMUC and cleared through the WCR_WDT2 register in the PMUC.

2.2.3 Software Reset Source

Software Reboot Reboot. Software can trigger a reboot by writing 1 to the CR_REBOOT register of the PMUC. After reboot, all modules except PMUC,RTC, and IWDT are reset. After a software reboot, the CR_REBOOT register of the PMUC remains at 1 and can serve as an indicator that a software reboot has occurred. To trigger another software reboot, the CR_REBOOT register must first be cleared to 0.

HCPU System Reset. Software can reset all internal modules of the HPSYS except HPSYS_AON, including HCPU, EPIC, DMAC1 and others by configuring the internal register of the HCPU to send a SYSRESETREQ. The system reset issued by the external debugger upon connection to HCPU is equivalent to the HCPU system reset.

LCPU system reset. By configuring the internal registers of LCPU and issuing SYSRESETREQ via software, all modules within LPSYS except LPSYS_AON can be reset, including LCPU , DMAC2 , and MAC . The system reset issued by the external debugger upon connection to 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 Wake-up Reset Sources

Hibernate Wake-up. The chip enters hibernatemode; upon wake-up, all modules except PMUC,RTC, and IWDT are reset.

HPSYS standby wake-up. HPSYS enters standby mode; upon wake-up, all internal modules of HPSYS except HPSYS_AON are reset, including HCPU , EPIC , DMAC1 , etc.

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

Table 2-1: Main reset sources of the chip

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

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

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

Reset Scope		Software Reset			Wakeup Reset		
		Reboot	HCPU sysrst	LCPU sysrst	hibernate Wakeup	HPSYS standby Wakeup	LPSYS standby Wakeup
HPSYS	HCPU	√	√	x	√	√	x
	SRAM ret	√	x	x	√	x	x
	SRAM noret	√	x	x	√	√	x
	HPSYS Periphera	√	√	x	√	√	x
	HPAON	√	x	x	√	x	x
LPSYS	LCPU	√	x	√	√	x	√
	SRAM ret	√	x	x	√	x	x
	SRAM noret	√	x	x	√	x	√
	LPSYS Periphera	√	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 following table. The clocks for each functional module 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_audpll	44.1MHz	clk_hxt48
clk_dbl96	96MHz	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 externalenvironmental 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 clk_lrc10 related configuration register is the LRC_CR register of 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 clk_lxt32 related configuration is disabled by default; the register is PMUC's LXT_CR.

clk_hrc48 is an RC oscillator clock generated internally by the chip. At chip startup, this clock is automatically enabled as the default operating clock for HCPU, but its frequency is uncalibrated at this time and is an unknown value below 48MHz. Before calibration, the operating clock of HCPU should be switched to another clock (such as clk_hxt48) before executing the calibration procedure. After calibration, the frequency of clk_hrc48 is 48MHz. The configuration register for clk_hrc48 is PMUC's HRC_CR; the calibration-related registers are HRCCAL1 and HRCCAL2 in HPSYS_RCC. when the chip HPSYS and LPSYS are both in 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 is automatically enabled upon chip start up and serves as the base clock for generating higher frequency clocks, as well as 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 HPSYS and LPSYS are both in low-power mode, this clock is disabled by default. clk_hrc48 related configuration registers are the HXT_CR1/2/3 registers of the PMUC.

clk_dll1/2/3 are internal chip DLL modules that generate high-frequency clocks based on clk_hxt48. These clocks are disabled by default and can be enabled individually as required to generate different frequencies independently. The clock frequency generated by the DLL modules is configurable in 24 MHz increments, with configuration registers located in HPSYS_RCC as DLL1CR, DLL2CR, and DLL3CR. When the chip HPSYS is in low-power mode, these clocks are disabled by default.

clk_audpll is an internal chip PLL module clock generated based on clk_hxt48, serving as the operating clock for audio-related modules; its frequency is adjustable and typically configured for 44.1MHz. This clock is disabled by default; the configuration register is located in theAUDCODEC_LPmodule.

clk_dbl96 is a clock generated by the chip's internal DBL module based on clk_hxt48, with a fixed frequency of 96 MHz. This clock is disabled by default; the related configuration register is DLB96_CR in the PMUC.

2.4 HPSYS Clock Structure

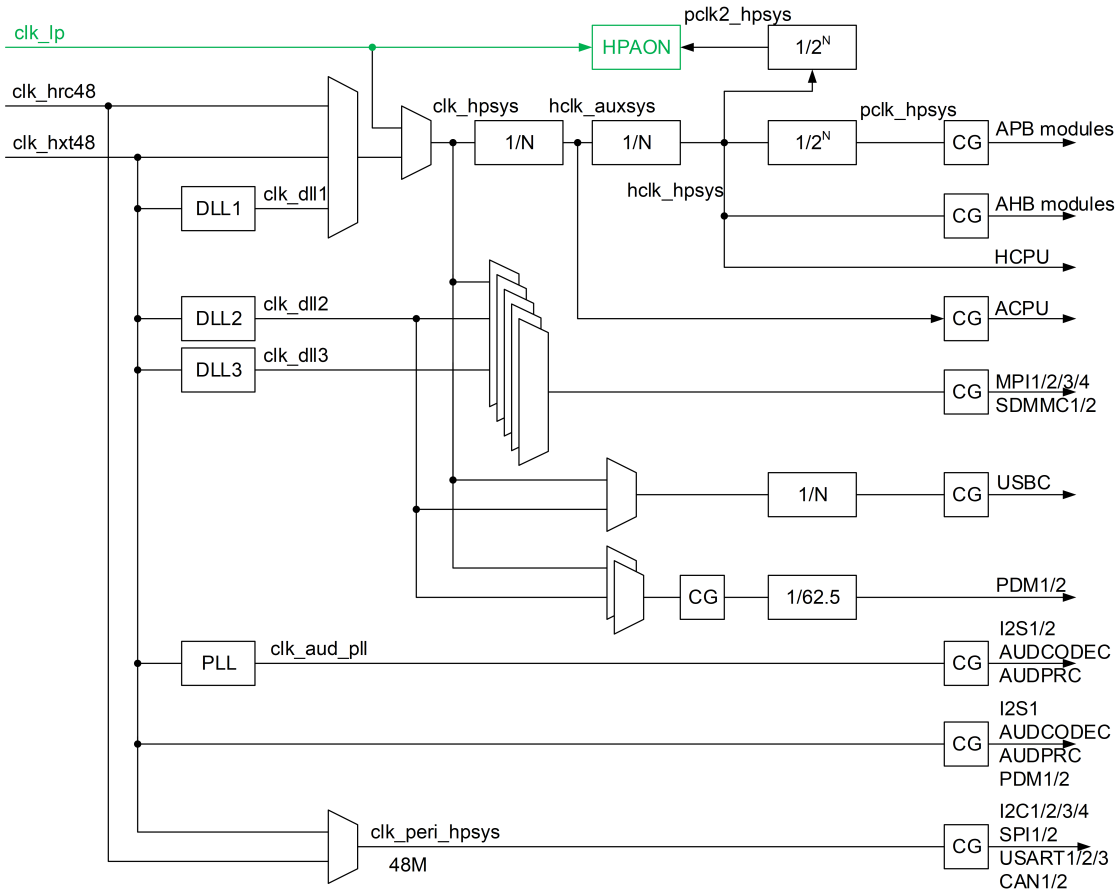


Figure 2-1: HPSYS Clock Structure

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

hclk_hpsys is generated by dividing clk_hpsys by a factor of N, where the division ratio is configured by CFGR_HDIV in HPSYS_RCC. hclk The maximum supported frequency is 240 MHz and it serves as the operating clock for AHB modules including HCPU, EPIC, DMAC1, as well as the AHB bus and SRAM.

pclk_hpsys is generated by dividing hclk_hpsys 1 by 2^N, with a division ratio of 2^{CFGR_PDIV1}. pclk_hpsys The maximum supported frequency is 120 MHz and it serves as the operating clock for APB modules including GPTIM1/2, BTIM1/2, as well as the APB bus clock. When the frequency of hclk_hpsys or pclk_hpsys changes, the operating clock frequency of modules such as GPTIM1/2 and BTIM1/2 also changes, which affects their functionality. Therefore, when the relevant modules are operating, the frequencies of hclk_hpsys and pclk_hpsys must remain constant.

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

clk_lp is a low-power clock selected from clk_lrc10 and clk_lxt32 (refer to the LPSYS clock structure block diagram), and it functions as the operating clock for the HPAON module.

The operating clocks for MPI1/2/3 can be selected from clk_hpsys, clk_dll2, and clk_dll3. The selection register is HPSYS_RCC with CSR_SEL_MPI1/2/3.

The operating clock of SDMMC1 can be selected from clk_hpsys, clk_dll2, and clk_dll3 . The selection register is HPSYS_RCC with CSR_SEL_SDMMC .

The operating clock of USB can be selected from clk_hpsys and clk_dll2 , with the selection register HPSYS_RCC and CSR_SEL_USB , and divided by a factor of 1 to N , where the division ratio is USBCR_DIV . It must be ensured that the operating clock of USB is 60MHz after division; otherwise, the USB will not function properly.

The operating clock clk_peri_hpsys for peripherals such as USART1/2/3 , SPI1/2 , CAN1 , and I2C1/2/3/4 can be selected from clk_hrc48 and clk_hxt48 , The frequency is 48MHz. Select the register HPSYS_RCC and the feld CSR_SEL_PERI within it. clk_peri_hpsys is independent of the system clock and is therefore unaffected during dynamic frequency adjustments of the system.

The audio modules I2S1 , AUDPRC , and AUDCODEC_HP(DAC) can select one of two working clocks for operation; the selection register is located within these modules. One working clock is clk_hxt48 , and the other is clk_audpll.

PDM1/2 can select one of two working clocks for operation; the selection register is located within the PDM module. One working clock is generated by clk_hxt48. Another routeclk_audpll 16division generated.

2.5 HPSYS Module enable

The ENR1 and ENR2 registers in HPSYS_RCC control the module enable for each HPSYS module. When the corresponding bit of the module is 1, the module registers are accessible, and the module can operate. When the corresponding bit of the module is 0, the module's working clock and bus clock are both disabled, the module stops operating, and the 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 module reset for each HPSYS module. When the module corresponding bit is 1 , the module registers and internal states are reset. When the module corresponding bit is 0 , the module reset is stopped.

2.7 HPSYS_RCC Register

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
[30:29]			RSVD	
[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	NNACC1	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
[23]			RSVD	
[22]	rw	1'h0	EXTDMA	0 - no reset; 1 - reset
[21]	rw	1'h0	SPI2	0 - no reset; 1 - reset
[20]	rw	1'h0	SPI1	0 - no reset; 1 - reset
[19]			RSVD	
[18]	rw	1'h0	BTIM2	0 - no reset; 1 - reset
[17]	rw	1'h0	BTIM1	0 - no reset; 1 - reset
[16]	rw	1'h0	GPTIM2	0 - no reset; 1 - reset
[15]	rw	1'h0	GPTIM1	0 - no reset; 1 - reset
[14]	rw	1'h0	TRNG	0 - no reset; 1 - reset
[13]	rw	1'h0	CRC1	0 - no reset; 1 - reset
[12]	rw	1'h0	AES	0 - no reset; 1 - reset
[11]	rw	1'h0	EFUSEC	0 - no reset; 1 - reset
[10]	rw	1'h0	SYSCFG1	0 - no reset; 1 - reset
[9]			RSVD	
[8]	rw	1'h0	I2S1	0 - no reset; 1 - reset
[7]	rw	1'h0	LCDC1	0 - no reset; 1 - reset
[6]	rw	1'h0	EPIC	0 - no reset; 1 - reset
[5]	rw	1'h0	EZIP1	0 - no reset; 1 - reset
[4]	rw	1'h0	USART2	0 - no reset; 1 - reset
[3]	rw	1'h0	USART1	0 - no reset; 1 - reset
[2]	rw	1'h0	PINMUX1	0 - no reset; 1 - reset
[1]	rw	1'h0	MAILBOX1	0 - no reset; 1 - reset
[0]	rw	1'h0	DMAC1	0 - no reset; 1 - reset
0x04			**RSTR2**	Reset Register 2
[31:26]			RSVD	
[25]	rw	1'h0	I2C4	0 - no reset; 1 - reset
[24:21]			RSVD	
[20]	rw	1'h0	AUDPRC	0 - no reset; 1 - reset
[19]	rw	1'h0	AUDCODEC	0 - no reset; 1 - reset
[18]			RSVD	
[17]	rw	1'h0	CAN1	0 - no reset; 1 - reset
[16]	rw	1'h0	SCI	0 - no reset; 1 - reset
[15]	rw	1'h0	FACC1	0 - no reset; 1 - reset
[14]	rw	1'h0	FFT1	0 - no reset; 1 - reset
[13]			RSVD	
[12]	rw	1'h0	USART3	0 - no reset; 1 - reset
[11:10]			RSVD	
[9]	rw	1'h0	ATIM1	0 - no reset; 1 - reset
[8]	rw	1'h0	I2C3	0 - no reset; 1 - reset
[7]			RSVD	
[6]	rw	1'h0	USBC	0 - no reset; 1 - reset
[5]	rw	1'h0	SDMMC2	0 - no reset; 1 - reset
[4]	rw	1'h0	SDMMC1	0 - no reset; 1 - reset
[3]	rw	1'h0	MPI3	0 - no reset; 1 - reset
[2]	rw	1'h0	MPI2	0 - no reset; 1 - reset
[1]	rw	1'h0	MPI1	0 - no reset; 1 - reset
[0]	rw	1'h0	GPIO1	0 - no reset; 1 - reset
0x08			**ENR1**	Enable Register 1
[31]	rw	1'h0	PTC1	0 - disabled; 1 - enabled
[30:29]			RSVD	
[28]	rw	1'h1	I2C2	0 - disabled; 1 - enabled

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[27]	rw	1'h1	I2C1	0 - disabled; 1 - enabled
[26]	rw	1'h0	PDM2	0 - disabled; 1 - enabled
[25]	rw	1'h0	PDM1	0 - disabled; 1 - enabled
[24]	rw	1'h0	NNACC1	0 - disabled; 1 - enabled
[23]			RSVD	
[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]			RSVD	
[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	CRC1	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]			RSVD	
[8]	rw	1'h0	I2S1	0 - disabled; 1 - enabled
[7]	rw	1'h1	LCDC1	0 - disabled; 1 - enabled
[6]	rw	1'h0	EPIC	0 - disabled; 1 - enabled
[5]	rw	1'h0	EZIP1	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
[31:26]			RSVD	
[25]	rw	1'h0	I2C4	0 - disabled; 1 - enabled
[24:21]			RSVD	
[20]	rw	1'h0	AUDPRC	0 - disabled; 1 - enabled
[19]	rw	1'h0	AUDCODEC	0 - disabled; 1 - enabled
[18]			RSVD	
[17]	rw	1'h0	CAN1	0 - disabled; 1 - enabled
[16]	rw	1'h0	SCI	0 - disabled; 1 - enabled
[15]	rw	1'h0	FACC1	0 - disabled; 1 - enabled
[14]	rw	1'h0	FFT1	0 - disabled; 1 - enabled
[13]			RSVD	
[12]	rw	1'h1	USART3	0 - disabled; 1 - enabled
[11:10]			RSVD	
[9]	rw	1'h0	ATIM1	0 - disabled; 1 - enabled
[8]	rw	1'h1	I2C3	0 - disabled; 1 - enabled
[7]			RSVD	
[6]	rw	1'h0	USBC	0 - disabled; 1 - enabled
[5]	rw	1'h0	SDMMC2	0 - disabled; 1 - enabled
[4]	rw	1'h0	SDMMC1	0 - disabled; 1 - enabled
[3]	rw	1'h1	MPI3	0 - disabled; 1 - enabled
[2]	rw	1'h1	MPI2	0 - disabled; 1 - enabled
[1]	rw	1'h1	MPI1	0 - disabled; 1 - enabled
[0]	rw	1'h1	GPIO1	0 - disabled; 1 - enabled

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x10			**CSR**	Clock Select Register
[31:18]			RSVD	
[17:16]	rw	2'h0	SEL_SDMMC	select SDMMC1 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[15]	rw	1'b0	SEL_USBC	select USB source clock 0 - clk_hpsys; 1 - clk_dll3
[14:13]			RSVD	
[12]	rw	1'h0	SEL_PERI	select clk_peri_hpsys source used by USART/SPI/CAN/I2C 0 - clk_hrc48; 1 - clk_hxt48
[11:10]			RSVD	
[9:8]	rw	2'h0	SEL_MPI3	selet MPI3 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[7:6]	rw	2'h0	SEL_MPI2	selet MPI2 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[5:4]	rw	2'h0	SEL_MPI1	selet MPI1 function clock 0 - clk_hpsys; 1 - reserved; 2 - clk_dll2; 3 - clk_dll3
[3]			RSVD	
[2]	rw	1'h0	SEL_SYS_LP	select clk_hpsys source 0 - selected by SEL_SYS; 1 - clk_lp
[1:0]	rw	2'h0	SEL_SYS	select clk_hpsys source 0 - clk_hrc48; 1 - clk_hxt48; 2 - reserved; 3 - clk_dll1
0x14			**CFGR**	Clock Configuration Register
[31:15]			RSVD	
[14:12]	rw	3'b111	PDIV2	pclk2_hpsys = hclk_hpsys / (2^{PDIV2}), by default divided by 128
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk_hpsys = hclk_hpsys / (2^{PDIV1}), by default divided by 2
[7:0]	rw	8'h1	HDIV	hclk_hpsys = clk_hpsys / HDIV if HDIV=0, hclk_hpsys = clk_hpsys
0x18			**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.
0x1C			**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 (STG+1)*24MHz if OUT_DIV2_EN=0 e.g. STG=9,DLL output is 240M
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x20			**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 (STG+1)*24MHz if OUT_DIV2_EN=0 e.g. STG=9,DLL output is 240M
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x24			**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	
[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 (STG+1)*24MHz if OUT_DIV2_EN=0 e.g. STG=9,DLL output is 240M
[1]	rw	1'b0	SW	
[0]	rw	1'b0	EN	0: dll disabled 1: dll enabled
0x28			**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
0x2C			**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
0x30			**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
[1:0]	rw	2'b0	CLK_SEL	for debug only
0x34			**DBGR**	Debug Register
[31:4]			RSVD	
[3]	rw	1'h0	FORCE_GPIO	for debug only
[2]	rw	1'h0	FORCE_BUS	for debug only
[1]	rw	1'h0	SYSCLK_SWLP	for debug only
[0]	rw	1'h0	SYSCLK_AON	for debug only
0x38			**DWCFGR**	Deep WFI mode Clock Configuration Register
[31:30]			RSVD	
[29]	rw	1'b0	DLL3_OUT_RSTB	for debug only

Continued on next page...

Table 2-4: HPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[28]	rw	1'b0	DLL3_OUT_EN	for debug only
[27]	rw	1'b0	DLL2_OUT_RSTB	for debug only
[26]	rw	1'b0	DLL2_OUT_EN	for debug only
[25]	rw	1'b0	DLL1_OUT_RSTB	for debug only
[24]	rw	1'b0	DLL1_OUT_EN	for debug only
[23:19]			RSVD	
[18]	rw	1'h1	SEL_SYS_LP	select clk_hpsys source during deep WFI 0 - selected by SEL_SYS; 1 - clk_lp
[17:16]	rw	2'h0	SEL_SYS	select clk_hpsys source during deep WFI 0 - clk_hrc48; 1 - clk_hxt48; 2 - reserved; 3 - clk_dll1
[15]	rw	1'h1	DIV_EN	enable PDIV1, PDIV2 and HDIV reconfiguration during deep WFI
[14:12]	rw	3'b001	PDIV2	pclk2_hpsys = hclk_hpsys / (2^{PDIV2}) during deep WFI
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk_hpsys = hclk_hpsys / (2^{PDIV1}) during deep WFI
[7:0]	rw	8'h1	HDIV	hclk_hpsys = clk_hpsys / HDIV during deep WFI

2.8 LPSYS Clock Structure

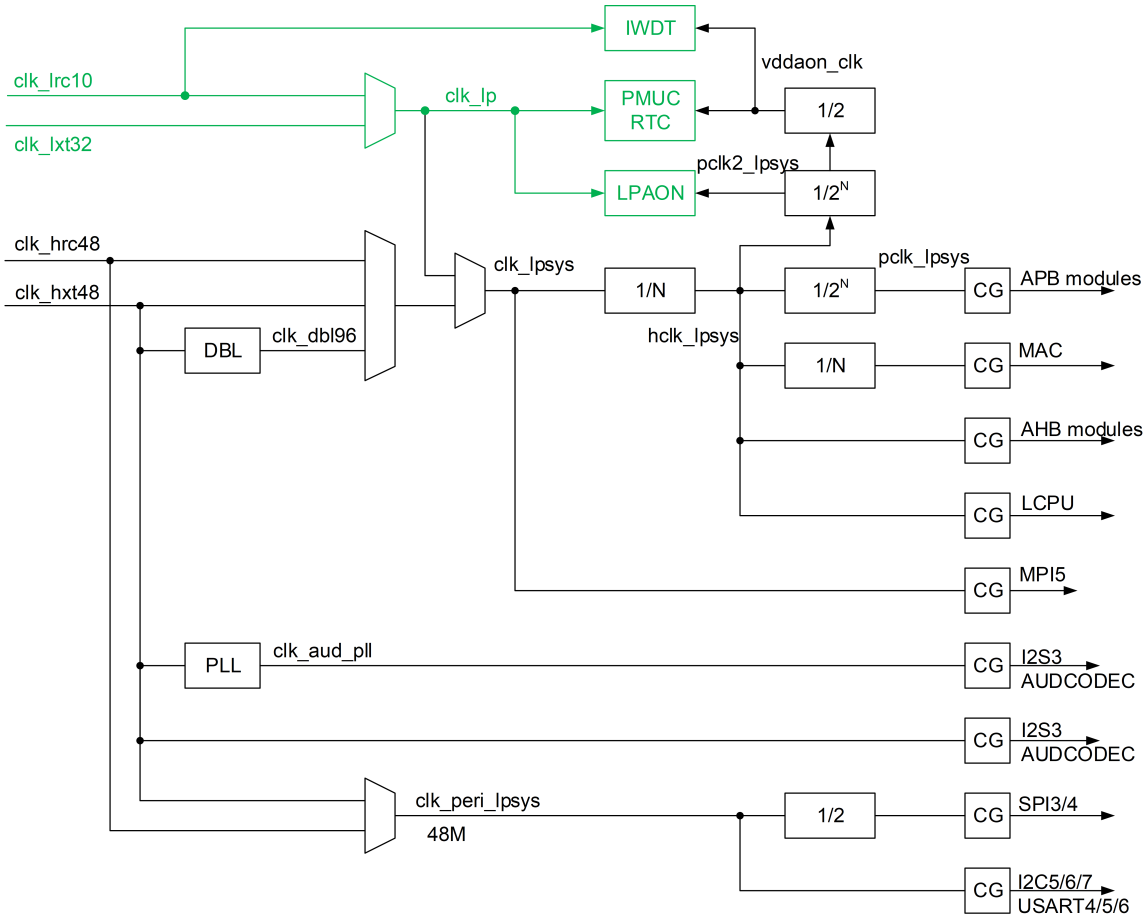


Figure 2-2: LPSYS Clock Structure

The system clock of LPSYS, clk_lpsys, can be selected from clk_hrc48, clk_hxt48, and clk_dbl96. The selection register is CSR_SEL_SYS in LPSYS_RCC . The maximum supported frequency of clk_lpsys is 48 MHz (base mode) or 96 MHz (enhanced mode).

hclk_lpsys is generated by dividing clk_lpsys 1 by N, with the division ratio defined by CFGR_HDIV1 in LPSYS_RCC. The maximum supported frequency of hclk_lpsys is 48 MHz (base mode) or 96 MHz (enhanced mode), serving as the operating clock for LCPU , DMAC2 , and other AHB modules, as well as the AHB bus and SRAM. CFGR_HDIV2 in LPSYS_RCC is used to adjust the wait cycles of SRAM. When the frequency of 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 to generate the clock required by the Bluetooth MAC, with the division ratio defined by CFGR_MACDIV in LPSYS_RCC.

pclk_lpsys is generated by dividing hclk_lpsys 1 by 2^N , with a division ratio of 2^{CFGR_PDIV1} . The maximum supported frequency of pclk_lpsys is 24 MHz (basemode) or 48 MHz(enhanced mode), serving as the operating clock for APB modules such as GPTIM3/4/5, BTIM3/4, GPADC, and also 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, GPADC also changes accordingly, functionality will be affected. Therefore, during the operation of related modules, the frequencies of hclk_lpsys and pclk_lpsys must remain constant.

pclk2_lpsys is generated by dividing hclk_lpsys by 2^N, with the division ratio defined by 2^{CFGR_PDIV2} . pclk2_lpsys supports a maximum frequency of 6 MHz and functions 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 LPAON, PMUC, and RTC, as well as the sleep clock for Bluetooth. The clk_lp select register is the CR_SEL_LPCLK in the PMUC module.

The working clock of IWDT is fixed to clk_lrc10 and is not affected by CR_SEL_LPCLK.

The working clock of MPI5 is generated by clk_lpsys.

The working clock clk_peri_lpsys for peripherals such as USART4/5/6 and I2C5/6/7 can be selected between clk_hrc48 and clk_hxt48 at a frequency of 48 MHz; the selection register is CSR_SEL_PERI in LPSYS_RCC. The working clock of SPI3/4 is the clk_peri_lpsys divided by two, which is 24 MHz. clk_peri_lpsys is independent of the system clock and is therefore unaffected during dynamic frequency adjustments of the system.

The audio module AUDCODEC_LP(ADC) can select one clock source from two working clocks; the selection register is located within the module. One working clock is clk_hxt48 , and the other is clk_audpll .

2.9 LPSYS Module Enable

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

2.10 LPSYS Module Reset

The RSTR1 and RSTR2 registers in LPSYS_RCC control the reset of each module. 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 disabled.

2.11 LPSYS_RCC Registers

Table 2-5: LPSYS_RCC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**RSTR1**	Reset Register 1
[31:30]			RSVD	
[29]	rw	1'h0	BUSMON2	0 - no reset; 1 - reset
[28]			RSVD	
[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]	rw	1'h0	AUDCODEC	0 - no reset; 1 - reset
[23]			RSVD	
[22]	rw	1'h0	GPADC	0 - no reset; 1 - reset
[21]			RSVD	
[20]	rw	1'h0	BTIM4	0 - no reset; 1 - reset
[19]	rw	1'h0	BTIM3	0 - no reset; 1 - reset
[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	I2C7	0 - no reset; 1 - reset
[13]	rw	1'h0	I2C6	0 - no reset; 1 - reset
[12]	rw	1'h0	I2C5	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	USART6	0 - no reset; 1 - reset
[6]	rw	1'h0	USART5	0 - no reset; 1 - reset
[5]	rw	1'h0	USART4	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 Register 2
[31:6]			RSVD	
[5]	rw	1'h0	CRC2	0 - no reset; 1 - reset
[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	MPI5	0 - no reset; 1 - reset
[0]	rw	1'h0	GPIO2	0 - no reset; 1 - reset
0x08			**ENR1**	Enable Register 1

Continued on next page...

Table 2-5: LPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:30]			RSVD	
[29]	rw	1'h0	BUSMON2	0 - disabled; 1 - enabled
[28]			RSVD	
[27]	rw	1'h0	PTC2	0 - disabled; 1 - enabled
[26]	rw	1'h1	TSEN	0 - disabled; 1 - enabled
[25]	rw	1'h1	LPCOMP	0 - disabled; 1 - enabled
[24]	rw	1'h1	AUDCODEC	0 - disabled; 1 - enabled
[23]			RSVD	
[22]	rw	1'h1	GPADC	0 - disabled; 1 - enabled
[21]			RSVD	
[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	I2C7	0 - disabled; 1 - enabled
[13]	rw	1'h1	I2C6	0 - disabled; 1 - enabled
[12]	rw	1'h1	I2C5	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	USART6	0 - disabled; 1 - enabled
[6]	rw	1'h1	USART5	0 - disabled; 1 - enabled
[5]	rw	1'h1	USART4	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 Register 2
[31:6]			RSVD	
[5]	rw	1'h1	CRC2	0 - disabled; 1 - enabled
[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	MPI5	0 - disabled; 1 - enabled
[0]	rw	1'h1	GPIO2	0 - disabled; 1 - enabled
0x10			**CSR**	Clock Select Register
[31:5]			RSVD	
[4]	rw	1'h0	SEL_PERI	select clk_peri_lpsys source used by USART/SPI/I2C 0 - clk_hrc48; 1 - clk_hxt48
[3]			RSVD	
[2]	rw	1'h0	SEL_SYS_LP	select clk_lpsys source 0 - selected by SEL_SYS; 1 - clk_lp
[1:0]	rw	2'h0	SEL_SYS	select clk_lpsys source 0 - clk_hrc48; 1 - clk_hxt48; 2 - clk_dbl96; 3 - reserved
0x14			**CFGR**	Clock Configuration Register
[31:25]			RSVD	
[24:20]	rw	5'h8	MACFREQ	clock frequency of MAC clock

Continued on next page...

Table 2-5: LPSYS_RCC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19:16]	rw	4'h3	MACDIV	MAC clock divider MACCLK = hclk_lpsys / MACDIV
[15]			RSVD	
[14:12]	rw	3'b101	PDIV2	pclk2_lpsys = hclk_lpsys / (2^{PDIV2}), by default divided by 32
[11]			RSVD	
[10:8]	rw	3'b001	PDIV1	pclk_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
0x20			**DBGR**	Debug Register
[31:6]			RSVD	
[5]	rw	1'h0	SYSCLK_SWBT	If set to 1, clk_lpsys will: switch from clk_hrc48 to clk_hxt48 when MAC active; switch from clk_hxt48 to clk_hrc48 when MAC sleep;
[4]	rw	1'h0	FORCE_GPIO	for debug only
[3]	rw	1'h0	FORCE_MAC	for debug only
[2]	rw	1'h0	FORCE_BUS	for debug only
[1]	rw	1'h0	SYSCLK_SWLP	for debug only
[0]	rw	1'h0	SYSCLK_AON	for debug only

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 requirements, 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 PA49 pin for direct measurement. The configuration method is:
   - HPSYS_PINMUX->PAD_PA49.FSEL = 9;
   - HPSYS_RCC->DBGCLKR = 5;
   - This measurement method is difficult to achieve high accuracy.
2. Measure the frequency of the RF carrier signal transmitted by Bluetooth and calculate the frequency of clk_hxt48.
3. 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.
4. 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_hrc48 through the PA49 pin for direct measurement. The configuration method is:
   - HPSYS_PINMUX->PAD_PA49.FSEL = 9;
   - HPSYS_RCC->DBGCLKR = 4;
2. 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 PB04/PB05/PB22/PB25/PB26 for direct measurement. The configuration method (taking PB04 as an example) is:
   - HPSYS_PINMUX->PAD_PB04=0x6; //Set PB04/05 to 6, and PB22/25/26 to 5.
   - LPSYS_CFG->DBGR=0x08000000;
   - RTC->CR_LPCKSEL = 1; // Select clk_lxt32 as clk_rtc
2. Lead out clk_lxt32 through pins PBR01~PBR03 for direct measurement. The configuration method (taking PBR01 as an example) is:
   - RTC->PBR1R = 0x1002; // SEL=1, OE=1
   - RTC->CR_LPCKSEL = 1; // Select clk_rtc as clk_lxt3
   - The clk_lxt32 output via this method remains active even when the chip enters any low-power mode except hibernate mode. It can be used to provide a 32.768 kHz clock signal to external devices.
3. 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_lrc10 through pins PB04/PB05/PB22/PB25/PB26 for direct measurement. The configuration method (taking PB04 as an example) is:
   - HPSYS_PINMUX->PAD_PA04 = 0x6; // Set PB04/05 to 6， and PB22/25/26 to 5
   - LPSYS_CFG->DBGR=0x08000000;
   - RTC->CR_LPCKSEL = 0; // Select clk_lrc10 as clk_rtc
2. Lead out clk_lrc32 through pins PBR01~PBR03 for direct measurement. The configuration method (taking PBR01 as an example) is:
   - RTC->PBR1R = 0x1002; // SEL=1, OE=1
   - RTC->CR_LPCKSEL = 0; // Select clk_rtc as clk_lrc10
3. 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.

2.12.5 Measurement of clk_dbl96

clk_dbl96 is a fixed-frequency 96 MHz clock derived from clk_hxt48, and its accuracy depends on the accuracy of clk_hxt48.

The frequency of clk_dbl96 cannot be adjusted. It can be measured using the following method:

- Route clk_dbl96 to one of the pins PB04, PB05, PB22, PB25, or PB26 for direct measurement. Configuration example (using PB04):
  - LPSYS_PINMUX->PAD_PB04 = 0x6; // Set PB04/PB05 to 0x6; set PB22/PB25/PB26 to 0x5
  - LPSYS_CFG->DBGR = 0x09000000;

3 Power Management

3.1 Introduction

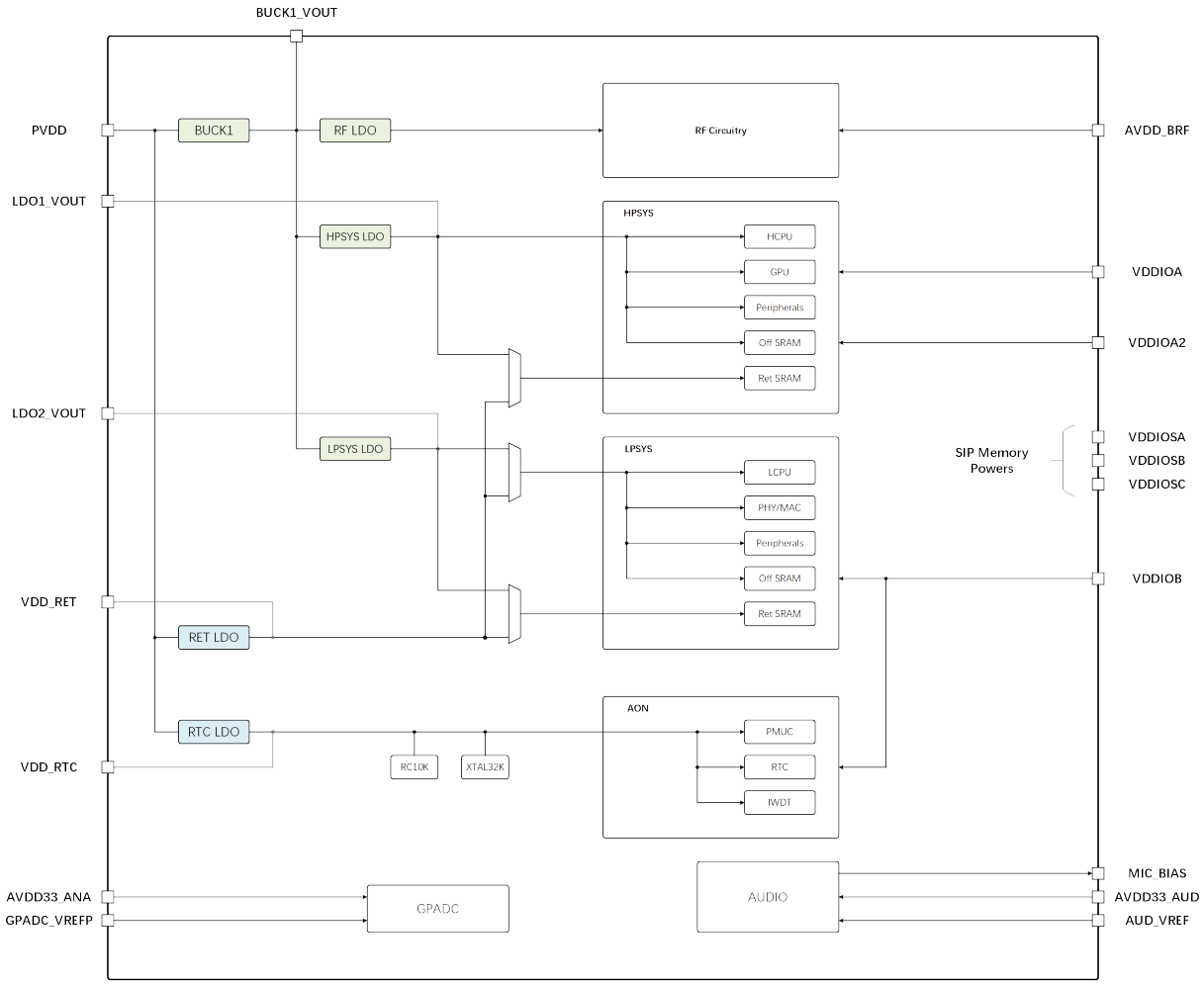


Figure 3-1: BGA Package Power Management Architecture

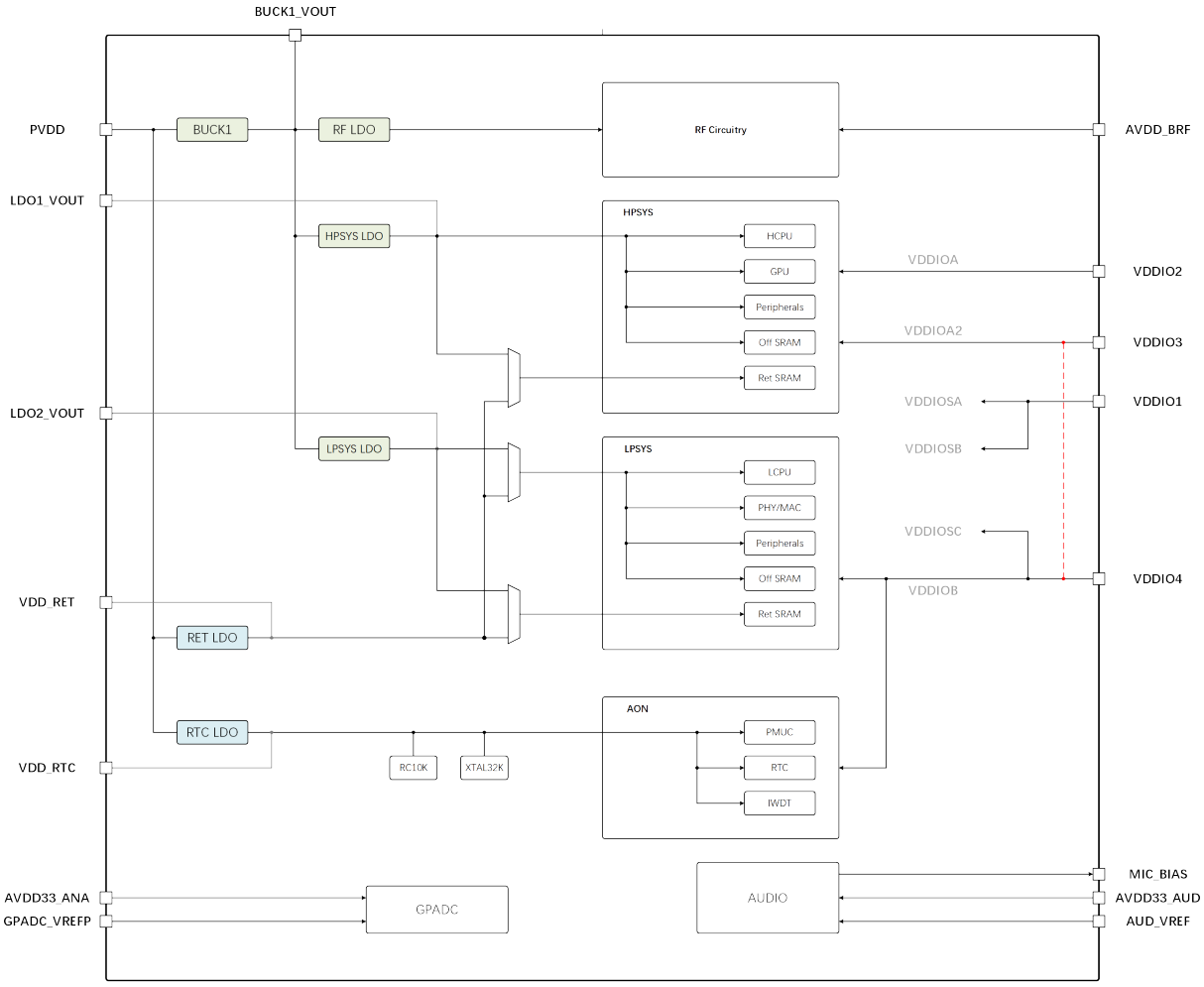


Figure 3-2: QFN Package Power Management Architecture

Notes:

Compared to the BGA package, the QFN package consolidates certain IO power and SIP Memory power domains, specifically:

- SF32LB561/563 : VDDIOSA and VDDIOSB are merged into VDDIO1, VDDIOSC and VDDIOB are merged into VDDIO4
- SF32LB560 : The aforementioned VDDIO4 is further merged with VDDIO3 into VDDIO3 (red dashed line)

3.2 PMUC Registers

Table 3-1: PMUC Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31:18]			RSVD	
[17:14]	rw	4'h1	PIN1_SEL	
[13:10]	rw	4'h0	PIN0_SEL	select one out of 14 Pins PBR[3:0], PA[4:0], PB[4:0]
[9:7]	rw	3'h0	PIN1_MODE	0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge
[6:4]	rw	3'h0	PIN0_MODE	4/6 - both edge (high-active detection), 5/7 - both edge (low-active detection)

Continued on next page...

Table 3-1: **PMUC Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[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
[0]	rw	1'h0	SEL_LPCLK	0 - LRC10, 1 - LXT32
0x04			**WER**	Wakeup Enable register
[31:5]			RSVD	
[4]	rw	1'b0	PIN1	Set 1 to enable PIN1 as wakeup source
[3]	rw	1'b0	PIN0	Set 1 to enable PIN0 as wakeup source
[2]	rw	1'b0	WDT2	Set 1 to enable WDT2 as reboot cause
[1]	rw	1'b0	WDT1	Set 1 to enable WDT1 as reboot cause
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x08			**WSR**	Wakeup Status register
[31:7]			RSVD	
[6]	r	1'b0	PWRKEY	
[5]	r	1'b0	IWDT	
[4]	r	1'b0	PIN1	
[3]	r	1'b0	PIN0	
[2]	r	1'b0	WDT2	Indicates reboot by WDT2
[1]	r	1'b0	WDT1	Indicates reboot by WDT1
[0]	r	1'b0	RTC	Indicates the wakeup status from RTC. Note: the status is masked by WER
0x0C			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:7]			RSVD	
[6]	w1c	1'b0	PWRKEY	Write 1 to clear PWRKEY reset flag
[5]			RSVD	
[4]	w1c	1'b0	PIN1	Write 1 to clear PIN1 wakeup flag.
[3]	w1c	1'b0	PIN0	Write 1 to clear PIN0 wakeup flag. Only valid if PIN wakeup is configured as edge trigger
[2]	w1c	1'b0	WDT2	Write 1 to clear WDT2 reboot flag
[1]	w1c	1'b0	WDT1	Write 1 to clear WDT1 reboot flag
[0]			RSVD	
0x10			**VRTC_CR**	VRTC Control Register
[31:13]			RSVD	
[12:9]	rw	4'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	
[0]	rw	1'h1	EN	
0x18			**LRC_CR**	RC10K Control Register
[31:9]			RSVD	

Continued on next page...

Table 3-1: PMUC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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'h0	RSN	
[0]	rw	1'h0	EN	
0x20			**BG1_CR**	BG1 Control Register
[31:24]			RSVD	
[23:20]	rw	4'h6	BG1_VREF_L	low voltage = 0.9V
[19:16]	rw	4'h9	BG1_VREF_M	middle voltage = 1.05V
[15:12]			RSVD	
[11]	rw	1'b0	LDOBG_EN	Force LDOBG enable
[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			**BUCK1_CR1**	BUCK1 Control Register 1
[31]	rw	1'h0	BUCK1_FORCE_RDY	
[30]	rw	1'h0	BUCK1_H2M_EN	
[29]	rw	1'h0	BUCK1_H2L_EN	
[28]	rw	1'h0	BUCK1_M2L_EN	
[27]	rw	1'h0	BUCK1_L2M_EN	
[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	
0x28			**BUCK1_CR2**	BUCK1 Control Register 2
[31:24]			RSVD	
[23:16]	rw	8'h0	L2M_CNT	
[15:8]	rw	8'h0	L2H_CNT	
[7:0]	rw	8'h0	M2H_CNT	
0x2C			**HPSYS_LDO**	HPSYS LDO Control Register
[31:17]			RSVD	
[16]	r	1'b0	RDY	

Continued on next page...

Table 3-1: PMUC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:10]	rw	6'h8	DLY	HPSYS_LDO power up delay in CLK_LP cycles
[9:6]	rw	4'ha	VREF2	Lower voltage for deep sleep mode
[5:2]	rw	4'hb	VREF	
[1]	rw	1'b0	BP	
[0]	rw	1'b1	EN	
0x30			**LPSYS_LDO**	LPSYS LDO Control Register
[31:17]			RSVD	
[16]	r	1'b0	RDY	
[15:10]	rw	6'h8	DLY	LPSYS_LDO power up delay in CLK_LP cycles
[9:6]	rw	4'h5	VREF2	Lower voltage for deep sleep mode
[5:2]	rw	4'h5	VREF	
[1]	rw	1'b0	BP	
[0]	rw	1'b1	EN	
0x34			**HPSYS_SWR**	HPSYS Switch Register
[31]	r	1'b0	RDY	
[30:8]			RSVD	
[7]	rw	1'b0	NORET	Cut off VHPRET entirely during standby. No retention
[6:4]	rw	3'h3	DLY	wait for N cycles before asserting RDY
[3:2]			RSVD	
[1:0]	rw	2'b10	PSW	[0] - RET_LDO; [1] - HPSYS_LDO
0x38			**LPSYS_SWR**	LPSYS Switch Register
[31]	r	1'b0	RDY	
[30:8]			RSVD	
[7]	rw	1'b0	NORET	Cut off VLPMEM entirely during standby. No retention
[6:4]	rw	3'h3	DLY	
[3:2]	rw	2'b01	PSW_RET	PSW value during DS/SB
[1:0]	rw	2'b10	PSW	[0] - RET_LDO; [1] - LPSYS_LDO
0x3C			**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
0x40			**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	
0x44			**HXT_CR1**	HXT48 Control Register 1
[31:30]			RSVD	
[29:20]	rw	10'h1ca	CBANK_SEL	
[19]	rw	1'b1	GM_EN	
[18:17]	rw	2'h3	LDO_FLT_RSEL	
[16:13]	rw	4'ha	LDO_VREF	
[12:11]	rw	2'h1	BUF_RF_STR	
[10:9]	rw	2'h1	BUF_AUD_STR	
[8]	rw	1'b0	BUF_AUD_EN	
[7:6]	rw	2'h1	BUF_DLL_STR	
[5]	rw	1'b0	BUF_DLL_EN	
[4:3]	rw	2'h1	BUF_DIG_STR	
[2]	rw	1'b1	BUF_DIG_EN	
[1]	rw	1'b1	BUF_EN	
[0]	rw	1'b1	EN	

Continued on next page...

Table 3-1: PMUC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x48			**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	
0x4C			**HXT_CR3**	HXT48 Control Register 3
[31]			RSVD	
[30:25]	rw	6'd31	DLY	
[24:23]	rw	2'h1	BUF_OSLO_STR	
[22:21]	rw	2'h1	BUF_DAC_STR	
[20:11]	rw	10'h32	OPT_IDAC	
[10]	rw	1'b0	OPT_IDAC_EN	
[9:0]	rw	10'h2da	OPT_CBANK_SEL	
0x50			**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	
0x54			**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	
[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'b0	OUT_EN	
[0]	rw	1'b0	EN	
0x58			**DBL96_CR**	DBL96 Control Register
[31:29]			RSVD	
[28:18]	rw	11'h1a6	DLY_SEL_EXT	
[17]	rw	1'b0	DLY_SEL_EXT_EN	

Continued on next page...

Table 3-1: PMUC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[16]	rw	1'b0	DLY_EXT_EN	
[15:12]	rw	4'h1	DLY_EN	
[11:8]	rw	4'b0	PH_EN	
[7]	rw	1'b0	LOOP_RSTB	
[6]	rw	1'b0	TOOSLO_EN	
[5]	rw	1'b0	TORF_EN	
[4:3]	rw	2'h1	TODIG_STR	
[2]	rw	1'b0	TODIG_EN	
[1]	rw	1'b0	OUT_EN	
[0]	rw	1'b0	EN	
0x5C			**DBL96_CALR**	DBL96 Calibration Register
[31:14]			RSVD	
[13]	r	1'h0	CAL_LOCK	
[12:2]	r	11'h0	CAL_OP	
[1]	rw	1'b0	CAL_CLOSE_EXT_EN	
[0]	rw	1'b0	CAL_EN	
0x60			**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	
0x64			**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	
0x68			**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	
0x6C			**WKUP_CNT**	Wakeup Count Register
[31:16]	rw	16'hffff	PIN1_CNT	
[15:0]	rw	16'hffff	PIN0_CNT	

4 Low-power Mode

4.1 Introduction

The chip supports multiple low-power modes to satisfy low-power requirements across various scenarios. Based on the task division between the high-performance processor and the low-power processor, as well as the requirements for power consumption and wake-up latency, HPSYS and LPSYS can be configured into distinct low-power modes 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	Subsystem	CPU	Peripheral	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 Retained	Level Hold	run	RTC, Wake-up PIN, IO(PA),LPTIM1, LPSYS, MAILBOX2	~250us
	LPSYS	stop	stop	Inaccessible, Fully Retained	Level Hold	run	RTC, Wake-up PIN IO(PB),LPTIM2, HPSYS, MAILBOX1, Bluetooth, LPCOMP	~250us
Standby	HPSYS	reset	reset	Inaccessible, Retained 64KB	Level Hold	run	RTC, Wake-up PIN, LPTIM1,LPSYS, MAILBOX2	~1.5ms
	LPSYS	reset	reset	Inaccessible, Fully Retained	Level Hold	run	RTC, Wake-up PIN, Bluetooth, LPTIM2, HPSYS, MAILBOX1	~1.5ms
Hibernate		reset	reset	Inaccessible, Not reserved	High impedance (excluding wake-up PIN)	reset	RTC,wake-up 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 MPI13	HP_AHB HP_APB	LP_RAM MPI5	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 enter Active mode. In Active mode, the CPU and peripherals within the subsystem operate normally, SRAM is accessible, IO can toggle normally, and resources can be accessed by other subsystems. To reduce runtime power consumption, appropriate clock sources and clock frequencies can be selected for the CPU and peripherals, and peripheral module enable should be activated only when the peripherals are 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 sleep mode, the CPU within the subsystem suspends 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, the HCPU can execute the WFI or WFE instruction to place HPSYS into 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. LCPU exits sleep mode upon receiving any enabled interrupt request 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. Most clocks in the subsystem during deepsleep mode are disabled, retaining only the low-power clock. Both the CPU and peripherals cease operation. SRAM is inaccessible , but its contents are preserved. IOs configured as output (except for PBR) cannot be toggled, maintaining the level state prior to entering deepsleep mode. Accessing subsystems in deepsleep mode by other subsystems will cause bus errors.

The procedure for HPSYSto enter deepsleepmode is as follows:

1. Ensure that all tasks of HPSYS peripherals (except for LPTIM1) have been completed;
2. HCPU processes all currently reported interrupts;
3. HCPU disables all interrupt enables via the NVICregister to prevent exceptions such as SysTick, SVCall,and PendSV,and sets the PRIMASK to 1;
4. Switch the clk_hpsys clock to clk_hrc48 and disable high-speed clocks such as clk_dll1/2/3;
5. At Set the wake-up source inHPSYS_AON;
6. Set ISSR_HP_ACTIVE of HPSYS_AON to 0;
7. Set configure the PMR_MODE register of HPSYS_AON to 2;
8. HCPU executes the WFIinstruction.

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

1. Ensure that all peripheral tasks of LPSYS (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 to 1；

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

After executing the above process, if interrupts remain uncleared causing interrupt or exception requests, the current entry into deepsleep mode fails, and the CPU will remain at in active mode and continue running. If this occurs, you can 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, IO (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, IO(PB), LPTIM2, Bluetooth, LPCOMP, 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 (approximately 250 µs), returning to active mode. CPU, peripherals, and memory states are preserved; the CPU resumes execution from the position prior to entering deepsleep mode (after the WFI instruction).

Upon exiting deepsleepmode, the subsystem generates an AONinterrupt.The CPUshould first execute the corresponding recovery procedure.

After HPSYS exits deepsleepmode, the recommended recovery procedure forHCPUis as follows:

1. Configure the PMR_MODE register of HPSYS_AON to 0;
2. Enable the AON interrupt via the NVIC register and set 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 the ISSR_HP_ACTIVE bit 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 the LCPU is:

1. Configure the PMR_MODEregister of LPSYS_AONto 0;
2. Enable the AON interrupt via the NVIC register and set PRIMASK to 0;
3. At Within the AON interrupt, query the wake-up source via the WSR register of LPSYS_AON and perform the corresponding processing, 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; other wake-up source flags must be cleared by configuring the corresponding modules that generate the wake-up sources.

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

From After exiting deepsleep mode, the register contents of each peripheral 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. Most clocks and power supplies within the subsystem in standby mode are disabled, retaining only the clocks and power supplies for low-power modules.CPU and peripherals are both reset.SRAM is inaccessible.HPSYS can retain 128KB of DTCM and 16KB of dedicated backup storage, while LPSYS can retain all TCM and SRAM. IOs configured as outputs (except PBR) cannot be toggled and maintain the level prior to entering standby mode. Bus errors occur when other subsystems access subsystems in standby mode.

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 SRAM contents and peripheral states in HPSYS DTCM or PSRAM;
3. HCPU handles all currently reported interrupts;
4. HCPU disables all interrupt enables via the NVICregister to prevent exceptions such asSysTick, SVCall,and PendSV,and sets the PRIMASK to 1;
5. Switch the clock clk_hpsys to clk_hrc48;
6. At Set the wake-up source inHPSYS_AON;
7. Set ISSR_HP_ACTIVEofHPSYS_AONto 0;
8. Set the ANACR register of HPSYS_AON to 3;
9. Set 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. 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 the PRIMASK to 1;
5. Clock clk_lpsys switches to clk_hrc48 and disables 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. LCPU executes the WFI instruction.

After executing the above procedure, if interrupts remain unmasked causing interrupt or exception requests, entry into standby mode fails, and theCPU remains in active mode to continue operation. If this occurs, it is possible to retry entering standby mode after completing 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 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, LPTIM1, as well as wake-up requests from LPSYS and MAILBOX2. Wake-up sources are configured via the WER register.

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

When a wake-up source is activated, the subsystem exits standby mode and returns to active mode after an initialization period of approximately 1.5 ms. The status of 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 standbymode, the subsystem generates an AONinterrupt. After the CPUexits the ROMprogram, the corresponding recovery procedure should be executed first.

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

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 the ISSR_HP_ACTIVE bit of HPSYS_AON to 1;
5. Re-enable the high-speed clock;
6. Restore peripheral configuration and SRAM contents from the DTCM or PSRAM of HPSYS;
7. Enable other HCPU interrupts via 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 through 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 SRAM of the LPSYS;
7. Enable other LCPU interrupts via the NVIC registers.

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; other wake-up source flags must be cleared by configuring the corresponding modules that generate the wake-up sources.

From after exiting standby mode, the subsystem 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 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 SRAM or PSRAM with retention capability prior to entering stand by mode; the backed-up content will not be cleared upon entering or exiting standby mode.

4.2.5 Hibernate Mode

The 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 not retained. IOs without wake-up PIN functionality enter a high-impedance state.

The procedure to enter Hibernate mode is as follows:

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

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

Wake-up sources for hibernate mode include the RTC and the wake-up PIN. IOs supporting the wake-up PIN function can maintain the effectiveness of pull-up and pull-down resistors (configuration is located in the RTC register and is therefore retained).

When a wake-up source is triggered, the chip exits hibernate mode after an initialization period (>2ms) and returns to active mode. HCPU executes from ROM0 Execution begins at the address. At this time, the PMUC's CR_HIBER_EN register remains set to 1, allowing the CPU to recognize that this startup is an exit from hibernate mode, perform corresponding processing, and clear CR_HIBER_EN to 0.

From after exiting hibernate mode, the subsystem's system clock is sourced from clk_hrc48; if switching to another clock is necessary, the required clock source must be re-enabled clock source.

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

4.2.6 Cross-Subsystem 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 when accessing each other. The accessed party shall implement bus protection mechanisms, while the access initiator shall follow the pre-wake-up procedure.

The accessed subsystem shall implement bus protection mechanisms when entering or exiting low-power mode. specifically, HPSYS shall set the ISSR_HP_ACTIVE bit of HPSYS_AON to 0 when entering deepsleep or standby modes. Subsequently, all master control modules of LPSYS, including LCPU, will receive bus errors upon accessing any address

space of HP-SYS (including HPSYS_AON). Normal access is only restored after HPSYS exits low-power mode and sets ISSR_HP_ACTIVE to 1. When LPSYS enters deepsleep or standby mode, LCPU shall set LPSYS_AONof ISSR_LP_ACTIVE to 0. Thereafter, any access by HCPU to any address space of LPSYS (including LPSYS_AON), and to any address space of the chip's low-power domains (such as PMUC, RTC, IWDT, etc.) will return a bus error. Normal access is only possible when LPSYS recovers from low-power modeand sets ISSR_LP_ACTIVE to 1. This bus error manifests as a HardFault exception on the CPU. If the above bus protection mechanism is not implemented, inappropriate cross-system access by the main control module may cause unrecoverable exceptions such as bus data errors or bus lockups.

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

Subsystems initiating cross-system access must follow the pre-wake procedure. An exception is that when the chip powers on or exits hibernate mode, both the HPSYS subsystem and the LPSYS subsystem 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 before any cross-system access.

The pre-wake-up procedure for HCPU access to the LPSYS subsystem and low-power domains (such as PMUC, RTC, IWDT, etc.) is as follows:

1. Set 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 low-power domains; access time and frequency are unrestricted. During this period, 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-wakeup procedure for 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; access time and frequency are unrestricted. During this period, HPSYS will not enter deepsleep or standby mode;
5. After access is completed, set ISSR_LP2HP_REQ of LPSYS_AON to 0, allowing HPSYS to enter deepsleep or standby mode.

4.2.7 Cross-System Data Interaction Method

When the HPSYSsubsystem and the LPSYSsubsystem 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 used. system data interaction mechanism.

To support the cross-system data interaction mechanism, the accessed party shall pre-enable the MAILBOX wake-up source by setting WER_LP2HP_IRQ of HPSYS_AON and WER_HP2LP_IRQofLPSYS_AON to 1.

The cross-system data interaction process is as follows:

1. The accessing party prepares the data signaling and stores it in the local system's SRAM;
2. The accessing party configures the local system's MAILBOX module, which issues a cross-system wake-up signal via

MAILBOX, then may enter sleepmode;
3. After the recipient receives the MAILBOX interrupt, it retrieves 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. Thereafter, 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 difculty increases, primarily manifested by the debugger disconnecting or failing to connect. By default, the debugger connects to the HCPU via two IOs, PB13 and PB15. When the register PMR_FORCE_LCPU in LPSYS_AON is set to 1, the connection switches to the LCPU. When the chip undergoes a reboot or exits hibernate mode, causing the LPSYS_AON registers to reset, the debugger will reconnect to the HCPU .

The debugger disconnects and cannot reconnect to the HCPU under the following conditions:

1. The debugger switches to the LCPU;
2. The IO functions of PB13 or PB15 are modifed;
3. LPSYS is in deepsleep or standby mode;
4. The ANACR_PB_ISO register of LPSYS_AON is set to 1 (configured by the 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 sleepmode.HPSYS from After waking up from deepsleep or standby mode and setting ISSR_HP_ACTIVE of HPSYS_AON to 1, the debugger can reconnect to HCPU and access the HPSYSaddress space.

When the debugger is connected to LCPU,the connection will be disconnected and cannot be re-established under the following conditions:

1. The debugger switches to HCPU;
2. The IO functions of PB13 or PB15 are modifed;
3. LPSYS is in deepsleep or standby mode;
4. The ANACR_PB_ISO register of LPSYS_AON is set to 1 (configured by the LCPU when LPSYS enters low-power mode);
5. The chip is in hibernatemode.

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

4.2.9 Determining 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 voltage remains at 0.9V (basic mode) or 1.1V (enhanced mode). When LPSYS is in standby mode, the voltage of LDO2_VOUT cannot be maintained and will gradually decrease to 0V. When the chip enters hibernate mode, LDO1_VOUT, LDO2_VOUT, and VDD_RET all drop to 0V.

Table 4-3: Power pin voltages in low-power modes

HPSYS	LPSYS	LDO1_VOUT	LDO2_VOUT	VDD_RET	VDD_RTC
active/sleep/deepsleep	active/sleep/deepsleep	1.1V	0.9V/1.1V	0.7V	1.1V
	standby	1.1V	0V	0.7V	1.1V
standby	active/sleep/deepsleep	0V	0.9V/1.1V	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 Register

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]	rw	1'b0	FORCE_HCPU	for debug only
[29:2]			RSVD	
[1:0]	rw	2'h0	MODE	Power Mode: 2'h0 - active; 2'h1 - light sleep; 2'h2 - deep sleep; 2'h3 - standby
0x04			**CR1**	Control Register 1
[31]	rw	1'h0	GTIM_EN	Enable global timer
[30:24]			RSVD	
[23:21]	rw	3'h0	PIN7_MODE	mode for wakeup PIN7 (PA52)
[20:18]	rw	3'h0	PIN6_MODE	mode for wakeup PIN6 (PA51)
[17:15]	rw	3'h0	PIN5_MODE	mode for wakeup PIN5 (PA50)
[14:12]	rw	3'h0	PIN4_MODE	mode for wakeup PIN4 (PB36)
[11:9]	rw	3'h0	PIN3_MODE	mode for wakeup PIN3 (PB35)
[8:6]	rw	3'h0	PIN2_MODE	mode for wakeup PIN2 (PB34)
[5:3]	rw	3'h0	PIN1_MODE	mode for wakeup PIN1 (PB33)
[2:0]	rw	3'h0	PIN0_MODE	mode for wakeup PIN0 (PB32) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x08			**CR2**	Control Register 2
[31:18]			RSVD	
[17:15]	rw	3'h0	PIN13_MODE	mode for wakeup PIN13 (PBR3)
[14:12]	rw	3'h0	PIN12_MODE	mode for wakeup PIN12 (PBR2)
[11:9]	rw	3'h0	PIN11_MODE	mode for wakeup PIN11 (PBR1)
[8:6]	rw	3'h0	PIN10_MODE	mode for wakeup PIN10 (PBR0)
[5:3]	rw	3'h0	PIN9_MODE	mode for wakeup PIN9 (PA54)
[2:0]	rw	3'h0	PIN8_MODE	mode for wakeup PIN8 (PA53) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x0C			**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
0x10			**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
0x14			**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
0x18			**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
0x1C			**WER**	Wakeup Enable register
[31:22]			RSVD	
[21]	rw	1'b0	PIN13	Set 1 to enable PBR3 as wakeup source
[20]	rw	1'b0	PIN12	Set 1 to enable PBR2 as wakeup source
[19]	rw	1'b0	PIN11	Set 1 to enable PBR1 as wakeup source
[18]	rw	1'b0	PIN10	Set 1 to enable PBR0 as wakeup source
[17]	rw	1'b0	PIN9	Set 1 to enable PA54 as wakeup source
[16]	rw	1'b0	PIN8	Set 1 to enable PA53 as wakeup source
[15]	rw	1'b0	PIN7	Set 1 to enable PA52 as wakeup source
[14]	rw	1'b0	PIN6	Set 1 to enable PA51 as wakeup source
[13]	rw	1'b0	PIN5	Set 1 to enable PA50 as wakeup source
[12]	rw	1'b0	PIN4	Set 1 to enable PB36 as wakeup source
[11]	rw	1'b0	PIN3	Set 1 to enable PB35 as wakeup source
[10]	rw	1'b0	PIN2	Set 1 to enable PB34 as wakeup source
[9]	rw	1'b0	PIN1	Set 1 to enable PB33 as wakeup source
[8]	rw	1'b0	PIN0	Set 1 to enable PB32 as wakeup source
[7]	rw	1'b0	LP2HP_IRQ	Set 1 to enable MAILBOX2 as wakeup source
[6]	rw	1'b0	LP2HP_REQ	Set 1 to enable LPSYS request as wakeup source
[5:3]			RSVD	
[2]	rw	1'b0	LPTIM1	Set 1 to enable LPTIM1 as wakeup source
[1]	rw	1'b0	GPIO1	Set 1 to enable IO(PA) as wakeup source
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x20			**WSR**	Wakeup Status register
[31:22]			RSVD	
[21]	r	1'b0	PIN13	Indicates the wakeup status from PBR3 request. Note: the status is masked by WER
[20]	r	1'b0	PIN12	Indicates the wakeup status from PBR2 request. Note: the status is masked by WER
[19]	r	1'b0	PIN11	Indicates the wakeup status from PBR1 request. Note: the status is masked by WER
[18]	r	1'b0	PIN10	Indicates the wakeup status from PBR0 request. Note: the status is masked by WER
[17]	r	1'b0	PIN9	Indicates the wakeup status from PA54 request. Note: the status is masked by WER
[16]	r	1'b0	PIN8	Indicates the wakeup status from PA53 request. Note: the status is masked by WER
[15]	r	1'b0	PIN7	Indicates the wakeup status from PA52 request. Note: the status is masked by WER
[14]	r	1'b0	PIN6	Indicates the wakeup status from PA51 request. Note: the status is masked by WER
[13]	r	1'b0	PIN5	Indicates the wakeup status from PA50 request. Note: the status is masked by WER
[12]	r	1'b0	PIN4	Indicates the wakeup status from PB36 request. Note: the status is masked by WER
[11]	r	1'b0	PIN3	Indicates the wakeup status from PB35 request. Note: the status is masked by WER

Continued on next page...

Table 4-4: HPSYS_AON Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	r	1'b0	PIN2	Indicates the wakeup status from PB34 request. Note: the status is masked by WER
[9]	r	1'b0	PIN1	Indicates the wakeup status from PB33 request. Note: the status is masked by WER
[8]	r	1'b0	PIN0	Indicates the wakeup status from PB32 request. Note: the status is masked by WER
[7]	r	1'b0	LP2HP_IRQ	Indicates the wakeup status from MAILBOX2. Note: the status is masked by WER
[6]	r	1'b0	LP2HP_REQ	Indicates the wakeup status from LPSYS request. Note: the status is masked by WER
[5:3]			RSVD	
[2]	r	1'b0	LPTIM1	Indicates the wakeup status from LPTIM1. Note: the status is masked by WER
[1]	r	1'b0	GPIO1	Indicates the wakeup status from IO(PA). Note: the status is masked by WER
[0]	r	1'b0	RTC	Indicates the wakeup status from RTC. Note: the status is masked by WER
0x24			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:22]			RSVD	
[21]	w1c	1'b0	PIN13	Write 1 to clear PBR3 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[20]	w1c	1'b0	PIN12	Write 1 to clear PBR2 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[19]	w1c	1'b0	PIN11	Write 1 to clear PBR1 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[18]	w1c	1'b0	PIN10	Write 1 to clear PBR0 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[17]	w1c	1'b0	PIN9	Write 1 to clear PA54 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[16]	w1c	1'b0	PIN8	Write 1 to clear PA53 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[15]	w1c	1'b0	PIN7	Write 1 to clear PA52 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[14]	w1c	1'b0	PIN6	Write 1 to clear PA51 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[13]	w1c	1'b0	PIN5	Write 1 to clear PA50 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[12]	w1c	1'b0	PIN4	Write 1 to clear PB36 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[11]	w1c	1'b0	PIN3	Write 1 to clear PB35 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[10]	w1c	1'b0	PIN2	Write 1 to clear PB34 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[9]	w1c	1'b0	PIN1	Write 1 to clear PB33 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[8]	w1c	1'b0	PIN0	Write 1 to clear PB32 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[7:0]			RSVD	
0x28			**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
0x2C			**ANACR**	Analog Control Register
[31:2]			RSVD	
[1]	rw	1'b0	VHP_ISO	Set 1 to force off all HPSYS related analog modules
[0]	rw	1'b0	PA_ISO	Set 1 to force IO(PA) into retention mode
0x30			**GTIMR**	Global Timer Register
[31:0]	r	32'h0	CNT	Global timer value
0x34			**RESERVE0**	Reserved Register 0
[31:0]	rw	32'b0	DATA	for debug only
0x38			**RESERVE1**	Reserved Register 1
[31:0]	rw	32'b0	DATA	for debug only

4.4 LPSYS_AON Register

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'b0	FORCE_LCPU	0: SWD connected to HCPU 1: SWD connected to LCPU
[29:3]			RSVD	
[2]	rw	1'b1	CPUWAIT	Stall CPU out of reset. Should be cleared before LCPU run
[1:0]	rw	2'h0	MODE	Power Mode: 2'h0 - active; 2'h1 - light sleep; 2'h2 - deep sleep; 2'h3 - standby
0x04			**CR1**	Control Register 1
[31]	rw	1'h0	GTIM_EN	Enable global timer
[30:28]	rw	3'h0	PBR_SEL1	for debug only
[27:25]	rw	3'h0	PBR_SEL0	for debug only
[24]			RSVD	
[23:21]	rw	3'h0	PIN7_MODE	mode for wakeup PIN7 (PA52)
[20:18]	rw	3'h0	PIN6_MODE	mode for wakeup PIN6 (PA51)
[17:15]	rw	3'h0	PIN5_MODE	mode for wakeup PIN5 (PA50)
[14:12]	rw	3'h0	PIN4_MODE	mode for wakeup PIN4 (PB36)
[11:9]	rw	3'h0	PIN3_MODE	mode for wakeup PIN3 (PB35)
[8:6]	rw	3'h0	PIN2_MODE	mode for wakeup PIN2 (PB34)
[5:3]	rw	3'h0	PIN1_MODE	mode for wakeup PIN1 (PB33)
[2:0]	rw	3'h0	PIN0_MODE	mode for wakeup PIN0 (PB32) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x08			**CR2**	Control Register 2
[31:18]			RSVD	
[17:15]	rw	3'h0	PIN13_MODE	mode for wakeup PIN13 (PBR3)
[14:12]	rw	3'h0	PIN12_MODE	mode for wakeup PIN12 (PBR2)
[11:9]	rw	3'h0	PIN11_MODE	mode for wakeup PIN11 (PBR1)
[8:6]	rw	3'h0	PIN10_MODE	mode for wakeup PIN10 (PBR0)
[5:3]	rw	3'h0	PIN9_MODE	mode for wakeup PIN9 (PA54)
[2:0]	rw	3'h0	PIN8_MODE	mode for wakeup PIN8 (PA53) 0 - high level, 1 - low level, 2 - pos edge, 3 - neg edge, 4/5/6/7: pos or neg edge
0x0C			**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
[29:11]			RSVD	

Continued on next page...

Table 4-5: LPSYS_AON Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	rw	1'b0	DS_DTCM	for debug only
[9]	rw	1'b0	DS_ITCM	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	
0x10			**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	
0x14			**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	
0x18			**SBCR**	Standby Mode Ctrl Register
[31:11]			RSVD	
[10]	rw	1'h0	PD_DTCM	for debug only
[9]	rw	1'h0	PD_ITCM	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	
0x1C			**WER**	Wakeup Enable register
[31:22]			RSVD	
[21]	rw	1'b0	PIN13	Set 1 to enable PBR3 as wakeup source
[20]	rw	1'b0	PIN12	Set 1 to enable PBR2 as wakeup source
[19]	rw	1'b0	PIN11	Set 1 to enable PBR1 as wakeup source
[18]	rw	1'b0	PIN10	Set 1 to enable PBR0 as wakeup source
[17]	rw	1'b0	PIN9	Set 1 to enable PA54 as wakeup source
[16]	rw	1'b0	PIN8	Set 1 to enable PA53 as wakeup source
[15]	rw	1'b0	PIN7	Set 1 to enable PA52 as wakeup source
[14]	rw	1'b0	PIN6	Set 1 to enable PA51 as wakeup source
[13]	rw	1'b0	PIN5	Set 1 to enable PA50 as wakeup source
[12]	rw	1'b0	PIN4	Set 1 to enable PB36 as wakeup source
[11]	rw	1'b0	PIN3	Set 1 to enable PB35 as wakeup source
[10]	rw	1'b0	PIN2	Set 1 to enable PB34 as wakeup source

Continued on next page...

Table 4-5: LPSYS_AON Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'b0	PIN1	Set 1 to enable PB33 as wakeup source
[8]	rw	1'b0	PIN0	Set 1 to enable PB32 as wakeup source
[7]	rw	1'b0	HP2LP_IRQ	Set 1 to enable MAILBOX1 as wakeup source
[6]	rw	1'b0	HP2LP_REQ	Set 1 to enable HPSYS request as wakeup source
[5]	rw	1'b0	BT	Set 1 to enable BT as wakeup source
[4]	rw	1'b0	LPCOMP2	Set 1 to enable LPCOMP2 as wakeup source
[3]	rw	1'b0	LPCOMP1	Set 1 to enable LPCOMP1 as wakeup source
[2]	rw	1'b0	LPTIM2	Set 1 to enable LPTIM2 as wakeup source
[1]	rw	1'b0	GPIO2	Set 1 to enable IO(PB) as wakeup source
[0]	rw	1'b0	RTC	Set 1 to enable RTC as wakeup source
0x20			**WSR**	Wakeup Status register
[31:22]			RSVD	
[21]	r	1'b0	PIN13	Indicates the wakeup status from PBR3 request. Note: the status is masked by WER
[20]	r	1'b0	PIN12	Indicates the wakeup status from PBR2 request. Note: the status is masked by WER
[19]	r	1'b0	PIN11	Indicates the wakeup status from PBR1 request. Note: the status is masked by WER
[18]	r	1'b0	PIN10	Indicates the wakeup status from PBR0 request. Note: the status is masked by WER
[17]	r	1'b0	PIN9	Indicates the wakeup status from PA54 request. Note: the status is masked by WER
[16]	r	1'b0	PIN8	Indicates the wakeup status from PA53 request. Note: the status is masked by WER
[15]	r	1'b0	PIN7	Indicates the wakeup status from PA52 request. Note: the status is masked by WER
[14]	r	1'b0	PIN6	Indicates the wakeup status from PA51 request. Note: the status is masked by WER
[13]	r	1'b0	PIN5	Indicates the wakeup status from PA50 request. Note: the status is masked by WER
[12]	r	1'b0	PIN4	Indicates the wakeup status from PB36 request. Note: the status is masked by WER
[11]	r	1'b0	PIN3	Indicates the wakeup status from PB35 request. Note: the status is masked by WER
[10]	r	1'b0	PIN2	Indicates the wakeup status from PB34 request. Note: the status is masked by WER
[9]	r	1'b0	PIN1	Indicates the wakeup status from PB33 request. Note: the status is masked by WER
[8]	r	1'b0	PIN0	Indicates the wakeup status from PB32 request. Note: the status is masked by WER
[7]	r	1'b0	HP2LP_IRQ	Indicates the wakeup status from MAILBOX1. Note: the status is masked by WER
[6]	r	1'b0	HP2LP_REQ	Indicates the wakeup status from HPSYS request. Note: the status is masked by WER
[5]	r	1'b0	BT	Indicates the wakeup status from BT. Note: the status is masked by WER
[4]	r	1'b0	LPCOMP2	Indicates the wakeup status from LPCOMP2. Note: the status is masked by WER
[3]	r	1'b0	LPCOMP1	Indicates the wakeup status from LPCOMP1. Note: the status is masked by WER
[2]	r	1'b0	LPTIM2	Indicates the wakeup status from LPTIM2. Note: the status is masked by WER
[1]	r	1'b0	GPIO2	Indicates the wakeup status from IO(PB). 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
0x24			**WCR**	Wakeup Clear register
[31]	w1c	1'b0	AON	Write 1 to clear the AON wakeup IRQ status
[30:22]			RSVD	

Continued on next page...

Table 4-5: LPSYS_AON Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21]	w1c	1'b0	PIN13	Write 1 to clear PBR3 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[20]	w1c	1'b0	PIN12	Write 1 to clear PBR2 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[19]	w1c	1'b0	PIN11	Write 1 to clear PBR1 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[18]	w1c	1'b0	PIN10	Write 1 to clear PBR0 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[17]	w1c	1'b0	PIN9	Write 1 to clear PA54 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[16]	w1c	1'b0	PIN8	Write 1 to clear PA53 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[15]	w1c	1'b0	PIN7	Write 1 to clear PA52 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[14]	w1c	1'b0	PIN6	Write 1 to clear PA51 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[13]	w1c	1'b0	PIN5	Write 1 to clear PA50 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[12]	w1c	1'b0	PIN4	Write 1 to clear PB36 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[11]	w1c	1'b0	PIN3	Write 1 to clear PB35 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[10]	w1c	1'b0	PIN2	Write 1 to clear PB34 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[9]	w1c	1'b0	PIN1	Write 1 to clear PB33 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[8]	w1c	1'b0	PIN0	Write 1 to clear PB32 wakeup source. Only valid if PIN wakeup is configured as edge trigger
[7:0]			RSVD	
0x28			**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
0x2C			**TARGET**	BT sleep time target
[31:28]			RSVD	
[27:0]	rw	28'h0	SLEEP_TARGET	bt sleep time target in cycles of clk_lp
0x30			**ACTUAL**	BT actual sleep time
[31:28]			RSVD	
[27:0]	r	28'h0	SLEEP_CNT	bt actual sleep time in cycles of clk_lp. If not woken up by software or external interrupt, sleep_cnt counts up every clk_lp cycle, until reaches sleep_target
0x34			**PRE_WKUP**	time before bt 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**	BT 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

Continued on next page...

Table 4-5: **LPSYS_AON Register map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[1:0]			RSVD	
0x3C			**SLP_CTRL**	BT sleep control
[31:7]			RSVD	
[6]	r	1'h1	BT_WKUP	bt wakeup source. 1 means bt has not enter sleep or has enter wakeup procedure
[5]	r	1'h1	XTAL_REQ	xtal request status. 1 means bt is requiring xtal.
[4]	r	1'h0	SLEEP_STATUS	bt sleep status. 1 means bt is sleeping and sleep_cnt is counting up
[3:2]			RSVD	
[1]	w1s	1'h0	WKUP_REQ	software request to wakeup bt. Will be cleared automatically
[0]	w1s	1'h0	SLEEP_REQ	bt sleep request. Will be cleared automatically
0x40			**ANACR**	Analog Control Register
[31:2]			RSVD	
[1]	rw	1'b0	VLP_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 120 configurable IO pins, including 79 general-purpose IOs (PA00~PA78) located in HPSYS , 37 general-purpose IOs (PB00~PB36) located in LPSYS , and 4 low-power IOs (PBR00~PBR03). The number of usable IO pins varies depending on the chip package. Each general-purpose IO can independently select input or output functions and independently configure drive strength and pull-up/pull-down resistors. When configured as a GPIO function, each general-purpose IO can trigger interrupts based on high or low level or edge transitions, and can wake the system from certain low-power modes.

5.2 IO Structure

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

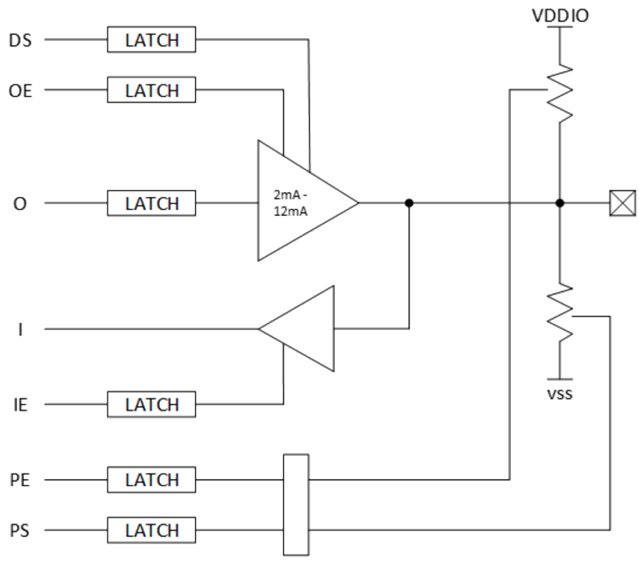


Figure 5-1: IO Structure

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

IO output is determined by O and OE ; different functions are selected via the FSEL register corresponding to the IO in the PINMUX, and automatically mapped to the output of the corresponding function.

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

corresponding to the IO. 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 of the pull-up and pull-down resistors is approximately 10k~40k ohms, depending on the external circuitry and interface level of the IO.

5.3 Input/Output Selection

By configuring the FSEL register of the PINMUX corresponding to the IO, the input/output function of the configurable IO can be mapped to one of several functions. The functions that each IO can map to 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. Regardless of the function to which the IO is mapped, theIO input can be read from the GPIO module and can generate a GPIO interrupt.

When a specific PA is mapped to the PA_I2C_UART function, that IO can serve as an interface signal for any I2C or USART within the HPSYS. The specific interface signal must be designated in the HPSYS_CFG register. For example, to configure PA07 as the TXD for USART2, the PINMUX register's PAD_PA07_FSEL must be set to 4 (corresponding to PA_I2C_UART), and the HPSYS_CFG register's USART2_PINR_TXD_PIN must be set to 7 (corresponding to PA07). It should be noted that assigning multiple interface signals to the same IO will cause functional errors

When a specific PA is mapped to the PA_TIM function, this IO can be used as an interface signal for any GPTIM within the HPSYS. The specific interface signal must be further specifed in the HPSYS_CFG register. For example, to configure PA60 as CH3 of GPTIM2, the PINMUX register's PAD_PA60_FSEL must be set to 5 (corresponding to PA_TIM), and the HPSYS_CFG register's GPTIM2_PINR_CH3_PIN must be set to 60 (corresponding to PA60). It should be noted that assigning multiple interface signals to the same IO will cause functional errors.

When a specific PB is mapped to the PB_I2C_UART function, this IO can serve as an interface signal for any I2C or USART within the LPSYS. The exact interface signal must be specifed in the LPSYS_CFG register. For example, to configure PB28 as the SDA of I2C5, the PAD_PB28_FSEL feld in the PIN-MUX register must be set to 2 (corresponding to PB_I2C_UART), and the I2C5_PINR_SDA_PIN feld in the LPSYS_CFG register must be configured accordingly. For 28 (corresponding to PB28). It should be noted that assigning multiple interface signals to the same IO will cause functional errors.

When a specific PB is mapped to the PB_TIM function, this IO can serve as an interface signal for any GPTIM within the LPSYS. The specific interface signal to be used must be specifed in the LPSYS_CFG register. For example, to use PB12 as the ETR input of GPTIM5, set the PAD_PB12_FSEL feld in the PINMUX register to 3 (corresponding to PB_TIM), and set the ETR_PINR_ETR5_PIN feld in the LPSYS_CFG register to 12 (corresponding to PB12). It should be noted that assigning multiple interface signals to the same IO will cause functional errors.

5.4 IO High Impedance

After chip power-up, IO pins have default pull-up or pull-down resistors (PE default value is 1); software must configure the IO to the required state. To configure the IO as high impedance, set PE to 0, configure the IO as GPIO function, and disable GPIO output enable.

5.5 I2C Mode

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

IOs configurable for I2C mode include:

PA17，PA18，PA48，PA49，PB00，PB01，PB23，PB24。

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 IOs (PA00~PA78) of HPSYS are controlled by HPSYS_GPIO, while the general-purpose IOs (PB00~PB36) of LPSYS are controlled by LPSYS_GPIO. 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 14 corresponds to PA78.

The DOERx register directly controls the OE signal of the IO; when set to 1, the IO output is enabled, and when set to 0, the IO output is disabled.

The software can directly configure the DOERx Register, or configure DOESRx or DOECRx for bit operations to avoid affecting other IOs.

The DORx Register directly controls the O signal of the IO; when set to 1, the output of the IO is high level; when set to 0, the output of the IO is low level. The software can directly configure the DORx Register, or configure DOSRx or DOCRx for bit operations to avoid affecting other IOs.

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 and the O must be set to 0; that is, the corresponding bit of DOERx is configured to 1 and the corresponding bit of DORx is configured to 0

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

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

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

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

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

5.7 GPIO Input

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

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

Even if the IO is not configured as a GPIO function, IO interrupts can still be generated; therefore, when IO interrupts are not required, the interrupt enable must be disabled.

The conditions for generating GPIO interrupts include IO input signals being high level, low level, rising edge, falling edge, or both edges, as shown in the following table.

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 IPHRxregister, or configure IPHSRxor IPHCRxfor 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 register, or configure IPLSRx or IPLCRx for bit operations to avoid affecting other IO.

The IO generating the interrupt can be queried via 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) pin list

Pin Number			Pin Name	Type	Default PU/PD Setting	Sel #	Function
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	R9	PA00	I/O	PD	0	GPIO_A0
						2	WLAN_ACTIVE
						3	ATIM1_CH1
						4	PA_I2C_UART
						5	PA_TIM
						6	SCI_CLK
						Others	Reserved
-	-	P9	PA01	I/O	PD	0	GPIO_A1
						2	BT_ACTIVE
						3	ATIM1_CH1N
						4	PA_I2C_UART
						5	PA_TIM
						6	SCI_DIO
						7	SPI1_DI
						Others	Reserved
-	-	M9	PA02	I/O	PD	0	GPIO_A2
						2	BT_COLLISION
						4	PA_I2C_UART
						5	PA_TIM
						6	SCI_RST
						7	SPI1_CS
						Others	Reserved
-	-	L9	PA03	I/O	PD	0	GPIO_A3
						3	ATIM1_CH2
						4	PA_I2C_UART
						5	PA_TIM
						6	CAN1_TXD
						7	SPI1_DO
						8	SPI1_DIO
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	N8	PA04	I/O	PD	0	GPIO_A4
						2	BT_PRIORITY
						3	ATIM1_CH2N
						4	PA_I2C_UART
						5	PA_TIM
						6	CAN1_RXD
						7	SPI1_CLK
						Others	Reserved
-	26	P8	PA05	I/O	PD	0	GPIO_A5
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
24	24	M8	PA06	I/O	PU	0	GPIO_A6
						1	SD2_DIO2
						2	MPI3_CS
						3	I2S1_MCLK
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
23	23	L7	PA07	I/O	PD	0	GPIO_A7
						1	SD2_DIO3
						2	MPI3_DIO1
						3	I2S1_SDI
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
22	22	R7	PA08	I/O	PD	0	GPIO_A8
						1	SD2_CLK
						2	MPI3_DIO2
						3	I2S1_SDO
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
21	21	P7	PA09	I/O	PU	0	GPIO_A9
						1	SD2_CMD
						2	MPI3_DIO0
						3	I2S1_BCK
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
20	20	J7	PA10	I/O	PU	0	GPIO_A10
						1	SD2_DIO0
						2	MPI3_CLK
						3	I2S1_LRCK
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
19	19	M6	PA11	I/O	PD	0	GPIO_A11
						1	SD2_DIO1
						2	MPI3_DIO3
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	R6	PA12	I/O	PU	0	GPIO_A12
						1	SD1_DIO2
						2	MPI3_CS
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	N5	PA13	I/O	PD	0	GPIO_A13
						1	SD1_DIO6
						2	BT_ACTIVE
						3	ATIM1_CH3
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R1
						Others	Reserved
-	-	M5	PA14	I/O	PD	0	GPIO_A14
						1	SD1_DIO7
						2	WLAN_ACTIVE
						3	ATIM1_CH3N
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R0
						Others	Reserved
-	-	R5	PA15	I/O	PD	0	GPIO_A15
						1	SD1_DIO1
						2	MPI3_DIO3
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	L5	PA16	I/O	PD	0	GPIO_A16
						3	ATIM1_CH4
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R2
						Others	Reserved
17	17	P5	PA17	I/O	PU	0	GPIO_A17
						2	#USB11_DP
						3	ATIM1_BKIN
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
16	16	P4	PA18	I/O	PD	0	GPIO_A18
						2	#USB11_DM
						3	ATIM1_BKIN2
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	P3	PA19	I/O	PD	0	GPIO_A19
						1	SD1_DIO5
						2	BT_PRIORITY
						3	ATIM1_ETR
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R4
						Others	Reserved
-	-	P1	PA20	I/O	PD	0	GPIO_A20
						1	SD1_DIO3
						2	MPI3_DIO1
						3	SPI1_CLK
						4	PA_I2C_UART
						5	PA_TIM
						7	I2S1_LRCK
						8	I/OM1_DATA
						Others	Reserved
-	-	R2	PA21	I/O	PD	0	GPIO_A21
						1	SD1_DIO4
						2	BT_COLLISION
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R5
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	R3	PA22	I/O	PD	0	GPIO_A22
						1	SD1_DIO0
						2	MPI3_CLK
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	P2	PA23	I/O	PU	0	GPIO_A23
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R6
						Others	Reserved
-	-	N4	PA24	I/O	PD	0	GPIO_A24
						2	ATIM1_CH1
						3	SPI1_CS
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R3
						7	I2S1_SDI
						8	I/OM1_CLK
						Others	Reserved
-	-	N3	PA25	I/O	PD	0	GPIO_A25
						1	SD1_CLKIN
						2	ATIM1_CH1N
						3	SPI1_DI
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_R7
						7	I2S1_BCK
						8	I/OM2_DATA
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	N1	PA26	I/O	PD	0	GPIO_A26
						1	SD1_CLK
						2	MPI3_DIO2
						3	SPI1_DO
						4	PA_I2C_UART
						5	PA_TIM
						7	I2S1_SDO
						8	I/OM2_CLK
						Others	Reserved
-	-	N2	PA27	I/O	PU	0	GPIO_A27
						1	SD1_CMD
						2	MPI3_DIO0
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
15	15	M3	PA28	I/O	PU	0	GPIO_A28
						2	SPI2_CS
						3	SWDIO
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G0
						7	LCDC1_JDI_B2
						8	LCDC1_8080_DIO2
						Others	Reserved
-	-	L4	PA29	I/O	PD	0	GPIO_A29
						2	SPI2_DI
						3	ATIM1_CH2
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G5
						7	LCDC1_JDI_R1
						8	LCDC1_8080_DIO3
						Others	Reserved
-	-	M2	PA30	I/O	PD	0	GPIO_A30
						2	SPI2_CLK
						3	ATIM1_CH2N
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G1
						7	LCDC1_JDI_B1
						8	LCDC1_8080_DIO4
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
14	14	K5	PA31	I/O	PD	0	GPIO_A31
						3	SWCLK
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G6
						7	LCDC1_JDI_R2
						8	LCDC1_8080_DIO5
						Others	Reserved
-	-	L3	PA32	I/O	PD	0	GPIO_A32
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G2
						7	LCDC1_JDI_G2
						8	LCDC1_8080_DIO6
						Others	Reserved
13	13	L1	PA33	I/O	PD	0	GPIO_A33
						1	LCDC1_SPI_TE
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G3
						8	LCDC1_8080_TE
						Others	Reserved
-	-	L2	PA34	I/O	PU	0	GPIO_A34
						2	SPI2_DO
						3	SPI2_DIO
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G4
						7	LCDC1_JDI_G1
						8	LCDC1_8080_DIO7
						Others	Reserved
-	-	K1	PA35	I/O	PD	0	GPIO_A35
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_G7
						Others	Reserved
12	12	K4	PA36	I/O	PU	0	GPIO_A36
						1	LCDC1_SPI_CS
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B0
						7	LCDC1_JDI_HCK
						8	LCDC1_8080_CS
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
11	11	J5	PA37	I/O	PD	0	GPIO_A37
						1	LCDC1_SPI_CLK
						2	I2S1_MCLK
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B1
						8	LCDC1_8080_WR
						Others	Reserved
10	10	K2	PA38	I/O	PD	0	GPIO_A38
						1	LCDC1_SPI_DIO0
						2	I2S1_SDI
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B2
						7	LCDC1_JDI_HST
						8	LCDC1_8080_RD
						Others	Reserved
9	9	H5	PA39	I/O	PD	0	GPIO_A39
						1	LCDC1_SPI_DIO1
						2	I2S1_SDO
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B5
						7	LCDC1_JDI_XRST
						8	LCDC1_8080_DC
						Others	Reserved
8	8	H4	PA40	I/O	PD	0	GPIO_A40
						1	LCDC1_SPI_DIO2
						2	I2S1_BCK
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B6
						7	LCDC1_JDI_VST
						8	LCDC1_8080_DIO0
						Others	Reserved
7	7	J2	PA41	I/O	PD	0	GPIO_A41
						1	LCDC1_SPI_DIO3
						2	I2S1_LRCK
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B4
						7	LCDC1_JDI_VCK
						8	LCDC1_8080_DIO1
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	G5	PA42	I/O	PD	0	GPIO_A42
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_VSYNC
						Others	Reserved
-	-	H3	PA43	I/O	PD	0	GPIO_A43
						1	LCDC1_SPI_RSTB
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B3
						7	LCDC1_JDI_ENB
						8	LCDC1_8080_RSTB
						Others	Reserved
-	-	G3	PA44	I/O	PD	0	GPIO_A44
						3	ATIM1_CH3
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_HSYNC
						Others	Reserved
-	-	G2	PA45	I/O	PD	0	GPIO_A45
						3	ATIM1_CH3N
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_CLK
						Others	Reserved
-	-	G1	PA46	I/O	PD	0	GPIO_A46
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_B7
						Others	Reserved
-	-	G4	PA47	I/O	PD	0	GPIO_A47
						3	ATIM1_CH4
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_DE
						Others	Reserved
5	5	F3	PA48	I/O	PU	0	GPIO_A48
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
4	4	F2	PA49	I/O	PU	0	GPIO_A49
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
3	3	E2	PA50	I/O	PD	0	GPIO_A50
						2	#WKUP_PIN5
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_SD
						Others	Reserved
2	2	E1	PA51	I/O	PD	0	GPIO_A51
						2	#WKUP_PIN6
						4	PA_I2C_UART
						5	PA_TIM
						6	LCDC1_DPI_CM
						Others	Reserved
-	-	C3	PA52	I/O	PD	0	GPIO_A52
						2	#WKUP_PIN7
						3	ATIM1_BKIN
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D2	PA53	I/O	PD	0	GPIO_A53
						2	#WKUP_PIN8
						3	ATIM1_BKIN2
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D3	PA54	I/O	PD	0	GPIO_A54
						2	#WKUP_PIN9
						3	ATIM1_ETR
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
61	61	A5	PA55	I/O	-	0	GPIO_A55
						1	#XTAL32K_XI
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
60	60	A6	PA56	I/O	-	0	GPIO_A56
						1	#XTAL32K_XO
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	D6	PA57	I/O	PD	0	GPIO_A57
						1	SPI1_CLK
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D4	PA58	I/O	PU	0	GPIO_A58
						1	SPI1_CS
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	F4	PA59	I/O	PD	0	GPIO_A59
						1	SPI1_DI
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	C6	PA60	I/O	PD	0	GPIO_A60
						3	ATIM1_CH1
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	E6	PA61	I/O	PD	0	GPIO_A61
						1	SPI1_DO
						2	SPI1_DIO
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	C7	PA62	I/O	PU	0	GPIO_A62
						3	ATIM1_CH1N
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D7	PA63	I/O	PU	0	GPIO_A63
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	B7	PA64	I/O	PD	0	GPIO_A64
						1	I2S1_SDO
						2	I/OM1_DATA
						4	PA_I2C_UART
						5	PA_TIM
						6	SPI2_DO
						7	SPI2_DIO
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	A7	PA65	I/O	PD	0	GPIO_A65
						1	I2S1_MCLK
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	E7	PA66	I/O	PD	0	GPIO_A66
						3	ATIM1_CH2
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D8	PA67	I/O	PD	0	GPIO_A67
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	E8	PA68	I/O	PD	0	GPIO_A68
						3	ATIM1_CH2N
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	B8	PA69	I/O	PD	0	GPIO_A69
						1	I2S1_SDI
						2	I/OM1_CLK
						4	PA_I2C_UART
						5	PA_TIM
						6	SPI2_DI
						Others	Reserved
-	-	E9	PA70	I/O	PD	0	GPIO_A70
						3	ATIM1_CH3
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	B9	PA71	I/O	PD	0	GPIO_A71
						1	I2S1_LRCK
						2	I/OM2_DATA
						3	ATIM1_CH3N
						4	PA_I2C_UART
						5	PA_TIM
						6	SPI2_CS
						Others	Reserved
-	-	E10	PA72	I/O	PD	0	GPIO_A72
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	A9	PA73	I/O	PD	0	GPIO_A73
						1	I2S1_BCK
						2	I/OM2_CLK
						3	ATIM1_CH4
						4	PA_I2C_UART
						5	PA_TIM
						6	SPI2_CLK
						Others	Reserved
-	-	E11	PA74	I/O	PD	0	GPIO_A74
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	C11	PA75	I/O	PU	0	GPIO_A75
						3	ATIM1_BKIN
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	D10	PA76	I/O	PU	0	GPIO_A76
						3	ATIM1_BKIN2
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	E12	PA77	I/O	PD	0	GPIO_A77
						3	ATIM1_ETR
						4	PA_I2C_UART
						5	PA_TIM
						Others	Reserved
-	-	C10	PA78	I/O	PD	0	GPIO_A78
						4	PA_I2C_UART
						5	PA_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	L10	PB00	I/O	PU	0	GPIO_B0
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	L11	PB01	I/O	PU	0	GPIO_B1
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	P10	PB02	I/O	PD	0	GPIO_B2
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	M11	PB03	I/O	PD	0	GPIO_B3
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_COLLISION
						Others	Reserved
26	-	H10	PB04	I/O	PD	0	GPIO_B4
						1	TWI_CLK
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
27	-	G10	PB05	I/O	PD	0	GPIO_B5
						1	TWI_DIO
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	P11	PB06	I/O	PD	0	GPIO_B6
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_ACTIVE
						7	LPCOMP1_OUT
						Others	Reserved
-	-	N11	PB07	I/O	PD	0	GPIO_B7
						2	PB_I2C_UART
						3	PB_TIM
						4	WLAN_ACTIVE
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	P13	PB08	I/O	PU	0	GPIO_B8
						1	SPI4_CS
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_PRIORITY
						Others	Reserved
-	-	J12	PB09	I/O	PD	0	GPIO_B9
						1	SPI4_CLK
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	N13	PB10	I/O	PD	0	GPIO_B10
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	K12	PB11	I/O	PD	0	GPIO_B11
						1	SPI4_DI
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	R13	PB12	I/O	PD	0	GPIO_B12
						1	SPI4_DO
						2	PB_I2C_UART
						3	PB_TIM
						4	SPI4_DIO
						Others	Reserved
28	28	R12	PB13	I/O	PD	0	SWDIO
						1	GPIO_B13
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
-	-	N14	PB14	I/O	PU	0	GPIO_B14
						2	PB_I2C_UART
						3	PB_TIM
						Others	Reserved
29	29	P12	PB15	I/O	PD	0	SWCLK
						1	GPIO_B15
						2	PB_I2C_UART
						3	PB_TIM
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
30	30	N12	PB16	I/O	PD	0	GPIO_B16
						2	PB_I2C_UART
						3	PB_TIM
						7	#DCTEST0
						Others	Reserved
31	31	M12	PB17	I/O	PU	0	GPIO_B17
						2	PB_I2C_UART
						3	PB_TIM
						7	#DCTEST1
						Others	Reserved
32	32	P14	PB18	I/O	PU	0	GPIO_B18
						1	SPI3_CS
						2	PB_I2C_UART
						3	PB_TIM
						7	#LPCOMP1_P
						Others	Reserved
33	33	P15	PB19	I/O	PU	0	GPIO_B19
						1	SPI3_CLK
						2	PB_I2C_UART
						3	PB_TIM
						7	#LPCOMP1_N
						Others	Reserved
34	34	P16	PB20	I/O	PD	0	GPIO_B20
						1	SPI3_DI
						2	PB_I2C_UART
						3	PB_TIM
						7	#LPCOMP2_P
						Others	Reserved
35	35	R15	PB21	I/O	PD	0	GPIO_B21
						1	SPI3_DO
						2	PB_I2C_UART
						3	PB_TIM
						4	SPI3_DIO
						7	#LPCOMP2_N
						Others	Reserved
36	36	K13	PB22	I/O	PD	0	GPIO_B22
						2	PB_I2C_UART
						3	PB_TIM
						4	TWI_CLK
						7	#GPADC_CH0
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
37	37	L13	PB23	I/O	PU	0	GPIO_B23
						2	PB_I2C_UART
						3	PB_TIM
						4	TWI_DIO
						7	#GPADC_CH1
						Others	Reserved
38	38	L14	PB24	I/O	PU	0	GPIO_B24
						2	PB_I2C_UART
						3	PB_TIM
						7	#GPADC_CH2
						Others	Reserved
39	39	N15	PB25	I/O	PD	0	GPIO_B25
						2	PB_I2C_UART
						3	PB_TIM
						7	#GPADC_CH3
						Others	Reserved
40	40	M14	PB26	I/O	PD	0	GPIO_B26
						2	PB_I2C_UART
						3	PB_TIM
						7	#GPADC_CH4
						Others	Reserved
41	41	N16	PB27	I/O	-	0	GPIO_B27
						2	PB_I2C_UART
						3	PB_TIM
						7	#GPADC_CH5
						Others	Reserved
-	-	M15	PB28	I/O	PD	0	GPIO_B28
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_COLLISION
						7	#GPADC_CH6
						Others	Reserved
-	-	C15	PB29	I/O	PD	0	GPIO_B29
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_PRIORITY
						7	LPCOMP2_OUT
						Others	Reserved
-	-	B16	PB30	I/O	PD	0	GPIO_B30
						2	PB_I2C_UART
						3	PB_TIM
						4	BT_ACTIVE
						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
SF32LB560	SF32LB561 SF32LB563	SF32LB565 SF32LB566 SF32LB567					
QFN68L	QFN68L	BGA175					
-	-	B15	PB31	I/O	PD	0	GPIO_B31
						2	PB_I2C_UART
						3	PB_TIM
						4	WLAN_ACTIVE
						Others	Reserved
54	54	D13	PB32	I/O	PD	0	GPIO_B32
						1	#WKUP_PIN0
						7	#GPADC_CH7
						Others	Reserved
55	55	D14	PB33	I/O	PD	0	GPIO_B33
						1	#WKUP_PIN1
						7	#ACTEST0
						Others	Reserved
56	56	C14	PB34	I/O	PD	0	GPIO_B34
						1	#WKUP_PIN2
						7	#ACTEST1
						Others	Reserved
-	-	A15	PB35	I/O	PD	0	GPIO_B35
						1	#WKUP_PIN3
						Others	Reserved
-	-	B14	PB36	I/O	PD	0	GPIO_B36
						1	#WKUP_PIN4
						Others	Reserved
59	59	A10	PBR00	I/O	PD	0	PWR_REQ
						1	GPO
						2	LPTIM3_OUT
						3	LPTIM3_OUT_BAR
						Others	Reserved
58	58	B10	PBR01	I/O	PD	0	GPO
						1	CLK_LP
						2	LPTIM3_OUT
						3	LPTIM3_OUT_BAR
						Others	Reserved
57	57	D12	PBR02	I/O	PD	0	GPO
						1	CLK_LP
						2	LPTIM3_OUT
						3	LPTIM3_OUT_BAR
						Others	Reserved
-	-	B11	PBR03	I/O	PD	0	GPO
						1	CLK_LP
						2	LPTIM3_OUT
						3	LPTIM3_OUT_BAR
						Others	Reserved

5.10 Package-integrated IO

ge-integ Package-integrated IO is not counted in the configurable IO number and is dedicated to MPI access to the package-integrated (SiP) NOR flash or pSRAM. The bonded IO shares the same structure as the general-purpose IO and

allows individual configuration of each IO's function as well as pull-up and pull-down settings. The bonded IO comprises IO(SA), IO(SB), and IO(SC). IO(SA) and IO(SB) are configured within HPSYS_PINMUX, whereas IO(SC) is configured within LPSYS_PINMUX.

IO(SA) includes SA00~SA12 and is used by MPI1 to access the bonded 8-line pSRAM.

IO(SB) includes SB00~SB12 and is used by MPI2 to access another bonded 8-line pSRAM.

IO(SC) includes SC00~SC05 and is used by MPI5 to access the bonded 4-line NOR Flash.

5.11 Low-power IO (PBR)

Low-power IO (PBR) is a specialized IO type capable of maintaining input and output functionality while the chip is in low-power mode. The input enable, output enable, pull-up and pull-down resistor functions of IO (PBR) can be configured via the RTC's PBRxR register. IO(PBR) can continuously output a fixed level. The low-power PWM generated by the low-power clock clk_lp or LPTIM3 is not affected by the system entering or exiting low-power mode (in hibernate mode, LPTIM3 ceases operation and thus cannot output PWM).

5.12 IO Power Supply

The voltage level standard of the IO depends on its supply voltage.

The power supply for combined package IO also serves as the power supply for the corresponding combined package memory. 3 The power supplies for discrete package IOs are independent of each other.

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

The power supply for IO(PA) is divided into two groups, with PA00~PA11 powered separately via VDDIOA2, enabling more flexible device selection.

IO Category	SF32LB565/566/567 Power Supply	SF32LB561/563 Power Supply	SF32LB560 Power Supply
SA	VDDIOSA	VDDIO1	VDDIO1
SB	VDDIOSB	VDDIO1	VDDIO1
SC	VDDIOSC	VDDIO4	VDDIO3
PA00~PA11	VDDIOA2	VDDIO3	VDDIO3
PA12~PA78	VDDIOA	VDDIO2	VDDIO2
PB	VDDIOB	VDDIO4	VDDIO3
PBR	VDDIOB	VDDIO4	VDDIO3

5.13 Wake-up PIN

The chip supports up to 14 wake-up PINs, including general-purpose IOs PB32~PB36, PA50~PA54, and low-power IOs PBR00~PBR03. The wake-up PIN can awaken the chip from hibernate mode, standby mode, or deepsleep mode. Wake-up trigger methods include high-level, low-level, rising edge, falling edge, and dual-edge triggers. 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.

Hibernate mode supports selecting up to two of these 14 wake-up PINs to be mapped as actual PIN wake-up sources. The selection register is located in the PMUC. CR_PIN0_SEL and CR_PIN1_SEL.

The standbymode supports up to 14wake-upPINssimultaneously as wake-up sources.

The deepsleep mode, in addition to supporting up to 14 wake-up PINs simultaneously as wake-up sources, also supports wake-up from any general-purpose IO within the subsystem. For example, after the HP-SYS enters deepsleep mode, it can be awakened not only by any one of the 14 wake-up PINs (wake-up status bits are HPSYS_AON WSR_PIN0~13), but also by any IO(PA) wake-up (wake-up status bit is HPSYS_AON WSR_GPIO1; the specific wake-up IO requires at GPIO Register query).

General-purpose IO (PB32~PB36, PA50~PA54) supporting the wake-up PIN function, in addition to the original IO input path, includes an additional 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 IO pins do not remain in a floating state when not toggled; otherwise, leakage current may occur.

When the chip is in hibernate mode, all general-purpose IO pull-up and pull-down resistor configurations in the PINMUX register are invalid. Therefore, if the external circuitry cannot guarantee that the IOs (PB32~PB36, PA50~PA54) supporting the wake-up PIN function remain at a defined high or low level, additional configuration of the PAWKUP and PBWKUP registers in the RTC to enable pull-up or pull-down resistors is required before entering hibernate mode. It should be noted that the pull-up and pull-down resistor configurations in the PINMUX register and the RTC register take effect simultaneously. To avoid redundancy and conflicts, it is recommended to configure only via the RTC register.

5.14 IO Status in Low-power Mode

In HPSYS active mode and sleep mode, IO(PA), IO(SA), and IO(SB) operate normally; outputs can toggle and generate IO interrupts.

After HPSYS enters deepsleep mode, IO(PA), IO(SA), and IO(SB) enter sleep mode; input enable, output enable, and pull-up/pull-down resistor configurations remain unchanged; outputs retain their previous state, stop toggling, cannot generate IO interrupts, but IO(PA) wake-up can be triggered. IO(PA) supporting the wake-up PIN function can also generate a PIN wake-up. The operation of IO(PB) and IO(PBR) is unaffected.

After HPSYS enters standby mode, IO(PA) , IO(SA) , and IO(SB) enter 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 and IO(PA) wake-up cannot be generated IO(PA) supporting the wake-up PIN function can generate a PIN wake-up. The operation of IO(PB) and IO(PBR) is unaffected.

In the active and sleep modes of LPSYS, IO(PB) and IO(SC) operate normally; output can toggle normally, and IO interrupts can be generated.

After LPSYS enters deepsleep mode, IO(PB) and IO(SC) enter sleep mode; input enable, output enable, and pull-up/pull-down resistor configurations remain unchanged, output holds the previous value, toggling stops, IO interrupts cannot be generated, but IO(PB) wake-up can be triggered. IO(PB) pins supporting wake-up PIN functionality can also generate PIN wake-up. The operation of IO(PA) and IO(PBR) remains unaffected.

After LPSYS enters standby mode, IO(PB) and IO(SC) enter retention mode; input enable, output enable, and pull-up/pull-down resistor configurations remain unchanged, output holds the previous value, toggling stops, IO interrupts and IO(PB) wake-up cannot be generated. IO (PB) sup porting the wake-up PIN function can generate PIN wake-up. The operation

of IO(PA) and IO(PBR) remains unaffected.

After the chip enters hibernate mode, IO (SA), IO (SB), and IO (PA) and IO (PB) without the wake-up PIN function enter shutdown mode; input enable and output enable are disabled, pull-up and pull-down resistors are disabled, IO behaves as high impedance, and IO interrupts and wake-up cannot be generated. IO (SC) enters retention mode; input enable, output enable, and pull-up/pull-down resistor configurations remain unchanged, output holds the previous value and stops toggling. IO (PA) and IO (PB) supporting the wake-up PIN function enter wake-up mode and can generate PIN wake-up; pull-up and pull-down resistors are configured by the RTC register. IO(PBR) operation is unaffected; output can be maintained(except for LPTIM3 PWM) , and PIN wake-up can be generated

Table 5-3: IO Operating Status

IO	function	active	sleep	deepsleep	standby	hibernate
PA00~PA49 PA55~PA78 PB00~PB31	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
	IO Wake-up	Present	Present	Present	None	None
	PIN Wake-up	None	None	None	None	None
PA50~PA54 PB32~PB36	Input Enable	Active	Active	hold	hold	Invalid
	Output Enable	Active	Active	hold	hold	Invalid
	Output Level	Flippable	Flippable	hold	hold	Invalid
	Pull-up/Pull-down Resistor	Present	Present	Present	Present	Present
	GPIO Interrupt	Present	Present	None	None	None
	IO Wake-up	Present	Present	Present	None	None
	PIN Wake-up	Present	Present	Present	Present	Present
PBR00~PBR03	Input Enable	Active	Active	Active	Active	Active
	Output Enable	Active	Active	Active	Active	Active
	Output Level	Flippable	Flippable	Flippable	Flippable	Flippable
	Pull-up/Pull-down Resistor	Present	Present	Present	Present	Present
	GPIO Interrupt	None	None	None	None	None
	IO Wake-up	None	None	None	None	None
	PIN Wake-up	Present	Present	Present	Present	Present
SA00~SA12 SB00~SB12	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	Present	Present	hold	hold	Invalid
	GPIO Interrupt	None	None	None	None	None
	IO Wake-up	None	None	None	None	None
	PIN Wake-up	None	None	None	None	None
SC00~SC05	Input Enable	Active	Active	hold	hold	hold
	Output Enable	Active	Active	hold	hold	hold
	Output Level	Flippable	Flippable	hold	hold	hold
	Pull-up/Pull-down Resistor	Present	Present	hold	hold	hold
	GPIO Interrupt	None	None	None	None	None
	IO Wake-up	None	None	None	None	None
	PIN Wake-up	None	None	None	None	None

* (1) The operating status of PA, SA and SB is affected by the low-power mode of HPSYS and not affected by the low-power mode of LPSYS. (2) The operating status of PB and SC is affected by the low-power mode of LPSYS and not affected by the low-power mode of HPSYS.

5.15 Avoid IOLeakage Current

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

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

For the second type of leakage, IOs configured as module output or GPIO push-pull output can have their pull-up and pull-down resistors 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 general purpose IOs (PB32~PB36, PA50~PA54) supporting wake-up PIN functionality, pull-up and pull-down resistors must be configured via the PAWKUP and PBWKUP registers in the RTC, and the pull-up/pull-down resistor configuration in the PIN-MUX registers must be disabled to prevent configuration conflicts.

For the fourth category of leakage current, except for IOs with explicitly defined input levels as high or low, the following configuration methods shall be applied:

- General-purpose IOs without input functionality shall have input enable disabled (IE set to 0).
- IOs (PBR) not used as wake-up PINs shall have input enable disabled (IE set to 0).
- For IO (PBR) used as a wake-up PIN, configure the chip's internal pull-up or pull-down resistor.
- General-purpose IOs supporting wake-up PIN functionality (PB32~PB36, PA50~PA54) configure pull-up and pull-down resistors via the
- PAWKUP and PBWKUP registers in the RTC.

If IO leakage is suspected, each IO can be checked as follows.

Table 5-4: IO Operating Status

IO	If configured as 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~PA49 PA55~PA78 PB00~PB31	Verify whether the output value matches the pull-up or pull-down configuration in PINMUX. Verify whether the output value corresponds to the pull-up and pull-down configuration of the external circuit. Verify if a high level is output while the power supply to the external device connected to the IO is 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 for level conflicts with the external circuit
PA50~PA54 PB32~PB36	Verify whether the output value corresponds to the pullup and pull-down configuration of the RTC Verify whether the pull-up and pull-down configurations of the PINMUX and RTC conflict Verify whether the output value corresponds to the pull-up and pull-down configuration of the external circuit. Verify if a high level is output while the power supply to the external device connected to the IO is off	Verify whether the external device connected to the IO can provide a defined logic level Verify whether the PINMUX and RTC are 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 PINMUX and RTC are configured with appropriate pull-up and pull-down settings
PBR00~PBR03	Verify whether the output value corresponds to the pull-up and pull-down configuration of the RTC Verify whether the output value corresponds to the pull-up and pull-down configuration of the external circuit Verify if a high level is output while the power supply to the external device connected to the IO is off	Verify whether the external device connected to the IO can provide a defined logic level Verify whether the RTC is configured with appropriate pull-up and pull-down resistors.	Verify the RTC configuration of pull-up and pull-down resistors and external circuit level conflicts.

The chip provides a large number of IOs. When it is uncertain which IOs are leaking current, all general-purpose IOs (PB00~PB36, PA00~PA78) can first be configured in PINMUX to a non-input high-impedance state (IE=0, PE=0, OE=0). All low-power IOs (PBR00~PBR03) can be configured in the RTC as no-input high-impedance state (IE=0, PE=0, OE=0), and

general-purpose IOs (PB32~PB36, PA50~PA54) supporting wake-up PIN functions should have appropriate pull-up and pull-down resistors configured in the RTC registers. Under this condition, the chip IOs will not generate leakage current. Restore the IO configuration in batches, identify the IO causing leakage current, and eliminate the leakage factors.

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

5.16 Prevent leakage current in bonded IO

The power supply for bonded IO also supplies the corresponding bonded memory; therefore, leakage current in bonded memory will also appear on the power supply of bonded IO. The following two types of bonded IO leakage current are common:

1. The chip enters low-power mode, but the bonded memory does not enter a low-power state. The embedded NOR flash or pSRAM also consumes current in the standby state without access (dependent on the memory model, typically on the order of 10~200uA) . This leakage current becomes more pronounced after the chip enters low-power mode; therefore, it is recommended that the embedded memory enter a low-power state (leakage approximately on the order of 1~10uA) before the chip enters deepsleep or standby mode, and that the embedded memory exit the low-power state after the chip wakes up. The embedded pSRAM can enter half sleep mode via instructions, and the embedded NOR flash can enter deep power down mode via instructions. These modes can signifcantly reduce memory leakage current. Co-packaged IO uncontrollably causes leakage in the co-packaged memory. When the chip enters hibernate mode, the co-packaged IO(SA) and IO(SB) present high-impedance floating levels. If the power supply to the co-packaged IO(SA) and IO(SB) remains active at this time, it may cause the co-packaged memory to enter a high leakage state (for example, if the CS pin is floating at a low or intermediate level, leakage currents ranging from hundreds of uA to several mA may occur). Therefore, before the chip enters hibernate mode, the power supply to the co-packaged IO(SA) and IO(SB) must be turned off to prevent leakage in the co-packaged memory. It can be used to maintain the power supply of IO(SA) and IO(SB), controlled by IO(PBR), which still have output capability in hibernate mode. The software shall turn off the power to IO(SA) and IO(SB) via IO(PBR) before entering hibernate mode, and turn on the power via IO(PBR) after wake-up to prevent leakage current during hibernate mode If the power supply to the encapsulated IO(SA) and IO(SB) can only maintain constant power and cannot be controlled, the chip's hibernate mode should be avoided, and standby mode may be used instead.

After the chip enters hibernate mode, the encapsulated IO(SC) can maintain its level, thereby preventing leakage current in the encapsulated NOR Flash. In hibernate mode, power to IO(SC) must be continuously maintained to ensure access to the sealed NOR Flash upon wake-up. To further reduce power consumption, the NOR Flash may be placed into deep power down mode prior to entering hibernate mode.

5.17 HPSYS_PINMUX Register

Table 5-5: HPSYS_PINMUX Register map

Offset	Attribute	Reset Value	Register Name	Register Description
0x0			**PAD_SA00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x4			**PAD_SA01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x8			**PAD_SA02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC			**PAD_SA03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x10			**PAD_SA04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x14			**PAD_SA05**	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

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

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 week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x28			**PAD_SA10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x2C			**PAD_SA11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x30			**PAD_SA12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x34			**PAD_SB00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x38			**PAD_SB01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate

Continued on 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, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x3C			**PAD_SB02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x40			**PAD_SB03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x44			**PAD_SB04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x48			**PAD_SB05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x4C			**PAD_SB06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x50			**PAD_SB07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x54			**PAD_SB08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x58			**PAD_SB09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x5C			**PAD_SB10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x60			**PAD_SB11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x64			**PAD_SB12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x68			**PAD_PA00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x6C			**PAD_PA01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x70			**PAD_PA02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x74			**PAD_PA03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x78			**PAD_PA04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x7C			**PAD_PA05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x80			**PAD_PA06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x84			**PAD_PA07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x88			**PAD_PA08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x8C			**PAD_PA09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x90			**PAD_PA10**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x94			**PAD_PA11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x98			**PAD_PA12**	
[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	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x9C			**PAD_PA13**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA0			**PAD_PA14**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA4			**PAD_PA15**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xA8			**PAD_PA16**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4'h0	FSEL	Function Select
0xAC			**PAD_PA17**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB0			**PAD_PA18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB4			**PAD_PA19**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xB8			**PAD_PA20**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xBC			**PAD_PA21**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input

Continued on next page...

Table 5-5: HPSYS_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 select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC0			**PAD_PA22**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC4			**PAD_PA23**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xC8			**PAD_PA24**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xCC			**PAD_PA25**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xD0			**PAD_PA26**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xD4			**PAD_PA27**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xD8			**PAD_PA28**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xDC			**PAD_PA29**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xE0			**PAD_PA30**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xE4			**PAD_PA31**	

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xE8			**PAD_PA32**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xEC			**PAD_PA33**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xF0			**PAD_PA34**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xF4			**PAD_PA35**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down

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 week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xF8			**PAD_PA36**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0xFC			**PAD_PA37**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x100			**PAD_PA38**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x104			**PAD_PA39**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x108			**PAD_PA40**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate

Continued on 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, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x10C			**PAD_PA41**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x110			**PAD_PA42**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x114			**PAD_PA43**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x118			**PAD_PA44**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x11C			**PAD_PA45**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x120			**PAD_PA46**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x124			**PAD_PA47**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x128			**PAD_PA48**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x12C			**PAD_PA49**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x130			**PAD_PA50**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x134			**PAD_PA51**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x138			**PAD_PA52**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x13C			**PAD_PA53**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x140			**PAD_PA54**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x144			**PAD_PA55**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x148			**PAD_PA56**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x14C			**PAD_PA57**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x150			**PAD_PA58**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x154			**PAD_PA59**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x158			**PAD_PA60**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x15C			**PAD_PA61**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x160			**PAD_PA62**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x164			**PAD_PA63**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x168			**PAD_PA64**	
[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	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x16C			**PAD_PA65**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x170			**PAD_PA66**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x174			**PAD_PA67**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x178			**PAD_PA68**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4'h0	FSEL	Function Select
0x17C			**PAD_PA69**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x180			**PAD_PA70**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x184			**PAD_PA71**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x188			**PAD_PA72**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x18C			**PAD_PA73**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input

Continued on next page...

Table 5-5: HPSYS_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 select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x190			**PAD_PA74**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x194			**PAD_PA75**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x198			**PAD_PA76**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x19C			**PAD_PA77**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3:0]	rw	4'h0	FSEL	Function Select
0x1A0			**PAD_PA78**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength

Continued on next page...

Table 5-5: HPSYS_PINMUX Register map (continued)

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

5.18 HPSYS_CFG Register

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**BMR**	Boot Mode Register
[31:1]			RSVD	
[0]	r	1'h0	BOOT_MODE	0 - normal, 1 - download mode
0x04			**IDR**	ID Register
[31:24]	r	8'hC2	SID	Series ID
[23:16]	r	8'h03	CID	Chip ID
[15:8]	r	8'h0	PID	Package ID
[7:0]	r	8'h0	REVID	Revision ID
0x08			**SCR**	Security Control Register
[31:1]			RSVD	
[0]	rw	1'h1	FKEY_MODE	reserved for debug
0x0C			**USBCR**	USB Control register
[31:24]			RSVD1	
[23:16]			RSVD0	
[15:13]	rw	3'h0	DC_TR	reserved for debug
[12]	rw	1'h0	DC_TE	reserved for debug
[11]			RSVD	
[10:8]	rw	3'h0	TX_RTUNE	TX outp impedance tuning 0 = 50 Ohm, 1 = 46 Ohm, 2 = 43 Ohm, 3 = 40 Ohm, 4 = 37.5 Ohm, 5 = 35 Ohm, 6 = 33 Ohm, 7 = 31.5 Ohm
[7]			RSVD	
[6]	rw	1'h0	DP_EN	0:disable dp pull up or pull down 1:enable dp pull or pull down
[5]	rw	1'h0	DM_PD	enable DM 15k Ohm pull down resistor
[4]	rw	1'h0	LDO_LP_EN	2.5V LDO low power mode enable. 0 = 240 uA, 1 = 50 uA
[3:1]	rw	3'h0	LDO_VSEL	2.5V LDO output voltage setting 0 = 2.40 V, 1 = 2.47 V, 2 = 2.53 V, 3 = 2.60 V, 4 = 2.60 V, 5 = 2.67 V, 6 = 2.73 V, 7 = 2.8 V
[0]	rw	1'h0	USB_EN	USB PHY enable, turn on power swith, power up LDO and bias
0x10			**MCR**	Memory Control register
[31]	rw	1'b0	FORCE_ON	reserved for debug
[30:14]			RSVD	
[13]	rw	1'b0	PD_OTHER	reserved for debug
[12:7]			RSVD	
[6]	rw	1'b0	PD_RAM3	reserved for debug
[5]	rw	1'b0	PD_RAM2	reserved for debug
[4]	rw	1'b0	PD_RAM1	reserved for debug
[3]	rw	1'b0	PD_RAM0	reserved for debug

Continued on next page...

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2]	rw	1'b0	PD_CACHE	reserved for debug
[1]	rw	1'b0	PD_ITCM	reserved for debug
[0]	rw	1'b0	PD_ROM	reserved for debug
0x14			**ULPMCR**	ULP Memory Control register
[31]	rw	1'h0	FORCE_ON	reserved for debug
[30:13]			RSVD	
[12:10]	rw	3'b000	WPULSE	reserved for debug
[9:7]	rw	3'b100	WA	reserved for debug
[6:5]	rw	2'b00	RA	reserved for debug
[4]	rw	1'b1	RME	reserved for debug
[3:2]			RSVD	
[1:0]	rw	2'b11	RM	reserved for debug
0x18			**RTC_TR**	Mirrored RTC Time Register
[31]	r	1'h0	PM	AM/PM notation 0: AM 1: PM
[30:29]	r	2'h0	HT	Hour tens in BCD format
[28:25]	r	4'h0	HU	Hour units in BCD format
[24:22]	r	3'h0	MNT	Minute tens in BCD format
[21:18]	r	4'h0	MNU	Minute units in BCD format
[17:15]	r	3'h0	ST	Second tens in BCD format
[14:11]	r	4'h0	SU	Second units in BCD format
[10]			RSVD	
[9:0]	r	10'h0	SS	Sub-second counter
0x1C			**RTC_DR**	Mirrored RTC Date Register
[31]	r	1'h0	ERR	reserved for debug
[30:25]			RSVD	
[24]	r	1'h0	CB	Century flag
[23:20]	r	4'h0	YT	Year tens in BCD format
[19:16]	r	4'h0	YU	Year units in BCD format
[15:13]	r	3'h1	WD	Week day units 000: forbidden 001: Monday ... 111: Sunday
[12]	r	1'h0	MT	Month tens in BCD format
[11:8]	r	4'h1	MU	Month units in BCD format
[7:6]			RSVD	
[5:4]	r	2'h0	DT	Date tens in BCD format
[3:0]	r	4'h1	DU	Date units in BCD format
0x20			**DBGR**	Debug Select Register
[31]			RSVD	
[30]	r	1'h0	LP2HP_NMIF	LP2HP NMI interrupt flag
[29]	rw	1'h0	LP2HP_NMIE	LP2HP NMI interrupt enable
[28]	rw	1'h0	HP2LP_NMI	set 1 to send NMI interrupt to LCPU
[27]	rw	1'b0	CLK_EN	reserved for debug
[26:24]	rw	3'b0	CLK_SEL	reserved for debug
[23:16]	rw	8'h0	BITEN_H	reserved for debug
[15:8]	rw	8'h0	BITEN_L	reserved for debug
[7:4]	rw	4'h0	SEL_H	reserved for debug
[3:0]	rw	4'h0	SEL_L	reserved for debug
0x24			**CAU2_CR**	CAU2 Control Register

Continued on next page...

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:13]			RSVD	
[12:10]	rw	3'h0	DC_MR	reserved for debug
[9:7]	rw	3'h0	DC_BR	reserved for debug
[6:4]	rw	3'h0	DC_TR	reserved for debug
[3:2]			RSVD	
[1]	rw	1'b0	HPBG_EN	reserved for debug
[0]	rw	1'b0	HPBG_VDDPSW_EN	reserved for debug
0x28			**CAU2_RSVD1**	CAU2 RSVD Register1
[31:24]			RSVD	
[23:16]	rw	8'h0	RESERVE2	reserved for debug
[15:8]	rw	8'h0	RESERVE1	reserved for debug
[7:0]	rw	8'h0	RESERVE0	reserved for debug
0x2C			**CAU2_RSVD2**	CAU2 RSVD Register2
[31:24]			RSVD	
[23:16]	r	8'h0	RESERVE5	reserved for debug
[15:8]	r	8'h0	RESERVE4	reserved for debug
[7:0]	r	8'h0	RESERVE3	reserved for debug
0x38			**SYS_RSVD**	HPSYS RSVD Register
[31:24]	r	8'h0	RESERVE3	reserved for debug
[23:16]	rw	8'hff	RESERVE2	reserved for debug
[15:8]	rw	8'h0	RESERVE1	reserved for debug
[7:0]	rw	8'h0	RESERVE0	reserved for debug
0x3C			**LPIRQ**	Interrupt Selection for LCPU
[31]	rw1c	1'h0	IF3	hp2lp3 interrupt status. Write 1 to clear.
[30]			RSVD	
[29:24]	rw	6'h0	SEL3	select hp2lp3 interrupt source
[23]	rw1c	1'h0	IF2	hp2lp2 interrupt status. Write 1 to clear.
[22]			RSVD	
[21:16]	rw	6'h0	SEL2	select hp2lp2 interrupt source
[15]	rw1c	1'h0	IF1	hp2lp1 interrupt status. Write 1 to clear.
[14]			RSVD	
[13:8]	rw	6'h0	SEL1	select hp2lp1 interrupt source
[7]	rw1c	1'h0	IF0	hp2lp0 interrupt status. Write 1 to clear.
[6]			RSVD	
[5:0]	rw	6'h0	SEL0	select hp2lp0 interrupt source
0x5C			**SYSCR**	System Configure Register
[31:5]			RSVD	
[4]	rw	1'h0	SDNAND	0: MPI3 AHB space is allocated to MPI3 1: MPI3 AHB space is allocated to SDMMC2
[3]	rw	1'h0	REMAP	reserved for debug
[2]	rw	1'h0	HEXOKAYS	reserved for debug
[1]	rw	1'h0	HEXOKAYC	reserved for debug
[0]	rw	1'h0	WDT1_REBOOT	If set to 1, WDT1 reset will reboot the whole chip
0x60			**I2C1_PINR**	I2C1 Pin Register
[31:15]			RSVD	
[14:8]	rw	7'h7f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.

Continued on next page...

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x64			**I2C2_PINR**	I2C2 Pin Register
[31:15]			RSVD	
[14:8]	rw	7'h7f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x68			**I2C3_PINR**	I2C3 Pin Register
[31:15]			RSVD	
[14:8]	rw	7'h7f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x6C			**I2C4_PINR**	I2C4 Pin Register
[31:15]			RSVD	
[14:8]	rw	7'h7f	SDA_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	SCL_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x70			**USART1_PINR**	USART1 Pin Register
[31]			RSVD	
[30:24]	rw	7'h7f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[23]			RSVD	
[22:16]	rw	7'h7f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[15]			RSVD	
[14:8]	rw	7'h7f	RXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	TXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x74			**USART2_PINR**	USART2 Pin Register
[31]			RSVD	
[30:24]	rw	7'h7f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[23]			RSVD	
[22:16]	rw	7'h7f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.

Continued on next page...

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15]			RSVD	
[14:8]	rw	7'h7f	RXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	TXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x78			**USART3_PINR**	USART3 Pin Register
[31]			RSVD	
[30:24]	rw	7'h7f	CTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[23]			RSVD	
[22:16]	rw	7'h7f	RTS_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[15]			RSVD	
[14:8]	rw	7'h7f	RXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	TXD_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x7C			**GPTIM1_PINR**	GPTIM1 Pin Register
[31]			RSVD	
[30:24]	rw	7'h7f	CH4_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[23]			RSVD	
[22:16]	rw	7'h7f	CH3_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[15]			RSVD	
[14:8]	rw	7'h7f	CH2_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	CH1_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x80			**GPTIM2_PINR**	GPTIM2 Pin Register
[31]			RSVD	
[30:24]	rw	7'h7f	CH4_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[23]			RSVD	
[22:16]	rw	7'h7f	CH3_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[15]			RSVD	

Continued on next page...

Table 5-6: HPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14:8]	rw	7'h7f	CH2_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	CH1_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
0x84			**ETR_PINR**	GPTIM ETR Pin Register
[31:15]			RSVD	
[14:8]	rw	7'h7f	ETR2_PIN	Connect GPTIM2_ETR to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	ETR1_PIN	Connect GPTIM1_ETR to selected IO(PA). 0 to 44 for PA00 to PA44. Other values for floating.
0x88			**LPTIM1_PINR**	LPTIM1 Pin Register
[31:23]			RSVD	
[22:16]	rw	7'h7f	ETR_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[15]			RSVD	
[14:8]	rw	7'h7f	OUT_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.
[7]			RSVD	
[6:0]	rw	7'h7f	IN_PIN	Connect function pin to selected IO(PA). 0 to 78 for PA00 to PA78. Other values for floating.

5.19 HPSYS_GPIO Register

Table 5-7: 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			**DOSR0**	Data Output Set Register
[31:0]	w	32'h0	DOS	set 1 to pull up output of corresponding GPIO[31:0]
0x0C			**DOCR0**	Data Output Clear Register
[31:0]	w	32'h0	DOC	set 1 to pull down output of corresponding GPIO[31:0]
0x10			**DOER0**	Data Output Enable Register
[31:0]	rw	32'h0	DOE	GPIO[31:0] output enable
0x14			**DOESR0**	Data Output Enable Set Register
[31:0]	w	32'h0	DOES	set 1 to enable output of corresponding GPIO[31:0]
0x18			**DOECR0**	Data Output Enable Clear Register
[31:0]	w	32'h0	DOEC	set 1 to disable output of corresponding GPIO[31:0]
0x1C			**IER0**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[31:0] interrupt enable

Continued on next page...

Table 5-7: HPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x20			**IESR0**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[31:0]
0x24			**IECR0**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[31:0]
0x28			**ITR0**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[31:0] interrupt type
0x2C			**ITSR0**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[31:0]
0x30			**ITCR0**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[31:0]
0x34			**IPHR0**	Interrupt Polarity High Register
[31:0]	rw	32'h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x38			**IPHSR0**	Interrupt Polarity High Set Register
[31:0]	w	32'h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x3C			**IPHCR0**	Interrupt Polarity High Clear Register
[31:0]	w	32'h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x40			**IPLR0**	Interrupt Polarity Low Register
[31:0]	rw	32'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x44			**IPLSR0**	Interrupt Polarity Low Set Register
[31:0]	w	32'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x48			**IPLCR0**	Interrupt Polarity Low Clear Register
[31:0]	w	32'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x4C			**ISR0**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[31:0]
0x60			**OEMR0**	output mode Register
[31:0]	rw	32'h0	OEM	output mode of corresponding GPIO[31:0]
0x64			**OEMSR0**	output mode Set Register
[31:0]	w	32'h0	OEMS	output mode Set of corresponding GPIO[31:0]
0x68			**OEMCR0**	output mode Clear Register
[31:0]	w	32'h0	OEMC	output mode Clear of corresponding GPIO[31:0]
0x80			**DIR1**	Data Input Register
[31:0]	r	32'h0	IN	GPIO[63:32] input value
0x84			**DOR1**	Data Output Register
[31:0]	rw	32'h0	OUT	GPIO[63:32] output value if output enabled
0x88			**DOSR1**	Data Output Set Register
[31:0]	w	32'h0	DOS	set 1 to pull up output of corresponding GPIO[63:32]
0x8C			**DOCR1**	Data Output Clear Register
[31:0]	w	32'h0	DOC	set 1 to pull down output of corresponding GPIO[63:32]
0x90			**DOER1**	Data Output Enable Register
[31:0]	rw	32'h0	DOE	GPIO[63:32] output enable
0x94			**DOESR1**	Data Output Enable Set Register
[31:0]	w	32'h0	DOES	set 1 to enable output of corresponding GPIO[63:32]
0x98			**DOECR1**	Data Output Enable Clear Register
[31:0]	w	32'h0	DOEC	set 1 to disable output of corresponding GPIO[63:32]
0x9C			**IER1**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[63:32] interrupt enable

Continued on next page...

Table 5-7: HPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xA0			**IESR1**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[63:32]
0xA4			**IECR1**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[63:32]
0xA8			**ITR1**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[63:32] interrupt type
0xAC			**ITSR1**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[63:32]
0xB0			**ITCR1**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[63:32]
0xB4			**IPHR1**	Interrupt Polarity High Register
[31:0]	rw	32'h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[63:32]
0xB8			**IPHSR1**	Interrupt Polarity High Set Register
[31:0]	w	32'h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[63:32]
0xBC			**IPHCR1**	Interrupt Polarity High Clear Register
[31:0]	w	32'h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[63:32]
0xC0			**IPLR1**	Interrupt Polarity Low Register
[31:0]	rw	32'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[63:32]
0xC4			**IPLSR1**	Interrupt Polarity Low Set Register
[31:0]	w	32'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[63:32]
0xC8			**IPLCR1**	Interrupt Polarity Low Clear Register
[31:0]	w	32'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[63:32]
0xCC			**ISR1**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[63:32]
0xE0			**OEMR1**	output mode Register
[31:0]	rw	32'h0	OEM	output mode of corresponding GPIO[63:32]
0xE4			**OEMSR1**	output mode Set Register
[31:0]	w	32'h0	OEMS	output mode Set of corresponding GPIO[63:32]
0xE8			**OEMCR1**	output mode Clear Register
[31:0]	w	32'h0	OEMC	output mode Clear of corresponding GPIO[63:32]
0x100			**DIR2**	Data Input Register
[31:15]			RSVD	
[14:0]	r	15'h0	IN	GPIO[78:64] input value
0x104			**DOR2**	Data Output Register
[31:15]			RSVD	
[14:0]	rw	15'h0	OUT	GPIO[78:64] output value if output enabled
0x108			**DOSR2**	Data Output Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	DOS	set 1 to pull up output of corresponding GPIO[78:64]
0x10C			**DOCR2**	Data Output Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	DOC	set 1 to pull down output of corresponding GPIO[78:64]
0x110			**DOER2**	Data Output Enable Register
[31:15]			RSVD	
[14:0]	rw	15'h0	DOE	GPIO[78:64] output enable
0x114			**DOESR2**	Data Output Enable Set Register

Continued on next page...

Table 5-7: HPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:15]			RSVD	
[14:0]	w	15'h0	DOES	set 1 to enable output of corresponding GPIO[78:64]
0x118			**DOECR2**	Data Output Enable Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	DOEC	set 1 to disable output of corresponding GPIO[78:64]
0x11C			**IER2**	Interrupt Enable Register
[31:15]			RSVD	
[14:0]	rw	15'h0	IER	GPIO[78:64] interrupt enable
0x120			**IESR2**	Interrupt Enable Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	IES	set 1 to enable interrupt of corresponding GPIO[78:64]
0x124			**IECR2**	Interrupt Enable Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	IEC	set 1 to disable interrupt of corresponding GPIO[78:64]
0x128			**ITR2**	Interrupt Type Register
[31:15]			RSVD	
[14:0]	rw	15'h0	ITR	GPIO[78:64] interrupt type
0x12C			**ITSR2**	Interrupt Type Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[78:64]
0x130			**ITCR2**	Interrupt Type Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[78:64]
0x134			**IPHR2**	Interrupt Polarity High Register
[31:15]			RSVD	
[14:0]	rw	15'h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[78:64]
0x138			**IPHSR2**	Interrupt Polarity High Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[78:64]
0x13C			**IPHCR2**	Interrupt Polarity High Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[78:64]
0x140			**IPLR2**	Interrupt Polarity Low Register
[31:15]			RSVD	
[14:0]	rw	15'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[78:64]
0x144			**IPLSR2**	Interrupt Polarity Low Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[78:64]
0x148			**IPLCR2**	Interrupt Polarity Low Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[78:64]
0x14C			**ISR2**	Interrupt Status Register
[31:15]			RSVD	
[14:0]	rw	15'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[78:64]
0x160			**OEMR2**	output mode Register
[31:15]			RSVD	

Continued on next page...

Table 5-7: HPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14:0]	rw	15'h0	OEM	output mode of corresponding GPIO[78:64]
0x164			**OEMSR2**	output mode Set Register
[31:15]			RSVD	
[14:0]	w	15'h0	OEMS	output mode Set of corresponding GPIO[78:64]
0x168			**OEMCR2**	output mode Clear Register
[31:15]			RSVD	
[14:0]	w	15'h0	OEMC	output mode Clear of corresponding GPIO[78:64]

5.20 LPSYS_PINMUX Register

Table 5-8: LPSYS_PINMUX Register map

Offset	Attribute	Reset Value	Register Name	Register Description
0x0			**PAD_SC00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x4			**PAD_SC01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x8			**PAD_SC02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0xC			**PAD_SC03**	

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x10			**PAD_SC04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x14			**PAD_SC05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x18			**PAD_PB00**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x1C			**PAD_PB01**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x20			**PAD_PB02**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x24			**PAD_PB03**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x28			**PAD_PB04**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x2C			**PAD_PB05**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x30			**PAD_PB06**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x34			**PAD_PB07**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x38			**PAD_PB08**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x3C			**PAD_PB09**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x40			**PAD_PB10**	

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x44			**PAD_PB11**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x48			**PAD_PB12**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x4C			**PAD_PB13**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x50			**PAD_PB14**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x54			**PAD_PB15**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x58			**PAD_PB16**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x5C			**PAD_PB17**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x60			**PAD_PB18**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x64			**PAD_PB19**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x68			**PAD_PB20**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x6C			**PAD_PB21**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x70			**PAD_PB22**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x74			**PAD_PB23**	

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x78			**PAD_PB24**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS	Drive Select. Logic LOW selects 4mA drive,logic HIGH selects 20mA drive
[9]			RSVD	
[8]	rw	1'h0	MODE	Mode Select. Logic LOW enables GPIO mode,logic HIGH enables I2C mode
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h1	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x7C			**PAD_PB25**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x80			**PAD_PB26**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x84			**PAD_PB27**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h0	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h0	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x88			**PAD_PB28**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x8C			**PAD_PB29**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x90			**PAD_PB30**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x94			**PAD_PB31**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x98			**PAD_PB32**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0x9C			**PAD_PB33**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0xA0			**PAD_PB34**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0xA4			**PAD_PB35**	
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select
0xA8			**PAD_PB36**	

Continued on next page...

Table 5-8: LPSYS_PINMUX Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:12]			RSVD	
[11]	rw	1'h0	POE	Reserved. Always set to logic LOW
[10]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[9]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[8]	rw	1'h0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[7]	rw	1'h1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[6]	rw	1'h1	IE	Input Enable. Logic HIGH enables the input buffer
[5]	rw	1'h0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[4]	rw	1'h1	PE	Pull Enable. Logic HIGH enables week pull device
[3]			RSVD	
[2:0]	rw	3'h0	FSEL	Function Select

5.21 LPSYS_CFG Register

Table 5-9: LPSYS_CFG Register map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**SWCR**	SW Control Register
[31:1]			RSVD	
[0]	rw	1'h0	SWSEL	0: SWD connected to HCPU 1: SWD connected to LCPU
0x04			**ULPMCR**	ULP Memory Control register
[31]	rw	1'h0	FORCE_ON	reserved for debug
[30]	rw	1'h0	ROM_DIS	reserved for debug
[29:21]			RSVD	
[20]	rw	1'b1	ROM_RME	reserved for debug
[19:18]			RSVD	
[17:16]	rw	2'b10	ROM_RM	reserved for debug
[15:13]			RSVD	
[12:10]	rw	3'b000	RAM_WPULSE	reserved for debug
[9:7]	rw	3'b110	RAM_WA	reserved for debug
[6:5]	rw	2'b01	RAM_RA	reserved for debug
[4]	rw	1'b1	RAM_RME	reserved for debug
[3:2]			RSVD	
[1:0]	rw	2'b00	RAM_RM	reserved for debug
0x08			**RTC_TR**	Mirrored RTC Time Register
[31]	r	1'h0	PM	AM/PM notation 0: AM 1: PM
[30:29]	r	2'h0	HT	Hour tens in BCD format
[28:25]	r	4'h0	HU	Hour units in BCD format
[24:22]	r	3'h0	MNT	Minute tens in BCD format
[21:18]	r	4'h0	MNU	Minute units in BCD format
[17:15]	r	3'h0	ST	Second tens in BCD format
[14:11]	r	4'h0	SU	Second units in BCD format
[10]			RSVD	
[9:0]	r	10'h0	SS	Sub-second counter
0x0C			**RTC_DR**	Mirrored RTC Date Register
[31]	r	1'h0	ERR	reserved for debug
[30:25]			RSVD	

Continued on next page...

Table 5-9: LPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[24]	r	1'h0	CB	Century flag
[23:20]	r	4'h0	YT	Year tens in BCD format
[19:16]	r	4'h0	YU	Year units in BCD format
[15:13]	r	3'h1	WD	Week day units 000: forbidden 001: Monday ... 111: Sunday
[12]	r	1'h0	MT	Month tens in BCD format
[11:8]	r	4'h1	MU	Month units in BCD format
[7:6]			RSVD	
[5:4]	r	2'h0	DT	Date tens in BCD format
[3:0]	r	4'h1	DU	Date units in BCD format
0x10			**MDBGR**	Memory Debug Register
[31:9]			RSVD	
[8]	rw	1'b0	PD_CACHE	reserved for debug
[7]	rw	1'b0	DS_CACHE	reserved for debug
[6]	rw	1'b0	LS_ROM	reserved for debug
[5]	rw	1'b0	LS_CACHE	reserved for debug
[4]	rw	1'b0	LS_DTCM	reserved for debug
[3]	rw	1'b0	LS_ITCM	reserved for debug
[2]	rw	1'b0	LS_RAM2	reserved for debug
[1]	rw	1'b0	LS_RAM1	reserved for debug
[0]	rw	1'b0	LS_RAM0	reserved for debug
0x14			**DBGR**	Debug Register
[31]	rw	1'h0	READY	reserved for debug
[30]	r	1'h0	HP2LP_NMIF	HP2LP NMI interrupt flag
[29]	rw	1'h0	HP2LP_NMIE	HP2LP NMI interrupt enable
[28]	rw	1'h0	LP2HP_NMI	set 1 to send NMI interrupt to HCPU
[27]	rw	1'b0	CLK_EN	reserved for debug
[26:24]	rw	3'b0	CLK_SEL	reserved for debug
[23:16]	rw	8'h0	BITEN_H	reserved for debug
[15:8]	rw	8'h0	BITEN_L	reserved for debug
[7:4]	rw	4'h0	SEL_H	reserved for debug
[3:0]	rw	4'h0	SEL_L	reserved for debug
0x38			**SYSCR**	System Configure Register
[31:26]	r	6'h0	SCIO	reserved for debug
[25:2]			RSVD	
[1]	rw	1'h0	DBG_SWAP	reserved for debug
[0]	rw	1'h0	WDT2_REBOOT	If set to 1, WDT2 reset will reboot the whole chip
0x3C			**ANATR**	Analog Test Register
[31:8]			RSVD	
[7:5]	rw	3'h0	DC_UR_ATEST1	reserved for debug
[4]	rw	1'h0	DC_TE_ATEST1	reserved for debug
[3:1]	rw	3'h0	DC_UR_ATEST0	reserved for debug
[0]	rw	1'h0	DC_TE_ATEST0	reserved for debug
0x40			**I2C5_PINR**	I2C5 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	

Continued on next page...

Table 5-9: LPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x44			**I2C6_PINR**	I2C6 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x48			**I2C7_PINR**	I2C7 Pin Register
[31:14]			RSVD	
[13:8]	rw	6'h3f	SDA_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	SCL_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x4C			**USART4_PINR**	USART4 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	RXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	TXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x50			**USART5_PINR**	USART5 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	RXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	

Continued on next page...

Table 5-9: LPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:0]	rw	6'h3f	TXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x54			**USART6_PINR**	USART6 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	RTS_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	RXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	TXD_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x58			**GPTIM3_PINR**	GPTIM3 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CH4_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	CH3_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	CH2_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	CH1_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x5C			**GPTIM4_PINR**	GPTIM4 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CH4_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	CH3_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	CH2_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	

Continued on next page...

Table 5-9: LPSYS_CFG Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:0]	rw	6'h3f	CH1_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x60			**GPTIM5_PINR**	GPTIM5 Pin Register
[31:30]			RSVD	
[29:24]	rw	6'h3f	CH4_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[23:22]			RSVD	
[21:16]	rw	6'h3f	CH3_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	CH2_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	CH1_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x64			**ETR_PINR**	ETR Pin Register
[31:22]			RSVD	
[21:16]	rw	6'h3f	ETR5_PIN	Connect GPTIM5_ETR to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	ETR4_PIN	Connect GPTIM4_ETR to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	ETR3_PIN	Connect GPTIM3_ETR to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
0x68			**LPTIM3_PINR**	LPTIM3 Pin Register
[31:22]			RSVD	
[21:16]	rw	6'h3f	ETR_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[15:14]			RSVD	
[13:8]	rw	6'h3f	OUT_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.
[7:6]			RSVD	
[5:0]	rw	6'h3f	IN_PIN	Connect function pin to selected IO(PB). 0 to 31 for PB00 to PB31. Other values for floating.

5.22 LPSYS_GPIO Register

Table 5-10: 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			**DOSR0**	Data Output Set Register
[31:0]	w	32'h0	DOS	set 1 to pull up output of corresponding GPIO[31:0]
0x0C			**DOCR0**	Data Output Clear Register
[31:0]	w	32'h0	DOC	set 1 to pull down output of corresponding GPIO[31:0]
0x10			**DOER0**	Data Output Enable Register
[31:0]	rw	32'h0	DOE	GPIO[31:0] output enable
0x14			**DOESR0**	Data Output Enable Set Register
[31:0]	w	32'h0	DOES	set 1 to enable output of corresponding GPIO[31:0]
0x18			**DOECR0**	Data Output Enable Clear Register
[31:0]	w	32'h0	DOEC	set 1 to disable output of corresponding GPIO[31:0]
0x1C			**IER0**	Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[31:0] interrupt enable
0x20			**IESR0**	Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable interrupt of corresponding GPIO[31:0]
0x24			**IECR0**	Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable interrupt of corresponding GPIO[31:0]
0x28			**ITR0**	Interrupt Type Register
[31:0]	rw	32'h0	ITR	GPIO[31:0] interrupt type
0x2C			**ITSR0**	Interrupt Type Set Register
[31:0]	w	32'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[31:0]
0x30			**ITCR0**	Interrupt Type Clear Register
[31:0]	w	32'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[31:0]
0x34			**IPHR0**	Interrupt Polarity High Register
[31:0]	rw	32'h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x38			**IPHSR0**	Interrupt Polarity High Set Register
[31:0]	w	32'h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x3C			**IPHCR0**	Interrupt Polarity High Clear Register
[31:0]	w	32'h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[31:0]
0x40			**IPLR0**	Interrupt Polarity Low Register
[31:0]	rw	32'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x44			**IPLSR0**	Interrupt Polarity Low Set Register
[31:0]	w	32'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x48			**IPLCR0**	Interrupt Polarity Low Clear Register
[31:0]	w	32'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[31:0]
0x4C			**ISR0**	Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[31:0]
0x50			**IER0_EXT**	Extra Interrupt Enable Register
[31:0]	rw	32'h0	IER	GPIO[31:0] extra interrupt enable
0x54			**IESR0_EXT**	Extra Interrupt Enable Set Register
[31:0]	w	32'h0	IES	set 1 to enable extra interrupt of corresponding GPIO[31:0]
0x58			**IECR0_EXT**	Extra Interrupt Enable Clear Register
[31:0]	w	32'h0	IEC	set 1 to disable extra interrupt of corresponding GPIO[31:0]

Continued on next page...

Table 5-10: LPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x5C			**ISR0_EXT**	Extra Interrupt Status Register
[31:0]	rw	32'h0	IS	Interrupt status. Write 1 will clear extra interrupt status of corresponding GPIO[31:0]
0x60			**OEMR0**	output mode Register
[31:0]	rw	32'h0	OEM	output mode of corresponding GPIO[31:0]
0x64			**OEMSR0**	output mode Set Register
[31:0]	w	32'h0	OEMS	output mode Set of corresponding GPIO[31:0]
0x68			**OEMCR0**	output mode Clear Register
[31:0]	w	32'h0	OEMC	output mode Clear of corresponding GPIO[31:0]
0x80			**DIR1**	Data Input Register
[31:5]			RSVD	
[4:0]	r	5'h0	IN	GPIO[36:32] input value
0x84			**DOR1**	Data Output Register
[31:5]			RSVD	
[4:0]	rw	5'h0	OUT	GPIO[36:32] output value if output enabled
0x88			**DOSR1**	Data Output Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	DOS	set 1 to pull up output of corresponding GPIO[36:32]
0x8C			**DOCR1**	Data Output Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	DOC	set 1 to pull down output of corresponding GPIO[36:32]
0x90			**DOER1**	Data Output Enable Register
[31:5]			RSVD	
[4:0]	rw	5'h0	DOE	GPIO[36:32] output enable
0x94			**DOESR1**	Data Output Enable Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	DOES	set 1 to enable output of corresponding GPIO[36:32]
0x98			**DOECR1**	Data Output Enable Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	DOEC	set 1 to disable output of corresponding GPIO[36:32]
0x9C			**IER1**	Interrupt Enable Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IER	GPIO[36:32] interrupt enable
0xA0			**IESR1**	Interrupt Enable Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	IES	set 1 to enable interrupt of corresponding GPIO[36:32]
0xA4			**IECR1**	Interrupt Enable Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	IEC	set 1 to disable interrupt of corresponding GPIO[36:32]
0xA8			**ITR1**	Interrupt Type Register
[31:5]			RSVD	
[4:0]	rw	5'h0	ITR	GPIO[36:32] interrupt type
0xAC			**ITSR1**	Interrupt Type Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	ITS	set 1 for edge-sensitive interrupt mode of corresponding GPIO[36:32]
0xB0			**ITCR1**	Interrupt Type Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	ITC	set 1 for level-sensitive interrupt mode of corresponding GPIO[36:32]
0xB4			**IPHR1**	Interrupt Polarity High Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IPH	rising edge in edge mode, or high level in level mode of corresponding GPIO[36:32]

Continued on next page...

Table 5-10: LPSYS_GPIO Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xB8			**IPHSR1**	Interrupt Polarity High Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	IPHS	set 1 for rising edge in edge mode, or high level in level mode of corresponding GPIO[36:32]
0xBC			**IPHCR1**	Interrupt Polarity High Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	IPHC	set 1 for disable rising edge in edge mode, or high level in level mode of corresponding GPIO[36:32]
0xC0			**IPLR1**	Interrupt Polarity Low Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IPL	falling edge in edge mode, or low level in level mode of corresponding GPIO[36:32]
0xC4			**IPLSR1**	Interrupt Polarity Low Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	IPLS	set 1 for falling edge in edge mode, or low level in level mode of corresponding GPIO[36:32]
0xC8			**IPLCR1**	Interrupt Polarity Low Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	IPLC	set 1 for disable falling edge in edge mode, or low level in level mode of corresponding GPIO[36:32]
0xCC			**ISR1**	Interrupt Status Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[36:32]
0xD0			**IER1_EXT**	Extra Interrupt Enable Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IER	GPIO[36:32] extra interrupt enable
0xD4			**IESR1_EXT**	Extra Interrupt Enable Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	IES	set 1 to enable extra interrupt of corresponding GPIO[36:32]
0xD8			**IECR1_EXT**	Extra Interrupt Enable Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	IEC	set 1 to disable extra interrupt of corresponding GPIO[36:32]
0xDC			**ISR1_EXT**	extra Interrupt Status Register
[31:5]			RSVD	
[4:0]	rw	5'h0	IS	Interrupt status. Write 1 will clear interrupt status of corresponding GPIO[36:32]
0xE0			**OEMR1**	output mode Register
[31:5]			RSVD	
[4:0]	rw	5'h0	OEM	output mode of corresponding GPIO[36:32]
0xE4			**OEMSR1**	output mode Set Register
[31:5]			RSVD	
[4:0]	w	5'h0	OEMS	output mode Set of corresponding GPIO[36:32]
0xE8			**OEMCR1**	output mode Clear Register
[31:5]			RSVD	
[4:0]	w	5'h0	OEMC	output mode Clear of corresponding GPIO[36:32]

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.DMAC comprises a total of 8 independent channels. Each channel can be configured with source and destination address regions, mapped respectively to the address ranges of various memories or peripherals, thereby enabling high-effciency transfers between memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral, 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 to match the peripheral's bandwidth; In memory transfer mode, DMAC does not wait for peripheral DMA requests and completes data transfers as promptly as possible. When multiple channels are enabled simultaneously, DMAC performs transfers sequentially in order of descending priority; 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 PTC trigger 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 based on 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 3types 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 Request

Each channel selects any one of the 64 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 peripheral request table of

the DMAC is as follows.

Table 6-1: DMAC Peripheral Request Table

CSELR1/2_CxS	DMAC1	DMAC2
0	mpi1	usart4_tx
1	mpi2	usart4_rx
2	mpi3	usart5_tx
3	i2c4	usart5_rx
4	usart1_tx	usart6_tx
5	usart1_rx	usart6_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	atim1_update	spi3_tx
17	atim1_trigger	spi3_rx
18	atim1_cc1	spi4_tx
19	atim1_cc2	spi4_rx
20	atim1_cc3	mpi5
21	atim1_cc4	i2c5
22	i2c1	i2c6
23	i2c2	i2c7
24	i2c3	gptim4_update
25	atim1_com	gptim4_trigger
26	usart3_tx	gptim4_cc1
27	usart3_rx	gptim4_cc2
28	spi1_tx	audadc_ch0
29	spi1_rx	audadc_ch1
30	spi2_tx	gpadc
31	spi2_rx	/
32	i2s1_tx	/
33	i2s1_rx	/
34	sci_tx	/
35	sci_rx	/
36	pdm1_l	/
37	pdm1_r	/
38	pdm2_l	/
39	pdm2_r	/
40	/	/
41	dac0	/
42	dac1	/
43	gptim2_update	/
44	gptim2_trigger	/
45	gptim2_cc1	/
46	audprc_tx_out_ch1	/
47	audprc_tx_out_ch0	/
48	audprc_tx_ch3	/
49	audprc_tx_ch2	/

Continued on next page...

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

CSELR1/2_CxS	DMAC1	DMAC2
50	audprc_tx_ch1	/
51	audprc_tx_ch0	/
52	audprc_rx_ch1	/
53	audprc_rx_ch0	/
54	gptim2_cc2	/
55	gptim2_cc3	/
56	gptim2_cc4	/
57	sdmmc2	/
58	/	/
59	/	/
60	/	/
61	/	/
62	/	/
63	/	/

6.1.4 DMAC Functional Description

6.1.4.1 DMAC Block Diagram

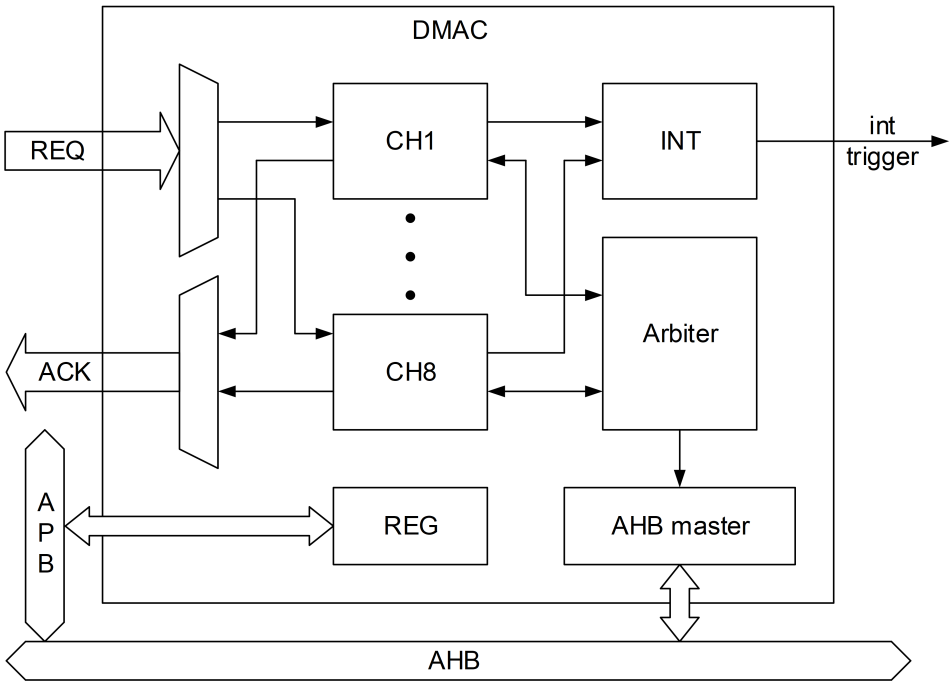


Figure 6-1: DMAC Block Diagram

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 cyclic 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, depending

on the target's AHB wait cycles.

6.1.4.3 Transfer Mode

Each channel of the DMAC can be independently configured for peripheral transfer mode or memory transfer mode, controlled by the CCRx_MEM2MEM register. The primary difference is whether the peripheral request/response mechanism is enabled. Peripheral transfer mode adapts to the peripheral's data bandwidth through the request/response mechanism, initiating transfers only after the peripheral is ready. This mode is typically used in scenarios such as UART transmission and reception, I2S audio input/output, and FLASH programming. Memory transfer mode does not require waiting for a request signal and transfers data at the maximum possible bandwidth, 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 DMACchannel;
2. The DMAC processes the request based on the priority of the relevant channel. When the channel holds the highest priority among all request channels, it initiates a single element transfer or block transfer on that channel;
3. After the completion of unit transfer or block transfer, DMAC sends an acknowledgment signal to the peripheral.
4. Upon receiving the acknowledgment signal from DMAC, the peripheral releases the request.
5. After 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, the transfer is initiated by software and repeatedly performs unit transfers until completion when the channel priority is sufcient.

6.1.4.5 Transfer Enable

The 8 channels of DMAC are enabled independently. When CCRx_EN is 1 and CNDTRx is not 0 , channel transfer starts; in peripheral transfer mode, it begins responding to peripheral requests, and in memory transfer mode, it starts the transfer 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 initiate the DMAC transfer. Prior to this, other parameters must be fully configured, or ensure that 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 comprises 3 steps:

1. Single bus read of source address data, configurable as single-byte/double-byte/four-byte;
2. Single write of the read data to the destination address, configurable as single-byte/double-byte/four-byte;
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 transfer unit counts, 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) , when CNDTRx decrements to 0, it immediately reloads the initial software-configured value and continues the transfer.

6.1.4.8 Circular Mode

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

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 transfers, access to the bus source and destination addresses can be independently configured as single-byte, double-byte, or four-byte types via CCRx_MSIZE and CCRx_PSIZE. DMAC does not provide data packing or unpacking functions; 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

Continued on next page...

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

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

6.1.4.11 Transfer Address

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

When configuring addresses, it is essential to note 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 divided into two separate transfers.

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

6.1.4.12 Channel Arbitration

The DMAC arbiter manages priority among different channels. When the arbiter grants priority to a valid channel (hardware request or software trigger), it initiates a unit transfer or block transfer for that channel. Subsequently, the arbiter re-arbitrates among the 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 lower CCRx_PL value has higher priority.
3. If both the transfer mode and CCRx_PL value are identical, the channel with the smaller channel number has higher priority (for example, channel 1has priority over channel 2).

During the transmission process of a given channel, if another channel with higher priority issues a transmission request,

the arbiter will switch the transmission to the higher priority channel after the current channel's unit transfer or block transfer completes.

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. When any of these three interrupts occur, a GIFx interrupt is generated. Channel interrupts can be cleared individually or collectively via CGIFx.

6.1.4.15 Channel configuration Procedure

When configuring the DMAC channel, the following steps must be followed:

1. Set the peripheral register address in the CPARx register.
   After a peripheral request occurs, or after enabling the channel in memory mode, data will be transferred from this address to memory or from memory to this address.
2. Set the memory address in the CM0ARx register.
   After a peripheral request occurs, or after enabling the channel in memory mode, data will be written to or read from memory.
3. Ensure CCRx_EN is 0, then write the number of data units to be transferred into the CNDTRx register.
   This value decrements after each data transfer.
4. At configure the following parameters in the CCRx register:
   - Channel Priority
   - Data Transfer Direction
   - Circular Mode
   - Peripheral and Memory Increment Mode
   - Peripheral and Memory Data Size
   - Interrupt enable for half-transfer completion and/or full completion, as well as transfer error occurrence
5. 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 0 to CCRx_EN 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 0to the CCRx_ENof the channel 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
[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

Continued on next page...

Table 6-4: DMAC Register Map (continued)

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

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

Continued on 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
0x20			**CNDTR2**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x24			**CPAR2**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x28			**CM0AR2**	

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

Continued on 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
0x34			**CNDTR3**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).

Continued on next page...

Table 6-4: DMAC Register Map (continued)

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

Continued on 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
0x48			**CNDTR4**	
[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).
0x4C			**CPAR4**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x50			**CM0AR4**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0x54			**CBSR4**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0x58			**CCR5**	
[31:15]			RSVD	
[14]	rw	1'h0	MEM2MEM	memory-to-memory mode 0: disabled 1: enabled
[13:12]	rw	2'h0	PL	priority level 00: low 01: medium 10: high 11: very high
[11:10]	rw	2'h0	MSIZE	memory size Defines the data size of each DMA transfer to the identified memory. In memory-to-memory mode, this field identifies the memory source if DIR = 1 and the memory destination if DIR = 0. In peripheral-to-peripheral mode, this field identifies the peripheral source if DIR = 1 and the peripheral destination if DIR = 0. 00: 8 bits 01: 16 bits 10: 32 bits 11: reserved

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

Continued on 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
0x70			**CNDTR6**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).
0x74			**CPAR6**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0x78			**CM0AR6**	

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

Continued on 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
0x84			**CNDTR7**	
[31:16]			RSVD	
[15:0]	rw	16'h0	NDT	number of data to transfer (0 to 2^{16} - 1) This field is updated by hardware when the channel is enabled: It is decremented after each single DMA 'read followed by write' transfer, indicating the remaining amount of data items to transfer. It is kept at zero when the programmed amount of data to transfer is reached, if the channel is not in circular mode (CIRC = 0 in the CCRx register). It is reloaded automatically by the previously programmed value, when the transfer is complete, if the channel is in circular mode (CIRC = 1). If this field is zero, no transfer can be served whatever the channel status (enabled or not).

Continued on next page...

Table 6-4: DMAC Register Map (continued)

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

Continued on 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
0x98			**CNDTR8**	
[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).
0x9C			**CPAR8**	
[31:0]	rw	32'h0	PA	peripheral address It contains the base address of the peripheral data register from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory destination address if DIR = 1 and the memory source address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral destination address DIR = 1 and the peripheral source address if DIR = 0.
0xA0			**CM0AR8**	
[31:0]	rw	32'h0	MA	peripheral address It contains the base address of the memory from/to which the data will be read/written. In memory-to-memory mode, this register identifies the memory source address if DIR = 1 and the memory destination address if DIR = 0. In peripheral-to-peripheral mode, this register identifies the peripheral source address DIR = 1 and the peripheral destination address if DIR = 0.
0xA4			**CBSR8**	
[31:8]			RSVD	
[7:0]	rw	8'h0	BS	burst size in non-m2m mode When BS>1, DMA will transfer for BS times for each request if left NDT is larger than BS, or else transfer for left NDT times. When BS=0 or 1, DMA will always do single transfer for each request. In memory-to-memory mode, BS is ignored.
0xA8			**CSELR1**	
[31:30]			RSVD	
[29:24]	rw	6'h0	C4S	DMA channel 4 selection
[23:22]			RSVD	
[21:16]	rw	6'h0	C3S	DMA channel 3 selection
[15:14]			RSVD	
[13:8]	rw	6'h0	C2S	DMA channel 2 selection
[7:6]			RSVD	
[5:0]	rw	6'h0	C1S	DMA channel 1 selection
0xAC			**CSELR2**	
[31:30]			RSVD	
[29:24]	rw	6'h0	C8S	DMA channel 8 selection
[23:22]			RSVD	
[21:16]	rw	6'h0	C7S	DMA channel 7 selection
[15:14]			RSVD	
[13:8]	rw	6'h0	C6S	DMA channel 6 selection
[7:6]			RSVD	
[5:0]	rw	6'h0	C5S	DMA channel 5 selection

6.2 ExtDMA

ExtDMA located in 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
[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

Continued on next page...

Table 6-5: ExtDMA Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
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

Continued on next page...

Table 6-5: ExtDMA Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.
[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 7 I2C interfaces, among which I2C1, I2C2, I2C3, and I2C4 are located in HPSYS, with input/output connected to IO(PA), and can send requests to DMAC1; I2C5, I2C6, and I2C7 are located in LPSYS, with input/output connected to IO(PB), and can send requests to DMAC2.

7.1.1 Introduction

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

7.1.2 Main Features

Can operate as both a master and a slave device simultaneously

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

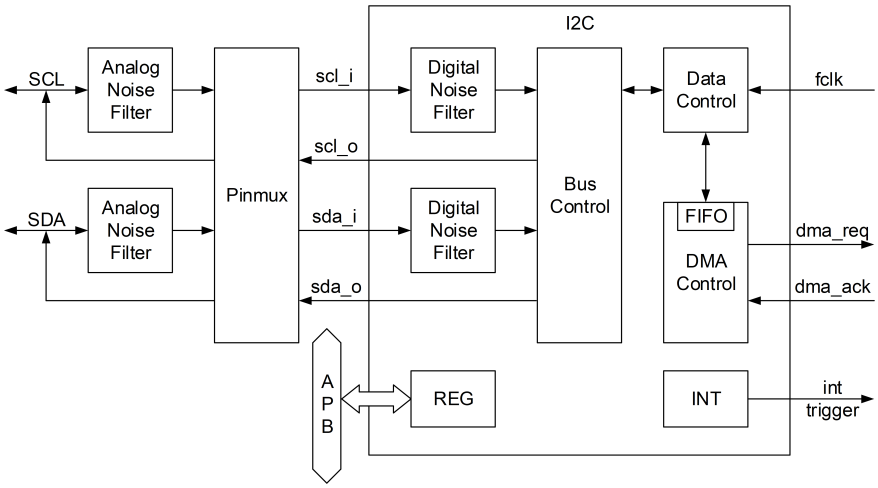


Figure 7-1: I2C Schematic Diagram

7.1.3 I2C Function Description

7.1.3.1 Two-Wire Transmission

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

7.1.3.2 Input Filter

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

7.1.3.3 Transmission Rate

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

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

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

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

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

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

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

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

7.1.3.4 Transmission sequence

A complete I2Ctransmission should follow the subsequent transmission sequence:

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

7.1.3.5 Operating Modes and States

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

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

7.1.3.6 I2C Initialization Process

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

7.1.3.7 Master Transmission Process

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

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

7.1.3.8 Master Receiving Process

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

7.1.3.9 Slave Transmisson process

1. When the address detection interrupt SAD is triggered, automatically respond with ACK.
2. SR_SAD Write 1 Clear the flag. Read SR_RWM , where 1 indicates sending and 0 indicates receiving. Assuming the current SR_RWM is 1 , enter the sending state.
3. Write the data to be transmitted into DBR.
4. TCR=TCR_TB.
5. Poll SR_TE until it reads 1, or wait for the TE interrupt.
6. Write 1 to SR_TE to clear the flag. Check SR_NACK ; if it reads 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 receiving process

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

7.1.3.11 DMA Transfer

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

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

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

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

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

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

7.1.3.12 Bus Exception Recovery

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

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

7.1.4 I2C Registers

Table 7-1: I2C Register Mapping Table

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

Continued on the next page...

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

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

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[15]	rw	1'h0	UFIE	FIFO Underflow Interrupt Enable 0 = FIFO Underflow interrupt is not enabled 1 = FIFO Underflow interrupt is enabled
[14]	rw	1'h0	OFIE	FIFO Overflow Interrupt Enable 0 = FIFO Overflow interrupt is not enabled 1 = FIFO Overflow interrupt is enabled
[13]	rw	1'h0	DMADONEIE	DMA Transaction Done Interrupt Enable 0 = DMA Transaction done interrupt is not enabled. 1 = DMA Transaction done interrupt is enabled.
[12]	rw	1'h0	MSDIE	Master Stop Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C unit to interrupt upon detecting a Master Stop sent by the I2C unit.
[11]			RSVD	
[10]	rw	1'h0	BEDIE	Bus Error Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt for the following I2C bus errors: As a master transmitter, no ACK was detected after a byte was sent. As a slave receiver, the I2C generated a NACK pulse. Software is responsible for guaranteeing that misplaced Start and Stop conditions do not occur.
[9]	rw	1'h0	SADIE	Slave Address Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt upon detecting a slave address match or a general call address.
[8]			RSVD	
[7]	rw	1'h0	RFIE	DBR Receive Full Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt when the DBR has received a data byte from the I2C bus.
[6]	rw	1'h0	TEIE	DBR Transmit Empty Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt after transmitting a byte onto the I2C bus.
[5]	rw	1'h0	ALDIE	Arbitration Loss Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt upon losing arbitration while in master mode.
[4]	rw	1'h0	SSDIE	Slave Stop Detected Interrupt Enable: 0 = Disable interrupt. 1 = Enables the I2C to interrupt when it detects a Stop condition while in slave mode.
[3:0]			RSVD	
0x0C			**SR**	Status register
[31:16]			RSVD	
[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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[17:9]	rw	9'h39	FLV	Decrementer Load value for Fast Mode (or Fast Mode Plus) SCL (master mode) for both high and low phase. Data rate is generated as Ffclk/(FLV+max(FLV,CNT*2+6)+7+DNF) approximately. 400kbps data rate is generated by default if fclk is 48MHz. FLV also controls setup time and hold time for START and STOP condition in Fast Mode(master mode). Thdsta=Tsusta=Tsusto=Tfclk*FLV
[8:0]	rw	9'hED	SLV	Decrementer Load value for Standard Mode SCL (master mode) for both high & low phase. Data rate is generated as Ffclk/(SLV+max(SLV,CNT*2+6)+7+DNF) approximately. 100kbps data rate is generated by default if fclk is 48MHz. SLV also controls setup time and hold time for START and STOP condition in Standard Mode(master mode). Thdsta=Tsusta=Tsusto=Tfclk*SLV
0x1C			**WCR**	Wait Count Register
[31:8]			RSVD	
[7:0]	rw	8'hA	CNT	Controls the counter values defining the setup and hold times in standard and fast mode Tvddat=Thddat=Tfclk*(CNT+2) Tsudat=max(Tlow-Thddat,Thddat) Lower counter values may violate setup and hold times.
0x20			**RCCR**	Bus Reset Cycle Counter Register
[31:4]			RSVD	
[3:0]	rw	4'h9	RSTCYC	The cycles of SCL during bus reset
0x24			**BMR**	Bus Monitor Register
[31:2]			RSVD	
[1]	r	1'h1	SCL	value of the SCL pin. Software can check bus level when the I2C bus is hung and the I2C unit must be reset.
[0]	r	1'h1	SDA	value of the SDA pin.
0x28			**DNR**	DMA number register
[31:9]			RSVD	
[8:0]	rw	9'h0	NDT	Write as number of data to transfer in byte. Read as left data number to transfer
0x30			**FIFO**	FIFO Register
[31:8]			RSVD	
[7:0]	rw	8'h0	DATA	Write to push send data into FIFO. Read to pop received data from FIFO

7.2 SPI

The chip contains 4 SPI units, among which SPI1 and SPI2 are located in HPSYS , with input/output connected to IO(PA) , capable of sending requests to DMAC1 ; SPI3 and SPI4 are located in LPSYS , with input/output connected to IO(PB) , capable of sending 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 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 / receive FIFO. Transmit FIFO.The transmit and receive FIFO share the same address; reading this address accesses the receive FIFO , while writing to this address accesses the transmit FIFO . The FIFO supports both software access mode and DMA access mode.

7.2.2 Main Features

- Supports 3communication formats: SSP/SPI/Microwire
- Supports data widths from 8 to 32Bit
- In SPI format, clock polarity and phase can be configured via registers SPOand SPH.
- Chip select signal polarity is configureable.
- FIFO depth is 32Bits×16Entries.
- Both transmit and receive support DMAmode.
- The maximum clock frequency of the SPIinHPSYSis 48MHz. The maximum clock frequency of the SPIinLPSYSis 24MHz.

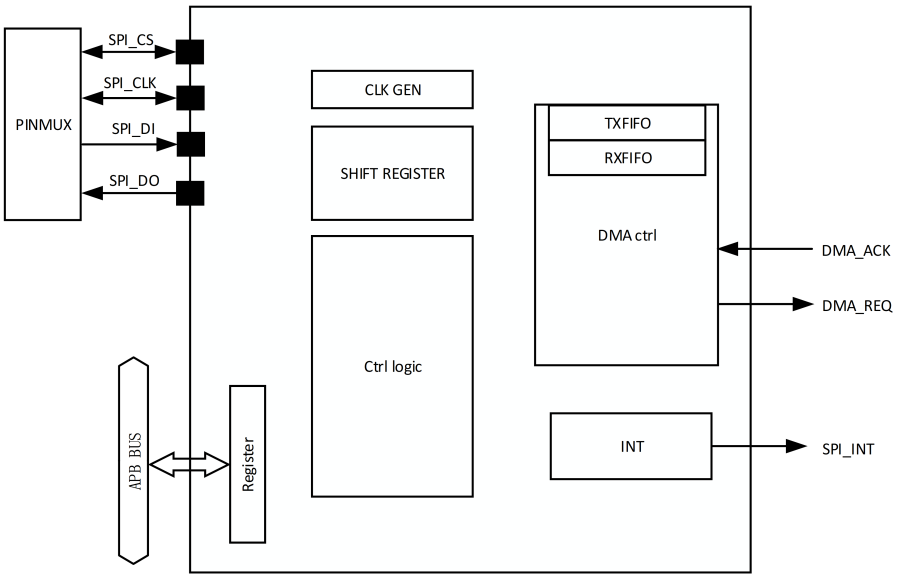


Figure 7-2: SPI Block Diagram

7.2.3 SPI Functional Description

7.2.4 Interface Signals

SPI_CS is used as the chip select signal or 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 transmitting one bit of data. 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 to communicate 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 a data transfer.

Under the Microwire protocol, SPI can only communicate as a Master; under this condition, 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 output data signal; data in theTX_FIFO is transmitted through SPI_DO in MSB first order. Regardless of operating as a Master or a Slave , it is always an output signal.

SPI_DI is the input data signal from external transmission; data received on SPI_DIis stored in the RX_FIFOand read by the CPU or DMA.

Whether operating as a Master or a Slave , it is always an input signal.

7.2.5 FIFO

The SPI controller contains TXFIFO and RXFIFO, both with a bit width of 32bits and a depth of 16entries. Both FIFOs can be accessed by the CPU or DMA

From the perspective of address mapping, both FIFOs share the same address; writing data to this address writes to the TXFIFO, while reading data from this address reads from the the earliest data in 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; for example,

If the bit width is not set to 32 bits, the SPI controller will ignore the bits above the set bit width within the 32 bits of the TXFIFO when transmitting. When receiving, the SPI controller will pad the bits above the set bit width with 0 before writing into the RXFIFO.

The FIFO can interact with the CPU via interrupts and software, or the CPU can poll the FIFO status registers to facilitate 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 necessary to properly configure settings to prevent RXFIFO overflow or TXFIFO underrun.

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 the SPI_CLK clock when the SPI controller is either disabled or in the IDLE state. When SPO is 0, the SPI_CLK polarity is low, and the first edge is a rising edge; When SPO is 1, the SPI_CLK polarity is high, and the first edge is a falling edge.SPH configures the timing of the clock edges for driving 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, SPI_DO outputs the MSB of the data to be transmitted); WhenSPH is 1, the SPI controller drives SPI_DO from the first edge of SPI_CLK and samples data on SPI_DI from the second edge of SPI_CLK . The specific communication timing is shown in Figures 7-3 and 7-4.

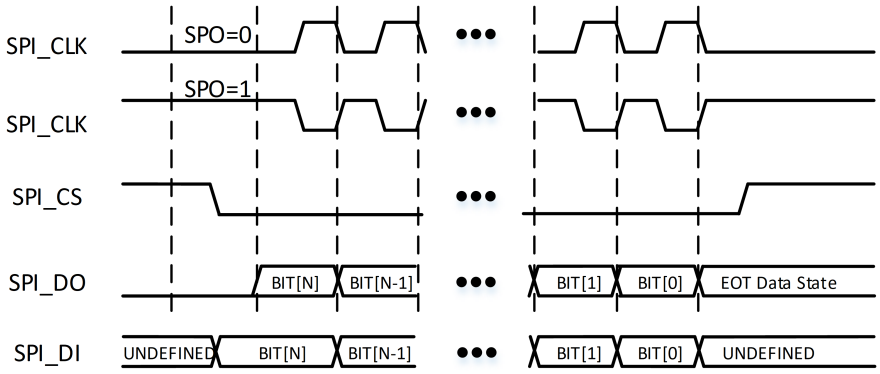


Figure 7-3: SPI communication when SPH is 0

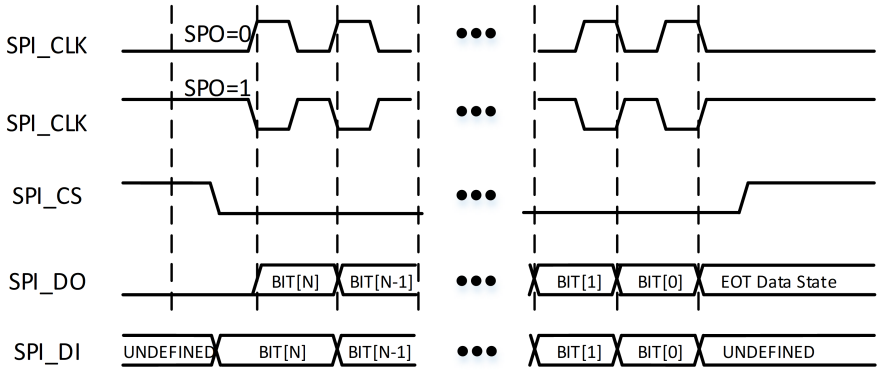


Figure 7-4: SPI communication when SPH is 1

The SPI protocol communication process is described below with reference to Figure 7-3 for the case of 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 transmission and remains low until the current data frame transmission is complete. After half a SPI_CLK cycle, the data for this transmission of MSB is driven onto the SPI_DO line. After another half SPI_CLK cycle, the SPI_CLK rises to initiate the first rising edge and sample SPI_DI .

Before all data for this transmission is fully sent, the SPI_CLK continues toggling at the configured frequency. After the fnal sampling, the SPI_CLK returns to a low level. After half a SPI_CLK cycle, the SPI_CS is pulled high to end this transmission.

When operating as a 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.

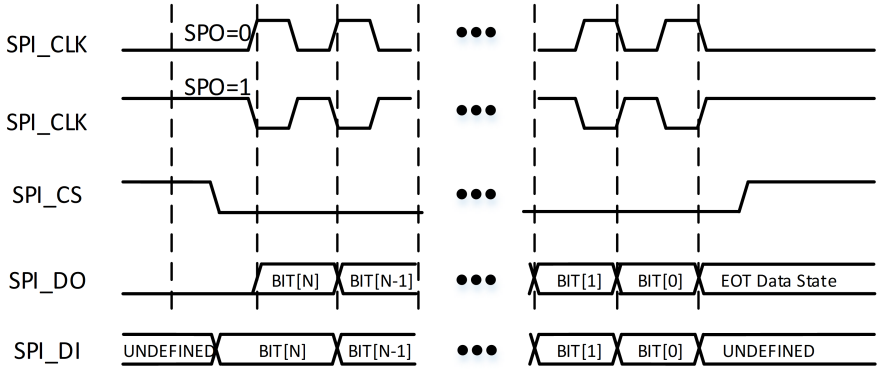


Figure 7-5: SPI Protocol Continuous Transmission Timing

In practical use, scenarios exist where SPI_CSis pulled low once to transmit multiple data frames. In such cases, if the TXFIFOdata is not flled promptly,it may cause SPI_CS 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, SPI_CS issues a high pulse with a width of one clock cycle to indicate the start of transmission. Subsequently, 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.

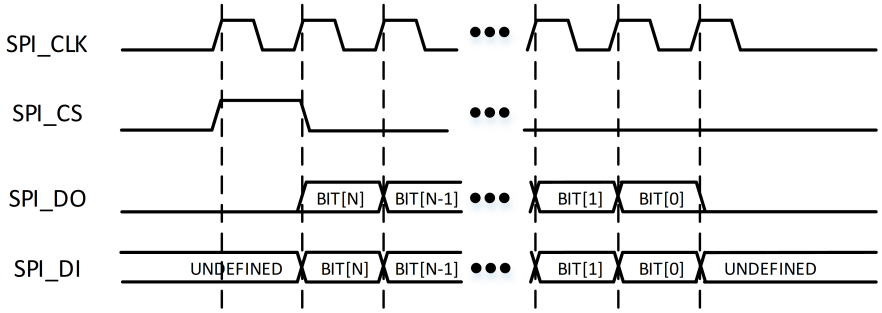


Figure 7-6: TI-SSP Protocol Communication Timing

When operating as a MASTER, 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.

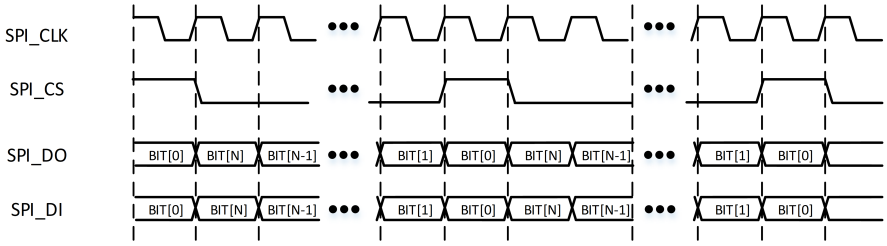


Figure 7-7: TI-SSP Protocol Continuous Communication Timing

If multiple data entries are transmitted consecutively, and data B immediately follows data A, during the last bit transmission of data A, SPI_CS is pulled high.

One cycle, and the MSB of data B is transmitted starting at the next rising edge of SPI_CLK.

- Micro wireprotocol

  The Microwire protocol is a half-duplex protocol. The SPI controller supports communication only as a Master. During communication, the Master first sends 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.

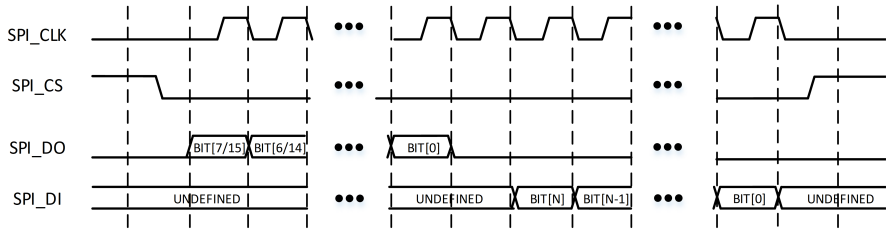


Figure 7-8: Timing diagram of a single communication in the Microwireprotocol

For consecutive transfers, the command word of the next transfer 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 consecutive transfers is shown in Figure 7-9.

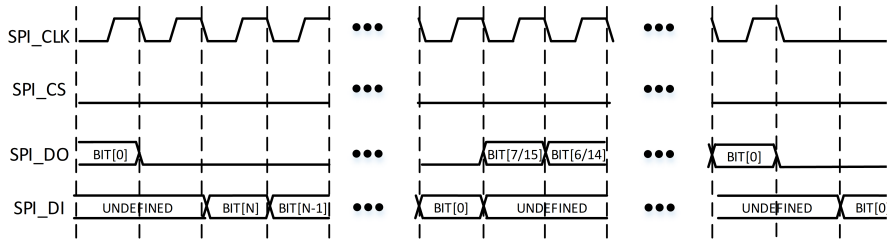


Figure 7-9: Microwire Protocol Continuous Transmission Timing

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 selected as 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 Configuration:
   - Configure the communication protocol via the FRF feld in the TOP_CTRL register.
   - FFRF=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 externally driven

signals.

- configure theMaster/Slavemode of SPI_CSvia the SFRMDIRfeld in the TOP_CTRLregister:
- SFRMDIR=0 ： Master mode, SFRMDIR=1:Slavemode
- configure theMaster/Slavemode of SPI_CLKvia the SCLKDIRfeld in the TOP_CTRLregister:
- SCLKDIR=0 ： Master mode,SCLKDIR=1:Slavemode
- Generally,SFRMDIRand SCLKDIRare configured to the same mode.

4. Clock frequency Configuration:
   - he procedure to configure the SPIclock is as follows:
     (a) Set the CLK_DIVfeld in the CLK_CTRLregister to configure the clock division ratio.
     (b) Set the CLK_SEL feld in the CLK_CTRL register to select the clock source,CLK_SEL=0 : select divided clock, CLK_SEL=1 : Select the source clock.
     (c) Set the CLK_SSP_EN feld in the CLK_CTRL register to enable the SPI clock, 1 : enable the SPI clock.
   - The SPI clock is derived from either the source clock or the divided source clock. The frequency of the divided clock is the source clock frequency divided by CLK_DIV. The source clock frequency is 48MHz in HPSYS and 24MHz in LPSYS.
5. Data width Configuration:
   - Supports data widths from 8 to 32 bits. The data bit width can be configured by the DSS feld 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 TXFIFO, and received data is read from RXFIFO.
   - Data operations can be performed either by direct CPU software access to the FIFO, or via DMA.
   - When data is operated by the CPU, the SPI controller typically notifes the CPU via interrupts to write toTXFIFO or read from RXFIFO. The interrupt corresponding to TXFIFO is enabled by setting the register bit TIE to 1 . Once enabled, when the number of data items in theTXFIFO is less than or equal to the value in the register of TFT , the SPI controller issues a TX interrupt notifcation to the CPU . The interrupt corresponding to RXFIFO is enabled by setting the register bit RIE to 1 . Once enabled, when the number of data items in the RXFIFO exceeds the value RFT in the register, the SPI controller issues an RX interrupt notifcation to the CPU .
   - The CPU can also operate the FIFO by polling the FIFO status. FIFO STATUS register

STATUS register	Meaning
TNF	0：TXFIFO is full；1：TXFIFO is not full。
TFL	The number of data items in the TXFIFO. When the read value is 0 , the TXFIFO is either full or empty; determination requires reference to the value of TNF.
TUR	1: An UNDERRUN has occurred in the TXFIFO , meaning the SPI controller performed a TXFIFO read operation while the TXFIFO was empty.
RNE	0: The RXFIFO is empty; 1: TheRXFIFOis not empty.
RFL	The number of data items in the RXFIFO. When the read value is 0xF , the RXFIFO is either empty or full; determination requires reference to the value of RNE.
ROR	1: An OVERRUN has occurred in the RXFIFO , meaning the SPI controller performed a RXFIFO write operation while the RXFIFO was full.

   - The CPU can add data to be transmitted into the TXFIFO by reading the STATUS register, provided that the TXFIFO is not full; Data received can be read from the RXFIFO when the RXFIFO is not empty.
   - When using DMA for data operations, the SPI controller initiates data handling by issuing a request to DMA

to start the DMA. Writing to RSRE 1 enables the DMA function for RX data. When the data in the RX FIFO exceeds RFT, theSPI issues a DMA request. Writing to TSRE 1 enables the DMAfunction for TXdata. When the data in theRX FIFO exceeds RFT,theSPIissues a DMA request.

7. Enable the SPIcontroller.
   - Set SSE in the register TOP_CTRL to 1 to enable the SPI controller.
8. Disable the SPIcontroller.
   - Query the STATUSregister for BUSY; if BUSY is 0, it indicates that the SPIcontroller is currently idle. Then write 0to SSE, which disables the SPIcontroller.
9. SPI_CS Signal Control
   - In certain communication scenarios, the SPI_CS signal must remain low throughout multiple data transmissions. By default, if the TXFIFO becomes empty during transmission, the SPI_CS signal may be pulled high intermittently and then pulled low again once the TXFIFO is non-empty. Software can be used to ensure that the SPI_CS signal maintains a stable low level during communication. Before initiating communication, set HOLD_FRAME_LOW in the TOP_CTRL register to 1 to ensure the SPI_CS signal remains low throughout the communication. Note to set HOLD_FRAME_LOW to 0 after the communication is completed 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 SPIcontroller
  7. Access the FIFOto start transmission and reception
  8. Complete transmission and reception, then disable the SPIcontroller

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, the SPI controller will toggle SPI_CLK regardless of whether there is data to be transmitted in the TXFIFO. Meanwhile, data received from SPI_DI will be stored in the RXFIFO. The usage of Receive-Only mode is configured as follows:

- Set the RWOT_CCMregister, whose value specifes the required number of SPI_CLKcycles.
- Set bothSET_RWOT_CYCLEand RWOT_CYCLEin the RWOT_CTRLregister to 1to enable the RWOT counter.
- Set RWOT in the RWOT_CTRL register to 1to enable theReceive-Onlymode.
- Set RWOT in the RWOT_CTRL register to 1to enable theReceive-Onlymode.

After completing the above Configurations and enabling the SPI controller, the SPI_CLK will 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 SPI controller supports three-wire mode with 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 function of this PIN 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, after transmission, software does not need to process the RXFIFO .
5. When receiving data, set TXD_OEN in the register TRIWIRE_CTRL to 1. If SPI_CLK operates in master mode, the SPI controller must drive SPI_CLK. There are two methods to drive SPI_CLK:
   - Write an equal amount of data to the TXFIFO as the data to be received in order to drive SPI_CLK.
   - Using Reserve-Only mode, the procedure is as follows:
     (a) Disable the SPIcontroller.
     (b) Set the RWOT_CCM register, whose value specifes the required number of SPI_CLKcycles.
     (c) Set both SET_RWOT_CYCLE and RWOT_CYCLE in the RWOT_CTRL register to 1to 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 stably low during the process. To maintain a stable 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	

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11]	rw	1'b0	SPH	Motorola SPI SPI_CLK phase setting 0: SPI_CLK is inactive until one cycle after the start of a frame and active until 1/2 cycle before the end of a frame 1: SPI_CLK is inactive until 1/2 cycle after the start of a frame and active until one cycle before the end of a frame
[10]	rw	1'b0	SPO	Motorola SPI SPI_CLK Polarity Setting 0: The inactive or idle state of SPI_CLK is low 1: The inactive or idle state of SPI_CLK is high
[9:5]	rw	5'h0	DSS	SPI controller Work data size, register bits value 7~31 indicated data size 8~32 bits, usually use data size 8bits, 16bits, 24bits, 32bits
[4]	rw	1'b0	SFRMDIR	SPI_CS Direction 0: Master mode, SPI controller drives SPI_CS 1: Slave mode, SPI controller receives SPI_CS
[3]	rw	1'b0	SCLKDIR	SPI_CLK Direction 0: Master mode, SPI controller drives SPI_CLK 1: Slave mode, SPI controller receives SPI_CLK
[2:1]	rw	2'h0	FRF	Frame Format 0x0: Motorola* Serial Peripheral Interface (SPI) 0x1: Texas Instruments* Synchronous Serial Protocol (SSP) 0x2: National Semiconductor Microwire* 0x3: RSVD
[0]	rw	1'b0	SSE	SPI controller Enable 0: SPI controller is disabled 1: SPI controller is enabled
0x04			**FIFO_CTRL**	FIFO Control Register
[31:18]			RSVD	
[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]

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13:12]	rw	2'h0	RXFIFO_RD_ENDIAN	apb_prdata Read from Rx FIFO Endian 0x0 = apb_prdata[31:0] = rxfifo_wdata[31:0] 0x1 = apb_prdata[31:0] = rxfifo_wdata[15:0], rxfifo_wdata[31:16] 0x2 = apb_prdata[31:0]= rxfifo_wdata[7:0], rxfifo_wdata[15:8], rxfifo_wdata[23:16], rxfifo_wdata[31:24] 0x3 = apb_prdata[31:0]= rxfifo_wdata[23:16], rxfifo_wdata[31:24], rxfifo_wdata[7:0], rxfifo_wdata[15:8]
[11]	rw	1'h0	RSRE	Receive Service Request Enable 0: RxFIFO DMA service request is disabled 1: RxFIFO DMA service request is enabled
[10]	rw	1'h0	TSRE	Transmit Service Request Enable 0: TxFIFO DMA service request is disabled 1: TxFIFO DMA service request is enabled
[9:5]	rw	5'h0	RFT	RXFIFO Trigger Threshold This field sets the threshold level at which RXFIFO asserts interrupt. The level should be set to the preferred threshold value minus 1.
[4:0]	rw	5'h0	TFT	TXFIFO Trigger Threshold This field sets the threshold level at which TXFIFO asserts interrupt. The level should be set to the preferred threshold value minus 1.
0x08			**INTE**	Interrupt Enable Register
[31:6]			RSVD	
[5]	rw	1'h0	TIM	Transmit FIFO Underrun Interrupt Mask 0 : TUR events generate an SPI interrupt 1 : TUR events do NOT generate an SPI interrupt
[4]	rw	1'h0	RIM	Receive FIFO Overrun Interrupt Mask 0: ROR events generate an SPI interrupt 1: ROR events do NOT generate an SPI interrupt
[3]	rw	1'h0	TIE	Transmit FIFO Interrupt Enable 0: TxFIFO threshold-level-reached interrupt is disabled 1: TxFIFO threshold-level-reached interrupt is enabled
[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	

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23]	r	1'h0	OSS	Odd Sample Status 0: RxFIFO entry has two samples 1: RxFIFO entry has one sample Note that this bit needs to be looked at only when FIFO Packing is enabled (FPCKE field in FIFO Control Register is set). Otherwise, this bit is zero. When SPI controller is in Packed mode and the CPU is used instead of DMA to read the RxFIFO, the CPU should make sure that [Receive FIFO Not Empty] = 1 AND this field = 0 before it attempts to read the RxFIFO.
[22]	r	1'h0	TX_OSS	TX FIFO Odd Sample Status When SPI controller is in packed mode, the number of samples in the TX FIFO is: ([Transmit FIFO Level]*2 + this field), when [Transmit FIFO Not Full] = 1 32, when [Transmit FIFO Not Full] = 0. The TX FIFO cannot accept new data when [Transmit FIFO Not Full] = 1 and [Transmit FIFO Level] = 15 and this field = 1. (The TX FIFO has 31 samples). 0: TxFIFO entry has an even number of samples 1: TxFIFO entry has an odd number of samples Note that this bit needs to be read only when FIFO Packing is enabled ([FIFO Packing Enable] in the FIFO Control Register is set). Otherwise, this bit is zero.
[21]			RSVD	
[20]	w1c	1'h0	ROR	Receive FIFO Overrun 0: RXFIFO has not experienced an overrun 1: Attempted data write to full RXFIFO, causes an interrupt request
[19]			RSVD	
[18:15]	r	4'h0	RFL	Receive FIFO Level This field is the number of entries minus one in RXFIFO. When the value 0xF is read, the RXFIFO is either empty or full, and software should read the [Receive FIFO Not Empty] field.
[14]	r	1'h0	RNE	Receive FIFO Not Empty 0: RXFIFO is empty 1: RXFIFO is not empty
[13]	r	1'h0	RFS	Receive FIFO Service Request 0: RXFIFO level is at or below RFT threshold (RFT) or SPI controller is disabled 1: RXFIFO level exceeds RFT threshold (RFT), causes an interrupt request
[12]	w1c	1'h0	TUR	Transmit FIFO Underrun 0: The TXFIFO has not experienced an underrun 1: A read from the TXFIFO was attempted when the TXFIFO was empty, causes an interrupt if it is enabled ([Transmit FIFO Underrun Interrupt Mask] in the INT EN Register is 0)
[11]			RSVD	
[10:7]	r	4'h0	TFL	Transmit FIFO Level This field is the number of entries in TXFIFO.When the value 0x0 is read, the TXFIFO is either empty or full, and software should read the [Transmit FIFO Not Full] field.

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	r	1'h0	TNF	Transmit FIFO Not Full 0: TXFIFO is full 1: TXFIFO is not full
[5]	r	1'h0	TFS	Transmit FIFO Service Request 0: TX FIFO level exceeds the TFT threshold (TFT + 1) or SPI controller is disabled 1: TXFIFO level is at or below TFT threshold (TFT + 1), causes an interrupt request
[4]			RSVD	
[3]	w1c	1'h0	TINT	Receiver Time-out Interrupt 0: No receiver time-out is pending 1: Receiver time-out pending, causes an interrupt request
[2]			RSVD	
[1]	r	1'h0	CSS	Clock Synchronization Status 0: SPI controller is ready for slave clock operations 1: SPI controller is currently busy synchronizing slave mode signals
[0]	r	1'h0	BSY	SPI controller Busy 0: SPI controller is idle or disabled 1: SPI controller is currently transmitting or receiving framed data
0x24			**RWOT_CTRL**	RWOT Control Register
[31:5]			RSVD	
[4]	rw	1'h0	MASK_RWOT_LAST_SAMPLE	Mask last_sample_flag in RWOT Mode 1: Mask 0: Unmask
[3]	rw	1'h0	CLR_RWOT_CYCLE	Clear Internal rwot_counter This field clears the rwot_counter to 0. This field is self cleared after SSE = 1. 1: Clear rwot_counter
[2]	rw	1'h0	SET_RWOT_CYCLE	Set RWOT Cycle This field is used to set the value of the RWOT_CCM register to the internal rwot_counter. This field is self-cleared after SSE = 1. 1: Set rwot_counter
[1]	rw	1'h0	CYCLE_RWOT_EN	Enable RWOT Cycle Counter Mode 1: Enable
[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

Continued on next page...

Table 7-2: SPI Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:9]			RSVD	
[8]	rw	1'h0	CLK_EN	enable clk for internal logic
[7]	rw	1'h0	CLK_SEL	0: select clk_div as clk for SPI controller 1: select clk_sys as clk for SPI controller
[6:0]	rw	7'h0	CLK_DIV	div ratio from clk_sys
0x54			**TRIWIRE_CTRL**	Three Wire Mode Control Register
[31:3]			RSVD	
[2]	rw	1'h0	WORK_WIDTH_DYN_CHANGE	WORK_WIDTH_DYN_CHNAGE 1: SW can dynamicly change TOP_CTRL[9:5] without disabling TOP_CTRL[0] and re-enabling TOP_CTRL[0]
[1]	rw	1'h0	TXD_OEN	TXD_OEN control when TRI-WIRE mode 1: SPI_DIO is input 0: SPI_DIO is output
[0]	rw	1'h0	SPI_TRI_WIRE_EN	SPI_THREE_WIRE_MODE_EN 0: normal mode 1: enable TRI-WIRE mode

7.3 UART

The chip contains a total of 6 USARTs, among which USART 1, USART 2, and USART 3 are located in HPSYS, with input/output connected to IO(PA), and can send requests to DMAC1; USART4, USART5, and USART6 are located in LPSYS, with input/output connected to IO(PB), enabling request transmission to DMAC.The Universal Asynchronous Receiver/Transmitter supports full-duplex mode, providing baud rates up to 6Mbps and multiple configurable data formats, offering a flexible and effcient data ex change method for communication with external standardized devices. It also supports DMA for multi-packet transmission and reception

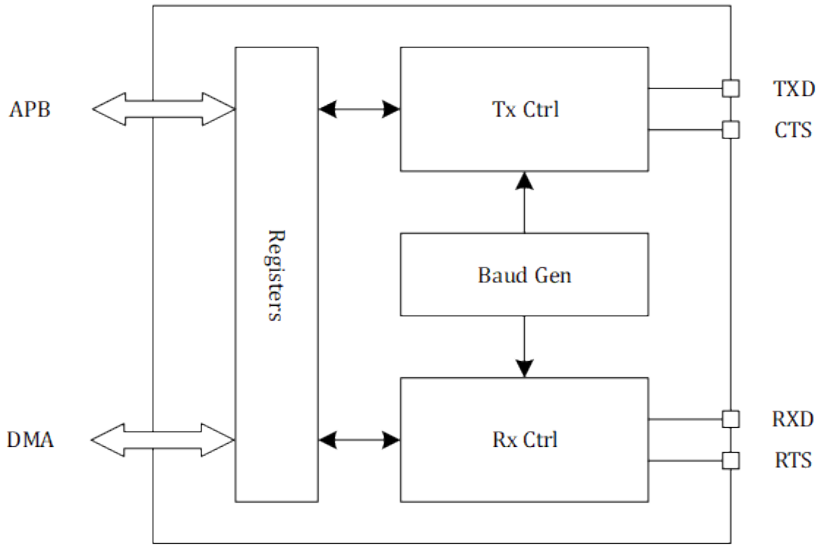


Figure 7-10: Universal Asynchronous Receiver/Transmitter

Key features of the Universal Asynchronous Receiver/Transmitter:

- Full-duplex asynchronous communication
- configurable 16times or 8times oversampling, with options to prioritize frequency or clock tolerance

- Flexible baud rate configuration; when the input clock is 48MHz and the oversampling rate is 16,the baud rate is 3Mbps
- Configurable packet length (7/8/9 bits)
- Configurable stop bits (1/2 bits)
- Hardware flow control (CTS/RTS)
- DMA multi-packet transmission and reception
- 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 fixed at 48 MHz, 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

Continued on next page...

Table 7-3: USART Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw	1'h0	TCIE	Transfer compelete interrupt enable 0: interrupt disabled 1: interrupt is generated whenever TC=1 in the ISR register
[5]	rw	1'h0	RXNEIE	Rx not empty interrupt enable 0: interrupt disabled 1: interrupt is generated whenever RXNE=1 in the ISR register
[4]	rw	1'h0	IDLEIE	Idle line interrupt enable 0: interrupt disabled 1: interrupt is generated whenever IDLE=1 in the ISR register
[3]	rw	1'h0	TE	Transmitter enable 0: transmitter is disabled 1: transmitter is enabled
[2]	rw	1'h0	RE	Receiver enable 0: receiver is disabled 1: receiver is enabled
[1]			RSVD	
[0]	rw	1'h0	UE	USART enable 0: disabled 1: enabled
0x04			**CR2**	Control Register 2
[31:24]			RSVD	
[23]			RSVD	
[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	

Continued on next page...

Table 7-3: USART Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13]			RSVD	
[12]	rw	1'h0	OVRDIS	Overrun disable 0: overrun error flag (ORE) will be set if new data received but previous data not read. New data will not overwrite the content in RDR register. 1: overrun disabled. If new data is received before previous data is read, the new data will overwrite the content in RDR register and ORE flag remains unset.
[11]	rw	1'h0	ONEBIT	One bit sampling mode 0: 3-bit sampling mode, the sampling value is determined by the voted result out of 3 bits 1: 1-bit sampling mode
[10]	rw	1'h0	CTSIE	CTS interrupt enable 0: interrupt disabled 1: interrupt is generated whenever CTSIF=1 in the ISR register
[9]	rw	1'h0	CTSE	CTS enable 0: CTS hardware flow control disabled 1: CTS hardware flow control enabled, data is transmitted only when CTS input is asserted low
[8]	rw	1'h0	RTSE	RTS enable 0: RTS hardware flow control disabled 1: RTS hardware flow control enabled, RTS output is asserted low when new data can be received
[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	

Continued on next page...

Table 7-3: USART Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]			RSVD	
[0]			RSVD	
0x1C			**ISR**	Interrupt and Status Register
[31:26]			RSVD	
[25]			RSVD	
[24:23]			RSVD	
[22]			RSVD	
[21]			RSVD	
[20]			RSVD	
[19]			RSVD	
[18]			RSVD	
[17]			RSVD	
[16]			RSVD	
[15]			RSVD	
[14]			RSVD	
[13]			RSVD	
[12]			RSVD	
[11]			RSVD	
[10]	r	1'h0	CTS	CTS input. Read this bit to get the raw status of the CTS line.
[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

Continued on next page...

Table 7-3: USART Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x20			**ICR**	Interrupt flag Clear Register
[31:21]			RSVD	
[20]			RSVD	
[19:18]			RSVD	
[17]			RSVD	
[16:13]			RSVD	
[12]			RSVD	
[11]			RSVD	
[10]			RSVD	
[9]	w1c	1'h0	CTSCF	CTS clear flag. Writing 1 to this bit clears the CTSIF flag in the ISR register.
[8]			RSVD	
[7]			RSVD	
[6]	w1c	1'h0	TCCF	Transmission complete clear flag. Writing 1 to this bit clears the TC flag in the ISR register.
[5]			RSVD	
[4]	w1c	1'h0	IDLECF	Idle line detected clear flag. Writing 1 to this bit clears the IDLECF flag in the ISR register.
[3]	w1c	1'h0	ORECF	Overrun error clear flag. Writing 1 to this bit clears the ORE flag in the ISR register.
[2]	w1c	1'h0	NCF	Noise detected clear flag. Writing 1 to this bit clears the NF flag in the ISR register.
[1]	w1c	1'h0	FECF	Framing error clear flag. Writing 1 to this bit clears the FE flag in the ISR register.
[0]	w1c	1'h0	PECF	Parity error clear flag. Wriring 1 to this bit clears the PE flag in the ISR register.
0x24			**RDR**	Receive Data Register
[31:9]			RSVD	
[8:0]	r	9'h0	RDR	Received data
0x28			**TDR**	Transmit Data Register
[31:9]			RSVD	
[8:0]	rw	9'h0	TDR	Transmit data
0x2C			**MISCR**	Miscellaneous Register
[31]	rw	1'h0	AUTOCAL	
[30:8]			RSVD	
[7:4]	rw	4'h2	RTSBIT	assert RTS ahead of the frame completion (in number of bits) Reserved-Do not modify
[3:0]	rw	4'h6	SMPLINI	initial sample count, count down from this value to zero to reach the middle of the start bit in Rx Reserved-Do not modify
0x30			**DRDR**	Debug Receive Data Register
[31:0]	r	32'h0	DATA	DbgUART is still a preliminary version in this project.
0x34			**DTDR**	Debug Receive Data Register
[31:0]	rw	32'h0	DATA	
0x38			**EXR**	Mutual Exclusive Register
[31:5]			RSVD	
[4]	r	1'b0	ID	
[3:1]			RSVD	
[0]	rw	1'h1	BUSY	

7.4 PTC

The chip contains 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 sequence to execute 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 channel can select an independent trigger source and configure 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 activate tasks on other channels. Each channel can be configured for the number of triggers. Some channels support executing tasks after a configurable delay following the trigger.

7.4.2 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 spaces, supporting only word-aligned access.
- Supports direct data writing or read-modify-write operations
- Supports 32-bit XOR/AND/OR/addition operations
- configurable trigger count: 1 1023, or infnite triggers
- configurable trigger delay: 0 65535 HCLK cycles
- Fixed priority arbitration; lower channel number indicates higher priority
- 4 word register space allocated for data buffering

7.4.3 Functional description

7.4.3.1 Channel trigger

Each channel can select one trigger source from 128 options, configured by register 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 channel completion events of the PTC itself. The polarity of the trigger source can be selected via TCRx_TRIGPOL. When selecting the trigger source IO input signals, only 4 out of every 32 IO can be simultaneously selected as trigger sources. The specific selection is configured through registers such as PTC GPIO31_0, GPIO63_32. IO trigger sources do not support debounce.

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

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	USB_RX	USB_TX	I2S1_OF	I2S1_UF	/	TRNG_RANDGEN	TRNG_SEEDGEN	HCPU_SLEEPDEEP	112
111	EPIC_LINEHIT	EPIC_DONE	NNACC1_DONE	LCDC1_ERR	LCDC1_LINE	LCDC1_LINEHIT	LCDC1_FMARK	LCDC1_DONE	104
103	LCDC1_BUSY	/	EZIP1_ROW	EZIP1_DONE	USART3_TXBYTE	USART3_RXBYTE	EXTDMA_HT	EXTDMA_TC	96
95	/	/	/	/	USART2_TXBYTE	USART2_RXBYTE	USART1_TXBYTE	USART1_RXBYTE	88
87	SDMMC2_DATIDLE	SDMMC2_CMDBUSY	SDMMC1_TC	SDMMC1_CC	SPI2_START	SPI2_DONE	SPI1_START	SPI1_DONE	80
79	I2C3_RF	I2C3_TE	I2C3_DMADONE	FACC1_DONE	I2C2_RF	I2C2_TE	I2C2_DMADONE	AES_DONE	72
71	I2C1_RF	I2C1_TE	I2C1_DMADONE	FFT1_DONE	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_C4INT7	MAILBOX1_C3INT7	MAILBOX1_C2INT7	MAILBOX1_C1INT7	/	/	/	/	48
47	BUSMON1_OF8	BUSMON1_OF7	BUSMON1_OF6	BUSMON1_OF5	BUSMON1_OF4	BUSMON1_OF3	BUSMON1_OF2	BUSMON1_OF1	40
39	DMAC1_HT8	DMAC1_HT7	DMAC1_HT6	DMAC1_HT5	DMAC1_HT4	DMAC1_HT3	DMAC1_HT2	DMAC1_HT1	32
31	DMAC1_TC8	DMAC1_TC7	DMAC1_TC6	DMAC1_TC5	DMAC1_TC4	DMAC1_TC3	DMAC1_TC2	DMAC1_TC1	24
23	/	ATIM1_COM	ATIM1_CH4	ATIM1_CH3	ATIM1_CH2	ATIM1_CH1	ATIM1_TRIG	ATIM1_UPDATE	16
15	BTIM2_UPDATE	BTIM1_UPDATE	GPTIM2_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	HCPU_SLEEPING	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	/	TSEN_DONE	/	/	RF_UNLOCK	RF_PKTDET	112
111	BT_EDRSYNC	BT_RCCALDONE	BT_ISOSYNC	BT_SLPSTATUS	BT_FRAME	BT_BLEEVENT	BT_PKTDET	BT_CRCERR	104
103	BT_RXDONE	BT_PHYRX	BT_RFRX	BT_TXDONE	BT_PHYTX	BT_RFTX	BT_KICKOFF	BT_INPROCESS	96
95	/	/	USART6_TXBYTE	USART6_RXBYTE	USART5_TXBYTE	USART5_RXBYTE	USART4_TXBYTE	USART4_RXBYTE	88
87	BT_MODE	BT_COLLISION	BT_PRIORITY	BT_ACTIVE	SPI4_START	SPI4_DONE	SPI3_START	SPI3_DONE	80
79	I2C7_RF	I2C7_TE	I2C7_DMADONE	/	I2C6_RF	I2C6_TE	I2C6_DMADONE	/	72
71	I2C5_RF	I2C5_TE	I2C5_DMADONE	/	/	/	/	/	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_C2INT7	MAILBOX2_C2INT6	MAILBOX2_C2INT5	MAILBOX2_C2INT4	MAILBOX2_C1INT7	MAILBOX2_C1INT6	MAILBOX2_C1INT5	MAILBOX2_C1INT4	48
47	BT_EDR3	BT_EDR2	BT_BRLASTBIT	BT_EDRLASTBIT	BUSMON2_OF4	BUSMON2_OF3	BUSMON2_OF2	BUSMON2_OF1	40
39	DMAC2_HT8	DMAC2_HT7	DMAC2_HT6	DMAC2_HT5	DMAC2_HT4	DMAC2_HT3	DMAC2_HT2	DMAC2_HT1	32
31	DMAC2_TC8	DMAC2_TC7	DMAC2_TC6	DMAC2_TC5	DMAC2_TC4	DMAC2_TC3	DMAC2_TC2	DMAC2_TC1	24
23	LCPU_SLEEPDEEP	LCPU_SLEEPING	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 contained in the TDRx register is directly written to the address pointed to by the TARx register. When TCRx_OP is 0x4 0x7, after the channel is triggered, the data at the address pointed to by the TARx register is first read, then XOR, AND, OR, or ADD operations are 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 a peripheral register address; therefore, the usual 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 status.

By default,TCRx_REPEN is 0 , allowing the channel to be triggered to execute tasks an unlimited number of times. When TCRx_REPEN is 1 , the number of times the channel executes tasks can be specifed through the RCRx_REP register, with a maximum of 1023 times. When RCRx_REP is greater than 0 , each trigger generated by the channel executes the task once and decrements RCRx_REP by 1 ; Once RCRx_REP decrements to 0 , even if trigger conditions occur, task execution will not be initiated.

After the channel performs a write operation to the address pointed to by the TARx register, a channel completion event

is generated, which can serve as a PTC trigger source for another channel. Simultaneously, the ISR_TCIfix flag is set, and when IER_TCIEx is 1 , an interrupt is generated. When TCRx_REPIRQ is 1, an interrupt is generated only after all tasks specifed by RCRx_REP are completed; otherwise, an interrupt is generated after each task completion. When TCRx_REPTRIG is 1, PTC triggers indicating channel completion are generated only after all tasks specifed by RCRx_REP are completed; otherwise, a PTC trigger is generated after each task completion.

If the address pointed to by the TARx Register is a bus address inaccessible by the PTC, a bus error occurs during channel access, which sets the ISR_TEIfix flag and generates an interrupt when IER_TEIE is 1.

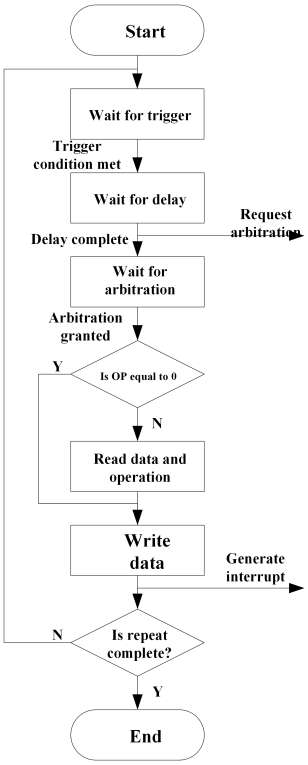


Figure 7-11: PTC Channel Execution Flowchart

7.4.3.3 Channel Arbitration

PTC has a total of 8 channels, arbitrated according to the principle that channels with smaller channel numbers have 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 start task execution first, while the other channels remain in a pending state until the task of the channel with the smallest number 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

Continued on next page...

Table 7-6: PTC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[22]	r	1'h0	TEIF7	transfer error flag for task 7
[21]	r	1'h0	TEIF6	transfer error flag for task 6
[20]	r	1'h0	TEIF5	transfer error flag for task 5
[19]	r	1'h0	TEIF4	transfer error flag for task 4
[18]	r	1'h0	TEIF3	transfer error flag for task 3
[17]	r	1'h0	TEIF2	transfer error flag for task 2
[16]	r	1'h0	TEIF1	transfer error flag for task 1
[15:8]			RSVD	
[7]	r	1'h0	TCIF8	task complete interrupt flag for task 8
[6]	r	1'h0	TCIF7	task complete interrupt flag for task 7
[5]	r	1'h0	TCIF6	task complete interrupt flag for task 6
[4]	r	1'h0	TCIF5	task complete interrupt flag for task 5
[3]	r	1'h0	TCIF4	task complete interrupt flag for task 4
[2]	r	1'h0	TCIF3	task complete interrupt flag for task 3
[1]	r	1'h0	TCIF2	task complete interrupt flag for task 2
[0]	r	1'h0	TCIF1	task complete interrupt flag for task 1
0x04			**ICR**	interrupt clear register
[31:17]			RSVD	
[16]	w1s	1'h0	CTEIF	clear transfer error flag
[15:8]			RSVD	
[7]	w1s	1'h0	CTCIF8	clear task complete interrupt flag for task 8
[6]	w1s	1'h0	CTCIF7	clear task complete interrupt flag for task 7
[5]	w1s	1'h0	CTCIF6	clear task complete interrupt flag for task 6
[4]	w1s	1'h0	CTCIF5	clear task complete interrupt flag for task 5
[3]	w1s	1'h0	CTCIF4	clear task complete interrupt flag for task 4
[2]	w1s	1'h0	CTCIF3	clear task complete interrupt flag for task 3
[1]	w1s	1'h0	CTCIF2	clear task complete interrupt flag for task 2
[0]	w1s	1'h0	CTCIF1	clear task complete interrupt flag for task 1
0x08			**IER**	interrupt enable register
[31:17]			RSVD	
[16]	rw	1'h0	TEIE	enable transfer error flag
[15:8]			RSVD	
[7]	rw	1'h0	TCIE8	enable task complete interrupt for task 8
[6]	rw	1'h0	TCIE7	enable task complete interrupt for task 7
[5]	rw	1'h0	TCIE6	enable task complete interrupt for task 6
[4]	rw	1'h0	TCIE5	enable task complete interrupt for task 5
[3]	rw	1'h0	TCIE4	enable task complete interrupt for task 4
[2]	rw	1'h0	TCIE3	enable task complete interrupt for task 3
[1]	rw	1'h0	TCIE2	enable task complete interrupt for task 2
[0]	rw	1'h0	TCIE1	enable task complete interrupt for task 1
0x10			**TCR1**	task 1 control register
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0

Continued on next page...

Table 7-6: PTC Register Map (Continued)

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

Continued on 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
0x24			**TAR2**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x28			**TDR2**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x2C			**RCR2**	task 2 repetition and delay counter register
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x30			**TCR3**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x34			**TAR3**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x38			**TDR3**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x3C			**RCR3**	task 3 repetition and delay counter register

Continued on next page...

Table 7-6: PTC Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x40			**TCR4**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x44			**TAR4**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x48			**TDR4**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x4C			**RCR4**	task 4 repetition and delay counter register
[31:16]	rw	16'h0	DLY	Delay time before task operation after triggered 0: no delay others: delay DLY HCLK cycles before task operation DLY is read as left delay time. DLY will be reloaded automatically after each operation.
[15:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x50			**TCR5**	
[31:24]			RSVD	

Continued on next page...

Table 7-6: PTC Register Map (Continued)

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

Continued on 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
0x64			**TAR6**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x68			**TDR6**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x6C			**RCR6**	task 6 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.
0x70			**TCR7**	
[31:24]			RSVD	
[23]	rw	1'h0	REPIRQ	repetition interrupt 0: interrupt will be generated after each operation 1: interrupt will be generated after operation for REP times
[22]	rw	1'h0	REPTRIG	repetition trigger 0: ptc trigger will be generated after each operation 1: ptc trigger will be generated after operation for REP times
[21]	rw	1'h0	REPEN	repetition enable 0: task will be triggerd no matter what value REP is 1: task will only be triggerd when REP is not 0
[20]	w1s	1'h0	SWTRIG	software trigger task will be triggerd at once after SWTRIG set. SWTRIG will be cleared automatically.
[19]	rw	1'h0	TRIGPOL	trigger polarity 0: select positive edge of trigger 1: select negative edge of trigger
[18:16]	rw	3'h0	OP	task operation 3'b000: direct write data 3'b100: read then XOR with data and write back 3'b101: read then OR with data and write back 3'b110: read then AND with data and write back 3'b111: read then add with data and write back
[15:8]			RSVD	
[7:0]	rw	8'h0	TRIGSEL	select trigger source
0x74			**TAR7**	
[31:0]	rw	32'h0	ADDR	peripheral address to access to
0x78			**TDR7**	
[31:0]	rw	32'h0	DATA	data value for task operation
0x7C			**RCR7**	task 7 repetition counter register
[31:10]			RSVD	
[9:0]	rw	10'h0	REP	Repetition counter value if REPEN is 1, task will only be triggerd when REP is not 0. when REP is larger than 0, it will be decrease by 1 automatically each time task triggered.

Continued on next page...

Table 7-6: PTC Register Map (Continued)

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

Continued on next page...

Table 7-6: PTC Register Map (Continued)

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

7.5 USB

USB is located in HPSYS.

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

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

8 Analog Peripheral

8.1 GPADC

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

8.1.1 Introduction

GPADC is a 12-bit precision SAR ADC that supports 0–3.3 V input voltage and outputs 12-bit data. The input voltage can be single-ended or differential, and the output data can be read via the APB bus or DMA interface

8.1.2 Key Features

- Input voltage range:0~3.3V, 12-bitresolution
- Supports single-ended mode and differential mode
- Supports 8 channels of single-ended analog input or 4pairs of differential analog input
- Supports single measurement mode and continuous measurement mode
- Each measurement can be divided into 8time slots; each time slot can be independently configured with an analog input channel
- Supports software (register write) and hardware (e.g., timer) trigger methods
- Supports DMA channels
- Sampling frequency configurable, maximum sampling frequency 4 MHz
- Interrupt generated upon conversion completion

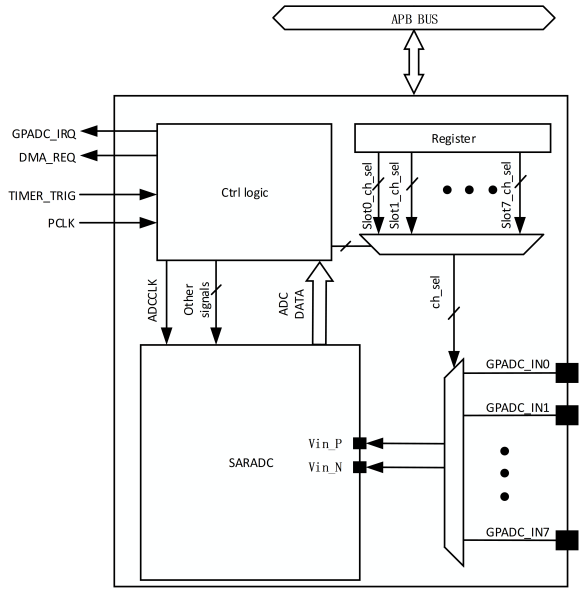


Figure 8-1: GPADC block diagram

8.1.3 Functional description

8.1.3.1 GPADC Clock Generation

The GPADC clock is generated by dividing the PCLK frequency. The ADCCLK frequency is configured by setting DATA_SAMP_DLY in ADC_CTRL_REG and CONV_WIDTH and SAMP_WIDTH in ADC_CTRL_REG2. The calculation formula is as follows:

$$f\_ADCCLK = f\_PCLK/(DATA\_SAMP\_DLY + CONV\_WIDTH + SAMP\_WIDTH + 2)$$

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

8.1.3.2 Slot configuration

In each GPADC sampling cycle, eight slots operate sequentially. Each slot can be independently enabled or disabled by setting SLOT_EN in ADC_SLOT*_REG; The input channels for each slot can be independently configured by setting PCHNL_SEL and NCHNL_SEL in ADC_SLOT*_REG.

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 to single-ended input mode; the input channel is selected by setting PCHNL_SEL in the corresponding slot configuration register.

Setting the ANAU_GPADC_SE bit to 0 in the ADC_CFG_REG register configures the GPADC to differential input mode; Vin_P and Vin_N input channels are selected by setting PCHNL_SEL and NCHNL_SEL in the corresponding slot configuration registers.

8.1.3.4 Input Channel Selection

The GPADC has a total of 8 input channels. By configuring PCHNL_SEL and NCHNL_SEL in the ADC_SLOT*_REG registers, the input channels to be sampled for each slot can be selected.

In single-ended mode, only set PCHNL_SELto select the input channel.

8.1.3.5 Sampling Mode

If ADC_OP_MODE in ADC_CTRL_REG is set to 0,the GPADC operates in single-sample mode. In this mode, each time the GPADC is started, the GPADC completes one round of sampling according to the configuration of each time slot, then returns to the wait-for-trigger state.

If ADC_OP_MODE in ADC_CTRL_REG is set to 1, the GPADC operates in continuous sampling mode. In this mode, each time the GPADC is started, it continuously samples in cycles according to the configuration of each time slot. Setting ADC_STOP in ADC_CTRL_REG to 1 causes the GPADC to return to the wait-for-trigger state.

8.1.3.6 Start GPADC

- Start by writing to the register
  Setting ADC_START bit to 1 in register ADC_CTRL_REG starts the GPADC.
  After triggering the GPADC, if it is in single sampling mode, it returns to the waiting trigger state upon completing one sampling cycle. If the GPADC is in continuous sampling mode, the ADC_STOP bit in ADC_CTRL_REG must be set to 1 to return the GPADC to the waiting trigger state
- Timer Trigger
  TheGPADC supports TIMER trigger. Enabling the TIMERtrigger function requires setting TIMER_TRIG_ENbit in-ADC_CTRL_REG to 1. There are 8 trigger sources in total, selectable via TIMER_TRIG_SRC_SEL in register 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 the GPADC, if it is in single sampling mode, it returns to the waiting trigger state upon completing one sampling cycle. If the GPADC is in continuous sampling mode, the ADC_STOP bit in ADC_CTRL_REG must be set to 1 to return the GPADC to the waiting trigger state

8.1.3.7 Data Access

GPADC The converted data can be accessed by the following two methods:

- Register Read

Software can directly read the GPADC conversion result by reading the register. Data for each time slot is stored in the register ADC_RDATA*. The correspondence between registers and time slot data is shown in the table below:

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

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

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

- DMA Mode
  Read the conversion result of GPADC via DMAC2 in LPSYS. The usage procedure is as follows:
  Set the DMA_EN bit of the ADC_CTRL_REG register to 1.
  Set the DMA source address is set to 0x50016034. The alignment of ADC data at this address is:

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

  Refer to the DMAC section for other DMA configurations.

8.1.3.8 Notifcation Mechanismc

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 of the GPADC_IRQ register to 1.

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

8.1.3.9 Configuration Startup Procedure

Using GPADC generally follows the procedure below:

- Configure PINMUX.
- Configure the GPADC clock frequency, input channel selection, and related parameters.
- Set ANAU_GPADC_EN_BG in register ADC_CFG_REG1 to 1 to enable the Bandgap.
- Set ANAU_GPADC_LDOREF_EN in register ADC_CFG_REG1 to 1 to enable the LDO that provides the reference voltage to the GPADC.
- Set FRC_ADC_EN in register ADC_CFG_REG1 to 1 to enable the GPADC.
- Trigger the GPADC to start sampling and read the data.
- After sampling is complete, set FRC_EN_ADC in register ADC_CTRL_REG to 0 to disable the GPADC module. If operating in continuous sampling mode, it is necessary to first set ADC_STOP in the register ADC_CTRL_REG to 1

and then to 0 to interrupt the sampling process.

- Set ANAU_GPADC_LDOREF_EN in the register ADC_CFG_REG1 to 0 to disable the LDO providing the reference voltage to the GPADC.
- Set ANAU_GPADC_EN_BG in the register ADC_CFG_REG1 to 0 to disable the Bandgap.

The procedure requires compliance with 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, enable the LDO providing the reference voltage; the LDO requires a 200us stabilization time.
- After enabling GPADC,a minimum startup time of 200us is required.
- Finally, initiate GPADC via register write trigger or 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
[31:30]			RSVD	
[29:25]	rw	5'h16	ANAU_GPADC_CMM	Tune CDAC CM voltage 375mV range (increasing) / 25mV step, 8: for 0.5V Vcm,in
[24:22]	rw	3'h3	ANAU_GPADC_CMPCL	Tune ADC comparator CL= 3: 40f, range: 10fF (0) ~ 80fF (7) / 10fF step
[21:20]	rw	2'h2	ANAU_GPADC_VSP	Set comparator input CM in sampling phase, 0.539V (0) / 0.578V (1) / 0.642V (2) / 0.784V (3)
[19]	rw	1'h0	ANAU_GPADC_LDOREF_EN	Enable LDORF for ADC VREF
[18:15]	rw	4'hA	ANAU_GPADC_LDOVREF_SEL	Set reference voltage for LDOREF, range = 0.35V(0) ~ 0.65V(15), step = 20mV
[14:12]	rw	3'h1	ANAU_GPADC_SEL_PCH	Select P-side input channel for GPADC, 0 for channel 0, 7 for channel 7, effective when force on
[11:9]	rw	3'h0	ANAU_GPADC_SEL_NCH	Select N-side input channel for GPADC, 0 for channel 0, 7 for channel 7, effective when force on
[8]	rw	1'h0	ANAU_GPADC_MUTE	Short GPADC P & N input to CMREF, i.e., VREF/2
[7]	rw	1'h0	ANAU_GPADC_SE	Set GPADC in single-ended mode, signal range at P-input: 0 ~ VREF
[6]	rw	1'h0	ANAU_GPADC_EN_V18	
[5:3]	rw	3'h2	ANAU_GPADC_CL_DLY	
[2]	rw	1'h0	ANAU_GPADC_P_INT_EN	
[1]			RSVD	
[0]	rw	1'h0	ANAU_GPADC_CMREF_FAST_EN	
0x04			**ADC_Slot0_REG**	ADC Slot0 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x08			**ADC_Slot1_REG**	ADC Slot1 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	

Continued on next page...

Table 8-1: GPADC Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x0C			**ADC_Slot2_REG**	ADC Slot2 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x10			**ADC_Slot3_REG**	ADC Slot3 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x14			**ADC_Slot4_REG**	ADC Slot4 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x18			**ADC_Slot5_REG**	ADC Slot5 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x1C			**ADC_Slot6_REG**	ADC Slot6 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x20			**ADC_Slot7_REG**	ADC Slot7 Config Register
[31:14]			RSVD	
[13:11]	rw	3'h1	NCHNL_SEL	
[10:8]	rw	3'h0	PCHNL_SEL	
[7:1]			RSVD	
[0]	rw	1'h1	SLOT_EN	
0x24			**ADC_RDATA0**	ADC Read Data0
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT1_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT0_RDATA	
0x28			**ADC_RDATA1**	ADC Read Data1
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT3_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT2_RDATA	
0x2C			**ADC_RDATA2**	ADC Read Data2
[31:28]			RSVD	
[27:16]	r	12'h0	SLOT5_RDATA	
[15:12]			RSVD	
[11:0]	r	12'h0	SLOT4_RDATA	
0x30			**ADC_RDATA3**	ADC Read Data3

Continued on next page...

Table 8-1: GPADC Register Map (continued)

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

8.2 TSEN

TSEN is located within LPSYS.

8.2.1 Introduction

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

8.2.2 Key Features

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

8.2.3 Functional description

8.2.3.1 Clock signal

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

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

8.2.3.2 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_IRSRbit in the register TSEN_IRQ. When the value of TSEN_IRSRis 1, it indicates that the temperature data conversion is complete.

After reading, set TSEN_ICR in the register TSEN_IRQ to 1, and clear TSEN_IRSR.

The process of reading via interrupt is as follows:

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

8.2.3.3 Usage Procedure

The operation of the TSEN must follow the procedure below.

1. configure the prescaler according to the PCLK frequency; the clock frequency of the TSEN should be 1 MHz or 2 MHz.
2. Set EN in the register BGR 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 clock of TSEN.
5. Set ANAU_TSEN_PU to 1 and ANAU_TSEN_RUN to 0 in register TSEN_CTRL_REG.
6. Set ANAU_TSEN_RSTB in register TSEN_CTRL_REG to 0 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 by setting ANAU_TSEN_RUN/ANAU_TSEN_PU/ANAU_TSEN_EN in register TSEN_CTRL_REG to 0. Set ANAU_TSEN_RSTB to 1.
9. Set ANAU_IARY_EN in register ANAU_ANA_TP to 0 to disable the Iarray.
10. Set EN in register BGR to 0 to disable the Bandgap.

8.2.4 TSEN Register

Table 8-2: 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 hclk by anau_tsen_clk_div
[11]	rw	1'h0	ANAU_TSEN_EN	Enable tsen digital module
[10]	r	1'h0	ANAU_TSEN_RDY	tsen ready
[9]	rw	1'h0	ANAU_TSEN_SER_PAR_SEL	serial-parallel output selection
[8]	rw	1'b0	ANAU_TSEN_SGN_EN	signature-mode enable
[7:6]	rw	2'h1	ANAU_TSEN_FCK_SEL	select internal clock frequency
[5:3]	rw	3'h1	ANAU_TSEN_IG_VBE	bias current selection to tune vba
[2]	rw	1'h0	ANAU_TSEN_RUN	enable tsen run
[1]	rw	1'h1	ANAU_TSEN_RSTB	resetb for tsen
[0]	rw	1'h0	ANAU_TSEN_PU	power up tsen
0x04			**TSEN_RDATA**	Tsen Read Data
[31:12]			RSVD	
[11:0]	r	12'h0	TSEN_RDATA	
0x08			**TSEN_IRQ**	Tsen IRQ Register
[31:4]			RSVD	
[3]	r	1'b0	TSEN_ISR	
[2]	r	1'b0	TSEN_IRSR	
[1]	rw	1'b0	TSEN_IMR	
[0]	w1s	1'b0	TSEN_ICR	
0x10			**ANAU_ANA_TP**	Tsen IRQ Register
[31:7]			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 ATIM

ATIM1 is located in HPSYS, with input/output connected to IO(PA), and can send requests to DMAC1.

9.1.1 Introduction

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

9.1.2 Main Features

- 32 bit increment, decrement, increment/decrement auto-reload counter
- 16 bit programmable (can be modifed in real-time)prescaler, with a division factor for the counter clock frequency ranging from 1 to 65536 for any value
- 16 bit configurable repeat count
- Supports one pulse mode (OPM), which automatically stops the counter upon completion of the repeat count.
- 6 independent channels
  - Channels 1 to 3 can be configured as input or output modes, with each channel capable of outputting two complementary PWM signals with dead-time protection.
  - Channel 4 can be configured as either input or output mode, capable of outputting a single PWM
  - Channels 5~6 can be configured for Output Compare Mode
- Input Mode
  - Rising Edge/Falling Edge Capture
  - PWM Pulse Width and Period Capture (requires two channels)
  - Optional one of 4 input ports or 1 external trigger port, supporting debounce 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 multiple counter interconnections, enabling it to generate control signals as a master device while being controlled by external inputs or other master devices as a slave.
  - Control modes include reset, trigger, gating, and others.
  - Supports synchronous starting and resetting of multiple counters.
- Encoding mode input for controlling counter increment/decrement counting.
- Supports Hall sensor circuits for positioning.
- 2 Brake inputs, featuring debounce filtering, can quickly place the output in a safe state. Brake signal sources include:：
  - CPU Exception
  - Comparator
  - External Input
  - Software Trigger
- An interrupt/DMArequest/PTC is generated when the following events occur:
  - Update: Counter increment overflow/decrement overflow, counter initialization (via software or internal/external trigger)
  - Trigger Events (counter start, stop, initialization, or counting triggered by internal/external sources)
  - Input Capture
  - Output Compare
  - Brake
  - Communitation

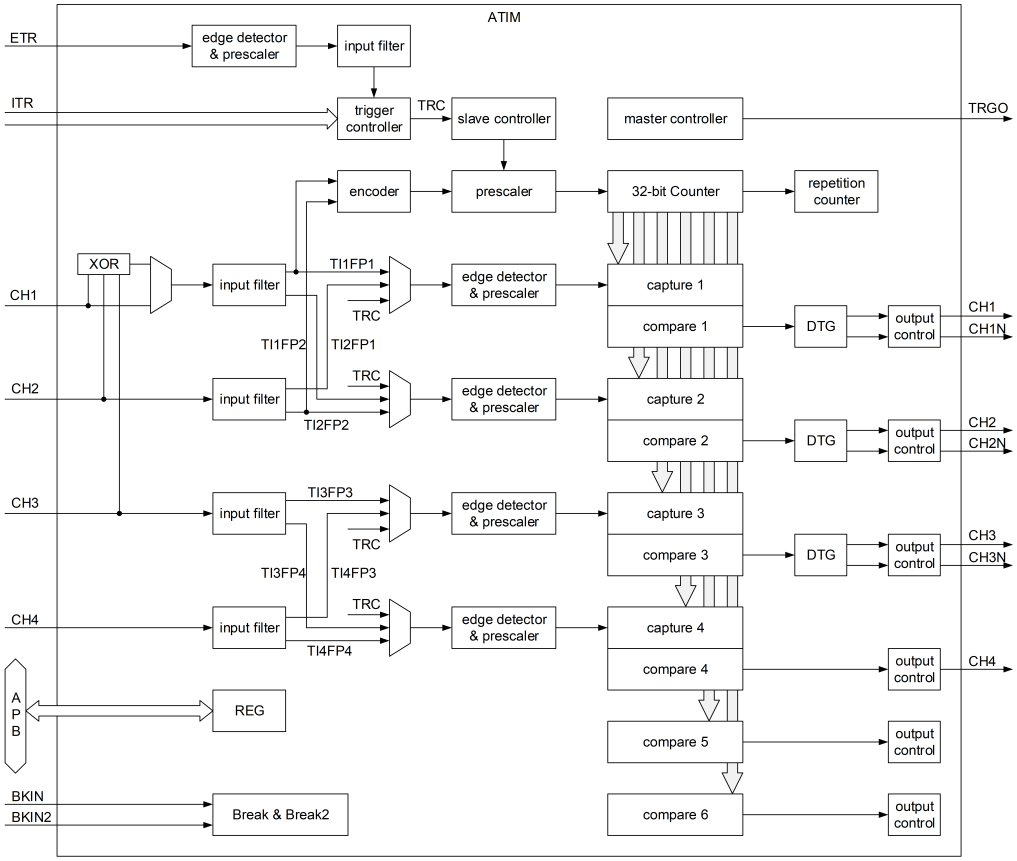


Figure 9-1: ATIM Structure Diagram

9.1.3 ATIM Function Description

9.1.3.1 Counter

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

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

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

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

9.1.3.2 Update Event(UEV)

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

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

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

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

9.1.3.3 Repeated Counting

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

The current value of the Repeat Counter cannot be read.

9.1.3.4 Shadow Register

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

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

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

9.1.3.5 Master-Slave Mode

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

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

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

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

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

9.1.3.6 Channel Input and Output

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

In input capture mode, when the corresponding trigger signal is valid, the value of the counter is recorded into CCRx , and an interrupt or other notifcation signal is generated. The trigger signal can be selected from ETR , ITR , and channel input CHx , and the signal polarity can be selected, along with pre-scaling, 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 counter value 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, along with generating interrupt and other notifcation signals. The parameters of the PWM signal, including the number of pulses, frequency, duty cycle, and phase, are adjustable. Multiple channels can also collaborate to produce specific relationships of PWM combinations, such as six-channel complementary PWM with dead time protection. The notifcation signals generated by the channel include interrupts, DMA requests, and PTC triggers.

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

9.1.3.7 Input Capture Mode

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

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

1. Select the valid input: Channel 1 is to be connected to the CH1 input, so write 01 to CCMR1_CC1S.
2. Configure the desired input filtering bandwidth based on the signal connected to the timer.
   Assuming that the CH1 signal edge changes with a maximum jitter of 5 PCLK cycles, the 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 confrmed.
3. Set CCER_CC1Pand CCER_CC1NP to 0 to select the valid conversion edge on CH1as the rising edge.
4. Program the input prescaler.
   In this example, we want to perform a capture operation on every valid conversion; therefore, the prescaler is disabled (set CCMR1_IC1PS to 00).
5. Set CCER_CC1E to 1 to enable channel 1and allow the counter value to be captured in CCR1.
6. If necessary, set DIER_CC1IE to 1to enable the corresponding interrupt request, or set DIER_CC1DE to 1to enable DMA requests.

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

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

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

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

9.1.3.8 PWM Input Capture

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

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

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

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

9.1.3.9 Output Compare Mode

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

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

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

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

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

9.1.3.10 Basic PWM Output

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

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

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

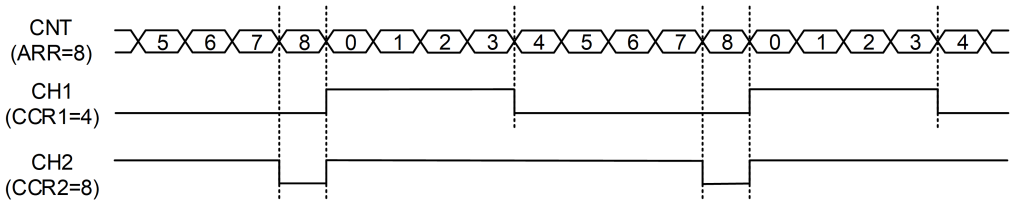


Figure 9-2: PWM output in increment counting mode

In center-aligned counting mode, if CCMR1_OC1M and CCMR1_OC2M are configured to 0110(0x6) , the PWM output is illustrated in Figure 9-3 . When the increment stage counting value CNT is less than CCR1/2 , the output is high; otherwise, it is low. When the decrement stage counting value CNT is greater than CCR1/2 , the output is low; otherwise, it is high.

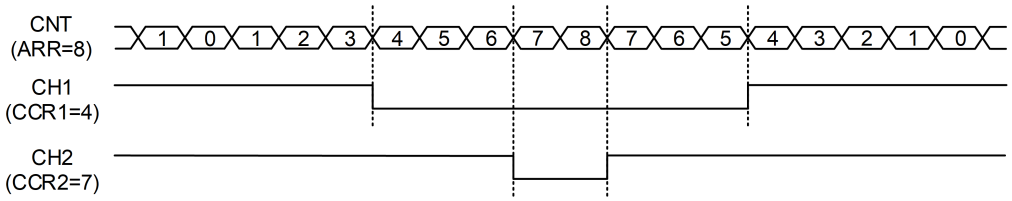


Figure 9-3: PWM output in center-aligned counting mode

9.1.3.11 Asymmetric PWM Output

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

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

If CCMR1_OC1M and CCMR2_OC3M are configured to 1110(0xe) , the PWM output is illustrated in Figure 9-4 . During the

increasing phase (0->ARR-1) , when the count value CNT is less than CCR1/3 , a high-level output is generated; otherwise, a low-level output is produced. During the decreasing phase (ARR->1) , when the count value CNT is greater than CCR2/4 , a low-level output is generated; otherwise, a high-level output is produced.

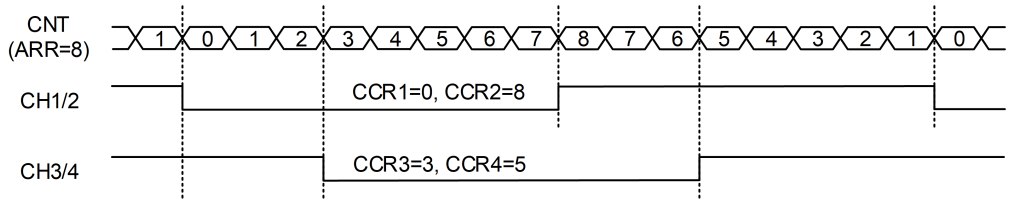


Figure 9-4: Asymmetric PWM Output

9.1.3.12 Combined PWM Output

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

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

For example, if CCMR1_OC1M is configured to 1101(0xd),CCMR1_OC2M to 0110(0x6),CCMR2_OC3M to 1100(0xc),CCMR2_OC4M is 0111(0x7) , the PWM output is illustrated in Figure 9-5 . When the count value CNT is less than CCR1/4 , OC1REF/OC4REF is low; otherwise, it is high. When the count value CNT is less than CCR2/3 , OC2REF/OC3REF is high; otherwise, it is low. The CH1 output represents the logical AND of OC1REF and OC2REF . The CH3 output represents the logical OR of OC3REF and OC4REF .

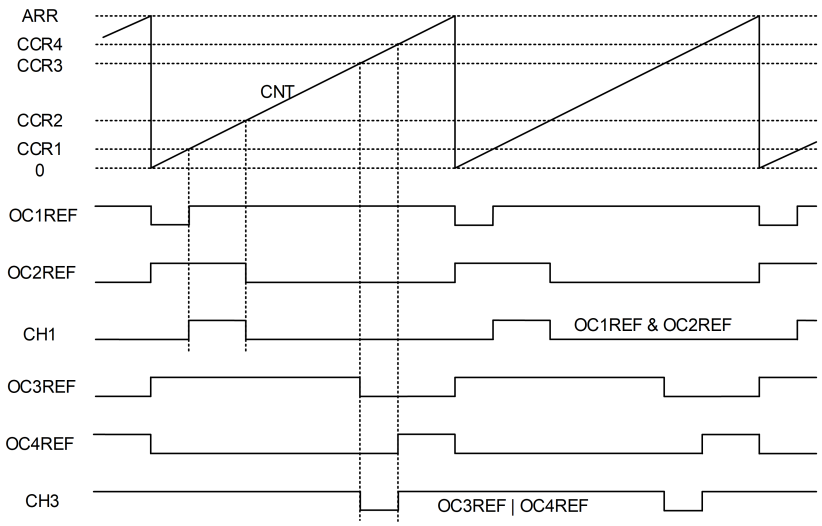


Figure 9-5: Combined PWM Output

9.1.3.13 Complementary PWM output with dead time

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

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

An example of complementary output is illustrated in Figure 9-6. The rising edge of CHx has a dead time delay relative to the rising edge of the reference output OCxREF generated by that channel, while the rising edge of CHxN has a dead time delay relative to the falling edge of OCxREF for that channel.

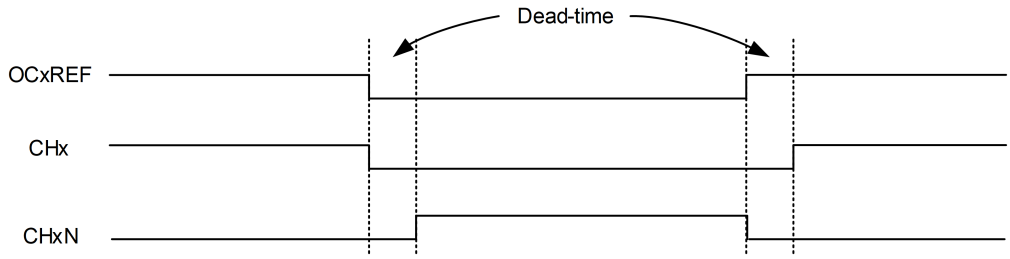


Figure 9-6: Complementary PWM output with dead time

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

9.1.3.14 Emergency Cut-off

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

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

After reset, the break function of ATIM is disabled, and BDTR_MOE is at a low level. Setting BDTR_BKE/BKE2 to 1enables the break function.

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

The internal circuit of the open circuit also implements write protection to ensure application security. Through this fea-

ture, users can freeze multiple parameter configurations, such as dead time duration, output polarity, and state when disabled, PWM mode, open circuit enable, and polarity, among others. This feature is achieved by writing to the AF1_LOCK register, allowing selection from 3 levels of protection.

9.1.3.15 6 Step PWM

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

9.1.3.16 One Pulse Mode

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

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

1. CCMR1_CC2S=01 to map 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.

9.1.3.17 Encoder Interface Mode

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

If the counter counts only on the rising edge of CH1 , configure SMCR_SMS to 0001; if the counter counts only on the rising edge of CH2, configure SMCR_SMS to 0010(0x2); if the counter counts on the rising edges of both CH1 and CH2, 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.

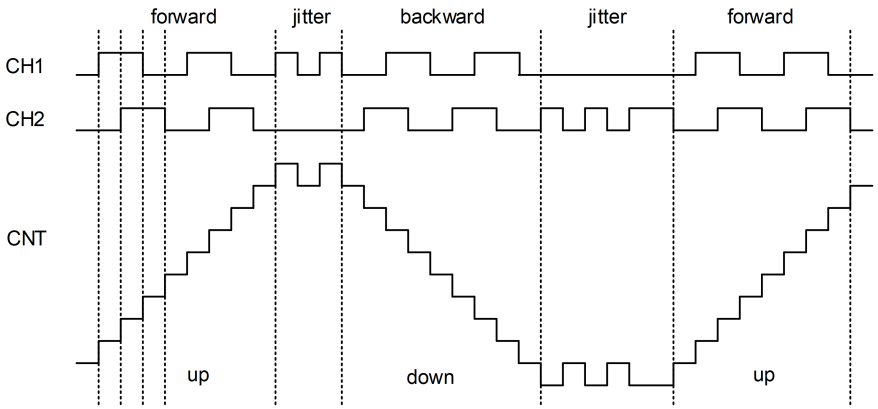
In Encoder Interface Mode, the counting events of the counter are the decoded outputs of the quadrature encoder interface. The counter only performs continuous counting between 0 and ARR (incrementing from 0 to ARR or decrementing from ARR to 0, depending on the specific counting direction). Therefore,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

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

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

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

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



9.1.3.18 Timer Synchronization

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

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

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

timers.

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

9.1.3.19 Notifcation Mechanism

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

9.1.4 ATIM Register

Table 9-1: ATIM Register Mapping Table

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

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

9.2 BTIM

The chip contains 4 BTIMs, where BTIM1 and BTIM2 are located in HPSYS and can send requests to DMAC1; BTIM 3 and BTIM4 are located in LPSYS and can send requests to DMAC2.

9.2.1 Introduction

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

9.2.2 Main Features

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

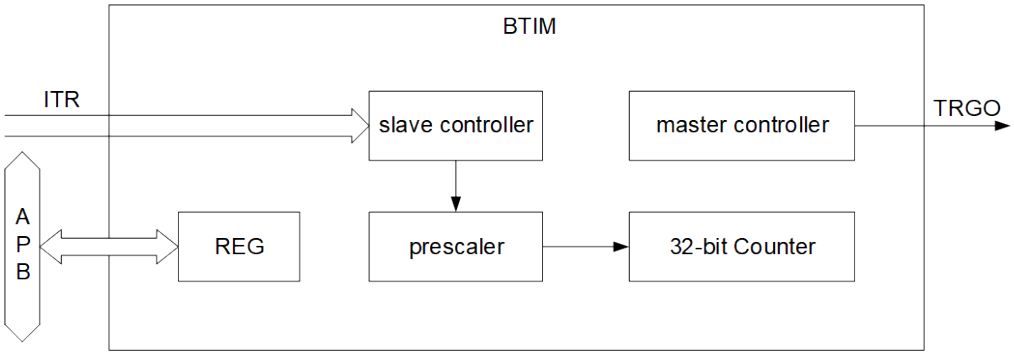


Figure 9-7: BTIM Block Diagram

9.2.3 BTIM Function Description

9.2.3.1 Counter

The functions of BTIM are based on a 32-bit counter. The counter operates on an event-counting basis, with the most fundamental event being a 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 .

9.2.3.2 Update Event(UEV)

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

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

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

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

9.2.3.3 Shadow Register

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

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

9.2.3.4 Master-Slave Mode

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

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

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

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

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

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

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

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

9.2.3.5 One Pulse Mode

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

9.2.3.6 Timer Synchronization

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

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

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

9.2.3.7 Notifcation Mechanism

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

9.2.4 BTIM Register

Table 9-2: BTIM Register Mapping Table

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

Continued on the next page...

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

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

Continued on the next page...

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

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

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[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.

9.3 GPTIM

The chip contains a total of 5 GPTIMs, of which GPTIM 1 and GPTIM 2 are located in HPSYS , with input/output connected to IO(PA) , capable of sending requests to DMAC1 ; GPTIM 3 , GPTIM4 and GPTIM5 are located in LPSYS , with input/output connected to IO(PB) , capable of sending requests to DMAC2 .

9.3.1 Introduction

GPTIM (General Purpose Timer) is based on a 16 -bit counter, which can perform timing, measure the pulse length of input signals (input capture), or generate output waveforms (output compare and PWM), among other functions. The counter itself can increment, decrement, or perform up/down counting, with the counting clock selectable from the system PCLK, IO input signals, or cascaded input signals, and can be prescaled by 1~65536 times. GPTIM has a total of 4 channels, which can be independently configured for input capture or output mode. The results of counting, input capture, and output compare can generate interrupts, DMA requests, or PTC triggers. GPTIM includes a master-slave mode interface, enabling multi-level cascading to achieve multi-level counting or synchronized triggering functions.

9.3.2 Main Features

- 16 bit increment, decrement, increment/decrement auto-reload counter, with a maximum count of 65535.
- 16 bit programmable (can be modifed in real-time)prescaler, with a division factor for the counter clock frequency ranging from 1 to 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

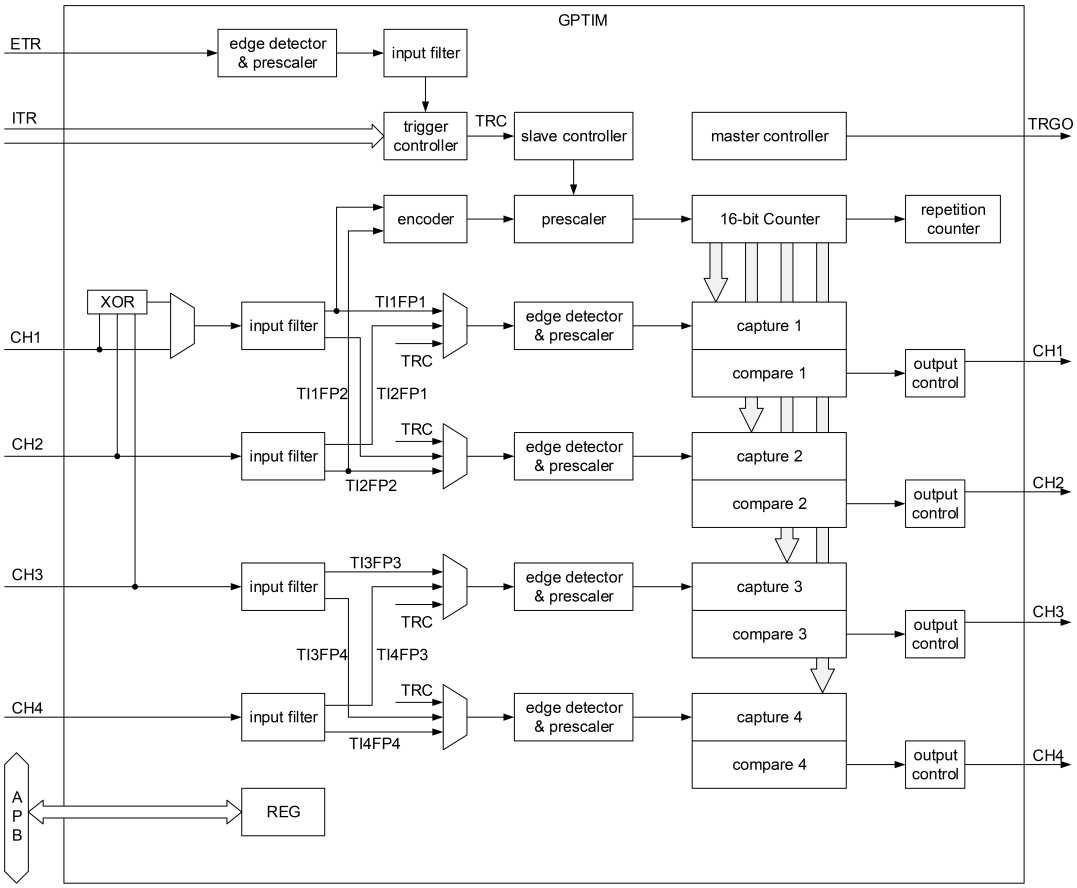


Figure 9-8: GPTIM Block Diagram

9.3.3 GPTIM Function Description

9.3.3.1 Counter

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

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

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

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

9.3.3.2 Update Event(UEV)

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

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

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

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

9.3.3.3 Repeat Counting

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

The current value of the Repeat Counter cannot be read.

9.3.3.4 Shadow Register

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

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

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

9.3.3.5 Master-Slave Mode

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

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

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

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

9.3.3.6 Channel Input and Output

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

In input capture mode, when the corresponding trigger signal is valid, the value of the counter is recorded into CCRx, and an interrupt or other notifcation signal is generated. The trigger signal can be selected from ETR , ITR , and channel input CHx , and the signal polarity can be selected, along with pre-scaling, 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.

9.3.3.7 Input Capture Mode

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

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

1. Select valid input: Channel 1 is to be connected to CH1input; therefore, write 01 to CCMR1_CC1S.
2. Configure the required input 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.。

9.3.3.8 PWM Input Capture

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

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

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

After configuration, on each rising edge of CH1 , the counter's value is recorded in CCR1 , while the counter is reset and begins counting again; on each falling edge of CH1 , the counter's value is recorded in CCR2 . Multiplying the value of CCR1 by the period of PCLK calculates the period of PWM. Set Multiplying the value ofCCR2 by the period of PCLK calculates the duration of the high level of PWM, thus determining the duty cycle of PWM.

9.3.3.9 Output Compare Mode

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

Specifically, the channel will execute the following operations when CCRxmatches the counter:

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

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

9.3.3.10 Basic PWM Output

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

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

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

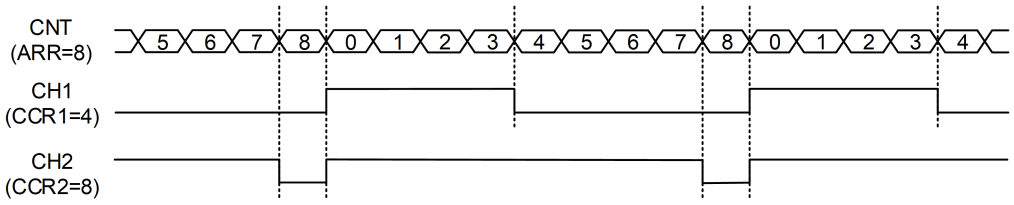


Figure 9-9: PWM output in increment counting mode

In center-aligned counting mode, if you configure CCMR1_OC1M and CCMR1_OC2M to 0110(0x6) , the PWM output is illustrated in Figure 9-10 . When the counting value CNT in the increment phase is less than CCR1/2 , the output is high; otherwise, it is low. When the counting value CNT in the decrement phase exceeds CCR1/2 , the output is low; otherwise, it is high.

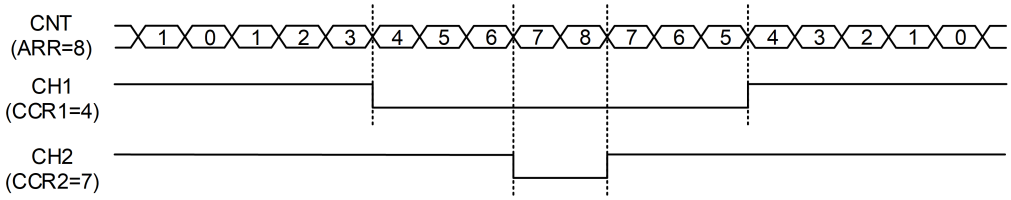


Figure 9-10: PWM output in center-aligned counting mode

9.3.3.11 Asymmetric PWM output

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

CH1/2 and CH3/4 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 9-11. In the increment phase (0->ARR-1) , when the counting value CNT is less than CCR1/3 , the output is high; otherwise, it is low. In the decrement phase (ARR->1) , when the counting value CNT exceeds CCR2/4 , the output is low; otherwise, it is high.

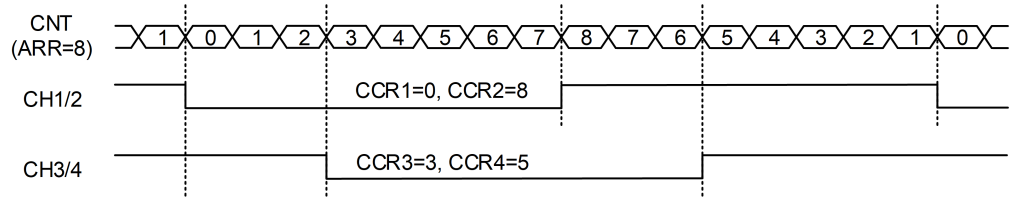


Figure 9-11: Asymmetric PWMO utput

9.3.3.12 Combined PWMO utput

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

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

For example, if CCMR1_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 9-12. When the count value CNT is less than CCR1/4, OC1REF/OC4REF is low; otherwise, it is high. When the count value CNT is less than CCR2/3, OC2REF/OC3REF is

high; otherwise, it is low. The output of CH1 is the logical AND of OC1REF and OC2REF. The output of CH3 is the logical OR of OC3REF and OC4REF.

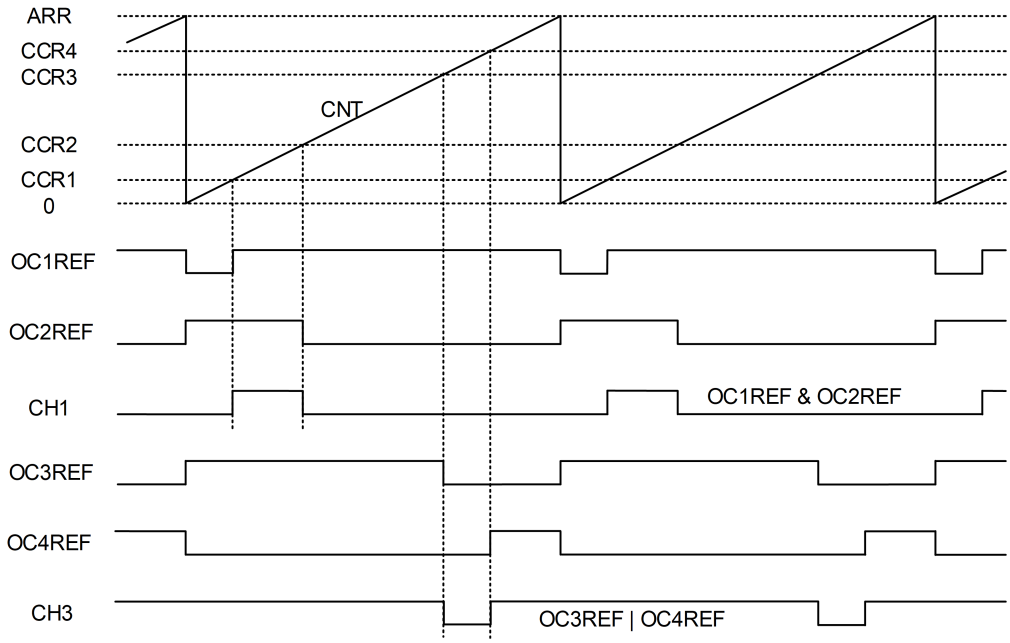


Figure 9-12: Combined PWM Output

9.3.3.13 One Pulse Mode

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

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

1. CCMR1_CC2S=01 to map 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.

9.3.3.14 Encoder Interface Mode

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

external encoder.

If the counter counts only on the rising edge of CH1, 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.

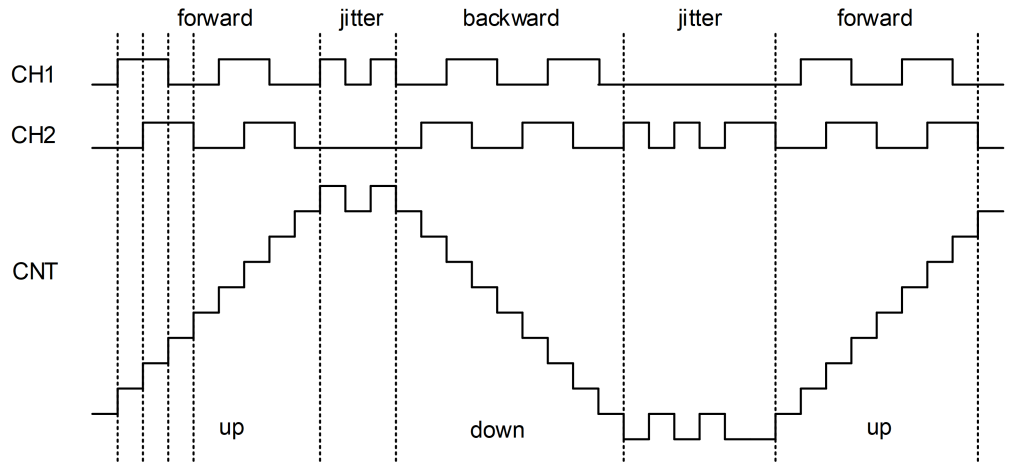
In Encoder Interface Mode, the counting events of the counter are the decoded outputs of the quadrature encoder interface. The counter only performs continuousounting between 0 and ARR (incrementing from 0 to ARR or decrementing from ARR to 0, depending on the specific counting direction). Therefore,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



9.3.3.15 Timer Synchronization

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

Set the master mode timer's TRGO to update event (CR2_MMS=010) , connected to another timer configured for external clock slave mode (SMCR_SMS = 0111) , to enable timer cascading counting. In this configuration, the master mode timer serves as the prescaler for the slave mode timer, and the total counting bit width is the sum of the bit widths of both timers.

Set the master mode timer's TRGO to count enable (CR2_MMS=001) , and configure the slave mode to trigger slave mode (SMCR_SMS=0110) , while also connecting to another timer set to trigger slave mode (SMCR_SMS=0110) , to achieve synchronized triggering of multiple timers, thereby aligning the start timing of all timers. In this scenario, the master mode timer must also set SMCR_MSM to 1 .

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

9.3.3.16 Notifcation Mechanism

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

9.3.4 GPTIM Register

Table 9-3: GPTIM Register Mapping Table

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

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[6:4]	rw	3'h0	MMS	Master mode selection These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - the UG bit from the EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset. 001: Enable - the Counter enable signal is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected. 010: Update - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO) 100: Compare - OC1REF signal is used as trigger output (TRGO) 101: Compare - OC2REF signal is used as trigger output (TRGO) 110: Compare - OC3REF signal is used as trigger output (TRGO) 111: Compare - OC4REF signal is used as trigger output (TRGO)
[3]	rw	1'h0	CCDS	Capture/compare DMA selection 0: CCx DMA request sent when CCx event occurs 1: CCx DMA requests sent when update event occurs
[2:0]			RSVD	
0x08			**SMCR**	TIM slave mode control register
[31:20]			RSVD	
[19:16]	rw	4'h0	SMS	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input. 0000: Slave mode disabled. 0001: Encoder mode 1 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 0010: Encoder mode 2 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 0011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 0100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 0101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 0110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 0111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. 1000: Combined reset + trigger mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter, generates an update of the registers and starts the counter.
[15]	rw	1'h0	ETP	External trigger polarity 0: ETR is non-inverted, active at high level or rising edge 1: ETR is inverted, active at low level or falling edge

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[14]	rw	1'h0	ECE	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal.
[13:12]	rw	2'h0	ETPS	External trigger prescaler External trigger signal ETRP frequency must be at most 1/4 of CK_INT frequency. A prescaler can be enabled to reduce ETRP frequency. It is useful when inputting fast external clocks. 00: Prescaler OFF 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
[11:8]	rw	4'h0	ETF	External trigger filter This bit-field then defines the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[7]	rw	1'h0	MSM	Master/Slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers on a single external event.
[6:4]	rw	3'h0	TS	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF)
[3:0]			RSVD	
0x0C			**DIER**	TIM DMA/Interrupt enable register
[31:15]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[14]	rw	1'h0	TDE	Trigger DMA request enable 0: Trigger DMA request disabled. 1: Trigger DMA request enabled.
[13]			RSVD	
[12]	rw	1'h0	CC4DE	Capture/Compare 4 DMA request enable 0: CC4 DMA request disabled. 1: CC4 DMA request enabled
[11]	rw	1'h0	CC3DE	Capture/Compare 3 DMA request enable 0: CC3 DMA request disabled. 1: CC3 DMA request enabled.
[10]	rw	1'h0	CC2DE	Capture/Compare 2 DMA request enable 0: CC2 DMA request disabled. 1: CC2 DMA request enabled.
[9]	rw	1'h0	CC1DE	Capture/Compare 1 DMA request enable 0: CC1 DMA request disabled. 1: CC1 DMA request enabled.
[8]	rw	1'h0	UDE	Update DMA request enable 0: Update DMA request disabled. 1: Update DMA request enabled
[7]			RSVD	
[6]	rw	1'h0	TIE	Trigger interrupt enable 0: Trigger interrupt disabled. 1: Trigger interrupt enabled
[5]			RSVD	
[4]	rw	1'h0	CC4IE	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled. 1: CC4 interrupt enabled
[3]	rw	1'h0	CC3IE	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled. 1: CC3 interrupt enabled
[2]	rw	1'h0	CC2IE	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled. 1: CC2 interrupt enabled.
[1]	rw	1'h0	CC1IE	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled. 1: CC1 interrupt enabled
[0]	rw	1'h0	UIE	Update interrupt enable 0: Update interrupt disabled. 1: Update interrupt enabled
0x10			**SR**	TIM status register
[31:13]			RSVD	
[12]	rw0c	1'h0	CC4OF	Capture/Compare 4 overcapture flag
[11]	rw0c	1'h0	CC3OF	Capture/Compare 3 overcapture flag
[10]	rw0c	1'h0	CC2OF	Capture/Compare 2 overcapture flag
[9]	rw0c	1'h0	CC1OF	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. It is cleared by software by writing it to '0'. 0: No overcapture has been detected. 1: The counter value has been captured in CCR1 register while CC1IF flag was already set
[8:7]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	rw0c	1'h0	TIF	Trigger interrupt flag This flag is set by hardware on trigger event (active edge detected on TRGI input when the slave mode controller is enabled in all modes but gated mode). It is set when the counter starts or stops when gated mode is selected. It is cleared by software. 0: No trigger event occurred. 1: Trigger interrupt pending.
[5]			RSVD	
[4]	rw0c	1'h0	CC4IF	Capture/Compare 4 interrupt flag
[3]	rw0c	1'h0	CC3IF	Capture/Compare 3 interrupt flag
[2]	rw0c	1'h0	CC2IF	Capture/Compare 2 interrupt flag
[1]	rw0c	1'h0	CC1IF	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This flag is set by hardware when the counter matches the compare value. It is cleared by software. 0: No match. 1: The content of the counter CNT has matched the content of the CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the CCR1 register. 0: No input capture occurred. 1: The counter value has been captured in CCR1 register (An edge has been detected on IC1 which matches the selected polarity).
[0]	rw0c	1'h0	UIF	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred 1: Update interrupt pending. This bit is set by hardware when the registers are updated: At overflow or underflow and if UDIS=0 in the CR1 register. When CNT is reinitialized by software using the UG bit in EGR register, if URS=0 and UDIS=0 in the CR1 register. When CNT is reinitialized by a trigger event, if URS=0 and UDIS=0 in the CR1 register.
0x14			**EGR**	Event generation register
[31:7]			RSVD	
[6]	w	1'h0	TG	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in SR register. Related interrupt or DMA transfer can occur if enabled.
[5]			RSVD	
[4]	w	1'h0	CC4G	Capture/compare 4 generation
[3]	w	1'h0	CC3G	Capture/compare 3 generation
[2]	w	1'h0	CC2G	Capture/compare 2 generation

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	w	1'h0	CC1G	Capture/compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel 1: If channel CC1 is configured as output: CC1IF flag is set, Corresponding interrupt or DMA request is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in CCR1 register. The CC1IF flag is set, the corresponding interrupt or DMA request is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
[0]	w	1'h0	UG	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (anyway the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (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 9-3: GPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:20]	rw	4'h0	OC1M	Output compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 0000: Frozen - The comparison between the output compare register CCR1 and the counter CNT has no effect on the outputs.(this mode is used to generate a timing base). 0001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter CNT matches the capture/compare register 1 (CCR1). 0010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter CNT matches the capture/compare register 1 (CCR1). 0011: Toggle - OC1REF toggles when CNT=CCR1. 0100: Force inactive level - OC1REF is forced low. 0101: Force active level - OC1REF is forced high. 0110: PWM mode 1 - In upcounting, channel 1 is active as long as CNT<CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0) as long as CNT>CCR1 else active (OC1REF=1). 0111: PWM mode 2 - In upcounting, channel 1 is inactive as long as CNT<CCR1 else active. In downcounting, channel 1 is active as long as CNT>CCR1 else inactive. 1000: Retriggerable OPM mode 1 - In up-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. In down-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes inactive again at the next update. 1001: Retriggerable OPM mode 2 - In up-counting mode, the channel is inactive until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 2 and the channels becomes inactive again at the next update. In down-counting mode, the channel is active until a trigger event is detected (on TRGI signal). Then, a comparison is performed as in PWM mode 1 and the channels becomes active again at the next update. 1010: Reserved, 1011: Reserved, 1100: Combined PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC is the logical OR between OC1REF and OC2REF. 1101: Combined PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC is the logical AND between OC1REF and OC2REF. 1110: Asymmetric PWM mode 1 - OC1REF has the same behavior as in PWM mode 1. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down. 1111: Asymmetric PWM mode 2 - OC1REF has the same behavior as in PWM mode 2. OC1REFC outputs OC1REF when the counter is counting up, OC2REF when it is counting down.
[19]	rw	1'h0	OC1PE	Output compare 1 preload enable 0: Preload register on CCR1 disabled. CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on CCR1 enabled. Read/Write operations access the preload register. CCR1 preload value is loaded in the active register at each update event.
[18:17]			RSVD	
[16]	rw	1'h0	OC1CE	Output compare 1 clear enable 0: OC1Ref is not affected by the ETRF input 1: OC1Ref is cleared as soon as a High level is detected on ETRF input

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[15:12]	rw	4'h0	IC2F	Input capture 2 filter
[11:10]	rw	2'h0	IC2PSC	Input capture 2 prescaler
[9:8]	rw	2'h0	CC2S	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS bit (SMCR register)
[7:4]	rw	4'h0	IC1F	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fCLK 0001: fSAMPLING=fCLK, N=2 0010: fSAMPLING=fCLK, N=4 0011: fSAMPLING=fCLK, N=8 0100: fSAMPLING=fCLK/2, N=6 0101: fSAMPLING=fCLK/2, N=8 0110: fSAMPLING=fCLK/4, N=6 0111: fSAMPLING=fCLK/4, N=8 1000: fSAMPLING=fCLK/8, N=6 1001: fSAMPLING=fCLK/8, N=8 1010: fSAMPLING=fCLK/16, N=5 1011: fSAMPLING=fCLK/16, N=6 1100: fSAMPLING=fCLK/16, N=8 1101: fSAMPLING=fCLK/32, N=5 1110: fSAMPLING=fCLK/32, N=6 1111: fSAMPLING=fCLK/32, N=8
[3:2]	rw	2'h0	IC1PSC	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=0. 00: no prescaler, capture is done each time an edge is detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
[1:0]	rw	2'h0	CC1S	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (SMCR register)
0x1C			**CCMR2**	TIM capture/compare mode register 2
[31:28]	rw	4'h0	OC4M	Output compare 4 mode
[27]	rw	1'h0	OC4PE	Output compare 4 preload enable
[26:25]			RSVD	
[24]	rw	1'h0	OC4CE	Output compare 4 clear enable
[23:20]	rw	4'h0	OC3M	Output compare 3 mode
[19]	rw	1'h0	OC3PE	Output compare 3 preload enable

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	CC1P	Capture/Compare 1 output Polarity. CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: CC1NP/CC1P bits select TI1FP1 and TI2FP1 polarity for trigger or capture operations. 00: noninverted/rising edge Circuit is sensitive to TIxFP1 rising edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode, encoder mode). 01: inverted/falling edge Circuit is sensitive to TIxFP1 falling edge (capture, trigger in reset, external clock or trigger mode), TIxFP1 is inverted (trigger in gated mode, encoder mode). 10: reserved, do not use this configuration. 11: noninverted/both edges Circuit is sensitive to both TIxFP1 rising and falling edges (capture, trigger in reset, external clock or trigger mode), TIxFP1 is not inverted (trigger in gated mode). This configuration must not be used for encoder mode.
[0]	rw	1'h0	CC1E	Capture/Compare 1 output enable. CC1 channel configured as output: 0: Off - OC1 is not active 1: On - OC1 signal is output on the corresponding output pin CC1 channel configured as input: This bit determines if a capture of the counter value can actually be done into the input capture/compare register 1 (CCR1) or not. 0: Capture disabled 1: Capture enabled
0x24			**CNT**	Counter
[31]	r	1'h0	UIFCPY	Value depends on IUFREMAP in CR1. If UIFREMAP = 1 UIFCPY: UIF Copy This bit is a read-only copy of the UIF bit of the ISR register
[30:16]			RSVD	
[15:0]	rw	16'h0	CNT	counter value
0x28			**PSC**	Prescaler
[31:16]			RSVD	
[15:0]	rw	16'h0	PSC	Prescaler value The counter clock frequency is equal to fCLK / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of EGR register or through trigger controller when configured in 'reset mode').
0x2C			**ARR**	Auto-reload register
[31:16]			RSVD	
[15:0]	rw	16'h0	ARR	Auto-reload value ARR is the value to be loaded in the actual auto-reload register.
0x30			**RCR**	Repetition counter register
[31:8]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[7:0]	rw	8'h0	REP	Repetition counter value These bits allow the user to set-up the update rate of the compare registers (i.e. periodic transfers from preload to active registers) when preload registers are enable, as well as the update interrupt generation rate, if this interrupt is enable. Each time the REP_CNT related downcounter reaches zero, an update event is generated and it restarts counting from REP value. As REP_CNT is reloaded with REP value only at the repetition update event, any write to the RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to the number of PWM periods in edge-aligned mode.
0x34			**CCR1**	Capture/Compare register 1
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR1	Capture/Compare 1 value If channel CC1 is configured as output: CCR1 is the value to be loaded in the actual capture/compare 1 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC1PE). Else the preload value is copied in the active capture/compare 1 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signaled on OC1 output. If channel CC1is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1). The CCR1 register is read-only and cannot be programmed.
0x38			**CCR2**	Capture/Compare register 2
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR2	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR1 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2). The CCR2 register is read-only and cannot be programmed.
0x3C			**CCR3**	Capture/Compare register 3
[31:16]			RSVD	
[15:0]	rw	16'h0	CCR3	Capture/Compare value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC3 output. If channel CC3is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3). The CCR3 register is read-only and cannot be programmed.
0x40			**CCR4**	Capture/Compare register 4
[31:16]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[15:0]	rw	16'h0	CCR4	Capture/Compare value 1. if CC4 channel is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value).It is loaded permanently if the preload feature is not selected in the CCMR2 register (bit OC4PE). Else the preload value is copied in the active capture/compare 4 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter CNT and signalled on OC4 output. 2. if CC4 channel is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4). The CCR4 register is read-only and cannot be programmed.

9.3.5 Timer Cascade

Table 9-4: Timer Cascade

Subordinate Timer	Cascade Port	Superior Timer
ATIM1	ITR0	BTIM2
	ITR1	GPTIM2
	ITR2	GPTIM1
	ITR3	BTIM1
GPTIM1	ITR0	GPTIM2
	ITR1	BTIM2
	ITR2	ATIM1
	ITR3	BTIM1
GPTIM2	ITR0	GPTIM1
	ITR1	ATIM1
	ITR2	BTIM1
	ITR3	BTIM2
BTIM1	ITR0	BTIM2
	ITR1	/
	ITR2	ATIM1
	ITR3	GPTIM2
BTIM2	ITR0	GPTIM1
	ITR1	BTIM1
	ITR2	/
	ITR3	ATIM1
GPTIM3	ITR0	BTIM3
	ITR1	BTIM4
	ITR2	GPTIM4
	ITR3	GPTIM5
GPTIM4	ITR0	GPTIM3
	ITR1	GPTIM5
	ITR2	BTIM3
	ITR3	BTIM4
GPTIM5	ITR0	GPTIM3
	ITR1	GPTIM4

Continued on next page...

Table 9-4: Timer Cascade (Continued)

Subordinate Timer	Cascade Port	Superior Timer
	ITR2	BTIM3
	ITR3	BTIM4
BTIM3	ITR0	GPTIM3
	ITR1	GPTIM4
	ITR2	GPTIM5
	ITR3	BTIM4
BTIM4	ITR0	/
	ITR1	/
	ITR2	GPTIM4
	ITR3	BTIM3

9.4 LPTIM

The chip contains a total of 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/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/output connected to IO(PB) and IO(PBR).

9.4.1 Introduction

LPTIM (Low Power Timer) is based on a 24-bit incrementing counter, capable of timing, generating output waveforms (output compare and PWM) , and waking up the system, among other functions. The counting clock can be the system clock, low power clock, IO input signal, or comparator output, and can support up to 128 times prescaling and up to 256 cycles of counting. Based on the counting results, it can generate PWM outputs, trigger interrupts, or produce wake-up signals to wake the system from Low Power Mode. When using IO input signals as the counting clock, it supports counting and generating wake-up signals independently of the internal clock.

9.4.2 Main Features

- 24-bit up-counting auto-reload counter, maximum count of 16777215(2^{24}-1)
- Counting Clock Selection
  - Internal clock, PCLK2, or low power clock
  - Optional edge-triggered IO input signal or comparator output; can utilize the internal clock for debouncing, or count independently without relying on the internal clock
- 8-level pre-scaler, with a counting clock division factor of 2 raised to the power of 0 to 7
- 1 to 256 loop counts
- Counting Mode
  - Continuous Counting Mode
  - Single Pulse Mode; counting ends after the loop count is completed
- Configurable polarity output mode
  - PWM output, with adjustable pulse width and period
  - Single toggle output
  - Single pulse or a specifed number of pulse outputs

- Trigger mode
  - Software Trigger
  - IO input signal edge-triggered, supporting debounce 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

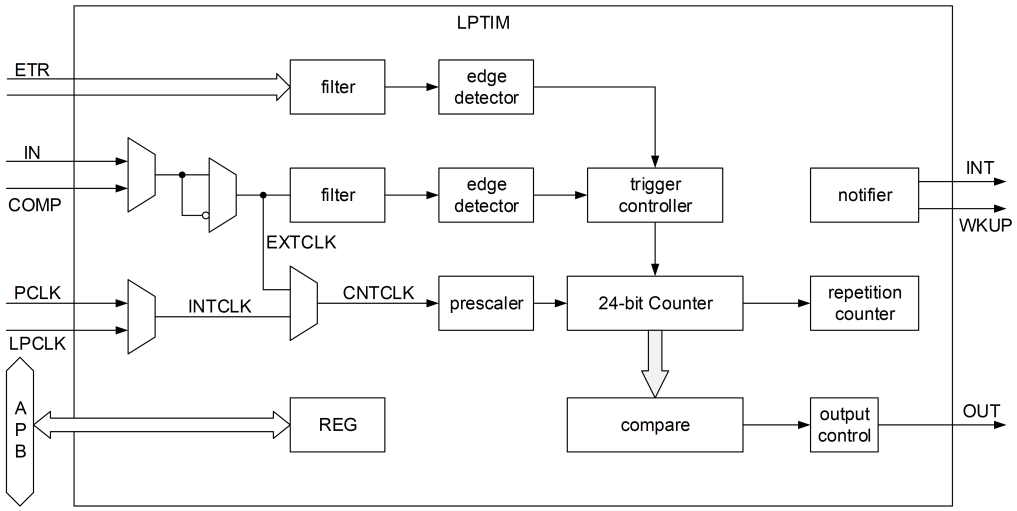


Figure 9-13: LPTIM structure diagram

9.4.3 LPTIM function description

9.4.3.1 counter

The functions of LPTIM are based on a 24 bit counter. The counter operates on event counting, including clock edges, external input edges, and comparator result edges.

Counting events will only be registered in the counter after undergoing pre-scaling processing. The number of pre-scales ranges from 1 to 128 ($2^{CFGR\_PRESCPSC}$), meaning that the counter's value will only change once after ($2^{CFGR\_PRESCPSC}$) counting events have occurred.

The counter is 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.

9.4.3.2 Counting Clock

The counting clock of the LPTIM CNTCLK can be selected from multiple sources. The most commonly used default mode is to select the internal low-power clock LPCLK , which allows the LPTIM to continue operating after the chip enters low-power sleep mode. When not in low-power sleep mode, the LPTIM can also select the internal PCLK as the clock. Additionally, the LPTIM can count using an external signal IN or a comparator output signal COMP without relying on the internal clock. The registers related to clock selection include CFGR_EXTCKSEL/INTCKSEL/CKSEL . The polarity of the external clock can be selected using CFGR_CKPOL .

When the internal clock is selected, it is also possible to count the events of the external signal IN or the comparator output signal COMP flipping, and to perform pre-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.

9.4.3.3 Update Event(UEV)

The update event is used to signify the end of a counting unit. The most basic update event occurs each time the counter overflows (when repeat counting is disabled) . Update events can generate interrupts and wake-up signals, serving as the most fundamental notifcation function of the timer

9.4.3.4 Repeat Counting

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

The current value of the Repeat Counter can be read through RCR . Since the counting clock is asynchronous with the APB clock, the values must match for two consecutive reads to be considered a valid data read.

9.4.3.5 Counter Trigger

The counter can be configured for single-level trigger mode(CFGR_TRIGEN=00)or dual-level trigger mode(CFGR_TRIGEN!=00).

Single-level trigger mode is software-triggered and includes both single trigger and continuous trigger types. Before triggering, the CR_ENABLE must be set to 1 to enable the counter. Then, set CR_SNGSTRT to 1 to initiate a single trigger, causing the counter to start immediately and stop after the Update Event; alternatively, set CR_CNTSTRT to 1 to initiate continuous triggering, causing the counter to start immediately and continue counting until it is disabled or reset.

The two-level trigger mode incorporates a hardware trigger mechanism in addition to software triggering, activating the counter only when the filtered ETRoptional edge arrives.

9.4.3.6 Timeout Monitoring

In the two-level trigger mode, if CFGR_TIMOUT is set to 1 , the counter will reset and restart each time a hardware trigger occurs. This feature can be utilized to monitor the interval between two consecutive trigger signals, generating a comparison or update event when the interval exceeds the expected duration.

9.4.3.7 PWM Output

LPTIM can generate a single-channel PWM output with controllable period and duty cycle to the OUT port. The period of the PWM output is determined by ARR , while the duty cycle is determined by CMP . The counter value CNT is compared with CMP to generate PWM , with polarity configured by CFGR_WAVEPOL . In single trigger mode, a single pulse or multiple pulses can be generated based on the value of RCR . In continuous trigger mode, a sustained PWM can be produced. If CFGR_WAVE is set to 1 , a single pulse waveform can be generated.

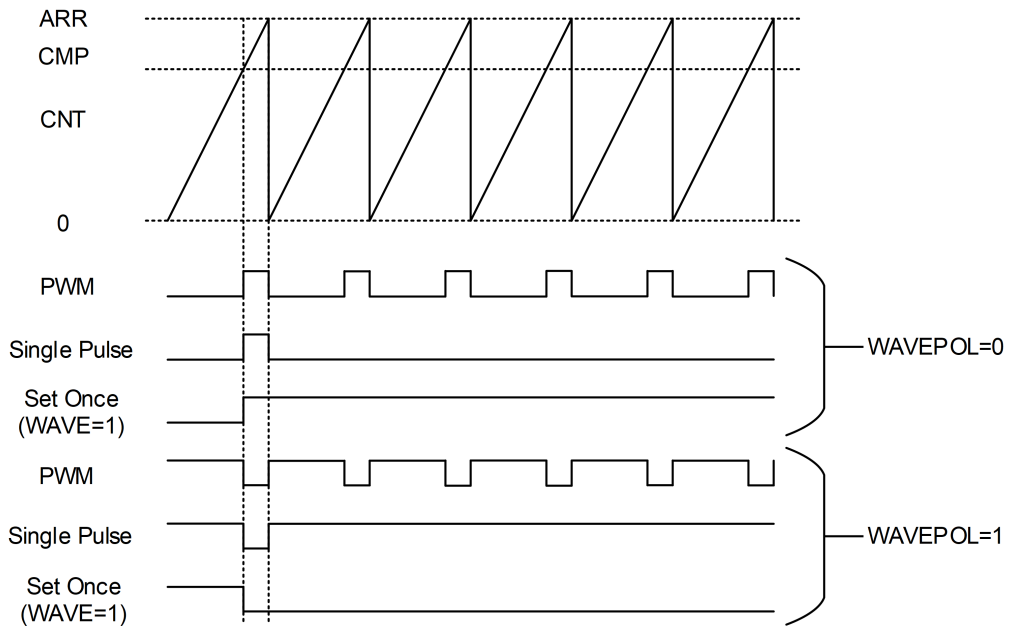


Figure 9-14: PWM Output

9.4.3.8 Notifcation Mechanism

LPTIM can generate notifcations such as interrupts and wake-up signals. Interrupts are generated only when the system is in a non-low power sleep state. Wake-up signals can be generated regardless of whether the system is in a low power sleep state and can wake the system. The events that can trigger notifcations primarily include overflow events, update events, trigger events, and comparator matches. The IER register can control whether various events generate interrupts and wake-ups. The status of each event can be queried in the ISR register.

9.4.4 LPTIM Register

Table 9-5: LPTIM Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**ISR**	LPTIM interrupt and status register
[31:11]			RSVD	
[10]	r	1'h0	OCWKUP	Indicates output compare wakeup occurred The OCWKUP bit is set by hardware when LPTIM_CNT register value reached the LPTIM_CMP register's value. To clear OCWKUP, first write 0 to the OCWE bit in the LPTIM_IER register to disable, then write 1 to the WKUPCLR bit in the LPTIM_ICR register.
[9]	r	1'h0	OFWKUP	Indicates overflow wakeup occurred OFWKUP is set by hardware when LPTIM_CNT register's value reached the LPTIM_ARR register's value and count from zero again. To clear OFWKUP, first write 0 to the OFWE bit in the LPTIM_IER register to disable, then write 1 to the WKUPCLR bit in the LPTIM_ICR register.
[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 9-5: 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 9-5: 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 9-5: LPTIM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:9]	rw	3'h0	PRESC	Clock prescaler The PRESC bits configure the prescaler division factor. It can be one among the following division factors: 000: /1 001: /2 010: /4 011: /8 100: /16 101: /32 110: /64 111: /128
[8]	rw	1'h0	EXTCKSEL	External clock source selector 0: external clock source is from lptim_in 1: external clock source is from LPCOMP (if LPCOMP integrated)
[7:6]	rw	2'h0	TRGFLT	Configurable digital filter for trigger The TRGFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an internal trigger before it is considered as a valid level transition. An internal clock source must be present to use this feature 00: any trigger active level change is considered as a valid trigger 01: trigger active level change must be stable for at least 2 clock periods before it is considered as valid trigger. 10: trigger active level change must be stable for at least 4 clock periods before it is considered as valid trigger. 11: trigger active level change must be stable for at least 8 clock periods before it is considered as valid trigger.
[5]	rw	1'h0	INTCKSEL	Internal clock source selector 0: internal clock source is clk_lp 1: internal clock source is pclk2
[4:3]	rw	2'h0	CKFLT	Configurable digital filter for external clock The CKFLT value sets the number of consecutive equal samples that should be detected when a level change occurs on an external clock signal before it is considered as a valid level transition. An internal clock source must be present to use this feature 00: any external clock signal level change is considered as a valid transition 01: external clock signal level change must be stable for at least 2 clock periods before it is considered as valid transition. 10: external clock signal level change must be stable for at least 4 clock periods before it is considered as valid transition. 11: external clock signal level change must be stable for at least 8 clock periods before it is considered as valid transition.
[2:1]	rw	2'h0	CKPOL	Clock Polarity If LPTIM is clocked by an external clock source, CKPOL bits is used to configure the active edge or edges used by the counter: 00: the rising edge is the active edge used for counting 01: the falling edge is the active edge used for counting 10: both edges are active edges. When both external clock signal edges are considered active ones, the LPTIM must also be clocked by an internal clock source with a frequency equal to at least four time the external clock frequency. 11: not allowed

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	CKSEL	Clock selector The CKSEL bit selects which clock source the LPTIM will use: 0: LPTIM is clocked by internal clock source, according to INTCKSEL 1: LPTIM is clocked by external clock source, according to EXTCKSEL
0x10			**CR**	LPTIM control register
[31:4]			RSVD	
[3]	rw	1'h0	COUNTRST	Counter reset This bit is set by software and cleared by hardware. When set to 1 this bit will trigger a synchronous reset of the CNT register. Due to the synchronous nature of this reset, it only takes place after a synchronization delay. COUNTRST must never be set to 1 by software before it is already cleared to 0 by hardware. Software should consequently check that COUNTRST bit is already cleared to 0 before attempting to set it to 1.
[2]	w	1'h0	CNTSTRT	Timer start in Continuous mode This bit is set by software and cleared by hardware. In case of software start (TRIGEN[1:0] = 00), setting this bit starts the LPTIM in Continuous mode. If the software start is disabled (TRIGEN[1:0] different than 00), setting this bit starts the timer in Continuous mode as soon as an external trigger is detected. If this bit is set when a single pulse mode counting is ongoing, then the timer will not stop at the next match between ARR and CNT registers and the LPTIM counter keeps counting in Continuous mode.
[1]	w	1'h0	SNGSTRT	LPTIM start in Single mode This bit is set by software and cleared by hardware. In case of software start (TRIGEN[1:0] = 00), setting this bit starts the LPTIM in single pulse mode. If the software start is disabled (TRIGEN[1:0] different than 00), setting this bit starts the LPTIM in single pulse mode as soon as an external trigger is detected. If this bit is set when the LPTIM is in continuous counting mode, then the LPTIM will stop at the following match between ARR and CNT registers. If this bit is set simultaneously with CNTSTRT, then LPTIM will be in continuous counting mode.
[0]	rw	1'h0	ENABLE	LPTIM enable The ENABLE bit is set and cleared by software. 0:LPTIM is disabled 1:LPTIM is enabled
0x14			**CMP**	LPTIM compare register
[31:24]			RSVD	
[23:0]	rw	24'h0	CMP	Compare value CMP is the compare value used by the LPTIM.
0x18			**ARR**	LPTIM autoreload register
[31:24]			RSVD	
[23:0]	rw	24'h0	ARR	Auto reload value ARR is the autoreload value for the LPTIM. This value must be strictly greater than the CMP[15:0] value.
0x1C			**CNT**	LPTIM counter register
[31:24]			RSVD	
[23:0]	r	24'h0	CNT	Counter value When the LPTIM is running with an asynchronous clock, reading the CNT register may return unreliable values. So in this case it is necessary to perform two consecutive read accesses and verify that the two returned values are identical.
0x20			**RCR**	LPTIM repetition register
[31:8]			RSVD	

Continued on the next page...

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

Offset	Attribute	Reset Value	Register Name	Register Description
[7:0]	rw	8'h0	REP	Repetition register value REP is the repetition value for the LPTIM. Read REP will return left repetition times. It should be noted that for a reliable REP register read access, two consecutive read accesses must be performed and compared. A read access can be considered reliable when the values of the two consecutive read accesses are equal.

9.5 WDT

The chip contains a total of 3 WDTs, among which WDT1 is located in HPSYS and stops operating when HPSYS enters deepsleep, standby, or hibernate modes; WDT2 is located in LPSYS and ceases operation when LPSYS enters deepsleep, standby, or hibernate modes. IWDT is located in AON and remains operational after the chip enters low-power mode. The specific reset scope of each WDT is detailed in the Clock and Reset chapter.

9.5.1 Introduction

The watchdog timer, as a type of counter, is primarily used to reset the system after a predetermined time to prevent software hangs.

Basic functions of the watchdog timer:

- Supports two operating modes:
  - mode0
    - The wdt does not generate an interrupt; it directly resets the system after the set time is reached.
    - Supports up to a 24-bitcounter.
  - mode1
    - Divided into two counting segments; an interrupt is generated after the first segment's set time is reached, and the system is reset after the second segment's set time is reached.
    - Each time segment supports up to a 24-bitcounter.
- Supports write protection to prevent erroneous operations on the wdt.

9.5.2 Operating modes of the WDT

There are two resetgeneration modes based on requirements:

Mode 1: Counts for one round only, directly generating a resetsignal at the end of the count.

Mode 2: Counts for two rounds, generating an interrupt at the end of the first round and a resetsignal at the end of the second round.

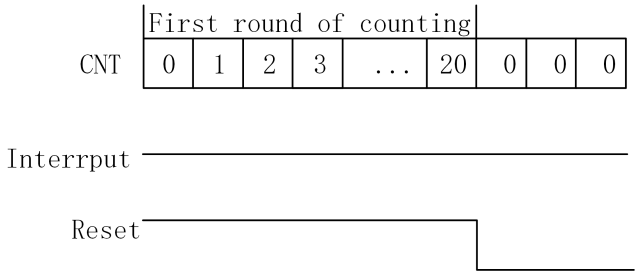


Figure 9-15: The relationship between the interrupt and resetsignal generation in Mode 1 and the counter (assuming a timeout value of 20, withresetactive low)

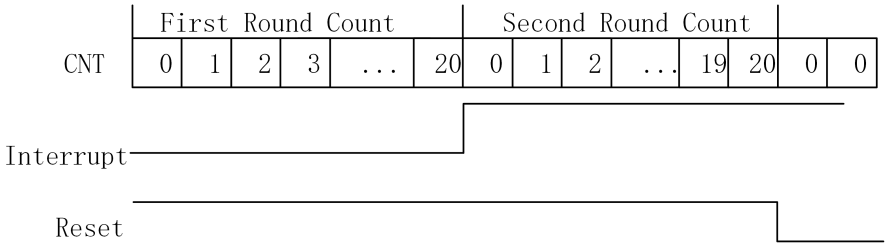


Figure 9-16: The relationship between the interrupt and resetsignal generation in Mode 2 and the counter (assuming a timeout value of 20, withresetactive low).

The 'dog feeding' behavior in both modes:

In Mode 1, if 'feeding the dog' occurs before the end of the first round of counting, the counter will restart counting from zero.

In mode 2 , if the first round of counting concludes before the 'dog is fed', the counter will restart from zero; if there is no 'dog feeding' action before the first round of counting concludes, the wdt will enter the second round of counting. At this point, either clearing the interrupt or performing the 'dog feeding' action will allow the wdt to restart counting from the beginning of the first round.

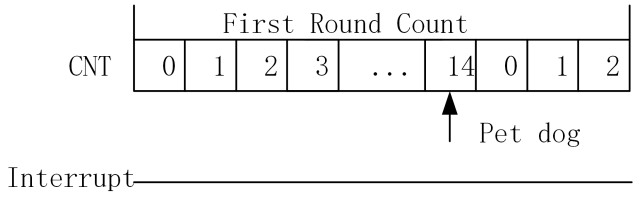


Figure 9-17: Mode 1 The effect of the 'dog feeding' action on the counter (assuming the timeout value is 20)

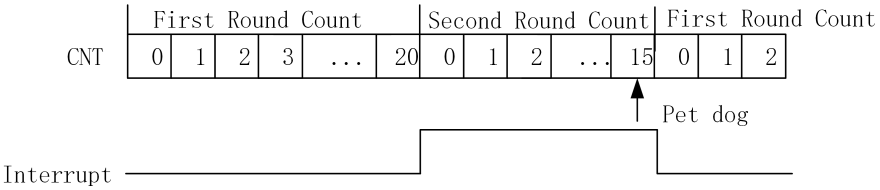


Figure 9-18: Mode 2 The effect of the 'dog feeding' operation on the counter during the second round of counting is the same as in the first round (assuming the timeout value is 20)

Methods to stop wdtin two modes:

In mode 1, configure register0x0C(counter_control)=0x34before the count ends, andwdtwill stop.

In mode 2 , configure register 0x0C(counter_control)=0x34 before the first round count ends, andwdt will stop; if in the second round counting stage, it is necessary to clear the interrupt or perform the 'feed the dog' operation to return wdt to the first round count before configuring register 0x0C(counter_control)=0x34 to stop wdt.

Method for Clearing Interrupts:

In mode 2, wdt will generate an interrupt after the first round of counting is completed, with 0x14 WDT_SR indicating int_assert as 1. At this point, it can be cleared by configuring 0x10's WDT_IDR with int_clr, or by configuring 0xc WDT_CCR with counter_control set to 0x76, which means feeding the watchdog to clear.

9.5.2.1 WDT Register Configuration Process

1. Select the working mode of wdt as needed. Configure 0x08 for the response_mode register, set 0x0 to select mode 1, and set 0x1 to select mode 2.
2. Configure 0x00 for count_value_0 register (the timeout value for the first round of counting in both modes) and 0x04 for count_value_1 (register for the timeout value of the second round counter in mode 2
3. Configure 0x08 for reset lengthr equirements.
4. Configure 0x0c in the counter_control Register (= 0x76) to trigger the wdt to start working.
   The sequence of steps 1 to 3 is not mandatory, provided it is completed before step 4.

9.5.2.2 Note

1. The wdtprovides awrite protectfeature to prevent accidental rewriting of the configurations inwdt, as follows:
   Set 0x18 for wrpt Register to 0x58ab99fc. When wrpt_st in 0x18 Register is 1, it indicates that write protection is enabled. In this state, all registers in wdt cannot be rewritten, but read operations are unaffected. To disable write protection, set 0x18 for wrpt Register to 0x51ff8621.
2. wdt's register address 0x1c, sync_fg register set to 1 indicates that the start , stop , irq clear , and reset flag clear operations have been synchronized from pclk to wdt clk and are now effective.
3. rst_fg set to 1 indicates that this wdt has undergone a reset; this register is only valid for iwdt.
4. If 0xc cannot write to counter_control , first check if the corresponding sys rcc wdt clock enable register is configurd to 1.

9.5.3 WDT Register

Table 9-6: WDT Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**WDT_CVR0**	WatchDog Counter Value 0
[31:24]			RSVD	
[23:0]	rw	24'hffffff	count_value_0	Count Value for 1st TimeOut
0x04			**WDT_CVR1**	WatchDog Counter Value 1
[31:24]			RSVD	
[23:0]	rw	24'hffffff	count_value_1	Count Value for 2nd TimeOut
0x08			**WDT_CR**	WatchDog Control Register
[31:5]			RSVD	
[4]	rw	1'b1	response_mode	0:reset only, 1:interrupt and reset
[3]			RSVD	

Continued on the next page...

Table 9-6: WDT Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[2:0]	rw	3'b000	reset_length	reset pulse length in number of wdt clock cycles
0x0C			**WDT_CCR**	WatchDog Counter Control Register
[31:8]			RSVD	
[7:0]	rw	8'h0	counter_control	SinglePulse /Write 8'h76 to restart, write8'h34 to stop, else do nothing
0x10			**WDT_ICR**	WatchDog Interrupt Clear Register
[31:1]			RSVD	
[0]	w1c	1'b0	int_clr	SinglePulse /A pulse to clear interrupt
0x14			**WDT_SR**	WatchDog Status Register
[31:2]			RSVD	
[1]	r	1'b0	wdt_active	Watchdog runs when 1, else 0
[0]	r	1'b0	int_assert	Interrupt assert when 1
0x18			**WDT_WP**	WatchDog Write Protect Register
[31]	r	1'b0	wrpt_st	1 indicates write protect is active
[30:0]	w	31'h0	wrpt	write 0x58ab99fc generate write_protect, write 0x51ff8621 to release
0x1C			**WDT_FG**	WatchDog Flag Register
[31:4]			RSVD	
[3]	r	1'b0	sync_fg	1 indicates one transition from system clk to wdt clk has complicated
[2]	w1c	1'b0	sync_fg_clr	SinglePulse/A pulse to clear sync flag
[1]	r	1'b0	rst_fg	1 indicates wdt has already reset system
[0]	w1c	1'b0	rst_fg_clr	SinglePulse/A pulse to clear reset flag

9.6 RTC

9.6.1 RTC Register

Table 9-7: RTC Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**TR**	Time Register
[31]	rw	1'h0	PM	AM/PM notation 0: AM 1: PM
[30:29]	rw	2'h0	HT	Hour tens in BCD format
[28:25]	rw	4'h0	HU	Hour units in BCD format
[24:22]	rw	3'h0	MNT	Minute tens in BCD format
[21:18]	rw	4'h0	MNU	Minute units in BCD format
[17:15]	rw	3'h0	ST	Second tens in BCD format
[14:11]	rw	4'h0	SU	Second units in BCD format
[10]			RSVD	
[9:0]	r	10'h0	SS	Sub-second counter
0x04			**DR**	Date Register
[31]	r	1'h0	ERR	reserved for debug
[30:25]			RSVD	
[24]	rw	1'h0	CB	century bit, 0 - 2000s, 1 - 1900s/2100s
[23:20]	rw	4'h0	YT	Year tens in BCD format
[19:16]	rw	4'h0	YU	Year units in BCD format

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:13]	rw	3'h1	WD	Week day units 000: forbidden 001: Monday ... 111: Sunday
[12]	rw	1'h0	MT	Month tens in BCD format
[11:8]	rw	4'h1	MU	Month units in BCD format
[7:6]			RSVD	
[5:4]	rw	2'h0	DT	Date tens in BCD format
[3:0]	rw	4'h1	DU	Date units in BCD format
0x08			**CR**	Control Register
[31:14]			RSVD	
[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	
[5:4]			RSVD	
[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	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[10]			RSVD	
[9:0]	rw	10'h0	SS	
0x1C			**ALRMDR**	Alarm Date Register
[31:30]			RSVD	
[29]	rw	1'h0	MSKWD	
[28]	rw	1'h0	MSKM	
[27]	rw	1'h0	MSKD	
[26]	rw	1'h0	MSKH	
[25]	rw	1'h0	MSKMN	
[24]	rw	1'h0	MSKS	
[23:20]	rw	4'h0	MSKSS	
[19:16]			RSVD	
[15:13]	rw	3'h1	WD	
[12]	rw	1'h0	MT	
[11:8]	rw	4'h1	MU	
[7:6]			RSVD	
[5:4]	rw	2'h0	DT	
[3:0]	rw	4'h1	DU	
0x20			**SHIFTR**	Shift Control Register
[31]	rw	1'b0	ADD1S	
[30:10]			RSVD	
[9:0]	rw	10'h0	SUBFS	
0x24			**TSTR**	Timestamp Time Register
[31]	r	1'h0	PM	
[30:29]	r	2'h0	HT	
[28:25]	r	4'h0	HU	
[24:22]	r	3'h0	MNT	
[21:18]	r	4'h0	MNU	
[17:15]	r	3'h0	ST	
[14:11]	r	4'h0	SU	
[10]			RSVD	
[9:0]	r	10'h0	SS	
0x28			**TSDR**	Timestamp Date Register
[31:16]			RSVD	
[15:13]	r	3'h1	WD	
[12]	r	1'h0	MT	
[11:8]	r	4'h1	MU	
[7:6]			RSVD	
[5:4]	r	2'h0	DT	
[3:0]	r	4'h1	DU	
0x2C			**OR**	
[31:0]			RSVD	
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	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x44			**BKP5R**	Backup 5 Register
[31:0]	rw	32'h0	BKP	
0x48			**BKP6R**	Backup 6 Register
[31:0]	rw	32'h0	BKP	
0x4C			**BKP7R**	Backup 7 Register
[31:0]	rw	32'h0	BKP	
0x50			**BKP8R**	Backup 8 Register
[31:0]	rw	32'h0	BKP	
0x54			**BKP9R**	Backup 9 Register
[31:0]	rw	32'h0	BKP	
0x58			**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	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xAC			**BKP31R**	Backup 31 Register
[31:0]	rw	32'h0	BKP	
0xB0			**PBRCR**	PBR Control Register
[31:2]			RSVD	
[1]	rw	1'b1	SNS	reserved for debug
[0]	rw	1'b1	RTO	reserved for debug
0xB4			**PBR0R**	PBR0 Register
[31]	rw	1'b0	FORCE1	1: force output value to 1 when output enabled (PBR0R_OE=1)
[30:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value when PBR0R.FORCE1=0 0: lpsys_ldo_req 1: value set by PBR0R.OUT 2: selected by CR1.PBR_SEL0 in LPSYS_AON (LPTIM3_OUT if PBR_SEL0=2, ~LPTIM3_OUT if PBR_SEL0=3) 3: selected by CR1.PBR_SEL1 in LPSYS_AON (LPTIM3_OUT if PBR_SEL1=2, ~LPTIM3_OUT if PBR_SEL1=3) others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b1	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR0R.SEL is 0
0xB8			**PBR1R**	PBR1 Register
[31:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR1R.OUT 1: clk_lp 2: selected by CR1.PBR_SEL0 in LPSYS_AON (LPTIM3_OUT if PBR_SEL0=2, ~LPTIM3_OUT if PBR_SEL0=3) 3: selected by CR1.PBR_SEL1 in LPSYS_AON (LPTIM3_OUT if PBR_SEL1=2, ~LPTIM3_OUT if PBR_SEL1=3) others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b1	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR1R.SEL is 0
0xBC			**PBR2R**	PBR2 Register
[31:15]			RSVD	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR2R.OUT 1: clk_lp 2: selected by CR1.PBR_SEL0 in LPSYS_AON (LPTIM3_OUT if PBR_SEL0=2, ~LPTIM3_OUT if PBR_SEL0=3) 3: selected by CR1.PBR_SEL1 in LPSYS_AON (LPTIM3_OUT if PBR_SEL1=2, ~LPTIM3_OUT if PBR_SEL1=3) others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b1	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR2R.SEL is 0
0xC0			**PBR3R**	PBR3 Register
[31:15]			RSVD	
[14:12]	rw	3'h0	SEL	select output value 0: value set by PBR3R.OUT 1: clk_lp 2: selected by CR1.PBR_SEL0 in LPSYS_AON (LPTIM3_OUT if PBR_SEL0=2, ~LPTIM3_OUT if PBR_SEL0=3) 3: selected by CR1.PBR_SEL1 in LPSYS_AON (LPTIM3_OUT if PBR_SEL1=2, ~LPTIM3_OUT if PBR_SEL1=3) others: reserved
[11:10]			RSVD	
[9]	r	1'b0	IN	Input value
[8]	rw	1'b0	DS1	Drive Select 1. Used to select output drive strength
[7]	rw	1'b1	DS0	Drive Select 0. Used to select output drive strength
[6]	rw	1'b0	SR	Slew Rate. Logic HIGH selects slow slew rate, logic LOW selects fast slew rate
[5]	rw	1'b1	IS	Input Select. Logic LOW selects CMOS input, logic HIGH selects Schmitt input
[4]	rw	1'b0	PS	Pull Select. Logic HIGH selects pull-up, logic LOW select pull-down
[3]	rw	1'b0	PE	Pull Enable. Logic HIGH enables week pull device
[2]	rw	1'b0	IE	Input Enable. Logic HIGH enables the input buffer
[1]	rw	1'b1	OE	Output enable. Active high
[0]	rw	1'h0	OUT	Output value if PBR3R.SEL is 0
0xC4			**PAWKUP**	PA Wakeup Register
[31:21]			RSVD	
[20:16]	rw	5'h1f	IS	Input Select of wakeup pin. Logic LOW selects CMOS input, logic HIGH selects Schmitt input. Keep effective during hibernate mode. bit 0: input select of PA50 bit 1: input select of PA51 bit 2: input select of PA52 bit 3: input select of PA53 bit 4: input select of PA54
[15:13]			RSVD	

Continued on the next page...

Table 9-7: RTC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[12:8]	rw	5'h0	PS	Pull select of wakeup pin. Logic HIGH selects pull-up, logic LOW select pull-down. Keep effective during hibernate mode. bit 0: pull select of PA50 bit 1: pull select of PA51 bit 2: pull select of PA52 bit 3: pull select of PA53 bit 4: pull select of PA54
[7:5]			RSVD	
[4:0]	rw	5'h0	PE	Pull enable of wakeup pin, active high. Keep effective during hibernate mode. Caution: Pull-functions of all pins can be configured in HPSYS_PINMUX registers, but in hibernate mode, such information will be lost, which may lead to wakup pin leakage. Pull-functions set here will keep active in hibernate mode for wakeup pins to avoid leakage. bit 0: pull enable of PA50 bit 1: pull enable of PA51 bit 2: pull enable of PA52 bit 3: pull enable of PA53 bit 4: pull enable of PA54
0xC8			**PBWKUP**	PB Wakeup Register
[31:21]			RSVD	
[20:16]	rw	5'h1f	IS	Input Select of wakeup pin. Logic LOW selects CMOS input, logic HIGH selects Schmitt input. Keep effective during hibernate mode. bit 0: input select of PB32 bit 1: input select of PB33 bit 2: input select of PB34 bit 3: input select of PB35 bit 4: input select of PB36
[15:13]			RSVD	
[12:8]	rw	5'h0	PS	Pull select of wakeup pin. Logic HIGH selects pull-up, logic LOW select pull-down. Keep effective during hibernate mode. bit 0: pull select of PB32 bit 1: pull select of PB33 bit 2: pull select of PB34 bit 3: pull select of PB35 bit 4: pull select of PB36
[7:5]			RSVD	
[4:0]	rw	5'h0	PE	Pull enable of wakeup pin, active high. Keep effective during hibernate mode. Caution: Pull-functions of all pins can be configured in LPSYS_PINMUX registers, but in hibernate mode, such information will be lost, which may lead to wakup pin leakage. Pull-functions set here will keep active in hibernate mode for wakeup pins to avoid leakage. bit 0: pull enable of PB32 bit 1: pull enable of PB33 bit 2: pull enable of PB34 bit 3: pull enable of PB35 bit 4: pull enable of PB36

10 Graphics

10.1 ePicasso™ High-performance 2.5Dgraphics engine

ePicasso is located in HPSYS.

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

10.1.1 Layer Blending

ePicasso™ supports up to four foreground layers, one dedicated mask layer, and one monochrome background layer for blending, with input and output formats including the commonly used RGB565、RGB888、ARGB8565、ARGB8888、L8、A8、A4. Each foreground layer possesses an independent blending mode and blending region; the mask layer is primarily used to extract specific shapes within the image. Furthermore, each layer offers a separate filter configuration option, enabling the layer to filter out a specific color. This feature can be utilized for basic image capture.

10.1.2 Graphics Scaling

ePicasso ™ features a layer called the function layer, which, in addition to supporting blending capabilities, can also perform graphics scaling. The maximum scaling ratio can reach 1024 times, with a precision of up to 1/65536 . In the X and Y directions, the scaling ratios can be configured independently to accommodate various requirements.

10.1.3 Graphic Rotation

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

10.2 LCDC

LCDC is located in HPSYS.

10.2.1 Introduction

LCDC, the full name LCD Controller, primarily functions to read image data from memory and then transmit the data to the corresponding display according to different screen interfaces. LCDC supports a wide range of screen interfaces, including:

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

In addition to a variety of interfaces, the LCDC supports multiple image data formats, including RGB565, RGB888, and ARGB8888. Building on basic functionality, the LCDC also supports simple layer aliasing, enabling fundamental image processing acceleration.

10.2.2 Architecture Introduction

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

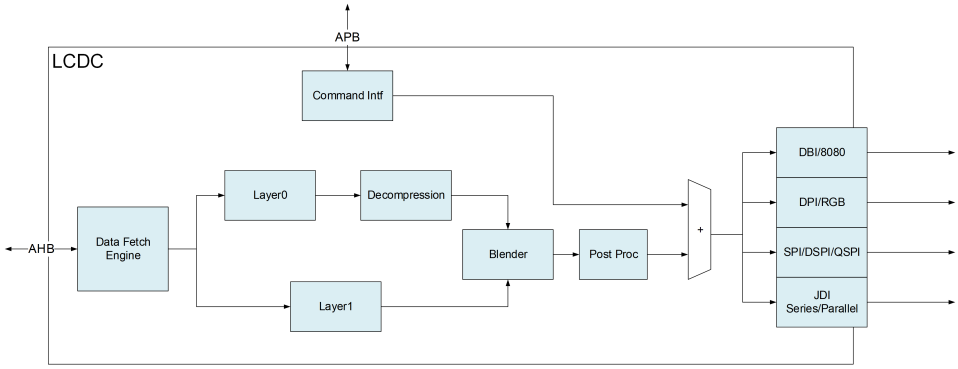


Figure 10-1: LCDC Architecture Diagram

The diagram contains two primary data paths.

1. configuration path: The upper layer transmits screen configuration commands via the APB bus. Upon receiving these commands, theLCDC forwards the corresponding data to the respective screen interfaces through the Command Intf module. This path operates at a lower speed and is mainly used for screen initialization configuration and fundamental functionality 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 data for two layers is prepared, Blender performs aliasing on the image data and sends it to the corresponding screen interface. This path is the primary channel through which the LCDC sends image data to the screen. It should also be noted that both Layer0 and Layer1 can be enabled independently. Even if neither is enabled, the Blender can be configured to send monochrome data to the screen.

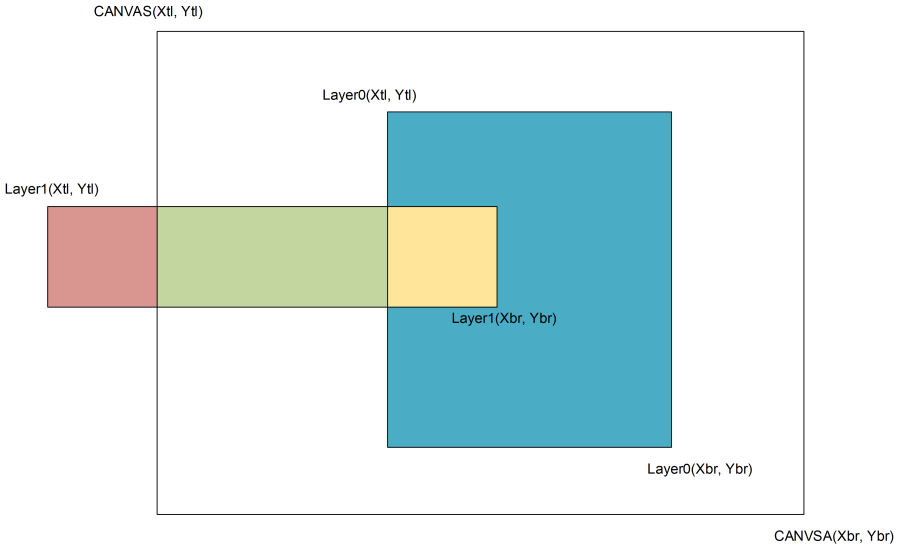
10.2.3 configuration Procedure

The configuration of the LCDC is mainly divided into two parts. The first part is layer configuration, which addresses the configuration of the two Layers and the Blender. The second part is interface configuration, which addresses the parameter configuration corresponding to different interfaces.

10.2.3.1 Layer configuration

Layer configuration includes the configurations of Layer0/Layer1 and the Blender module. Users can configure the Layer0_CONFIG and Layer1_CONFIG registers to enable Layer0 and Layer1 and set their color formats and aliasing 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 set to RGB888.

When layer data is sent to the Blender, the Blender performs aliasing 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. For details, please refer to the fgure below:



In the fgure, CANVAS, Layer0, and Layer1 each have two sets of coordinates representing their respective rectangular regions, (Xtl, Ytl) indicating the top-left corner coordinates of the rectangle, (Xbr, Ybr) indicating the bottom-right corner coordinates. It can be observed that Layer0 is entirely within the CANVAS, so its entire region participates in aliasing calculations. Layer1 consists of three parts: the red area lies outside the CANVAS and therefore does not participate in the calculations; the yellow area overlaps with both the CANVAS and Layer0, so the Blender computes these two layers together with the CANVAS; fnally, the green area overlaps only with the CANVAS, so it undergoes aliasing calculations 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 frame buffer (FrameBuffer) to the screen, it is sufcient to configure the layer coordinates to match the CANVAS coordinates.

After Blender outputs, it passes through the Post Processing module, whose Dithering adds a certain amount of noise to the original image, making the image visually smoother; additionally, this module supports horizontal mirroring of the image.

10.2.3.2 Interface configuration

Due to signifcant differences in transmission protocols across various interfaces, the corresponding configuration parameters also differ. The following lists the configuration parameters according to each interface.

DPI/RGB Interface:

DPI/RGB is a synchronous parallel interface driven by the PixelClock. Its main configuration parameters are listed in the table below:

Parameter Name	Description	configuration
Freq(pclk)	Pixel Clock Rate	PCLK_DIV , the Pixel Clock division ratio, with the source clock being the system clock
Hsync Width	Hsync Signal Active Period Count	HSW , the period corresponding to the Pixel Clock
Vsync Height	Vsync Signal Active Line Coun	VSH
Horizontal Active Width	Active Line Width	HAW , the number of active Pixels per line
Vertical Active Height	Active Line Count	VAH , the number of lines containing active 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 paramete	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 DPI interface configuration, configuring the DPI_EN signal will enable the DPI interface. Once enabled, only by setting DPI_EN to 0 can the LCDC data transmission be stopped; otherwise, the LCDC will continuously send the Framebuffer 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, theDBIparallel interface is divided into two types: TypeA and TypeB, which differ slightly in signals but can be logically converted between each other. When selecting the LCDCinterface, the corresponding TypeA and TypeB can also be selected.

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

Parameter Name	Description	configuration
PWH	Number of cycles for the WRX/RDX signal inactive state	The cycle corresponds to the system clock
PWL	Number of cycles for the WRX/RDX signal active state	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 constitute the primary configuration parameters for the DBI interface, where PWH and PWL determine the DBI interfacerate; their sum corresponds to the number of cycles per single data transfer.

It should be noted that all DBI interface signals are active low by default. To adjust the active phase, phase inversion can be implemented by configuring the POL register associated with the respective signal.

Besides the image transmission path, the DBI interface also supports configuration paths. Upon completing the DBI interface configuration, the user can first set the value to be written in the LCD_WR register, trigger read/write operations via the WR_TRIG and RD_TRIG registers, and subsequently retrieve the read value through the LCD_RD register.

SPI/DSPI/QSPI interfaces:

SPI,DSPI, and QSPI all belong to the SPI type of display interfaces, differing in the number of data lines. SPI typically has a single data line,DSPI has two, and QSPI has four. Apart from the difference in the number of data lines, the SPI interface is also classifed into 3-wire SPI and 4-wire SPI based on the transmission method of the D/C (Data/Command selection) signal. In 3-wire SPI, the D/C signal acts as a data bit transmitted via SDO, whereas in 4-wire SPI, the D/C has a dedicated line for indication; therefore, 3-wire SPI has a slightly lower effective bandwidth compared to 4-wire SPI.

The primary parameter configuration of the SPI interface is detailed in the following table:

Parameter Name	Description	configuration
CLK_DIV	SSPI Clock division ratio	The source clock is the system clock
LINE	SPI Mode	Different modes include3-wire/4-wireand 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 acquire data in time to send to the display, 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 acquire data in time to send to the display, it will automatically stop the SPI interface clock signal

In addition to the above primary parameters, other parameters mainly pertain to SPI interface read operations and the phase of the SPIsignal; please refer to the relevant documentation for details within the register table.

Similar to the DBI interface, the SPI interface also supports configurable pathways. After selecting the SPI interface, similar to DBI, the user can trigger read and write operations via the WR_TRIG and RD_TRIG registers. For SPI, the user also needs to configure WR_LEN and RD_LEN, which determine the number of bytes per single read/write operation on the configured SPI pathway. A value of 0 represents 1 byte, up to a maximum of 4 bytes.

JDI Serial/Parallel Interface:

The JDI Serial and JDI Parallel interfaces are dedicated interfaces for JDI reflective displays. The JDI Serial interface is a serial interface supporting relatively lower resolution screens, JDI Parallel As a parallel interface, it supports displays with higher resolution. JDI SerialThe interface configuration is relatively simple, with the following parameters:

Parameter Name	Description	configuration
CLK_DIV	JJDI 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 transmits a command sequence prior to sending 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 WR_LEN register of the JDI Serial to define the bit width of a single transmission, then send data via WR_TRIG.

The JDI Parallel interface supports relatively higher resolution displays, providing richer color reproduction. Its configuration parameters are more comprehensive; please refer to the JDI Parallel interface protocol to configure the corresponding parameters. The configuration parameters are as follows:

parameter	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	umber 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 signa	Based on system clock cycles
HST_DLY	Delay from VCK to HSTsignal	Based on system clock cycles
HCK_DLY	Delay from HST to 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 transmits it to the display. The JDI Parallel interface will only stop after the current frame data transmission is completed once the enable 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**	
[31:23]			RSVD	
[22]	rw1c	1'h0	LINE_DONE_RAW_STAT	raw_status of line process done interrupt
[21]	rw1c	1'h0	JDI_PAR_UDR_RAW_STAT	raw_status of jdi parallel interface under run interrupt
[20]	rw1c	1'h0	JDI_PARL_INTR_RAW_STAT	raw_status of jdi parallel interface line interrupt
[19]	rw1c	1'h0	DPI_UDR_RAW_STAT	raw status of dpi under run interrupt
[18]	rw1c	1'h0	DPIL_INTR_RAW_STAT	raw status of dpi line interrupt
[17]	rw1c	1'h0	ICB_OF_RAW_STAT	raw status of icb overflow interrupt
[16]	rw1c	1'h0	EOF_RAW_STAT	raw status of end of frame interrupt
[15:7]			RSVD	
[6]	rw1c	1'h0	LINE_DONE_STAT	line process done interrupt, masked by mask register
[5]	rw1c	1'h0	JDI_PAR_UDR_STAT	jdi parallel interface under run interrupt, masked by mask register
[4]	rw1c	1'h0	JDI_PARL_INTR_STAT	jdi parallel interface line interrupt, masked by mask register
[3]	rw1c	1'h0	DPI_UDR_STAT	dpi under run interrupt, masked by mask register
[2]	rw1c	1'h0	DPIL_INTR_STAT	dpi line interrupt, masked by mask register
[1]	rw1c	1'h0	ICB_OF_STAT	icb overflow interrupt, masked by mask register
[0]	rw1c	1'h0	EOF_STAT	end of frame interrupt, masked by mask register
0x0C			**SETTING**	
[31:27]			RSVD	
[26:16]	rw	11'h1ff	LINE_DONE_NUM	line number of line process done interrupt
[15:9]			RSVD	
[8]	rw	1'h1	AUTO_GATE_EN	auto clock gating enable
[7]			RSVD	
[6]	rw	1'h0	LINE_DONE_MASK	line process done interrupt, 0: mask the interrupt
[5]	rw	1'h0	JDI_PAR_UDR_MASK	jdi parallel interface under run interrupt mask, 0: mask the interrupt
[4]	rw	1'h0	JDI_PARL_INTR_MASK	jdi parallel interface line interrupt, 0: mask the interrupt
[3]	rw	1'h0	DPI_UDR_MASK	dpi under run interrupt mask, 0: mask the interrupt
[2]	rw	1'h0	DPIL_INTR_MASK	dpi line interrupt, 0: mask the interrupt
[1]	rw	1'h0	ICB_OF_MASK	icb overflow interrupt mask, 0: mask the interrupt
[0]	rw	1'h0	EOF_MASK	end of frame interrupt mask, 0: mask the interrupt
0x10			**CANVAS_TL_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	
0x14			**CANVAS_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	
0x18			**CANVAS_BG**	
[31:28]			RSVD	

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[27]	rw	1'h0	H_MIRROR	set 1 to do horizontal mirror for output image
[26]	rw	1'h0	LB_BYPASS	line buffer bypass. Set 1 to bypass line buffer.
[25]	rw	1'h0	ALL_BLENDING_BYPASS	if this bit is set, lcdc is in pure dma mode. No blending operation.
[24]	rw	1'h0	BG_BLENDING_BYPASS	if this bit is set, the layer is not blending with background. The alpha value will be reserved to output.
[23:16]	rw	8'h0	RED	Red color
[15:8]	rw	8'h0	GREEN	green color
[7:0]	rw	8'h0	BLUE	blue color
0x1C			**LAYER0_CONFIG**	
[31]			RSVD	
[30]	rw	1'h0	V_MIRROR	set 1 to do vertical mirror for the layer
[29]	rw	1'h0	ALPHA_BLEND	set 1 to enable alpha blending mode. Use layer alpha as blending factor for image with Alpha. Alpha_out = Layer_alpha * Image_alpha
[28]	rw	1'h0	ACTIVE	layer active flag
[27]	rw	1'h0	LINE_FETCH_MODE	line fetch mode 0: address skip every single line 1: address skip every two line
[26]	rw	1'h0	PREFETCH_EN	preload 64 bytes extra data when reading pixel from memory
[25:13]	rw	13'h0	WIDTH	source image width(including padding), unit is bytes
[12]	rw	1'h0	FILTER_EN	layer color filter enable
[11:4]	rw	8'h0	ALPHA	layer alpha value
[3]	rw	1'h0	ALPHA_SEL	alpha selection 1'b0: select alpha according to image format 1'b1: select layer alpha
[2:0]	rw	3'h0	FORMAT	overlay layer input format 3'h0: RGB565 3'h1: RGB888 3'h2: ARGB8888 3'h3: ARGB8565 3'h4: RGB332 3'h5: A8 3'h6: L8 others: reserved
0x20			**LAYER0_TL_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	Coordinate X-value
0x24			**LAYER0_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	Coordinate X-value
0x28			**LAYER0_FILTER**	
[31:24]	rw	8'h0	FILTER_MASK	layer color filter mask
[23:16]	rw	8'h0	FILTER_R	filter r color
[15:8]	rw	8'h0	FILTER_G	filter g color
[7:0]	rw	8'h0	FILTER_B	filter b color
0x2C			**LAYER0_SRC**	
[31:0]	rw	32'h0	ADDR	source image RGB data address[31:0]. For RGB565 format, address should be aligned to halfword. For ARGB8888 format, address should be aligned to word.

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x30			**LAYER0_FILL**	
[31:26]			RSVD	
[25]	rw	1'h0	ENDIAN	input 565 data format endian 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]
[24]	rw	1'h0	BG_MODE	not used
[23:16]	rw	8'h0	BG_R	background r color
[15:8]	rw	8'h0	BG_G	background g color
[7:0]	rw	8'h0	BG_B	background b color
0x34			**LAYER0_DECOMP**	
[31:24]			RSVD	
[23:13]	rw	11'h0	col_size	number of colums in a line of original image, max column size is 1024
[12:1]	rw	12'h0	target_words	size of a single channel data before decompression. Unit is half word. Each line has 3 channels. So for each line, the compressed data size is target_words * 3 * 2 bytes.
[0]	rw	1'h0	enable	decompression enable
0x38			**LAYER0_DECOMP_CFG0**	
[31:20]	rw	12'h10	cfg0_reserved	
[19:16]	rw	4'h5	lossless_qidx2	condition to decrease qidx
[15:12]	rw	4'h5	lossless_qidx1	up level for adjusted qidx value for low quality block
[11:8]	rw	4'h9	use_lossless_qidx	condition to increase qidx
[7:4]	rw	4'h8	extra_threshold	the threshold to distinguish high/low quality block
[3:0]	rw	4'h2	extra_high	extra bit for high quality bit
0x3C			**LAYER0_DECOMP_CFG1**	
[31:28]	rw	4'h8	extra_low	extra bit for low quality block
[27:24]	rw	4'h0	block_min_qidx	minimum qidx for block mode
[23:20]	rw	4'h0	line_min_qidx	minimum qidx for line mode
[19:16]	rw	4'h2	failover_bits_b	failover compression mode target bits(Blue)
[15:12]	rw	4'h3	failover_bits_g	failover compression mode target bits(Green)
[11:8]	rw	4'h3	failover_bits_r	failover compression mode target bits(Red)
[7:2]	rw	6'h1	cfg1_reserved	
[1]	rw	1'b1	dither	dithering function 0: off 1: on
[0]	rw	1'b1	block_width	block_size in pixel unit. 0: 16 pixels 1: 32 pixels Small block size will cause more blocks and more bits to store block information.
0x40			**LAYER0_DECOMP_STAT**	
[31:7]			RSVD	
[6:0]	r	7'h0	buf_max_depth	buf max usage
0x60			**LAYER1_CONFIG**	
[31]			RSVD	
[30]	rw	1'h0	V_MIRROR	set 1 to do vertical mirror for the layer
[29]	rw	1'h0	ALPHA_BLEND	set 1 to enable alpha blending mode. Use layer alpha as blending factor for image with Alpha. Alpha_out = Layer_alpha * Image_alpha
[28]	rw	1'h0	ACTIVE	layer active flag
[27]	rw	1'h0	LINE_FETCH_MODE	line fetch mode 0: address skip every single line 1: address skip every two line
[26]	rw	1'h0	PREFETCH_EN	preload 64 bytes extra data when reading pixel from memory

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[25:13]	rw	13'h0	WIDTH	source image width(including padding), unit is bytes
[12]	rw	1'h0	FILTER_EN	layer color filter enable
[11:4]	rw	8'h0	ALPHA	layer alpha value
[3]	rw	1'h0	ALPHA_SEL	alpha selection 1'b0: select alpha according to image format 1'b1: select layer alpha
[2:0]	rw	3'h0	FORMAT	overlay layer input format 3'h0: RGB565 3'h1: RGB888 3'h2: ARGB8888 3'h3: ARGB8565 3'h4: RGB332 3'h5: A8 3'h6: L8 others: reserved
0x64			**LAYER1_TL_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y0	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X0	Coordinate X-value
0x68			**LAYER1_BR_POS**	
[31:27]			RSVD	
[26:16]	rw	11'h0	Y1	Coordingate Y-value
[15:11]			RSVD	
[10:0]	rw	11'h0	X1	Coordinate X-value
0x6C			**LAYER1_FILTER**	
[31:24]	rw	8'h0	FILTER_MASK	layer color filter mask
[23:16]	rw	8'h0	FILTER_R	filter r color
[15:8]	rw	8'h0	FILTER_G	filter g color
[7:0]	rw	8'h0	FILTER_B	filter b color
0x70			**LAYER1_SRC**	
[31:0]	rw	32'h0	ADDR	source image RGB data address[31:0]. For RGB565 format, address should be aligned to halfword. For ARGB8888 format, address should be aligned to word.
0x74			**LAYER1_FILL**	
[31:26]			RSVD	
[25]	rw	1'h0	ENDIAN	input 565 data format endian 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]
[24]	rw	1'h0	BG_MODE	not used
[23:16]	rw	8'h0	BG_R	background r color
[15:8]	rw	8'h0	BG_G	background g color
[7:0]	rw	8'h0	BG_B	background b color
0x78			**DITHER_CONF**	
[31:13]			RSVD	
[12]	w1t	1'h0	lfsr_load	load lfsr init value
[11:10]	rw	2'h0	lfsr_load_sel	select lfsr 0: none 1: red 2: green 3: blue
[9:7]	rw	3'h0	w_r	red dither width
[6:4]	rw	3'h0	w_g	green dither width

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:1]	rw	3'h0	w_b	blue dither width
[0]	rw	1'h0	en	dither enable
0x7C			**DITHER_LFSR**	
[31:0]	rw	32'h0	init_val	lfsr init load value
0x80			**LCD_CONF**	
[31:19]			RSVD	
[18]	rw	1'h0	ENDIAN	LCD 565 data format endian, this bit would affect SPI, DPI, DBI and AHB interface 565 format 0: R[4:0], G[5:3], G[2:0], B[4:0] 1: G[2:0], R[4:0], B[4:0], G[5:3]
[17]	rw	1'h0	DIRECT_INTF_EN	when the target LCD is AHB LCD, this bit enable the direct interface to DSI module. Direct interface has higher bandwidth and speed than AHB interface.
[16:15]	rw	2'h0	JDI_SER_FORMAT	JDI serial format 2'b00: 3-bit mode 2'b01: 4-bit mode 2'b10: 1-bit mode 2'b11: reserved
[14:12]	rw	3'h0	DPI_LCD_FORMAT	DPI LCD format 3'b000: 16-bit conf1 3'b001: 16-bit conf2 3'b010: 16-bit conf3 3'b011: 18-bit conf1 3'b100: 18-bit conf2 3'b101: 24-bit others: Reserved
[11:10]	rw	2'h0	SPI_LCD_FORMAT	SPI LCD format 2'b00: 8-bit RGB 3:3:2 2'b01: 16-bit RGB 5:6:5 2'b10: 24-bit RGB 8:8:8 2'b11: Reserved
[9:8]	rw	2'h0	AHB_FORMAT	AHB LCD/RAM output format: 0: RGB565 1: RGB888 2: ARGB8888 3: RGB332
[7:5]	rw	3'h0	LCD_FORMAT	LCD output format: 3'b000: 8-bit RGB 3:3:2 3'b001: 16-bit RGB 5:6:5 over 8-bit bus, 2 cycles/pixel 3'b010: 12-bit RGB 4:4:4 3'b011: 16-bit RGB 5:6:5 3'b100: 18-bit RGB 6:6:6 3'b101: 24-bit RGB 8:8:8 3'b110: 24-bit RGB 8:8:8 over 16-bit bus, 1.5 cycles/pixel 3'b111: 24-bit RGB 8:8:8 over 8-bit bus, 3cycles/pixel others: Reserved
[4:2]	rw	3'h0	LCD_INTF_SEL	3'b000: 8080 DBI Type B 3'b001: SPI interface 3'b010: DBI to DSI interface 3'b011: DPI interface 3'b100: JDI serial interface 3'b101: JDI parallel interface 3'b110: 8080 DBI Type A 3'b111: DPI to DSI interface

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1:0]	rw	2'h0	TARGET_LCD	The Data can be sent to four destinations: 2'b00: LCD panel 0 2'b01: LCD panel 1 2'b10: AHB LCD 2'b11: AHB RAM
0x84			**LCD_IF_CONF**	
[31:26]			RSVD	
[25]	rw	1'h0	CTRL_DLY_SET	if this bit is set to 1, LCD control output will be delayed for 1 lcdc clock cycle
[24]	rw	1'h0	DO_DLY_SET	if this bit is set to 1, LCD data output will be delayed for 1 lcdc clock cycle
[23]	rw	1'h0	LCD_RSTB	LCD RSTB pin, direct to output
[22]	rw	1'h0	RD_POL	LCD RD pin polarity. RD is 0 for write operation, 1 for idle if polarity bit is set as 0. RD bit definition is opposite if polarity bit is set as 1.
[21]	rw	1'h0	WR_POL	LCD WR pin polarity. WR is 0 for write operation, 1 for idle if polarity bit is set as 0. WR bit definition is opposite if polarity bit is set as 1.
[20]	rw	1'h0	RS_POL	LCD RS pin polarity. RS is 1 for data access, 0 for command access if polarity bit is set as 0. RS bit definition is opposite if polarity bit is set as 1.
[19]	rw	1'h0	CS1_POL	LCD0 CS pin polarity. CS is 0 for LCD chip select if polarity bit is set as 0. CS bit definition is opposite if polarity bit is set as 1.
[18]	rw	1'h0	CS0_POL	LCD1 CS pin polarity. CS is 0 for LCD chip select if polarity bit is set as 0. CS bit definition is opposite if polarity bit is set as 1.
[17:12]	rw	6'h0	PWH	inactive cycles of LCD_WR/LCD_RD for consecutive write/read operation
[11:6]	rw	6'h0	PWL	active cycles of LCD_WR/LCD_RD
[5:3]	rw	3'h0	TAH	hold cycles, delay from LCD_WR/LCD_RD inactive to LCD_CS inactive
[2:0]	rw	3'h0	TAS	setup cycles, delay from LCD_CS active to LCD_WR/LCD_RD active
0x88			**LCD_MEM**	
[31:0]	rw	32'h0	ADDR	address for AHB LCD/AHB RAM
0x8C			**LCD_O_WIDTH**	
[31:16]			RSVD	
[15:0]	rw	16'h0	OFFSET	AHB RAM address offset for each line
0x90			**LCD_SINGLE**	
[31:4]			RSVD	
[3]	r	1'h0	LCD_BUSY	LCD/SPI LCD interface is busy for single access
[2]	w1t	1'h0	RD_TRIG	Single read operation trigger
[1]	w1t	1'h0	WR_TRIG	Single write operation trigger
[0]	rw	1'h0	TYPE	LCD access type, this bit could affect all LCD interface including SPI, parellel and AHB 1'b0: command 1'b1: data
0x94			**LCD_WR**	
[31:0]	rw	32'h0	DATA	LCD write data
0x98			**LCD_RD**	
[31:0]	r	32'h0	DATA	LCD read data
0x9C			**SPI_IF_CONF**	
[31]			RSVD	
[30]	rw	1'h0	SPI_CLK_INIT	SPI CLK idle state value 1'h0: high 1'h1: low
[29]	rw	1'h0	SPI_CLK_POL	SPI CLK polarity 1'h0: normal 1'h1: inverted

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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)
[19:17]	rw	3'h0	LINE	SPI line mode 0: 4-line 1: 4-line with 2 data line(support RGB565 and RGB888) 2: 4-line with 4 data line(support RGB565 and RGB888) 3: reserved 4: 3-line 5: 3-line with 2 data line(support RGB565 and RGB888) 6: 3-line with 4 data line(support RGB565 and RGB888) 7: reserved
[16:14]	rw	3'h0	DUMMY_CYCLE	SPI transaction dummy cycle
[13:6]	rw	8'ha	CLK_DIV	SPI clock divider
[5:0]	rw	6'h0	WAIT_CYCLE	SPI line wait cycle, wait cycle is after each line and is according to SPI clock. 0 refers to no wait cycle.
0xA0			**TE_CONF**	
[31:21]			RSVD	
[20]	rw	1'h0	FMARK_SOURCE	TE signal source 1: use TE signal from DSI 0: use TE signal from external pin
[19]	rw	1'h0	FMARK_MODE	TE signal trigger mode 1: edge trigger 0: pulse trigger
[18:3]	rw	16'h0	VSYNC_DET_CNT	vsync signal detect counter, used for mode 1 to detect vsync signal
[2]	rw	1'h0	MODE	0: vsync only TE mode 1: vsync+hsync TE mode
[1]	rw	1'h0	FMARK_POL	TE signal polarity
[0]	rw	1'h0	ENABLE	TE enable
0xA4			**TE_CONF2**	
[31:0]	rw	32'h0	DLY_CNT	TE delay counter
0xA8			**DPI_IF_CONF1**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HSW	dpi hsync width
[15:11]			RSVD	
[10:0]	rw	11'h0	VSH	dpi vsync height
0xAC			**DPI_IF_CONF2**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HBP	horizontal back porch
[15:11]			RSVD	
[10:0]	rw	11'h0	VBP	vertical back porch
0xB0			**DPI_IF_CONF3**	

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:27]			RSVD	
[26:16]	rw	11'h0	HFP	horizontal front porch
[15:11]			RSVD	
[10:0]	rw	11'h0	VFP	vertical front porch
0xB4			**DPI_IF_CONF4**	
[31:27]			RSVD	
[26:16]	rw	11'h0	HAW	horizontal active width
[15:11]			RSVD	
[10:0]	rw	11'h0	VAH	vertical active height
0xB8			**DPI_IF_CONF5**	
[31:24]			RSVD	
[23]	rw	1'h0	clk_force_on	1: force DPI clock on 0: DPI clock is controlled by hardware
[22:12]	rw	11'h7ff	INT_LINE_NUM	DPI interrupt line number
[11]	rw	1'h0	HSPOL	hsync polarity
[10]	rw	1'h0	VSPOL	vsync polarity
[9]	rw	1'h0	DEPOL	de polarity
[8]	rw	1'h0	PCLKPOL	pixel clock polarity
[7:0]	rw	8'h1	PCLK_DIV	pixel clock divider
0xBC			**DPI_CTRL**	
[31:4]			RSVD	
[3]	rw	1'h0	DPI_UC	dpi update config
[2]	rw	1'h0	DPI_SD	dpi shutdown
[1]	rw	1'h0	DPI_CM	dpi color mode
[0]	rw	1'h0	DPI_EN	dpi interface enable
0xC0			**DPI_STAT**	
[31:16]	r	16'h1	VPOS	dpi vertical position
[15:14]			RSVD	
[13:11]	r	3'h0	HSTAT	horizontal status 0: idle 1: prep 2: hsync 3: hbp 4: hact 5: hfp 6: wait
[10:0]	r	11'h1	HPOS	dpi horizontal position
0xC4			**JDI_SER_CONF1**	
[31:16]			RSVD	
[15:8]	rw	8'h2	CLK_DIV	jdi serial clock divider
[7:5]			RSVD	
[4:0]	rw	5'd1	WR_LEN	jdi single write bit length
0xC8			**JDI_SER_CONF2**	
[31:16]	rw	16'h0	INIT_LINE_CNT	jdi serial init line counter
[15:0]	rw	16'h0	WR_CMD	jdi serial data transfer write command
0xCC			**JDI_SER_CTRL**	
[31:2]			RSVD	
[1]	rw	1'h0	EXTCOMIN	jdi serial interface extcomin control
[0]	rw	1'h0	DISP	jdi serial interface disp control
0xD0			**JDI_PAR_CONF1**	
[31:16]	rw	16'h0	MAX_LINE	jdi parallel interface max line, line number start from 0
[15:0]	rw	16'h0	MAX_COL	jdi parallel interface max column, column number start from 0

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xD4			**JDI_PAR_CONF2**	
[31:16]	rw	16'h0	ST_LINE	jdi parallel interface start line, line number start from 0
[15:0]	rw	16'h0	END_LINE	jdi parallel interface end line, line number start from 0
0xD8			**JDI_PAR_CONF3**	
[31:16]	rw	16'h0	ST_COL	jdi parallel interface start column, column number start from 0
[15:0]	rw	16'h0	END_COL	jdi parallel interface end column, column number start from 0
0xDC			**JDI_PAR_CONF4**	
[31:16]	rw	16'h0	HCK_WIDTH	jdi parallel interface HCK width, HSK width = lcd_ck_cycle * HCK_WIDTH
[15:0]	rw	16'h0	HST_WIDTH	jdi parallel interface HST width, HST width = lcd_ck_cycle * HST_WIDTH
0xE0			**JDI_PAR_CONF5**	
[31:16]	rw	16'h0	VCK_WIDTH	jdi parallel interface VCK width, VCK width = lcd_ck_cycle * VCK_WIDTH
[15:0]	rw	16'h0	VST_WIDTH	jdi parallel interface VST width, VST width = lcd_ck_cycle * VST_WIDTH
0xE4			**JDI_PAR_CONF6**	
[31:16]	rw	16'h0	VCK_DLY	jdi parallel interface VST to VCK delay, VST2VCK delay = lcd_ck_cycle * VCK_DLY
[15:0]	rw	16'h0	HST_DLY	jdi parallel interface VCK to HST delay, VCK2HST delay = lcd_ck_cycle * HST_DLY
0xE8			**JDI_PAR_CONF7**	
[31:17]			RSVD	
[16]	rw	1'h0	DP_MODE	double pixel mode. Some jdi parallel screens use large pixel+small pixel structure. Set this bit to 1 to support this structure.
[15:0]	rw	16'h0	HCK_DLY	jdi parallel interface HST to HCK delay
0xEC			**JDI_PAR_CTRL**	
[31:16]	rw	16'hffff	INT_LINE_NUM	jdi parallel interface interrupt line number, line number start from 0.
[15:10]			RSVD	
[9]	rw	1'h0	VSTPOL	jdi parallel vst polarity
[8]	rw	1'h0	VCKPOL	jdi parallel vck polarity
[7]	rw	1'h0	HSTPOL	jdi parallel hst polarity
[6]	rw	1'h0	HCKPOL	jdi parallel hck polarity
[5]	rw	1'h0	ENBPOL	jdi parallel enb polarity
[4]	rw	1'h0	XRST	jdi parallel interface XRST
[3:1]			RSVD	
[0]	rw	1'h0	ENABLE	jdi parallel interface enable
0xF0			**JDI_PAR_STAT**	
[31:16]	r	16'h0	VPOS	jdi parallel vertical position
[15:0]	r	16'h0	HPOS	jdi parallel horizontal position
0xF4			**JDI_PAR_EX_CTRL**	
[31]	r	1'h0	VCOM	VCOM value
[30]	r	1'h0	FRP	FRP value
[29]	r	1'h0	XFRP	XFRP value
[28]	rw	1'h0	CNT_EN	VCOM/FRP/XFRP counter enable
[27:24]			RSVD	
[23:0]	rw	24'h0	MAX_CNT	VCOM/FRP/XFRP max counter
0xF8			**JDI_PAR_CONF8**	
[31:16]	rw	16'h0	ENB_ST_COL	jdi parallel interface enb start column, column number start from 0
[15:0]	rw	16'h0	ENB_END_COL	jdi parallel interface enb end column, column number start from 0
0xFC			**JDI_PAR_CONF9**	
[31:16]	rw	16'h0	ENB_ST_LINE	jdi parallel interface enb start line, line number start from 0
[15:0]	rw	16'h0	ENB_END_LINE	jdi parallel interface enb end line, line number start from 0
0x100			**JDI_PAR_CONF10**	
[31:16]	rw	16'h0	HC_ST_LINE	jdi parallel interface horizontal control start line, line number start from 0
[15:0]	rw	16'h0	HC_END_LINE	jdi parallel interface horizontal control end line, line number start from 0
0x110			**CANVAS_STAT0**	
[31:27]			RSVD	

Continued on next page...

Table 10-1: LRegister Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[26:16]	r	11'h0	y_cor	canvas y cordinate
[15:11]			RSVD	
[10:0]	r	11'h0	x_cor	canvas x cordinate
0x114			**CANVAS_STAT1**	
[31:12]			RSVD	
[11:9]	r	3'h0	fetch_stat	fetch status
[8:6]	r	3'h0	prec_stat	prec status
[5:3]	r	3'h0	postc_stat	postc_status
[2:0]	r	3'h0	fifo_cnt	pre calc fifo count
0x118			**OL0_STAT**	
[31:24]			RSVD	
[23:22]	r	2'h0	sc_lb0	
[21:20]	r	2'h0	sc_lb1	
[19:16]	r	4'h0	sc_fe	
[15:13]	r	3'h0	sc_be	
[12:11]	r	2'h0	sc_out	
[10:8]	r	3'h0	pf_pr	
[7:6]	r	2'h0	pf_df	
[5:4]	r	2'h0	data_conv	
[3:2]	r	2'h0	prefetch_read	
[1]	r	1'h0	prefetch_out	
[0]	r	1'h0	done_req	
0x11C			**OL1_STAT**	
[31:11]			RSVD	
[10:8]	r	3'h0	pf_pr	
[7:6]	r	2'h0	pf_df	
[5:4]	r	2'h0	data_conv	
[3:2]	r	2'h0	prefetch_read	
[1]	r	1'h0	prefetch_out	
[0]	r	1'h0	done_req	
0x120			**MEM_IF_STAT**	
[31:10]			RSVD	
[9:7]	r	3'h0	arb_main	
[6:4]	r	3'h0	arb_read_port	
[3:0]	r	4'h0	ahb	
0x124			**PERF_CNT**	
[31:0]	rw	32'h0	VAL	lcdc performance counter

10.3 eZip™ Lossless Compression Decoder

eZip is located in HPSYS.

The eZip ™ decoder is a real-time lossless decompression module based on proprietary algorithms, with compression ratios compara ble 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 reduce transmission time and power consumption.

Additionally,eZip ™ supports proprietary format image compression with compression ratios comparable to the PNG format, and supports independent DMA operations or integration with ePicasso ™ for reading. When operating independently, eZip ™ can flexibly decompress and transfer compressed images stored in Flash or RAM to the target buffer

via the DMA mechanism. In integration mode, ePicasso ™ utilizes the eZip ™ module to read images from storage and decompress them in real time, then per forms the required 2.5D computations according to the standard graphics pipeline, thereby eliminating the need for a temporary buffer for decompressed images.

Through the above mechanism,eZip ™ effectively reduces the storage capacity requirements for image assets, maximizes asset richness within lim ited storage, and decreases bandwidth demands on off-chip memory, thereby signifcantly enhancing overall system operational effciency.

The eZip ™ module decodes and outputs 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 be directly sent to the epic module for subsequent processing.

This module features the following characteristics:

- 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 cachefunction; decoding time can be reduced in the event of acachehit

11 Audio

11.1 PDM

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

11.1.1 Introduction

The PDM (Pulse Density Modulation) interface is primarily used to convert PDM audio signals collected from PDM microphones into PCM (Pulse Code Modulation) audio signals for subsequent audio processing.

Main functions:

- Simultaneously supports left and right stereo signals and can also collect mono signals individually.
- Available PDM microphone clock rate:3.072MHz. 、1.536MHz、0.768MHz、1.024MHz、2.4MHz、1.6MHz、0.8MHz Etc.
- Supported PCM data rate: 48kHz. 、32kHz、24kHz、16kHz、12kHz、8kHz Equivalent
- Supports 32 -bit 、24bit、16bit、8-bit PCM Signal
- Supports a resolution of 0.5 dB and adjustable gain from -15 dB to 45 dB.

11.1.2 Usage Instruction

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

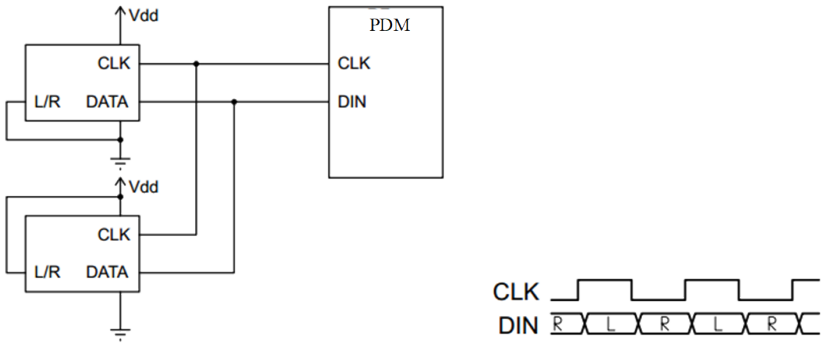


Figure 11-1: Typical connection of a digital microphone through the PDM Module

The above figure illustrates a typical connection of a pair of digital microphones through the PDM Module, where both microphones share the same bit stream clock and data line. Using the microphone configuration pins (L/R) , one microphone can provide valid data on the rising edge of CLK while the other microphone provides valid data on the falling edge of CLK.

11.1.2.1 Overall Structure of the PDM Module

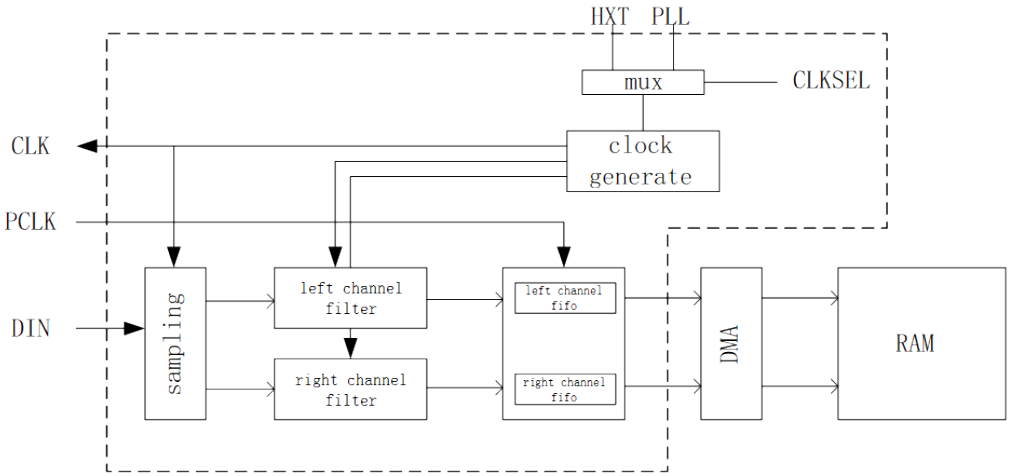


Figure 11-2: Overall Structure of the PDM Module

11.1.2.2 Clock Structure of the PDM Module

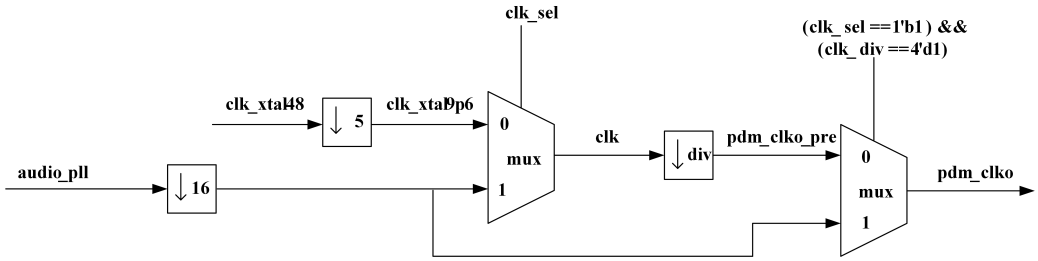


Figure 11-3: Clock Structure of the PDM Module

The clock source of the PDM module consists of two components: one is the system's 48mHz crystal oscillator, and the other is the system's audio PLL after a 16 fold frequency division. The 48mHz crystal oscillator first passes through a 5 fold divider within the PDM module to generate a 9.6mHz clock. This clock is selected between the audio PLL clock and then goes through a configurable divider to produce the fnal clock sent to the digital microphone, referred to as pdm_clko . The final output data rate of the PDM module is pdm_clko/(sinc_rate×lpf_downsample) . The parameters sinc_rate and lpf_downsample are configured according to the following table for the registers sinc_rate and lpf_ds to obtain the final data.

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

PDM_CLK(MHz)	Fs PCM (Output Rate, KHz)	OSR (Over Sampling Rate)	SINC RATE (CIC Down Sampling Rate)	LPF Post Down Sampling Rate	SINC ORDER
3.072	48	64	32	2	3
3.072	32	96	48	2	3
3.072	24	128	64	2	3
3.072	16	192	96	2	3
3.072	12	256	64	4	3
3.072	8	384	96	4	3
1.536	48	32	16	2	4
1.536	32	48	24	2	4

Continued on the next page...

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

PDM_CLK(MHz)	Fs PCM (Output Rate, KHz)	OSR (Over Sampling Rate)	SINC RATE (CIC Down Sampling Rate)	LPF Post Down Sampling Rate	SINC ORDER
1.536	24	64	32	2	3
1.536	16	96	48	2	3
1.536	12	128	64	2	3
1.536	8	192	96	2	3
0.768	24	32	16	2	4
0.768	16	48	24	2	4
0.768	12	64	32	2	3
0.768	8	96	24	4	4
1.024	32	32	16	2	4
1.024	16	64	32	2	3
1.024	8	128	64	2	3
2.4	48	50	25	2	4
2.4	24	100	50	2	3
2.4	16	150	75	2	3
2.4	12	200	100	2	3
2.4	8	300	75	4	3
1.6	32	50	25	2	4
1.6	16	100	50	2	3
1.6	8	200	100	2	3
0.8	16	50	25	2	4
0.8	8	100	50	2	3
2.4	32	75	75	LPF bypass	3
1.2	48	25	25	LPF bypass	4
1.2	24	50	25	2	4
1.2	16	75	75	LPF bypass	3
1.2	12	100	50	2	3
1.2	8	150	75	2	
2.8224	44.1	64	32	2	3
2.8224	22.05	128	64	2	3
2.8224	11.025	256	64	4	3
1.4112	44.14	32	16	2	4
1.4112	22.05	64	32	2	3
1.4112	11.025	128	64	2	3
0.7056	22.05	32	16	2	4
0.7056	11.025	64	32	2	3

Using the configuration in the first row of the table as an example, the register configuration for the PDM module is explained. The output clock is 3.072mHz , and the output data rate is 48kHz . The output data bit width is 16 bits, with stereo enabled.

Configuration process for the PDM registers:

1. Select the output rate according to the table. Choose the clock source based on the PDM_CLK in the table, and configure 0x00 in the clk_sel register. 0x0 indicates that the input clock for the PDM module is 9.6mHz , while 0x1 indicates that the input clock for the PDM module is the one-sixteenth frequency clock of the audio PLL. In this case, the configuration of clk_sel is 1 .(The configuration of the audio clock is not discussed here.)
2. According to the table, configure the SINC RATE and SINC ORDER in 0x08 for sinc_rate and sinc_order_sel registers, where sinc_order_sel is 1 corresponds to the table where SINC ORDER is 4 , and 0 corresponds to 3 . Based on the LPF post-downsampling rate, configure 0x34 for lpf_ds register and lpf_bypass register. Configuring lpf_ds register to 1 indicates a downsampling of 4 in excel for lpf , while configuring lpf_ds register to 0 indicates a downsampling

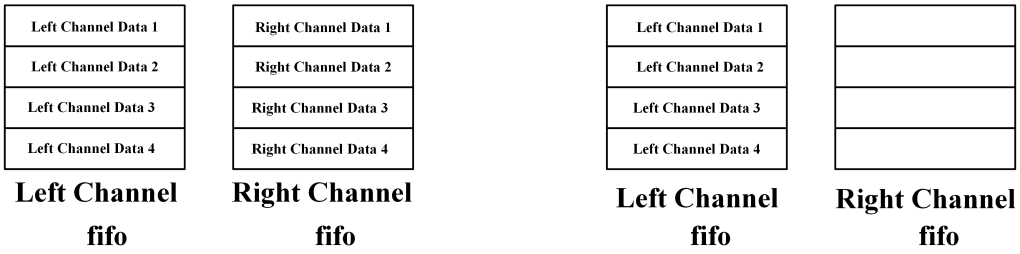
of 2 in excel for lpf . Configuring lpf_bypass to 1 corresponds to the LPF BYPASS in the table. In this example, 0x08 sinc_rate=64 , sinc_order_sel=0 , 0x34 lpf_ds=0

3. Enable the corresponding signal based on the channel configuration:

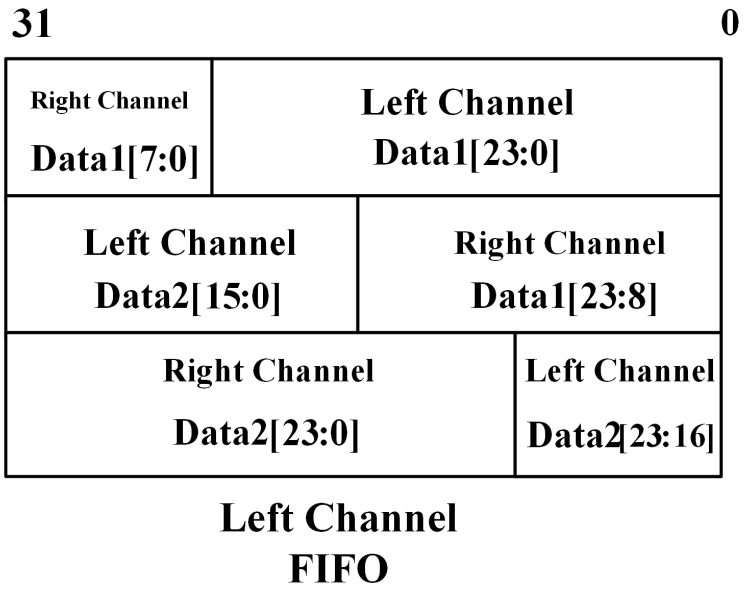
		Dual Channel		Left Channel	Right Channel
Register 0x0 CFG0	left_en	1	0	1	0
	right_en	1	0	0	1
	stereo_en	0	1	0	0

When the swap_en register at 0x0 is set to 0, the PDM module interprets the right channel as collecting data from the rising edge output of the PDM mic, while the left channel collects data from the falling edge output of the PDM mic. Conversely, when the swap_en register at 0x0 is set to 1, the right channel is interpreted as collecting data from the falling edge output of the PDM mic, and the left channel collects data from the rising edge output of the PDM mic. Configure according to requirements. In this example, 0x0 stereo_en=1 left_en=0 right_en=0 or stereo_en=0 left_en=1 right_en=1 .

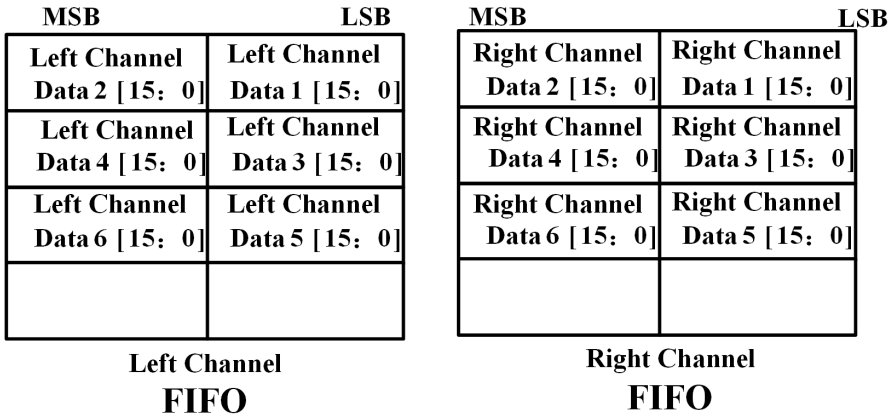
4. Configure according to the output data format of the pdm module. The different bit widths of the output data configure the 0x38 byte_trunc register. Set 0 for a 24-bit output, set 1 for a 16-bit output, set 2 for an 8-bit output, and set 3 for a 32-bit output. In this example, the 0x38 byte_trunc is 1 .
5. Configure the 0x38 byte_con register based on whether the left and right channel data needs to be mixed together. Set 0 to store the data of the left and right channels separately in their respective ffo , while set 1 will store the data of both channels in the left channel's fifo . A schematic diagram illustrates the differences in data storage corresponding to different byte_con register configurations.



If the data is less than 32 bits, the lower bits of the next data will automatically fill the higher bits of the previous data to form 32 bits of data. For 24 bit stereo data, when byte_con is set to 1, it is illustrated in the fgure below.



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



In this example, there are no specific requirements for the data storage format; users may select the configuration based on their needs.

6. Configure the DMA registers according to the DMA usage instructions to transfer data from the PDM FIFO, one 32-bit data at a time, to the specifed RAM address.
7. Set the PDMCOREEN register at 0x0 to enable the PDM module.

The order of steps 1 to 6 is not fixed, provided that they are completed before step 7.

11.1.2.3 Precaution

All interrupts generated by the PDM module are triggered by errors caused by fifo overflow.

11.1.3 PDM Register

PDM base address is 0x5009A000。

Table 11-2: PDM Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CFG0**	
[31:10]			RSVD	
[9]	rw	1'b0	swap_en	1: Swap right channel and left channel pdm data; 0: Not swap right channel and left channel pdm data
[8]	rw	1'b0	stereo_en	1:Enable double channels pdm data sampling; 0: Disable double channels pdm data sampling
[7]	rw	1'b0	right_en	1: Enable right channel pdm data sampling; 0: Disable right channel pdm data sampling
[6]	rw	1'b0	left_en	1: Enable left channel pdm data sampling; 0: Disable left channel pdm data sampling
[5:2]	rw	4'h4	clk_div	Clock frequency division ratio of 3.072MHz or 9.6MHz according to register clk_sel
[1]	rw	1'b0	clk_sel	1:Clk select dll 3.072MHz; 0: Clk selct xtal 9.6MHz
[0]	rw	1'b0	pdmcoreen	1:Enable pdm module; 0: Disable pdm module
0x04			**CFG1**	
[31:11]			RSVD	
[10:8]	rw	3'h0	sample_dly_r	The number of delay dff before the right data stream in processing
[7:5]	rw	3'h0	sample_dly_l	The number of delay dff before the left data stream in processing
[4:0]			RSVD	
0x08			**SINC_CFG**	
[31:9]			RSVD	
[8]	rw	1'b1	sinc_order_sel	1:select four differentiators in sinc filter; 0:select three differentiators in sinc filter
[7:0]	rw	8'd32	sinc_rate	dowmsampling rate of sinc filter
0x14			**HPF_CFG**	
[31:6]			RSVD	
[5]	rw	1'b1	hpf_rst	1:high-pass filter normal operation ; 0:reset high-pass filter
[4]	rw	1'b0	hpf_bypass	1:bypass-high pass filter ; 0: enable high-pass filter
[3:0]	rw	4'hd	hpf_coeff	coefficient of high-pass filter
0x18			**PGA_CFG**	
[31:14]			RSVD	
[13:7]	rw	7'd0	pga_gain_r	right channel gain control , the range is -15dB~45dB. Resolution is 0.5dB/LSB
[6:0]	rw	7'd0	pga_gain_l	left channel gain control , the range is -15dB~45dB. Resolution is 0.5dB/LSB
0x34			**LPF_CFG6**	
[31:14]			RSVD	
[13]	rw	1'b0	lpf_bypass	1:bypass low-pass filter ; 0: enable low-pass filter
[12]	rw	1'b0	lpf_ds	1:downsampling rate of low pass filter is two;0:No downsampling of low pass filter
[11:0]			RSVD	
0x38			**FIFO_CFG**	
[31:9]			RSVD	
[8]	rw	1'b0	lr_chg	1:exchange storage location of left and right channel; 0: don't exchange storage location of left and right channel
[7]	rw	1'b0	rx_dma_msk_l	1:disable left channel dma request; 0: enable left channel dma request
[6]	rw	1'b0	rx_dma_msk_r	1:disable right channel dma request; 0: enable right channel dma request
[5:3]	rw	3'h0	pdm_shift	the number of data left shift for higher data accuracy
[2:1]	rw	2'b0	byte_trunc	1: 16bits output ; 0: 24bits output ;2: 8bits output ; 3: 32bits output
[0]	rw	1'b0	byte_con	1: combine left channel and right channel; 0: not combine left channel and right channel
0x44			**FIFO_ST**	
[31:8]			RSVD	
[7]	r	1'h0	full_l	1 indicates left channel fifo is full

Continued on the next page...

Table 11-2: PDM Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[6]	r	1'b0	empty_l	1 indicates left channel fifo is empty
[5]	r	1'b0	almost_full_l	1 indicates left channel fifo is less than two full
[4]	r	1'b0	almost_empty_l	1 indicates left channel fifo is less than two datas left
[3]	r	1'h0	full_r	1 indicates right channel fifo is full
[2]	r	1'b0	empty_r	1 indicates right channel fifo is empty
[1]	r	1'b0	almost_full_r	1 indicates right channel fifo is less than two full
[0]	r	1'b0	almost_empty_r	1 indicates right channel fifo is less than two datas left
0x48			**INT_ST**	
[31:2]			RSVD	
[1]	r	1'b0	overflow_l	1 indicates left channel fifo has already overflowed and as irq at same time
[0]	r	1'b0	overflow_r	1 indicates right channel fifo has already overflowed and as irq at same time
0x4c			**INT_MSK**	
[31:2]			RSVD	
[1]	rw	1'b0	int_mask_l	1:disable left channel irq to system; 0: enable left channel irq to system
[0]	rw	1'b0	int_mask_r	1:disable right channel irq to system; 0: enable right channel irq to system
0x50			**INT_CLR**	
[31:2]			RSVD	
[1]	w1c	1'b0	int_clr_l	clear left channel irq
[0]	w1c	1'b0	int_clr_r	clear right channel irq

11.2 I2S

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

11.2.1 Introduction

The I2S bus (also known as IIS, which stands for Inter-IC Sound) is a standard established by Philips for audio data transmission between digital audio devices. This bus operates in a master/slave mode and is dedicated to data transmission between audio devices, making it widely used in various multimedia systems.

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

The standard I2S mode is illustrated in the fgure below:

Data is transmitted on the second rising edge of BCLK following LRCLK, with the MSB transmitted first, and the subsequent bits following in order until the LSB. The transmission is dependent on the word length, BCLK frequency, and sampling rate (BCLK = Fs x number of channels x number of sampling bits). There should be unused BCLK cycles between the LSB of each sample and the MSB of the next sample. LRCLK is 0 for transmitting left channel data, and LRCLK is 1 for transmitting right channel data.

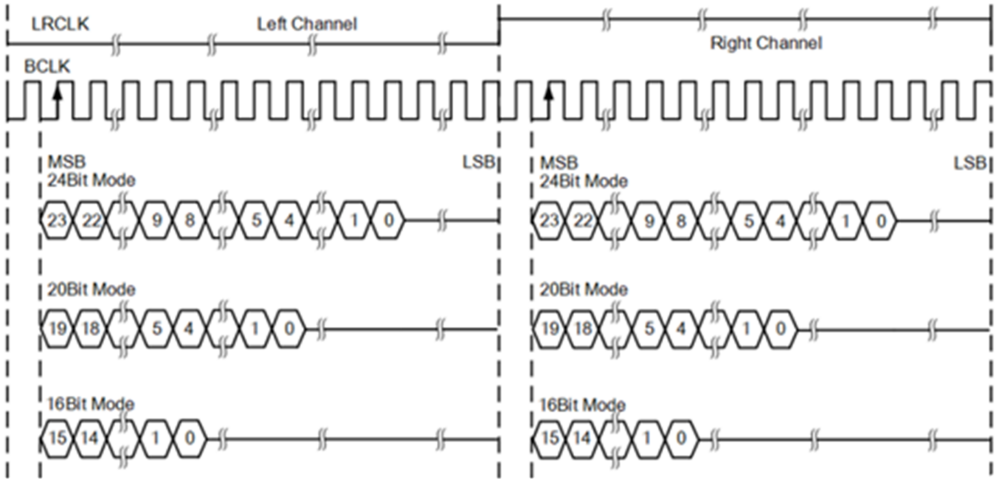


Figure 11-4: Standard I2S Mode

The left-aligned I2S mode is illustrated in the fgure below:

In the standard left-aligned format, the MSB of the data has no clock delay relative to BCLK. The MSB of the left and right channel data in the left-aligned format is valid on the first rising edge of BCLK after the LRCLK edge changes. LRCLK is 1 for transmitting left channel data and 0 for transmitting right channel data.

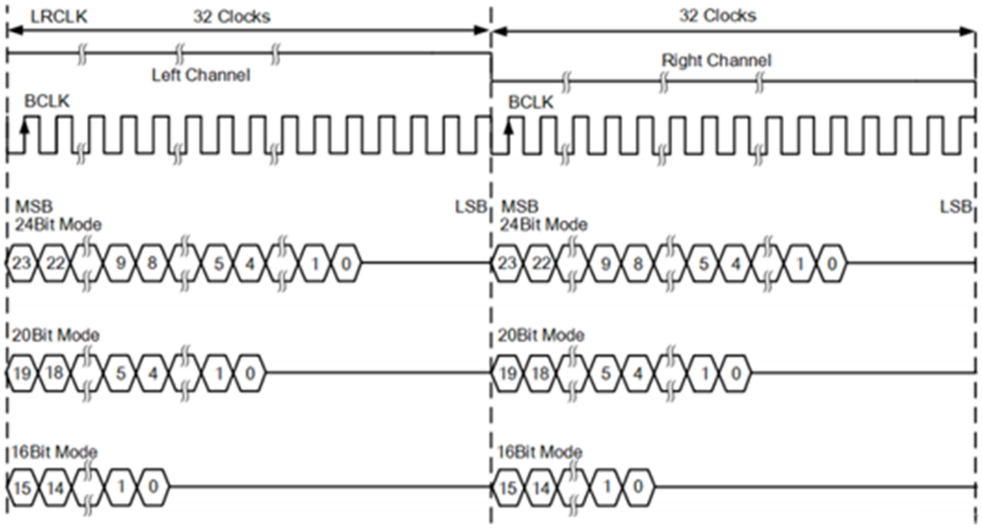


Figure 11-5: I2S Left Alignment

I2S Right alignment is illustrated in the following fgure:

Simultaneously with the completion of the sound data LSB transmission, LRCLK completes its second transition, with LRCLK being 1 for transmitting left channel data and 0 for transmitting right channel data.

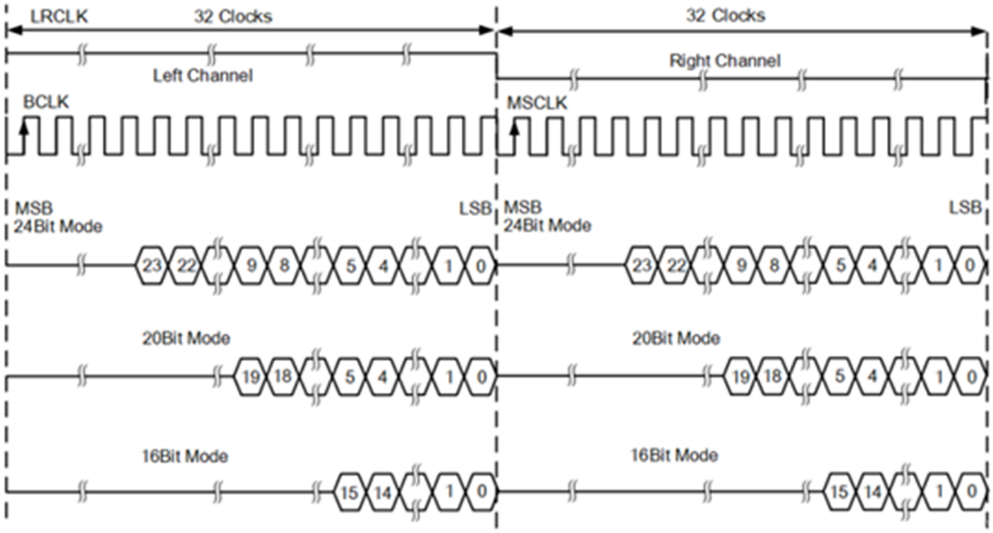


Figure 11-6: I2S Right Alignment

11.2.2 I2S Function Description

The I2S module supports both master and slave modes. In master mode, theMCU provides the sampling clock LRCK and the bit clockBCLK. In slave mode, theBCLK and LRCK are supplied externally, with the MCU responsible for the data transmission and reception of I2S.

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

The current version of the I2S module additionally provides an output for MCLK , which is a higher frequency clock synchronized with BCLK and LRCK , allowing for higher operating frequencies for I2S peripherals. Users can also define the frequency of MCLK according to their requirements. Furthermore, the I2S clock source has added the option of PLL , enhancing the flexibility of the I2S clock.

11.2.3 I2S Register

I2S base address is 0x50009000.

Table 11-3: I2S Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x10			**TX_PCM_FORMAT**	
[31:6]			RSVD	
[5]	rw	1'h0	track_flag	0: stereo 1: mono

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[4:0]	rw	5'h10	dw	tx source pcm data width N(N>=8) common value is 8,13,14,16,18,20,22,24 This data width indicate the tx fifo output data width. When writing to tx fifo, please refer to following format: Mono 8 bit: fifo_data[31:0] = L3,L2,L1,L0, each word contains 4 samples, so four samples need read one word Stereo 8 bit: fifo_data[31:0] = R1,L1,R0,L0 , each word contains 2 samples, so two samples need read one word Mono 13/14/16 bit: fifo_data[31:0] = L1,L0, each word contains 2 samples, so two samples need read one word Stereo 13/14/16 bit: fifo_data[31:0] = R0,L0, each word contains 1 samples, so each sample need read one word Mono 18/20/22/24 bit: fifo_data[31:0] = L0, each word contains 1 samples, so each sample need read one word Stereo 18/20/22/24 bit: fifo_data[31:0][0] = L0, fifo_data[31:0][1]=R0, each 2 words contain 1 samples, so each sample need read two word
0x20			**TX_PCM_SAMPLE_CLK**	
[31:13]			RSVD	
[12:0]	rw	13'd250	fs_duty	source PCM sample clock duty cycle(with GCLK=12MHz): 250 for 48K FS 272 for 44.1K FS 375 for 32K FS 500 for 24K FS 544 for 22.05K FS 750 for 16K FS 1000 for 12K FS 1088 for 11.025K FS 1500 for 8K FS
0x30			**TX_RS_SMOOTH**	
[31:1]			RSVD	
[0]	rw	1'h0	en	0: Disable TX re-sample smooth filter 1: Enable TX re-sample smooth filter This function is not implemented.
0x40			**TX_PCM_CH_SEL**	
[31:4]			RSVD	
[3:2]	rw	2'h0	left_channel_sel	TX re-sampling module setting: 00: TX left = source left 01: TX left = source right 10,11: TX left = (source left + source right)/2
[1:0]	rw	2'h0	right_channel_sel	TX re-sampling module setting: 00: TX right = source right 01: TX right = source left 10,11: TX right = (source left + source right)/2
0x50			**TX_VOL_CTRL**	
[31:4]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[3:0]	rw	4'hf	vol	volume control: 0000: +6dB, 0001: +4.5dB, 0010: +3dB, 0011: +1.5dB, 0100: 0dB, 0101: -1.5dB, 0110: -3.0dB, 0111: -4.5dB, 1000: -6.0dB, 1001: -7.5dB, 1010: -9dB, 1011: -10.5dB, 1100: -12dB, 1101: -13.5dB, 1110: -15dB, 1111: mute Note: 1) +1.5db = 20log(1+1/4-1/16+1/1024) 2) -1.5dB = 20log(1-1/8-1/32-1/512-1/2048)
0x60			**TX_LR_BAL_CTRL**	
[31:6]			RSVD	
[5:4]	rw	2'h0	en	LR balance enable: 00: both left and right in full volume 10: left channel balance volume adjustment enable 01: right channel balance volume adjustment enable 11: reserved, still kepp left and right in full volume
[3:0]	rw	4'h0	bal_vol	Balance volume control: 0000: Reserved, 0001: -1.5dB, 0010: -3.0dB, 0011: -4.5dB, 0100: -6.0dB, 0101: -7.5dB, 0110: -9.0dB, 0111: -10.5dB, 1000: -12dB, 1001: -13.5dB, 1010: -15dB, 1011: -16.5dB, 1100: -18dB, 1101: -19.5dB, 1110: -21dB, 1111: mute Note: 1) bit[5:0] = 101111 for left mute 2) bit[5:0] = 011111 for right mute 3) bit[5:4] = 00 or 11, bit[3:0] is don't care 4) +1.5db = 20log(1+1/4-1/16+1/1024) 5) -1.5dB = 20log(1-1/8-1/32-1/512-1/2048)
0x70			**AUDIO_TX_LRCK_DIV**	
[31:28]			RSVD	
[27:16]	rw	12'd125	duty_high	TX LRCK duty cycle high: 125 for 48K FS 136 for 44.1K FS 185 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS
[15:12]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:0]	rw	12'd125	duty_low	TX LRCK duty cycle low: 125 for 48K FS 136 for 44.1K FS 190 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS Note: 1)duty_cycle = 12M/FS
0x80			**AUDIO_TX_BCLK_DIV**	
[31:6]			RSVD	
[5:0]	rw	6'h5	duty	TX serial bit clock duty cycle 5 for 48K FS 4 for 44.1K FS 5 for 32KFS 10 for 24K FS 8 for 22.05K FS 15 for 16K FS 20 for 12K FS 16 for 11.025K FS 30 for 8KFs
0x90			**AUDIO_TX_FORMAT**	
[31:5]			RSVD	
[4:0]	rw	5'h10	pcm_data_width	I2S out pcm data width M >= 16, common value: 16, 18, 20, 22, 24
0xa0			**AUDIO_SERIAL_TIMING**	
[31:4]			RSVD	
[3]	rw	1'h0	lrck_pol	TX LRCK polarity control. 0: disable TX_LRCK inventor 1: enable TX_LRCK inventor for standard I2S, set tx_lrck_pol to low for Left/Right Justified, set tx_lrck_pol to hgih
[2]	rw	1'h0	slave_en	audio code transmit mode select. 0: master mode, 1: slave mode
[1:0]	rw	2'h0	timing	00: I2S mode 01: Left justified 10: right justified 11: reserved
0xb0			**AUDIO_TX_FUNC_EN**	
[31:2]			RSVD	
[1]	rw	1'h0	tx_intf_sel	1: select external tx interface 0: select internal apb tx interface
[0]	rw	1'h0	tx_en	1: enable 0:disable
0xc0			**AUDIO_TX_PAUSE**	
[31:1]			RSVD	
[0]	rw	1'h0	tx_pause	TX pause control when tx_enable = 1. 1: pause 0: TX work
0xc8			**AUDIO_I2S_SL_MERGE**	
[31:1]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	slave_timing_merge	when work as an I2S slave, and external I2S master TX/RX share an only BCLK/LRCK, we need set this bit high. 0: I2S slave use separated timing control port. TX_BCLK_IN/TX_LRCK_IN and RX_BCLK/RX_LRCK_IN are separated. 1: I2S slave use the same BCLK/LRCK, the TX_BCLK_IN/TX_LRCK also is used for RX controller.
0x100			**AUDIO_RX_FUNC_EN**	
[31:2]			RSVD	
[1]	rw	1'h0	rx_intf_sel	1: select external rx interface 0: select internal apb rx interface
[0]	rw	1'h0	rx_en	1: enable 0: disable
0x110			**AUDIO_RX_PAUSE**	
[31:1]			RSVD	
[0]	rw	1'h0	rx_pause	RX pause control when rx_enable = 1. 1: pause 0: RX work
0x120			**AUDIO_RX_SERIAL_TIMING**	
[31:4]			RSVD	
[3]	rw	1'h0	lrck_pol	RX LRCK polarity control. 0: disable RX_LRCK inventor 1: enable RX_LRCK inventor for standard I2S, set tx_lrck_pol to low for Left/Right Justified, set tx_lrck_pol to hgih
[2]	rw	1'h0	slave_en	audio code receiver mode select. 0: master mode, 1: slave mode
[1:0]	rw	2'h0	timing	00: I2S 01: Left justified 10: right justified 11: reserved
0x130			**AUDIO_RX_PCM_DW**	
[31:5]			RSVD	
[4:0]	rw	5'h10	pcm_data_width	For I2S and left justified mode, M can be 8,13,14,16 For right justified mode, M can be 8, 13, 14, 16, 18, 20, 22, 24
0x140			**AUDIO_RX_LRCK_DIV**	
[31:28]			RSVD	
[27:16]	rw	12'd125	duty_high	RX LRCK duty cycle high: 125 for 48K FS 136 for 44.1K FS 185 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS
[15:12]			RSVD	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[11:0]	rw	12'd125	duty_low	RX LRCK duty cycle low: 125 for 48K FS 136 for 44.1K FS 190 for 32K FS 250 for 24K FS 272 for 22.05K FS 375 for 16K FS 500 for 12K FS 544 for 11.025K FS 750 for 8K FS Note: 1)duty_cycle = 12M/FS
0x150			**AUDIO_RX_BCLK_DIV**	
[31:10]			RSVD	
[9:0]	rw	10'h5	duty	RX serial bit clock duty cycle 5 for 48K FS 4 for 44.1K FS 5 for 32KFS 10 for 24K FS 8 for 22.05K FS 15 for 16K FS 20 for 12K FS 16 for 11.025K FS 30 for 8KFs
0x160			**RECORD_DATA_SEL**	
[31:1]			RSVD	
[0]	rw	1'h0	rs_data_sel	0: I2S audio recording 1: BT recording
0x170			**RX_RE_SAMPLE_CLK_DIV**	
[31:13]			RSVD	
[12:0]	rw	13'd250	rs_duty	source PCM sample clock duty cycle: 250 for 48K FS 272 for 44.1K FS 375 for 32K FS 500 for 24K FS 544 for 22.05K FS 750 for 16K FS 1000 for 12K FS 1088 for 11.025K FS 1500 for 8K FS Note: 1)duty_cycle = 12M/FS
0x180			**RX_RE_SAMPLE**	
[31:1]			RSVD	
[0]	rw	1'h0	smooth_en	0: Disable RX re-sample smooth filter 1: Enable RX re-sample smooth filter
0x190			**RECORD_FORMAT**	
[31:2]			RSVD	
[1]	rw	1'h0	track	1: mono recording, 0: stereo recording

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	dw	0: 8bit 1: 16bit RX fifo data format: Mono 8 bit (unsigned): RX FIFO_DIN[31:0] = L3,L2,L1,L0, each four samples need one FIFO write operation Stereo 8 bit (unsigned): RX_FIFO_DIN[31:0] = R1,L1,R0,L0, each tow samples need one FIFO write operation Mono 16 bit (Signed 2's complement): RX_FIFO_DIN[31:0] = L1,L0, each two samples need one FIFO write operation Stereo 16 bit (Signed 2's complement): RX_FIFO_DIN[31:0] = R0,L0, each sample need one FIFO write operation
0x1a0			**RX_CH_SEL**	
[31:4]			RSVD	
[3:2]	rw	2'h0	left_channel_sel	RX re-sampling module setting: 00: RD left = RX left 01: RD left = RX right 10,11: RD left = (RX left + RX right)/2
[1:0]	rw	2'h0	right_channel_sel	RX re-sampling module setting: 00: RD right = RX right 01: RD right = RX left 10,11: RD right = (RX left + RX right)/2
0x200			**BT_PHONE_CTRL**	
[31:6]			RSVD	
[5]	rw	1'h0	bb_i2s_bps_to_cdc	bypass baseband I2S interface to audio codec i2s interface 0: no bypass, 1: bypass
[4]	rw	1'h0	bt_pcm_if_bps	bypass baseband PCM signals to BT VCI master: 0: no bypass, 1: bypass
[3]	rw	1'h0	bt_path_sel	BT path select 0: digital path, 1: analog path
[2]	rw	1'h0	bt_mix_smooth_filter_en	0: disable the smooth filter for background mixer 1: enable the smooth filer for background mixer
[1]	rw	1'h0	bt_back_mix_en	background mixer enable 0: disable, 1: enable
[0]	rw	1'h0	bt_ph_en	BT phone enable 0: disable, 1: enable
0x210			**BB_PCM_FORMAT**	
[31:11]			RSVD	
[10]	rw	1'h0	pcm_clk_pol	input BB pcm clock polarity: 0: rising edge for data transmitting, falling edge for data receiving 1: rising edge for data receiving, falling edge for data transmitting
[9]	rw	1'h0	i2s_lrck_pol	0: no bb_i2s_lrck input inventor 1: enable bb_i2s_lrck input inventor for standard I2S, set tx_lrck_pol to low for Left/Right Justified, set tx_lrck_pol to high
[8]	rw	1'h0	pcm_lsb_flag	Serial PCM data bit sequence. 0: MSB first, 1: LSB first
[7]	rw	1'h0	pcm_sync_flag	0: short sync, 1: long sync
[6:5]	rw	2'h0	pcm_tim_sel	00: I2S timing, 01: Left Justified 10: Right Justified, 11: PCM timing
[4:0]	rw	5'h8	pcm_dw	Baseband Master PCM data width (>=8) Common value: 8, 13,14, 16, 18, 20, 22, 24. for I2S/Left Justified/Right Kistified timing, bb_pcm_dw >=16 For PCM timing, only 8, 13, 14, 16 configure value is available.
0x220			**BT_PCM_DW**	

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:5]			RSVD	
[4:0]	rw	5'h10	dw	BT PCM master data width (>= 8), common value: 8, 13,14, 16
0x230			**BT_PCM_TIMING**	
[31:3]			RSVD	
[2]	rw	1'h0	clk_pol	BT PCM master output pcm clock polarity: 0: rising edge for data transmitting, falling edge for data receiving 1: rising edge for data receiving, falling edge for data transmitting
[1]	rw	1'h0	sync_flag	0: short sync, 1: long sync
[0]	rw	1'h0	lsb_flag	Serial PCM data bit sequence. 0: MSB first, 1: LSB first
0x240			**BT_PCM_CLK_DUTY**	
[31:10]			RSVD	
[9:0]	rw	10'h0	clk_duty	BT_PCM_CLK duty cycle <= (GCLK/(bt_pcm_sync*bt_pcm_dw))
0x250			**BT_PCM_SYNC_DUTY**	
[31:6]			RSVD	
[5:0]	rw	6'h0	sync_duty	PCM_SYNC duty cycle (bt_pcm_sync frequency = bt_pclk_clk/bt_pcm_sync_duty)
0x260			**BT_VOL_CTRL**	
[31:4]			RSVD	
[3]	rw	1'h0	vol_adj_en	BT volume adjust enable
[2:0]	rw	3'h0	vol	BT master volume
0x300			**INT_MASK**	
[31:2]			RSVD	
[1]	rw	1'h1	tx_fifo_int_mask	Interrupt mask for TX FIFO pop underflow, high active
[0]	rw	1'h1	rx_fifo_int_mask	Interrupt mask for RX FIFO push overflow, high active
0x310			**INT_STATUS**	
[31:2]			RSVD	
[1]	rw	1'h0	tx_fifo_underflow	TX FIFO pop underflow
[0]	rw	1'h0	rx_fifo_overflow	RX FIFO push overflow
0x400			**TX_DMA_ENTRY**	
[31:0]	w	32'h0	tx_dma_entry	TX DMA entry
0x440			**RX_DMA_ENTRY**	
[31:0]	r	32'h0	rx_dma_entry	RX DMA entry
0x480			**DMA_MASK**	
[31:2]			RSVD	
[1]	rw	1'h1	tx_dma_mask	TX DMA mask enable:1: mask0: do not mask
[0]	rw	1'h1	rx_dma_mask	RX DMA mask enable:1: mask0: do not mask
0x500			**DEBUG_LOOP**	
[31:24]			RSVD	
[23:16]	rw	8'h2	sp_clk_div	sp clock divider value
[15:9]			RSVD	
[8]	w1c	1'h0	sp_clk_div_update	update sp clock divider
[7:3]			RSVD	
[2]	rw	1'h0	sp_clk_sel	clock select 0: xtal clock 1: pll clock
[1]	rw	1'h0	ad2da_loop_back	RX–>TX Loop debug control: 0: disable 1: enable, internally connect RX Resampled PCM to TX Resample PCM input

Continued on the next page...

Table 11-3: I2S Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw	1'h0	da2ad_loop_back	TX–>RX Loop debug control: 0: disable 1: enable, internally connect TX SDTO to RX SDTI
0x600			**FIFO_STATUS**	
[31:8]			RSVD	
[7:0]	rw	8'h0	fifo_status_out	FIFO Status output: Bit [7:0] = tx_full,tx_empty,tx_almost_full,tx_almost_empty, rx_full,rx_empty,rx_almost_full,rx_almost_empty
0x700			**TX_EQUALIZER_EN**	
[31:1]			RSVD	
[0]	rw	1'h0	tx_equalizer_en	0: Disable TX equalizer 1: Enable TX equalizer equalizer is not implemented
0x710			**TX_EQUALIZER_GAIN1**	
[31:30]			RSVD	
[29:25]	rw	5'h0	band6_gain	
[24:20]	rw	5'h0	band5_gain	
[19:15]	rw	5'h0	band4_gain	
[14:10]	rw	5'h0	band3_gain	
[9:5]	rw	5'h0	band2_gain	
[4:0]	rw	5'h0	band1_gain	
0x720			**TX_EQUALIZER_GAIN2**	
[31:20]			RSVD	
[19:15]	rw	5'h0	band10_gain	
[14:10]	rw	5'h0	band9_gain	
[9:5]	rw	5'h0	band8_gain	
[4:0]	rw	5'h0	band7_gain	

11.3 Audprc

Audprc is located in HPSYS.

11.3.1 Introduction

The full name of the Audprc module is Audio Process Controller, whose primary function is to process collected and playback audiodata and then deliver the processed data to the designated module. The audio processing functions within Audprc include gain adjustment, mixing, equalization, and sample rate conversion. Users can configure these functions individually as required.

11.3.2 System Architecture

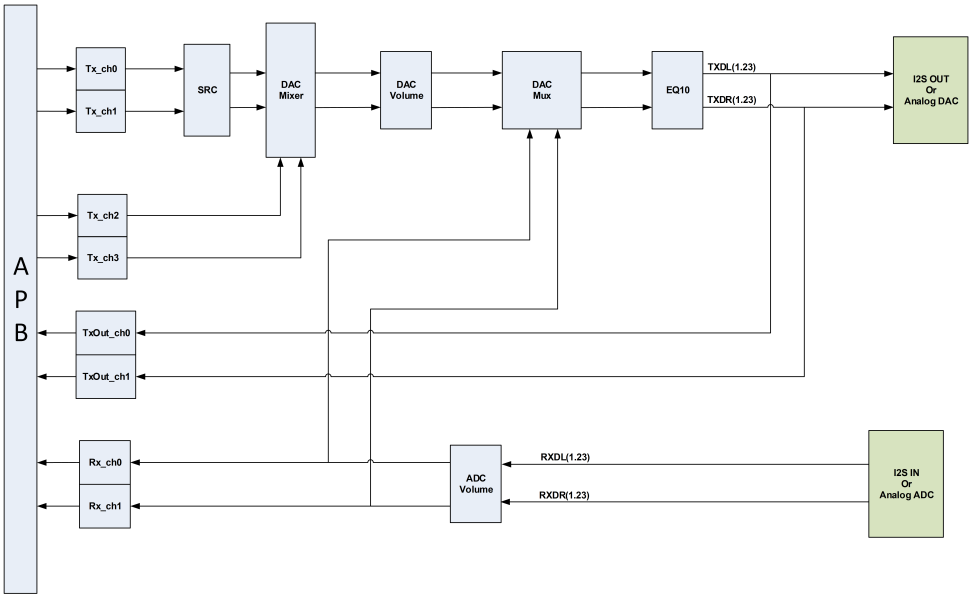


Figure 11-7: Audprc Block Diagram

Audprc mainly consists of two signal paths: the output TX path and the input RX path. The TX path transmits data from the memory space to the I2S output terminal or the analog DAC. The RX path receives data from the I2S input terminal or the analog ADC and stores it into the memory space.

11.3.3 Functional description

11.3.3.1 Sample Rate Conversion Module

The three stages of Half-band filters have identical structures and configuration methods. To achieve 2 fold upsampling, set Hbf enable to 1 and configure mode to 0. To achieve 2 fold downsampling, set Hbf enable to 1 and configure mode to 0. By default, the Hbf 's enable is 0 indicates that this stage of the Half-band filter is inactive and does not change the sampling rate. The configuration of the three-stage Half-band filter should approximate the final target sampling rate conversion ratio as closely as possible.

The last stage Sinc filter has two configuration parameters: the enable flag and the sampling rate conversion ratio Sinc_ratio. Sinc_ratio is a 31-bit fxed-point unsigned number, consisting of 1 integer bit and 30 fractional bits, corresponding to a precision of $1/2^{30}$.

The following example configures the SRC for conversion from 16KHz to 44.1KHz.

There are two relatively simple methods for converting from 16KHz to 44.1KHz.

- Scheme 1:16KHz->32KHz->44.1KHz
- Scheme 2:16KHz->32KHz->64KHz->44.1KHz

Both schemes are theoretically feasible. In Scheme 1, a first-stage Half-band filter is used for 2times upsampling, followed by a Sinc filter with a RatioFor of 1.378 times. In Scheme 2, two stages of Half-band filters are applied for a total of 4 times upsampling, followed by a Sinc filter with a Ratio of 0.689 times. Comparing the two schemes, the Sinc filter Ratio

in Scheme 2 is closer to 1, indicating better filter performance; therefore, Scheme 2 is selected.

11.3.3.2 Mixer Module

The Mixer Module(DAC Mixer, DAC Mux)is used on the DAC path to combine audio sources from different origins.

The input of the DAC Mixer consists of four TX channels; these four audio channels are recombined into two audio outputs. Each output channel is generated by summing two audio data streams, which can be selected from the four TX channels or muted. For details, refer to registers mixlsrc0, mixlsrc1, mixrsrc0, mixrsrc1.

The input of the DAC Mux includes two DAC channels and two ADC channels. The mixing method of the Mux is identical to that of the Mixer, combining four inputs into two outputs, with each audio source configurable independently. For details, please refer to registers muxlsrc0 , muxlsrc1 , muxrsrc0 , muxrsrc1 .

11.3.3.3 Gain Adjustment Module

The DAC and ADC paths are controlled by independent gain adjustment modules Volume, with separate adjustments for the left and right audio channels . The range is-18~ 13 dBwith a step size of 0.5 dB .

11.3.3.4 Equalizer Module

The equalizer module EQ supports up to 10 independent adjustment stages. Users must configure eq_stage to enable the desired number of stages. Each stage has 5 parameters, totaling 50 configurable parameters. Users can utilize tools to generate the desired equalizer effect, then configure the parameters into Audprc . Before using the equalizer, it is necessary to first enable eq_clr , wait until eq_clr_done equals 1 , then clear eq_clr . This step is intended to clear internal temporary data residues. If not cleared, the equalizer module's convergence time will be longer, and transient noise may occur upon activation.

11.3.4 Configuration Procedure

11.3.4.1 Configure Tx and Rx Channels

The Tx channel of Audprc supports mono 16bit/24bit and stereo 16bit audio data formats, with a maximum of 4 audio channels (stereo occupies two channels). Each Tx channel can support single-channel 16bit/24bit audio by configuring the format register. For stereo 16bit audio sources (each 32bit data contains 16bit data for both left and right channels), only Tx channels 0 and 2 are supported; mode configuration can be performed via the Mode register. When Tx channel 0 is configured to stereo 16bit mode, Tx channel 1 will be occupied and unavailable for use. Similarly, when Tx channel 2 is configured to stereo 16bit mode, Tx channel 3 will also be occupied and unavailable for use.

The Audprc Rx channel supports mono 16bit/24bit audio data formats, with a maximum of 2 audio channels. Each Rx channel can support single-channel 16bit/24bit audio by configuring the format register.

After configuring the audio data structures for both Txand Rx,it is necessary to configure the corresponding DMAto manage data transmission and reception for the Txand Rxchannels.

11.3.4.2 Configure the DAC path

The DAC path configuration includes the previously mentioned modules such as SRC, EQ, Mixer, Mux, and Volume. It should be noted that both SRC and EQ support independent enablement of two input channels; configuration requires

enabling only the input channels with audio data. Additionally, when configuring the Mux, if the data from the selected Rx channel is mixed with the data from the Tx channel, it is necessary to ensure that both have the same sampling rate; otherwise, audio mixing synchronization issues will occur.

After configuring the internal DAC module, the output target module of the DAC can be set via the register dst_sel. If dst_sel is 0, the DAC data will be sent to the Codec module. This path supports up to 24-bit stereo. After processing by the Codec module, the audio data will be converted into an analog audio signal output. If dst_sel is 1, the DAC data will be sent to the I2S module. This path supports up to 16-bit stereo, and the data will ultimately be output through the digital I2S interface. If dst_sel is 2, the DAC data will be sent to the Audprc Tx_out channel, which supports up to 24-bit stereo. Users can configure DMA to read data from the Tx_out channel and store it into memory.

Additionally, the DAC path requires configuration of the sampling clock source and sampling division ratio. The clock source can be selected from the chip's built-in audio PLL or the 48 MHz crystal oscillator. It is necessary to ensure that the clock source used by the DAC path matches the output target module, and that the divided clock corresponds to the sampling rate of the output module. If the output module is the Tx_out channel, this requirement does not apply.

11.3.4.3 Configure ADC path

The ADC path configures a Volumemodule. After configuring the Volume,the audio source of theADCcan be set via the register src_sel.If src_sel is 0, the ADC data source is the Codec module. This path supports up to 24-bit stereo. The Codec data originates from the external analog audio signal acquired by the ADC. If src_sel is 1, the ADC data source is the I2S module. This path supports up to 16-bit stereo, with audio data sourced from the external I2S digital interface.

Similar to the DACpath, the ADC path requires configuration of the sampling clock source and division ratio. The clock source must match the audio source, and the clock after division must Consistent with the audio sampling rate.

When ADC data is routed through the Mux to the DAC path, the path of ADC data to the Rx channel remains unaffected; users can simultaneously acquire ADC data from the enabled Rx channel.

11.3.5 Audprc

Table 11-4: Audprc Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**id**	
[31:0]	rw	32'hA0000	rev	revision id
0x04			**cfg**	
[31:21]			RSVD	
[20]	w1c	1'h0	audclk_div_update	audprc clock divider update, write 1 to update
[19:16]	rw	4'h2	audclk_div	audprc clock divider, 0 and 1 means divide by 1
[15:10]			RSVD	
[9]	rw	1'h0	stb_clk_sel	audio strobe clock select 0: use xtal clock to generate strobe 1: use pll clock to generate strobe
[8]	rw	1'h0	auto_gate_en	auto clock gating enable, high active
[7]	rw	1'h0	adc_path_en	adc path enable
[6]	rw	1'h0	dac_path_en	dac path enable
[5]	rw	1'h0	adc_path_sreset	adc path software reset, high active
[4]	rw	1'h0	dac_path_sreset	dac path software reset, high active
[3]	rw	1'h0	adc_path_flush	adc path fifo flush, high active
[2]	rw	1'h0	dac_path_flush	dac path fifo flush, high active

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	sreset	audprc software reset, high active
[0]	rw	1'h0	enable	audprc enable
0x08			**stb**	
[31:16]	rw	16'h1	adc_div	adc strobe divider
[15:0]	rw	16'h1	dac_div	dac strobe divider
0x0C			**irq**	
[31:26]			RSVD	
[25]	rw	1'h0	tx_out1_fifo_uf_mask	tx_out channel 1 fifo underflow mask, 0: mask the interrupt
[24]	rw	1'h0	tx_out0_fifo_uf_mask	tx_out channel 0 fifo underflow mask, 0: mask the interrupt
[23]	rw	1'h0	rx_in_fifo_of_mask	rx input fifo overflow mask, 0: mask the interrupt
[22]	rw	1'h0	tx_out_fifo_uf_mask	tx output fifo underflow mask, 0: mask the interrupt
[21]	rw	1'h0	rx1_fifo_uf_mask	rx channel 1 fifo underflow mask, 0: mask the interrupt
[20]	rw	1'h0	rx0_fifo_uf_mask	rx channel 0 fifo underflow mask, 0: mask the interrupt
[19]	rw	1'h0	tx3_fifo_of_mask	tx channel 3 fifo overflow mask, 0: mask the interrupt
[18]	rw	1'h0	tx2_fifo_of_mask	tx channel 2 fifo overflow mask, 0: mask the interrupt
[17]	rw	1'h0	tx1_fifo_of_mask	tx channel 1 fifo overflow mask, 0: mask the interrupt
[16]	rw	1'h0	tx0_fifo_of_mask	tx channel 0 fifo overflow mask, 0: mask the interrupt
[15:10]			RSVD	
[9]	rw1c	1'h0	tx_out1_fifo_uf	tx_out channel 1 fifo underflow, write 1 to clear
[8]	rw1c	1'h0	tx_out0_fifo_uf	tx_out channel 0 fifo underflow, write 1 to clear
[7]	rw1c	1'h0	rx_in_fifo_of	rx input fifo overflow, write 1 to clear
[6]	rw1c	1'h0	tx_out_fifo_uf	tx output fifo underflow, write 1 to clear
[5]	rw1c	1'h0	rx1_fifo_uf	rx channel 1 fifo underflow, write 1 to clear
[4]	rw1c	1'h0	rx0_fifo_uf	rx channel 0 fifo underflow, write 1 to clear
[3]	rw1c	1'h0	tx3_fifo_of	tx channel 3 fifo overflow, write 1 to clear
[2]	rw1c	1'h0	tx2_fifo_of	tx channel 2 fifo overflow, write 1 to clear
[1]	rw1c	1'h0	tx1_fifo_of	tx channel 1 fifo overflow, write 1 to clear
[0]	rw1c	1'h0	tx0_fifo_of	tx channel 0 fifo overflow, write 1 to clear
0x10			**tx_ch0_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch0
[2]	rw	1'h0	mode	tx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, tx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x14			**tx_ch0_entry**	
[31:0]	rw	32'h0	data	tx channel 0 data entry
0x18			**tx_ch1_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch1
[2]			RSVD	

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x1C			**tx_ch1_entry**	
[31:0]	rw	32'h0	data	tx channel 1 data entry
0x20			**tx_ch2_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch2
[2]	rw	1'h0	mode	tx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, tx channel 3 is not working, both left and right audio data comes from channel 2.
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x24			**tx_ch2_entry**	
[31:0]	rw	32'h0	data	tx channel 2 data entry
0x28			**tx_ch3_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx ch3
[2]			RSVD	
[1]	rw	1'h0	format	tx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx channel 0 enable
0x2C			**tx_ch3_entry**	
[31:0]	rw	32'h0	data	tx channel 3 data entry
0x30			**rx_ch0_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	rx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for rx ch0
[2]	rw	1'h0	mode	rx mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, rx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	rx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	rx channel 0 enable
0x34			**rx_ch0_entry**	
[31:0]	r	32'h0	data	rx channel 0 data entry
0x38			**rx_ch1_cfg**	
[31:8]			RSVD	

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7:4]	r	4'h0	fifo_cnt	rx fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for rx ch1
[2]			RSVD	
[1]	rw	1'h0	format	rx format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	rx channel 1 enable
0x3C			**rx_ch1_entry**	
[31:0]	r	32'h0	data	rx channel 1 data entry
0x40			**tx_out_ch0_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx out fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx out ch0
[2]	rw	1'h0	mode	tx out mode 1'h0: mono mode 1'h1: stereo mode This bit is only used for 16-bit mode, in 24-bit mode, channel can only be set in mono mode. In 16-bit stereo mode, rx channel 1 is not working, both left and right audio data comes from channel 0.
[1]	rw	1'h0	format	tx out format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx out channel 0 enable
0x44			**tx_out_ch0_entry**	
[31:0]	r	32'h0	data	tx out channel 0 data entry
0x48			**tx_out_ch1_cfg**	
[31:8]			RSVD	
[7:4]	r	4'h0	fifo_cnt	tx out fifo counter
[3]	rw	1'h0	dma_msk	1: mask the dma request for tx out ch1
[2]			RSVD	
[1]	rw	1'h0	format	tx out format 0: 16-bit mode 1: 24-bit mode
[0]	rw	1'h0	enable	tx out channel 1 enable
0x4C			**tx_out_ch1_entry**	
[31:0]	r	32'h0	data	tx out channel 1 data entry
0x50			**dac_path_cfg0**	
[31:26]			RSVD	
[25:24]	rw	2'h0	dst_sel	dac path destination select 2'h0: select audio codec 2'h1: select external interface 2'h2: select apb interface 2'h3: reserved
[23:21]	rw	3'h3	mixrsrc1	dac mixer right channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:tx ch2 3'h3:tx ch3 3'h4:mute other: mute

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[20:18]	rw	3'h2	mixrsrc0	dac mixer right channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:tx ch2 3'h3:tx ch3 3'h4:mute other: mute
[17:15]	rw	3'h1	mixlsrc1	dac mixer left channel input source1 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:tx ch2 3'h3:tx ch3 3'h4:mute other: mute
[14:12]	rw	3'h0	mixlsrc0	dac mixer left channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:tx ch2 3'h3:tx ch3 3'h4:mute other: mute
[11:6]	rw	6'h3f	vol_r	volume control from -18~13dB, step is 0.5dB 6'h0: -18dB 6'h1: -17.5dB ..... 6'h3e: 13dB 6'h3f: mute
[5:0]	rw	6'h3f	vol_l	volume control from -18~13dB, step is 0.5dB 6'h0: -18dB 6'h1: -17.5dB ..... 6'h3e: 13dB 6'h3f: mute
0x54			**dac_path_cfg1**	
[31:30]	rw	2'h0	src_ch_clr	clear src channal internal data
[29:28]	r	2'h0	src_ch_clr_done	src channel internal data clear done
[27]	rw	1'h0	src_hbf3_mode	3rd stage hbf mode: 0: upsampling 1: downsampling
[26]	rw	1'h0	src_hbf3_en	3rd stage hbf enable
[25]	rw	1'h0	src_hbf2_mode	2nd stage hbf mode: 0: upsampling 1: downsampling
[24]	rw	1'h0	src_hbf2_en	2nd stage hbf enable
[23]	rw	1'h0	src_hbf1_mode	1st stage hbf mode: 0: upsampling 1: downsampling
[22]	rw	1'h0	src_hbf1_en	1st stage hbf enable
[21:20]	rw	2'h0	src_ch_en	source rate converter channel enable
[19]	rw	1'h0	eq_clr	equalizer clear request
[18]	r	1'h0	eq_clr_done	equalizer clear done flag
[17:14]	rw	4'ha	eq_stage	set equalizer stage, max is 10.

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[13:12]	rw	2'h0	eq_ch_en	equalizer channel enable 2'b11: enable both channel 2'b10: enable right chanel only 2'b01: enable left channel only 2'b00: bypass equalizer
[11:9]	rw	3'h3	muxrsrc1	dac mux right channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:rx ch0 3'h3:rx ch1 3'h4:mute other: mute
[8:6]	rw	3'h2	muxrsrc0	dac mux right channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:rx ch0 3'h3:rx ch1 3'h4:mute other: mute
[5:3]	rw	3'h1	muxlsrc1	dac mux left channel input source1 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:rx ch0 3'h3:rx ch1 3'h4:mute other: mute
[2:0]	rw	3'h0	muxlsrc0	dac mux left channel input source0 select 3'h0:tx ch0 3'h1:tx ch1 3'h2:rx ch0 3'h3:rx ch1 3'h4:mute other: mute
0x58			**dac_path_cfg2**	
[31]	rw	1'h0	src_sinc_en	sinc filter enable
[30:0]	rw	31'h0	sinc_ratio	sinc filter ratio, s31.30 format. Range from 0~2
0x5C			**adc_path_cfg0**	
[31:15]			RSVD	
[14]	rw	1'h0	rx2tx_loopback	rx to tx loopback enable
[13]	rw	1'h0	data_swap	swap adc path left and right channel data
[12]	rw	1'h0	src_sel	adc path source select 1'h0: select audio codec 1'h1: select external interface
[11:6]	rw	6'h3f	vol_r	volume control from -18~13dB, step is 0.5dB 6'h0: -18dB 6'h1: -17.5dB ..... 6'h3e: 13dB 6'h3f: mute

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:0]	rw	6'h3f	vol_l	volume control from -18~13dB, step is 0.5dB 6'h0: -18dB 6'h1: -17.5dB ..... 6'h3e: 13dB 6'h3f: mute
0x70			**dac_eq_cfg0**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x74			**dac_eq_cfg1**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x78			**dac_eq_cfg2**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x7C			**dac_eq_cfg3**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x80			**dac_eq_cfg4**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x84			**dac_eq_cfg5**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x88			**dac_eq_cfg6**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x8C			**dac_eq_cfg7**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x90			**dac_eq_cfg8**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x94			**dac_eq_cfg9**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x98			**dac_eq_cfg10**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x9C			**dac_eq_cfg11**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xA0			**dac_eq_cfg12**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xA4			**dac_eq_cfg13**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xA8			**dac_eq_cfg14**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xAC			**dac_eq_cfg15**	

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xB0			**dac_eq_cfg16**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xB4			**dac_eq_cfg17**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xB8			**dac_eq_cfg18**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xBC			**dac_eq_cfg19**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xC0			**dac_eq_cfg20**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xC4			**dac_eq_cfg21**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xC8			**dac_eq_cfg22**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xCC			**dac_eq_cfg23**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD0			**dac_eq_cfg24**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD4			**dac_eq_cfg25**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xD8			**dac_eq_cfg26**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xDC			**dac_eq_cfg27**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE0			**dac_eq_cfg28**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE4			**dac_eq_cfg29**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xE8			**dac_eq_cfg30**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xEC			**dac_eq_cfg31**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xF0			**dac_eq_cfg32**	
[31:24]			RSVD	

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[23:0]	rw	24'h0	coef	
0xF4			**dac_eq_cfg33**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xF8			**dac_eq_cfg34**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0xFC			**dac_eq_cfg35**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x100			**dac_eq_cfg36**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x104			**dac_eq_cfg37**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x108			**dac_eq_cfg38**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x10C			**dac_eq_cfg39**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x110			**dac_eq_cfg40**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x114			**dac_eq_cfg41**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x118			**dac_eq_cfg42**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x11C			**dac_eq_cfg43**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x120			**dac_eq_cfg44**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x124			**dac_eq_cfg45**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x128			**dac_eq_cfg46**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x12C			**dac_eq_cfg47**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x130			**dac_eq_cfg48**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	
0x134			**dac_eq_cfg49**	
[31:24]			RSVD	
[23:0]	rw	24'h0	coef	

Continued on next page...

Table 11-4: Audprc Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x138			**RESERVED_IN**	
[31:24]			RSVD	
[23:16]	rw	8'hf	CTRL_2	reserved control 2
[15:8]	rw	8'hf	CTRL_1	reserved control 1
[7:0]	rw	8'hf	CTRL_0	reserved control 0
0x13C			**RESERVED_OUT**	
[31:8]			RSVD	
[7:0]	r	8'h0	STAT	reserved status

12 Accelerator

12.1 Neural Network Accelerator

12.1.1 Neural Network Matrix Convolution Accelerator (NNACC)

The matrix convolution accelerator is designed to meet the demand for underlying matrix computing power in machine learning computations and is broadly applicableto various neural network frameworks. The accelerator features a comprehensive memory access interface and provides flexible data address configuration. Supports a maximum input matrix of 255 × 255 and a maximum of 128 input/output channels. Supports 8-bit integer operations, capable of meeting most edge AI computing requirements, such as 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 in vokes this processor through dedicated co-processor instructions. The co-processor features include:

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

12.2 FFT Accelerator

FFT computations require high processing power; using an FFT accelerator can reduce CPU load and improve system performance. The HPSYS integrates an FFT accelerator, enabling it to meet FFT processing requirements across more scenarios.

The main features of the FFT accelerator are as follows:

- Supports FFT sizes from a minimum of 16 points to a maximum of 512 points
- Supports fixed-point signed input/output for 24points, 16points, and 8points, with independently configurable input and output bit widths
- Supports real-valued FFT computations
- Supports IFFT
- Supports windowed FFT/IFFT Supports DCT/IDCT

12.3 Cordic Coprocessor

The Cordic coprocessor is used to compute trigonometric and hyperbolic functions, as well as certain derived arithmetic operations. Each of the HPSYS and LPSYS integrates a Cordic coprocessor.

The Cordiccoprocessor features are as follows:

- SupportsARM coprocessorinstructions.
- Supports ARM Custom Datapath Extension instructions (only in HPSYS).
- Supports trigonometric operations:cos、sin、ang、mod、atan、rot
- Supports hyperbolic operations: cosh、sinh、atanh、angh、modh、mul、div、ln、exp、sqrt
- Supports 32-bit fixed-point input and output.

12.4 CRC

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

12.4.1 Introduction

CRC (Cyclic Redundancy Check) can perform CRC calculations with specific bit widths, arbitrary generating polynomials, and arbitrary initial values. Data can be input through CPU or DMA, with the minimum input unit being a single byte and no maximum byte limit. A single HCLK cycle can complete the calculation for a single byte input. The checksum result is obtained immediately after all data input is completed. It supports highlow bit reversal for input data and high-low bit reversal for output data. It also accommodates input data with varying valid bit widths.

12.4.2 Main Features

- 7/8/16/32 Bit CRCCalculation
- Arbitrary Custom Polynomial
- Arbitrary Initial Value
- Input data supports single-byte/double-byte/triple-byte/four-byte valid bit widths
- Input data supports byte/double-byte/four-byte high-low bit reversal
- Output data supports high-low bit reversal
- Calculation speed is 1 byte per HCLK cycle

12.4.3 CRC Configuration Method

Before starting the CRC calculation, it is necessary to pre-configure the corresponding Register, including polynomial width, valid data bit width, input-output reversal mode, polynomial, and initial value, etc. The mainstream CRC format configuration methods are shown in the table below.

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

Continued on the next page...

Table 12-1: CRC Configuration Method(continued)

CRC Algorithm	Polynomial Formula	POLYSIZE	POL	INIT	REV_IN	REV_OUT	Result XOR Value
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 CRC module does not perform the XOR operation on the result; this must be handled by software after reading the result.

12.4.4 Data Format

The basic data unit for CRC calculation is a byte. The data written to the DR register is fxed at 4 bytes, and the bytes that participate in the calculation are specifed by CR_DATASIZE . The gray cells in the table below indicate which bytes are included in the calculation.

Table 12-2: Data Participating in the Calculation

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

Each byte of data participating in the calculation can be specifed individually; however, it is important to note that changing CR_DATASIZE must occur when SR_DONE is 1 ; otherwise, it may affect the calculation result of the current data.

The operation sequence is executed in order by BYTE0, BYTE1, BYTE2, BYTE3. When calculating each byte, it defaults to the order from the most signifcant bit to the least signifcant bit. If the input inversion mode is configured, the order will follow the inverted sequence from the most signifcant bit to the least signifcant bit. The table below provides configuration examples, which can be flexibly adjusted according to the data format in memory.

Table 12-3: Operation Sequence

DATASIZE	REV_IN	DR	Inverted Input	First Beat Calculate Byte	Second Beat Calculate Byte	Third Beat Calculate Byte	Fourth Beat Calculate Byte
0	0	0x12345678	/	0x78	/	/	/
1	0	0x12345678	/	0x78	0x56	/	/
2	0	0x12345678	/	0x78	0x56	0x34	/
3	0	0x12345678	/	0x78	0x56	0x34	0x12
3	1	0x12345678	0x482C6A1E	0x1E	0x6A	0x2C	0x48
3	2	0x12345678	0x2C481E6A	0x6A	0x1E	0x48	0x2C
3	3	0x12345678	0x1E6A2C48	0x48	0x2C	0x6A	0x1E

12.4.5 Calculation Rate

CRC Completes the calculation of one byte per HCLKcycle. When data is continuously input, the time required for calculation is approximately the number of bytes×HCLK cycles.

12.4.6 CRC Configuration Process

1. Configure the CRC format and set POL according to requirements. , INIT , CR_POLYSIZE , CR_REV_IN , CR_REV_OUT , CR_DATASIZE 。
2. Set CR_RESET to 1to initialize the CRC.
3. Continuously transfer the required data to the DR Register using the CPU or DMA.
4. If the remaining number of data bytes does not match CR_DATASIZE, first check SR_DONE; if it is 1, change CR_DATASIZE and write the remaining data into the DR Register.
5. Read the DR Register to obtain the calculation result, and perform a software XOR operation as needed to derive the final CRC value.

12.4.7 CRC Register

Table 12-4: CRC Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**DR**	Data register
[31:0]	rw	32'hffffffff	DR	Data register bits. This register is used to write new data to the CRC calculator. It holds the previous CRC calculation result when it is read. If the data size is less than 32 bits, the least significant bits are used to write/read the correct value.
0x04			**SR**	Status register
[31:2]			RSVD	
[1]	r	1'h0	overflow	Overflow when new data arrive while last calculation not done yet
[0]	r	1'h0	done	Done flag. When DR written, done flag will be cleared automatically. The flag will assert after CRC operation of current DR finished.
0x08			**CR**	Control register
[31:8]			RSVD	

Continued on the next page...

Table 12-4: CRC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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.

12.5 FACC

Chip has one FACC, located within HPSYS.

12.5.1 Introduction

The chip contains one FACC located in HPSYS.

12.5.2 Key Features

- Supports 8/16-bitdata width
- Supports FIR/IIR filter algorithms
- Supports FIR mode with four algorithms: standard, convolution, symmetric, and antisymmetric
- Supports FIR mode maximum order of 32(16bit) and64(8bit)
- Supports IIR mode maximum order of 16/16(16bit) and32/32(8bit)

- Supports data parallel processing
- Input/output data addresses configurable via the AHB bus
- Supports data block processing
- Supports multiple sets of different data cross-processing

12.5.3 FIR standard mode

The FACC module supports the following standard FIR algorithm formula:

$$y[n] = \sum\_k = 0^N b[k]x[n-k]$$

x is the input array to be processed,y is the output filtered array, and b is the parameter for each order calculation.x array is read from the source address via the ahb bus, and they array is transmitted to the destination address via the ahb bus. The b array is directly configured by software into the module's internal fifo , with the fifo bit width of 32bit .

The data in the x array is arranged sequentially from low to high bits as $x[0]$,$x[1]$, $x[2]$, and so forth. The same applies to the y and b arrays.

12.5.4 FIR Standard Mode

The FACC module supports the following FIR convolution algorithm formula:

$$y\_n = \begin{cases} \sum\_m = 0^n h\_mx\_n - m, & \text{if } 0 \le n < M, \\ \sum\_m = 0^M h\_mx\_n - m, & \text{if } M \le n \le L-1, \\ \sum\_m = n - L + 1^M h\_mx\_n - m, & \text{if } L-1 < n \le L-1+M, \end{cases}$$

In the formula, the harray corresponds to the FACCmodule's b array. In convolution mode, the number of output data elements in the y array exceeds that of the input data elements in the x array.

12.5.5 FIR Symmetric Mode

The formula for the FIR symmetric mode supported by the FACC module is identical to that of the standard FIR mode; the distinction lies in b the data structure. In this mode, b data is distributed symmetrically. Based on the structure of the b data, the symmetric mode is further classifed into two types: odd symmetry and even symmetry.

Assuming the $b$array contains $n$data points, in the even symmetric mode, the new array after symmetry of thebarray is:

$$b[0]b[1]b[2].....b[n-2]b[n-1]b[n-1]b[n-2].....b[1]b[0]$$

The order of the array after even symmetry is $2n$. In the odd symmetric mode, the new array after symmetry of the b array is:

$$b[0]b[1]b[2].....b[n-2]b[n-1]b[n-2].....b[1]b[0]$$

The order of the array after odd symmetry is $2n-1$.

In this mode, configured to b For the array fifodata, it is unnecessary to write all symmetric data; only the original data prior to symmetry needs to be written. $b[0]b[1].....b[n-1]$ means affirmative. The order parameter is also configured according to the order prior to symmetry.

12.5.6 FIR Antisymmetric Mode

The difference between the FIR antisymmetric mode and symmetric mode supported by the FACC module is that the symmetric mode accumulates the calculated data after symmetry, whereas the antisymmetric mode subtracts the calculated data after symmetry. The symmetric structure of the b array corresponds to the FIR symmetric mode, comprising odd symmetry and even symmetry.

12.5.7 IIR Mode

The IIR algorithm formula supported by the FACC module is as follows:

$$y[n] = \sum\_k = 0^N b[k]x[n-k] + \sum\_j = 1^M a[j]y[n-j]$$

Compared to the FIR algorithm, the a data is added; during parameter configuration, the b array and a array must be configured separately. $a[0]$ data is not used, so the $a[0]$ data does not need to be written into the fifo within the a array.

12.5.8 Gain Adjustment

The result after filtering calculation can have its gain adjusted through register configuration.

12.5.9 Data Blocking

The FACC module supports block processing of input data, dividing input data into blocks according to specific rules; after processing one block, the next block processing is initiated. At the start of each data block, parameters such as source address, destination address, and data block length must be reconfigured. If operating in FIR convolution mode, when starting the last data block, the last_sel flag bit must also be configured to complete the fnal convolution operation.

Data chunking rule: Since the FACC internally processes data in 4-line parallel, the data length of all chunks except the last must be an integer multiple of 4. Data length is defined by the number of data units to be processed. For example, if the data to be processed is 8-bit, then 8-bit counts as 1 data length, and 16-bit counts as 2 data lengths. If the data to be processed is 16-bit, then 16-bit counts as 1 data length, and 32-bit counts as 2 data lengths.

12.5.10 Data Cache Mode

The FACC module supports data cache mode, which enables interleaved processing of multiple data sets. For example, with two data sets, after processing one block of the first set, the module processes one block of the second set, then another block of the first set, and so on. The principle applies similarly to more data sets. In data cache mode, each data group requires an external cache buffer to store intermediate computation results. Upon restart, the cached data is read back; therefore, data cache space must be reserved and the data cache address configured.

The size of the cache space for each data group is as follows:

Mode	Space size(byte)
FIR	68
FIR symmetric	132
FIR convolution	68
IIR	100

In data cache mode, buf_sel must be configured to 1 . When buf_sel=1 , the module first reads cached data from the cache address; therefore, the cache space data must be cleared when starting the first data block. After processing each data block, the data to be cached is sent to the cache space

In non-cache mode, intermediate data within one data group cannot be interleaved with another data group; the entire data group must be processed in a single operation before initiating computation of other data groups.

12.5.11 configuration Procedure in Non-Cache Mode

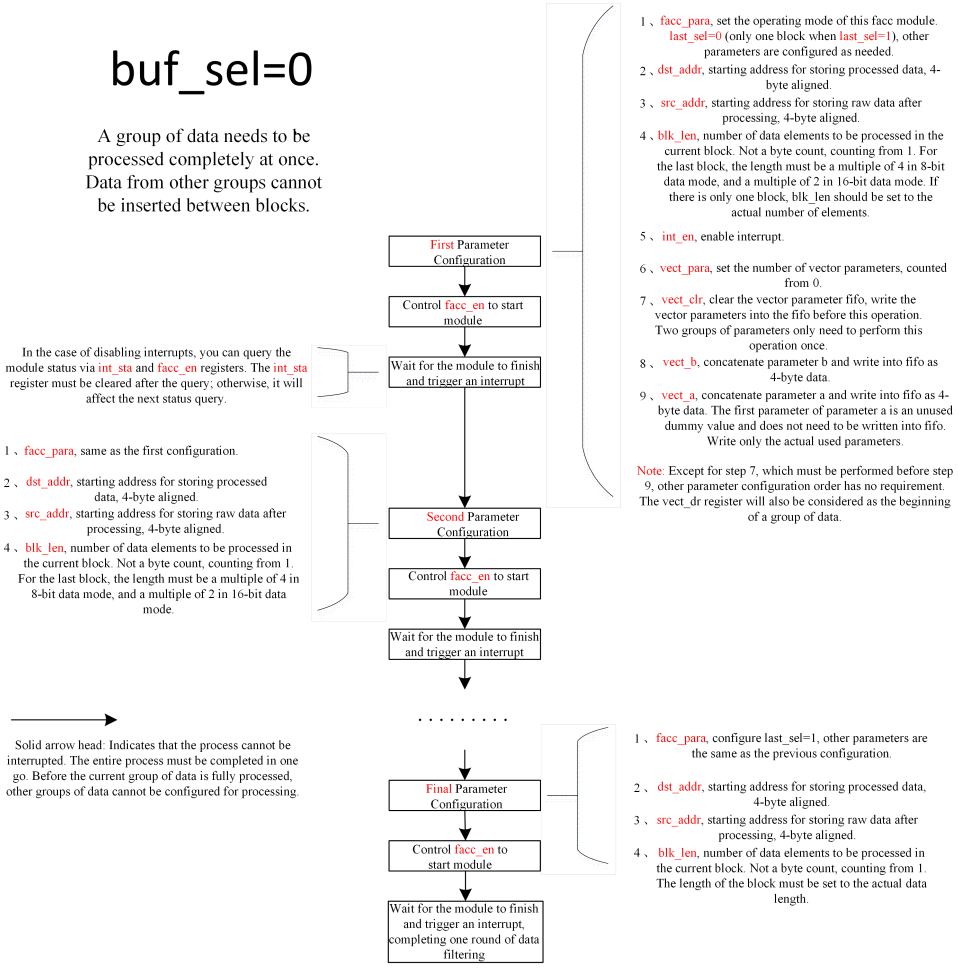


Figure 12-1: configuration Procedure Flowchart in Non-Cache Mode

12.5.12 configuration Procedure in Cache Mode

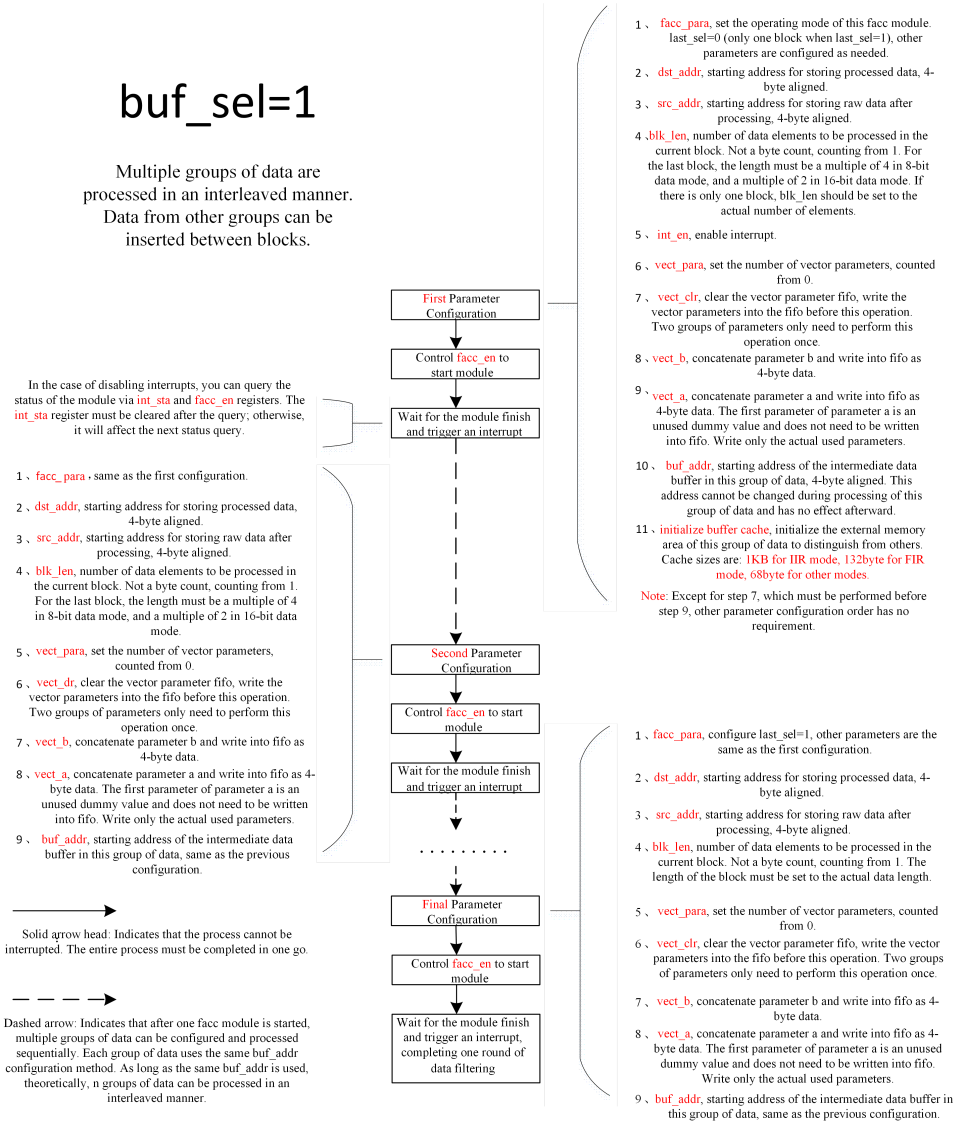


Figure 12-2: Flowchart of the configuration Procedure in Cache Mode

12.5.13 FACC Register

Table 12-5: FACC Register map

Offset	Attribute	Reset Value	Register Name	Register Description
0x000			**FACC_EN**	facc enable ctrl
[31:1]			RSVD	
[0]	rw1s	1'h0	facc_en	1:enable 0:stop
0x004			**FACC_PARA**	FACC parameter
[31:13]			RSVD	
[12]	rw	1'h0	anti_sel	when sym_sel=1,this bit is effective 0:disable 1:enable
[11]	rw	1'h0	buf_sel	0:only support one group data 1:store intermediate data to external buffer,support cross processing of multiple groups data
[10]	rw	1'h0	even_sel	when sym_sel=1, this bit is effective 0: odd 1: even
[9]	rw	1'h0	conv_sel	0: filter 1: convolution(only support fir mod)
[8]	rw	1'h0	fp_sel	0: 16bit 1: 8bit
[7]	rw	1'h0	sym_sel	0:normal 1:symmetric(only support fir mod)
[6]	rw	1'h0	mod_sel	0: fir 1: iir
[5]	rw	1'h0	first_sel	unused
[4]	rw	1'h0	last_sel	when conv_sel=1 ,it is necrssary 0:other data block 1:last data block
[3:0]	rw	4'h0	gain	facc gain
0x008			**SRC_ADDR**	source address
[31:0]	rw	32'h0	src_addr	source address
0x00C			**DST_ADDR**	destination address
[31:0]	rw	32'h0	dst_addr	destination address
0x010			**BLK_LEN**	source block length
[31:0]	rw	32'h0	blk_len	block data byte(8bit) or word(16bit) number,count from 1. if not the last data block,the block length must be a multiple of 4.
0x014			**BUF_ADDR**	buffer address
[31:0]	rw	32'h0	buf_addr	intermediate data buffer address
0x018			**DATA_SEQ**	the sequence of the first data
[31:0]	rw	32'h0	data_seq	unused
0x01C			**VECT_PARA**	vector number
[31:16]			RSVD	
[15:8]	rw	8'h0	m	IIR vector a step , count from 0
[7:0]	rw	8'h0	p	FIR/IIR vector b step , count from 0
0x020			**VECT_B**	vector b fifo
[31:0]	w	32'h0	vect_b	vector b
0x024			**VECT_A**	vector a fifo
[31:0]	w	32'h0	vect_a	vector a
0x028			**VECT_CLR**	vector fifo clr
[31:1]			RSVD	

Continued on next page...

Table 12-5: FACC Register map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[0]	rw1s	1'h0	vect_clr	vector fifo clr
0x02C			**INT_EN**	interrupt enable
[31:1]			RSVD	
[0]	rw	1'h0	int_en	interrupt enable
0x030			**INT_STA**	interrupt state
[31:1]			RSVD	
[0]	rw1c	1'h0	int_sta	interrupt state
0x034			**DBG_DATA**	debug data
[31:0]	r	32'h0	debug_data	13:inbuf_empty 12:inmem_full 11:inmem_empty 10:iir_en 9:iir_on 8:iir_ok 7:fir_end 6:conv_act 5:fir_on 4:blk_on 3~0 :ahb_state

13 Security

13.1 AES

AES locate in HPSYS。

13.1.1 Introduction

The AES_ACC module of the SF32LB56x is primarily designed to accelerate encryption and decryption operations of specialized algorithms in the security domain. Symmetric encryption algorithms include AES128 , AES192 , AES256 , and SM4 . Modes include ECB , CTR , and CBC . Hash algorithms include SHA1 , SHA224 , SHA256 , and SM3 . Upon activation, the AES_ACC module invokes the internal DMA to read raw data, and based on the algorithm, writes the corresponding results to the target address via the internal DMA or stores them in the module's internal registers.

13.1.2 AES Function Description

13.1.2.1 Symmetric Encryption Algorithm

The symmetric encryption algorithms primarily include AES and SM4 , where AES is further categorized into AES128 , AES192 , and AES256 based on the length of the Key . The SM4 algorithm is a national secret symmetric encryption algorithm. In terms of strength, a longer key length results in higher security; however, the time required for encryption and decryption also increases. The SM4 algorithm requires the most time.

13.1.2.2 Symmetric Encryption Mode

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

ECB Mode: The ECB mode encrypts and decrypts plaintext data directly using the KEY , processing data in groups of 16 bytes . Each encryption and decryption operation is performed on the entire 16 bytes. The advantage of this mode is that each group of data is independent, allowing for parallel computation and supporting random read and write operations by groups. The disadvantage is that if the plaintext of different groups is identical, the ciphertext will also be identical, making it susceptible to attacks.

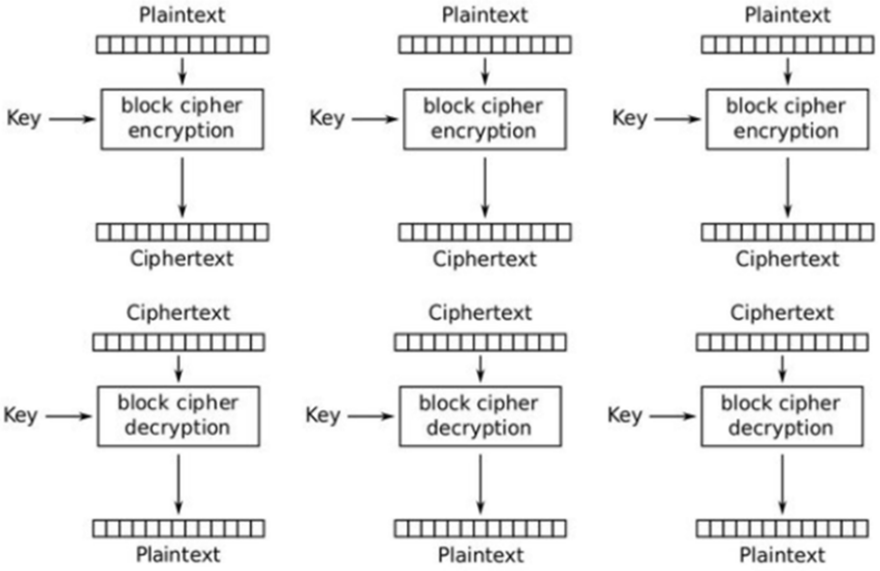


Figure 13-1: ECB Mode Decryption

CTR Mode: The CTR mode encrypts a vector composed of a NONCE and a COUNTER using a KEY , and then XORs the encrypted result with the plaintext data to produce the ciphertext, thereby achieving encryption. During decryption, the encrypted result of the same vector is XORed with the ciphertext to retrieve the plaintext data, thus completing the decryption process. In practical applications, the NONCE is typically represented by a constant, while the COUNTER uses the address of the data, ensuring that each data group has a different vector, which in turn makes the XOR data used in the encryption and decryption processes distinct. Understanding the composition of the vector used in CTR , the encryption and decryption of each data group are independent and can be computed in parallel, similar to the ECB mode. Additionally, this mode only involves encryption and XOR operations, making its structure relatively simple. The operations on the vector during encryption and decryption are independent of the data, which is why this mode is commonly used for encrypting and decrypting externally stored data.

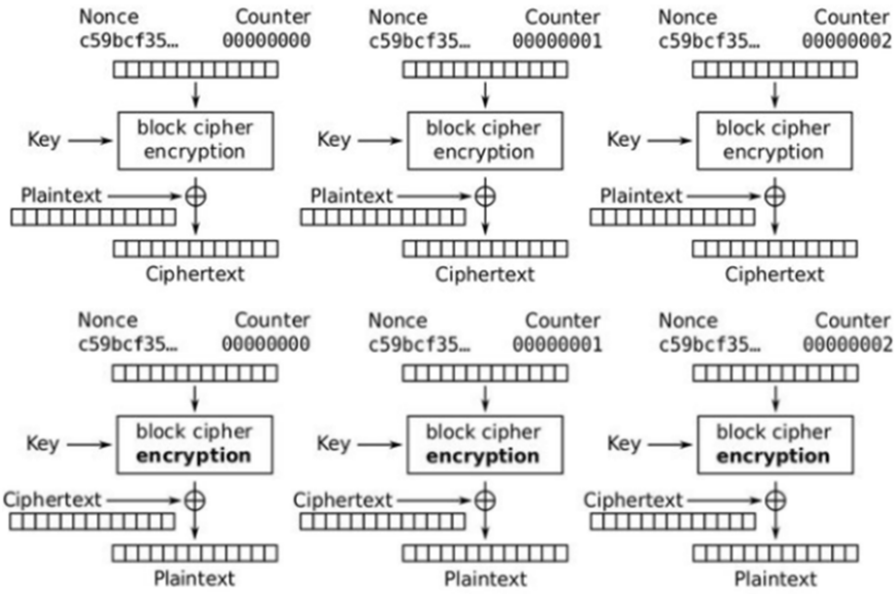


Figure 13-2: CTR Mode Decryption

CBC Mode: The CBC mode utilizes the ciphertext of the previous block as the initialization vector, which is then XORed with the current block before being encrypted with KEY to produce the ciphertext. During decryption, the ciphertext is decrypted using KEY and then XORed with the previous block's ciphertext to generate the plaintext. This mode can perform both encryption and decryption, and the last block of ciphertext can also serve as the MAC value for integrity verifcation. Due to the interrelation of each block of data during the encryption and decryption process, it provides enhanced security; however, it cannot execute parallel computations and does not allow for random read or write operations on individual blocks of data.

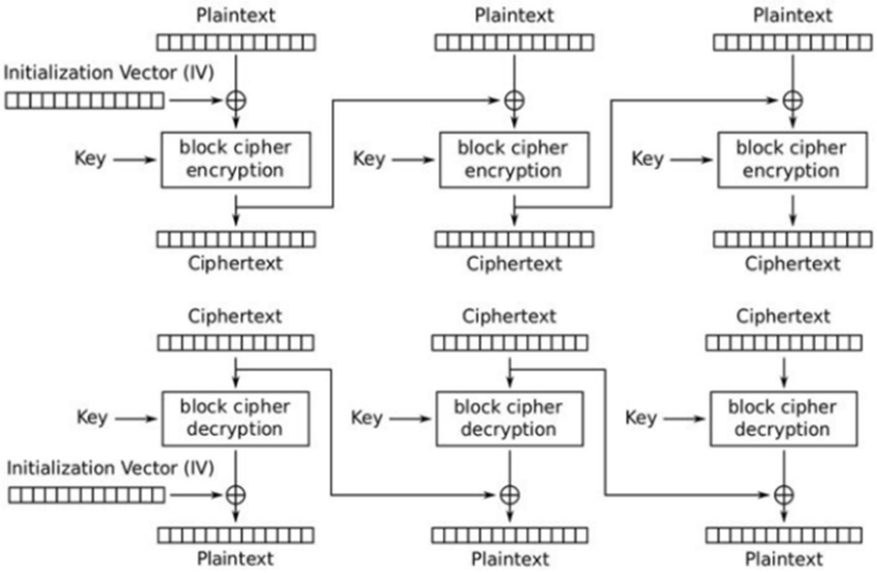


Figure 13-3: CBC Mode Decryption

13.1.2.3 Multiple Calls for Symmetric Encryption and Decryption

At times, it is necessary to perform encryption and decryption operations on large volumes of data. However, due to limitations in storage capacity or transmission rates, the data may need to be processed in multiple batches. When repeatedly invoking the symmetric encryption and decryption module for continuous processing of a large segment of data, it is crucial to consider the preservation and maintenance of the data context between calls.

ECB Mode:

The ECB mode conducts separate encryption and decryption operations for each group of 16 bytes of data. For the data to be encrypted or decrypted in a single operation, it must be ensured that the data volume is a multiple of 16 bytes. Any excess data can be merged and processed once the next batch is ready.

CTR Mode:

The CTR mode assigns a corresponding NONCE andCOUNTER value for each group of 16 bytes of data. Typically, theNONCE is a constant, while theCOUNTER serves as the identifer for the current data group, incrementing by 1 for each operation. For the data to be encrypted or decrypted in a single operation, the upper-level software must ensure that the data volume is an integer multiple of 16 bytes , while recording the COUNTER value based on the data volume. In the next call, the accumulated COUNTER value will be used as the initial vector input to the corresponding IV Register. Any excess data can be processed once the next batch is ready.

CBC Mode:

In CBC mode, the initial vector for each group of data is derived from the ciphertext of the previous group. For single encryption and decryption operations, the upper-level software must ensure that the data volume is an integer multiple of 16 bytes , and it is necessary to record the ciphertext of the last group of data from the current operation. This will serve as the initial vector input to the corresponding IV Register in the next operation. Any excess data will be processed once the next batch is ready.

13.1.2.4 Hash Algorithm

Hash algorithms include SHA1 , SHA224 , SHA256 , and SM3 , each with different digest lengths. The digest length for SHA1 is 160 bits , for SHA224 is 224 bits , and both SHA256 and SM3 are 256 bits . In terms of collision resistance, a longer digest results in a lower probability of collision.

13.1.2.5 Multiple Calls for Hash Value Calculation

At times, it is necessary to compute hash values for large datasets. However, due to limitations in storage capacity or transmission rates, the data may need to be divided into multiple batches for processing. When multiple calls to the hash value calculation module are required to handle a large segment of data, the following points should be considered.

First, except for the last processing instance, the amount of data processed each time must be a multiple of 4 bytes . Any excess data from a single processing instance must be saved and merged with the next batch of data for processing.

Second, for all hashing algorithms, before each calculation, the intermediate results of the previous hash calculation H0~H7 must be written into the corresponding registers. The hash value calculation module should then be configured to use the external initial values of H0 H7 and loaded into the module.

Third, each time HASH is called, the HASH LEN RESULT from the previous call must be updated to the current HASH LEN register and loaded into the module.

Fourth, except for the last call, the hash value calculation module should be configured not to use padding.

13.1.3 AES Register

Table 13-1: AES Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**COMMAND**	
[31:5]			RSVD	
[4]	rw	1'h0	AUTO_GATE	auto clock gating
[3]	rw	1'h0	HASH_RESET	HASH_ACC soft reset, 1'h1: reset the HASH_ACC block
[2]	w1t	1'h0	HASH_START	write 1 to trigger the HASH_ACC block
[1]	rw	1'h0	AES_ACC_RESET	AES_ACC soft reset, 1'h1: reset the AES_ACC block
[0]	w1t	1'h0	START	write 1 to trigger the AES_ACC block
0x04			**STATUS**	
[31:3]			RSVD	
[2]	r	1'h0	HASH_BUSY	HASH_ACC block is busy
[1]	r	1'h0	FLASH_KEY_VALID	flash key valid indicator
[0]	r	1'h0	BUSY	AES_ACC block is busy
0x08			**IRQ**	
[31:22]			RSVD	
[21]	rw1c	1'h0	HASH_PAD_ERR_RAW_STAT	HASH_ACC padding error raw status
[20]	rw1c	1'h0	HASH_BUS_ERR_RAW_STAT	HASH_ACC bus error raw status

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[19]	rw1c	1'h0	HASH_DONE_RAW_STAT	HASH_ACC done raw status
[18]	rw1c	1'h0	SETUP_ERR_RAW_STAT	AES_ACC setup error raw status
[17]	rw1c	1'h0	BUS_ERR_RAW_STAT	AES_ACC bus error raw status
[16]	rw1c	1'h0	DONE_RAW_STAT	AES_ACC done raw status
[15:6]			RSVD	
[5]	rw1c	1'h0	HASH_PAD_ERR_STAT	HASH_ACC padding error status
[4]	rw1c	1'h0	HASH_BUS_ERR_STAT	HASH_ACC bus error status
[3]	rw1c	1'h0	HASH_DONE_STAT	HASH_ACC done status
[2]	rw1c	1'h0	SETUP_ERR_STAT	AES_ACC setup error status
[1]	rw1c	1'h0	BUS_ERR_STAT	AES_ACC bus error status
[0]	rw1c	1'h0	DONE_STAT	AES_ACC done status
0x0C			**SETTING**	
[31:6]			RSVD	
[5]	rw	1'h0	HASH_PAD_ERR_MASK	HASH_ACC padding error interrupt mask, 0: mask the interrupt
[4]	rw	1'h0	HASH_BUS_ERR_MASK	HASH_ACC bus error interrpt mask, 0: mask the interrupt
[3]	rw	1'h0	HASH_DONE_MASK	HASH_ACC done interrupt mask, 0: mask the interrupt
[2]	rw	1'h0	SETUP_ERR_IRQ_MASK	AES_ACC setup error interrupt mask, 0: mask the interrupt
[1]	rw	1'h0	BUS_ERR_IRQ_MASK	AES_ACC bus error interrupt mask, 0: mask the interrupt
[0]	rw	1'h0	DONE_IRQ_MASK	AES_ACC done interrupt mask, 0: mask the interrupt
0x10			**AES_SETTING**	
[31:9]			RSVD	
[8]	rw	1'h0	AES_BYPASS	1'h0: normal operation 1'h1: bypass
[7]	rw	1'h0	AES_OP_MODE	1'h0: decryption 1'h1: encryption
[6]	rw	1'h0	ALGO_STANDARD	1'h0: AES 1'h1: SM4
[5]	rw	1'h0	KEY_SEL	1'h0: select key from AES_ACC key registers 1'h1: use internal root key
[4:3]	rw	2'h0	AES_LENGTH	AES Length: 2'h0: 128-bit 2'h1: 192-bit 2'h2: 256-bit 2'h3: Reserved
[2:0]	rw	3'h0	AES_MODE	AES Mode: 3'h0: ECB 3'h1: CTR 3'h2: CBC Others: Reserved
0x14			**DMA_IN**	
[31:0]	rw	32'h0	ADDR	AES_ACC input data address
0x18			**DMA_OUT**	
[31:0]	rw	32'h0	ADDR	AES_ACC output data address
0x1C			**DMA_DATA**	
[31:28]			RSVD	
[27:0]	rw	28'h0	SIZE	AES_ACC data block size, AES_ACC only support block aligned transaction. Each block contains 16 bytes.
0x20			**IV_W0**	
[31:0]	rw	32'h0	DATA	Initial Vector Word0
0x24			**IV_W1**	
[31:0]	rw	32'h0	DATA	Initial Vector Word1
0x28			**IV_W2**	

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	DATA	Initial Vector Word2
0x2C			**IV_W3**	
[31:0]	rw	32'h0	DATA	Initial Vector Word3
0x30			**EXT_KEY_W0**	
[31:0]	rw	32'h0	DATA	External Key Word0
0x34			**EXT_KEY_W1**	
[31:0]	rw	32'h0	DATA	External Key Word1
0x38			**EXT_KEY_W2**	
[31:0]	rw	32'h0	DATA	External Key Word2
0x3c			**EXT_KEY_W3**	
[31:0]	rw	32'h0	DATA	External Key Word3
0x40			**EXT_KEY_W4**	
[31:0]	rw	32'h0	DATA	External Key Word4
0x44			**EXT_KEY_W5**	
[31:0]	rw	32'h0	DATA	External Key Word5
0x48			**EXT_KEY_W6**	
[31:0]	rw	32'h0	DATA	External Key Word6
0x4C			**EXT_KEY_W7**	
[31:0]	rw	32'h0	DATA	External Key Word7
0x50			**HASH_SETTING**	
[31:9]			RSVD	
[8]	w1t	1'h0	HASH_LEN_LOAD	write 1 to load hash length
[7]	w1t	1'h0	HASH_IV_LOAD	write 1 to load hash iv
[6]	rw	1'h0	RESULT_ENDIAN	hash result endian setting: 1'h0: little endian 1'h1: big endian
[5]	rw	1'h0	DFT_IV_SEL	HASH default iv select. 1'h0: default iv according to hash mode 1'h1: default iv from HASH_IV_H* registers
[4]	rw	1'h0	BYTE_SWAP	HASH byte swap option. Set 1 to swap byte order when read data from memory.
[3]	rw	1'h0	DO_PADDING	HASH padding enable. Set 1 to do padding after data transfer.
[2:0]	rw	3'h0	HASH_MODE	HASH Mode: 3'h0: SHA-1 3'h1: SHA-224 3'h2: SHA-256 3'h3: SM3 Others: Reserved
0x54			**HASH_DMA_IN**	
[31:0]	rw	32'h0	ADDR	input data address
0x58			**HASH_DMA_DATA**	
[31:0]	rw	32'h0	SIZE	HASH input data byte size.
0x5C			**HASH_IV_H0**	
[31:0]	rw	32'h0	DATA	HASH IV H0
0x60			**HASH_IV_H1**	
[31:0]	rw	32'h0	DATA	HASH IV H1
0x64			**HASH_IV_H2**	
[31:0]	rw	32'h0	DATA	HASH IV H2
0x68			**HASH_IV_H3**	
[31:0]	rw	32'h0	DATA	HASH IV H3
0x6C			**HASH_IV_H4**	

Continued on the next page...

Table 13-1: AESRegister Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	rw	32'h0	DATA	HASH IV H4
0x70			**HASH_IV_H5**	
[31:0]	rw	32'h0	DATA	HASH IV H5
0x74			**HASH_IV_H6**	
[31:0]	rw	32'h0	DATA	HASH IV H6
0x78			**HASH_IV_H7**	
[31:0]	rw	32'h0	DATA	HASH IV H7
0x7C			**HASH_RESULT_H0**	
[31:0]	r	32'h0	DATA	HASH result H0
0x80			**HASH_RESULT_H1**	
[31:0]	r	32'h0	DATA	HASH result H1
0x84			**HASH_RESULT_H2**	
[31:0]	r	32'h0	DATA	HASH result H2
0x88			**HASH_RESULT_H3**	
[31:0]	r	32'h0	DATA	HASH result H3
0x8C			**HASH_RESULT_H4**	
[31:0]	r	32'h0	DATA	HASH result H4
0x90			**HASH_RESULT_H5**	
[31:0]	r	32'h0	DATA	HASH result H5
0x94			**HASH_RESULT_H6**	
[31:0]	r	32'h0	DATA	HASH result H6
0x98			**HASH_RESULT_H7**	
[31:0]	r	32'h0	DATA	HASH result H7
0x9C			**HASH_LEN_L**	
[31:0]	rw	32'h0	DATA	HASH load length l
0xA0			**HASH_LEN_H**	
[31:29]			RSVD	
[28:0]	rw	29'h0	DATA	HASH load length h
0xA4			**HASH_RESULT_LEN_L**	
[31:0]	r	32'h0	DATA	HASH result length l
0xA8			**HASH_RESULT_LEN_H**	
[31:29]			RSVD	
[28:0]	r	29'h0	DATA	HASH result length h

13.2 TRNG

TRNG locate in HPSYS。

13.2.1 Introduction

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

13.2.2 Module Architecture

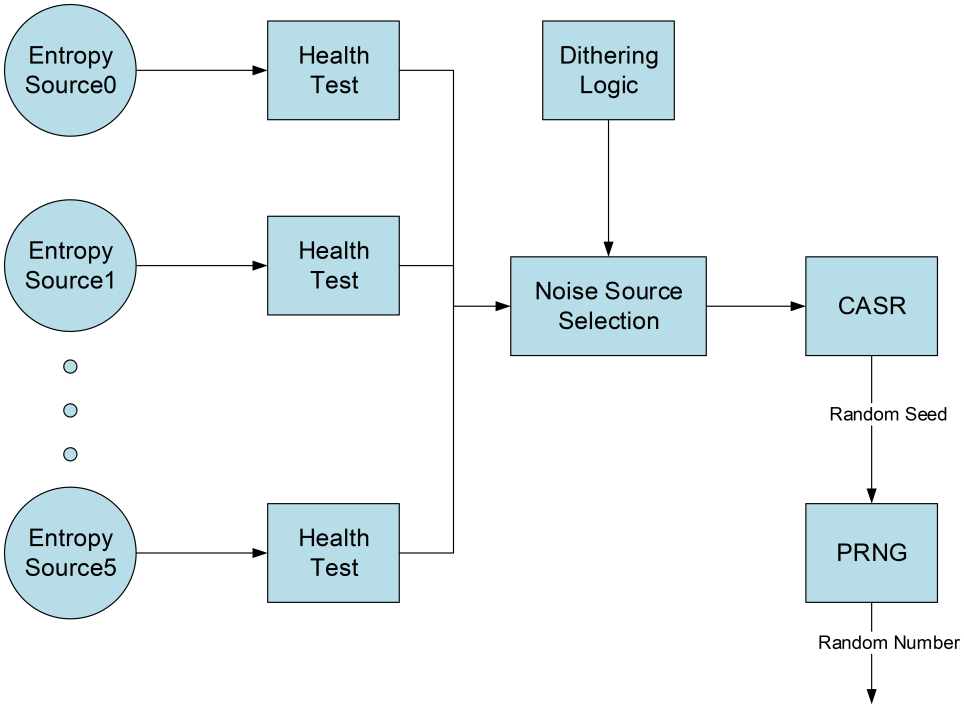


Figure 13-4: TRNG Schematic Diagram

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

13.2.3 Function Description

13.2.3.1 Entropy Source

The entropy source consists of six inverter chains. Each time a new random seed is generated, all inverter chains are activated, and a specific verifcation algorithm is employed to generate the random number seed. The generation time for the random number seed is relatively long, and it also consumes a signifcant amount of power. It is generally recommended to regenerate a new seed in two scenarios: first, when the original seed's lifecycle has expired, necessitating the periodic generation of new seeds to overwrite the old ones; and second, when the software requires a manual generation of a new seed under special circumstances.

The entropy source is activated each time a seed is generated. The VN verifer threshold is used to filter the entropy source; a higher threshold allows for a more lenient selection of the entropy source, resulting in faster seed generation, although the randomness may decrease.

Users can directly trigger gen_seed_startto activate the entropy source module for generating new random seeds.

13.2.3.2 Random Seed Generator

The random seed generator collects data from the entropy source and generates random seeds through the CASR (Cellular Automata Shift Register) module, which are provided for use by the next level PRNG. Users can also obtain random seeds

directly through the register. When the user configures use_ext_seed to 1, the next level PRNG module will not utilize the seeds generated by the random seed generator, but will instead use external seeds to generate random numbers.

13.2.3.3 Random Number Generator

The random number generator is a pseudo-random number generator based on existing random seeds, with a relatively short and fixed duration. It is important to note that the pseudo-random number generator has a probability of entering a self-locking state due to certain internal data sequences that can cause the linear feedback register to deadlock. When the prng_lockup state is detected, it is advisable to regenerate a new set of random seeds to produce new random numbers.

Users can trigger gen_rand_num_start to initiate the random number generator. If the current random seed has not been generated, the random number generator will first create a random seed and then generate random numbers.

13.2.3.4 Other Functional Modules

The TRNG module additionally provides measurement of the oscillation period of the inverter chain, which can be used to infer the process corner of the chip. The process begins by determining the number of the inverter chain used, as different inverter chains have varying lengths and corresponding oscillation periods. Next, the number of oscillation periods must be set, which corresponds to the frequency of the system pclk . Once the measurement is enabled, two counters will track the clock cycles of pclk and the clock cycles of the inverter chain, stopping when the number of clock cycles of pclk reaches the set threshold. At this point, the frequency of the inverter chain can be calculated using the formula Finv=Fpclk*Npclk/Ninv , based on the clock frequency Fpclk , the number of periods Npclk , and the number of periods of the inverter chain Ninv . This allows for inferring the process corner corresponding to the chip.

13.2.4 TRNG Register

Table 13-2: TRNG Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CTRL**	
[31:5]			RSVD	
[4]	rw	1'h0	gen_rand_num_suspend	Set 1 to suspend random number generation and update. Set 0 to recover the process.
[3]	rw	1'h0	gen_rand_num_stop	Set 1 to stop random number generation and update. This will reset the random number generation engine. After release the stop bit, user should write 1 to gen_rand_num_start to trigger the random number engine.
[2]	rw	1'h0	gen_seed_stop	Set 1 to stop random seed generation. This will reset the random seed generation engine. After release the stop bit, user should write 1 to gen_seed_start to trigger the random seed engine.
[1]	w1t	1'h0	gen_rand_num_start	write 1 to trigger the random number generation engine
[0]	w1t	1'h0	gen_seed_start	write 1 to trigger the random seed generation engine
0x04			**STAT**	
[31:4]			RSVD	
[3]	r	1'h0	rand_num_valid	random number valid flag
[2]	r	1'h0	rand_num_gen_busy	random number engine busy flag
[1]	r	1'h0	seed_valid	random seed valid flag
[0]	r	1'h0	seed_gen_busy	random seed engine busy flag
0x08			**CFG**	
[31:16]			RSVD	

Continued on the next page...

Table 13-2: TRNG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[15:8]	rw	8'ha	reject_threshold	random seed internal VN corrector check threshold
[7:2]			RSVD	
[1]	rw	1'h0	use_ext_seed	set 1 to use external seed to generate random number
[0]	rw	1'h0	auto_clock_enable	auto clock gating enable
0x0c			**IRQ**	
[31:19]			RSVD	
[18]	rw	1'h1	prng_lockup_msk	prng lockup interrupt mask
[17]	rw	1'h1	rand_num_avail_msk	random number available interrupt mask
[16]	rw	1'h1	seed_gen_done_msk	random seed generation done interrupt mask
[15:3]			RSVD	
[2]	rw1c	1'h0	prng_lockup	prng lockup raw interrupt
[1]	rw1c	1'h0	rand_num_avail	random number available raw interrupt
[0]	rw1c	1'h0	seed_gen_done	random seed generation done raw interrupt
0x10			**rand_seed0**	
[31:0]	rw	32'h0	val	random seed value0. If using external random seed, write value to this register will update the random seed in use.
0x14			**rand_seed1**	
[31:0]	rw	32'h0	val	random seed value1. If using external random seed, write value to this register will update the random seed in use.
0x18			**rand_seed2**	
[31:0]	rw	32'h0	val	random seed value2. If using external random seed, write value to this register will update the random seed in use.
0x1c			**rand_seed3**	
[31:0]	rw	32'h0	val	random seed value3. If using external random seed, write value to this register will update the random seed in use.
0x20			**rand_seed4**	
[31:0]	rw	32'h0	val	random seed value4. If using external random seed, write value to this register will update the random seed in use.
0x24			**rand_seed5**	
[31:0]	rw	32'h0	val	random seed value5. If using external random seed, write value to this register will update the random seed in use.
0x28			**rand_seed6**	
[31:0]	rw	32'h0	val	random seed value6. If using external random seed, write value to this register will update the random seed in use.
0x2c			**rand_seed7**	
[31:0]	rw	32'h0	val	random seed value7. If using external random seed, write value to this register will update the random seed in use.
0x30			**rand_num0**	
[31:0]	r	32'h0	val	random number value0
0x34			**rand_num1**	
[31:0]	r	32'h0	val	random number value1
0x38			**rand_num2**	
[31:0]	r	32'h0	val	random number value2
0x3c			**rand_num3**	
[31:0]	r	32'h0	val	random number value3
0x40			**rand_num4**	
[31:0]	r	32'h0	val	random number value4
0x44			**rand_num5**	
[31:0]	r	32'h0	val	random number value5
0x48			**rand_num6**	
[31:0]	r	32'h0	val	random number value6
0x4c			**rand_num7**	

Continued on the next page...

Table 13-2: TRNG Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[31:0]	r	32'h0	val	random number value7
0x50			**cal_cfg**	
[31:16]	rw	16'hff	length	calibration length
[15:6]			RSVD	
[5]	r	1'h0	done	calibration done
[4]	rw	1'h0	enable	calibration enable
[3:1]	rw	3'h0	osc_clk_sel	osc clock select
[0]	rw	1'h0	osc_clk_force_on	osc force enable
0x54			**cal_result**	
[31:16]	r	16'h0	osc_cnt	osc clock calibration counter result
[15:0]	r	16'h0	pclk_cnt	pclk calibration counter result

13.3 efusec

efusec locate in HPSYS。

13.3.1 Introduction

efusec refers to efuse control, which can control the corresponding bits of the efuse through write operations. The information of the blown bit is read back as 1, and the stored information value can be retrieved through read operations.

13.3.2 Main Features

- Can control 4 blocks of 256-bit efuse units
- Can read/write 1024-bit information
- Can control the read/write operation of one efuseunit at a time
- Read/write operations are single bitserial
- Completion of reading/writing can generate an interrupt signal
- The functionality for read and write masking can be implemented
- Certain stored information is directly output through the module interface
- The default value of efuse is all 0

13.3.3 Write Operation

The write operation of efusec is a programming operation for efuse , where the data written is a 1 bit. This bit is set to an open circuit by means of fusing, resulting in a read value of 1 . The information to be written is configured through registers PGM_DATA0~7 , with four units sharing 8 PGM_DATA* registers. The control flow of the write operation is as follows:

- Configure the BANKSEL register to select the efuse unit to be written
- Configure the MODE register to select the write operation
- Configure the IE register to enable interrupt status.
- Configure the PGM_DATA* register to write programming data.
- Configure the TIMER register to set the critical duration.
- Configure the EN register to initiate the write operation.

- Wait for the interrupt to arrive or query the SR register to obtain the write completion status.

The blown bit cannot be restored to a 0 state, while the unblown bit can be blown again through a write operation.

13.3.4 Read Operation

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

- Configure the BANKSEL register to select the efuse unit for reading
- Configure the MODE register to select the read operation
- Configure the IE register to enable interrupt status.
- Configure the TIMER register to set the critical duration.
- Configure the EN register to initiate the write operation.
- Wait for an interrupt to arrive or query the SRregister to obtain the read completion status
- Read the BANK*_DATA* register to obtain the read value
- A portion of the read value is directly transmitted through the interface signal

13.3.4.1 Read and Write Timing Control

The read and write timing of efuse has specific requirements, particularly regarding the fuse time. When the working clock of the efusec module changes, the clock count value must be adjusted to meet the read and write timing requirements of efuse . The timing control points that need to be addressed are as follows:

- TCKHP : Single bitfuse time, 10us
- THPCK : Hold time for enabling write, >20 ns
- THRCK : Hold time for enabling read, >500 ns

These three times can be configured based on the working clock count value through the TIMEregister. The reset value of the register corresponds to the count value at a 48 MHzworking clock.

13.3.4.2 Read/Write Masking Function

To prevent malicious read and write operations, after configuring the data and ceasing data updates, the read and write functions of the efuse modules in banks 1 to 3 can be masked by controlling the fxed bit of bank0. Once masked, writing to the corresponding efuse is not possible, and the read values cannot be updated.

- bank0[255:254]=2’b11, Mask bank3write function;
- bank0[253:252]=2’b11, Mask bank3read function;
- bank0[251:250]=2’b11, Mask bank2write function;
- bank0[249:248]=2’b11, Mask bank2read function;
- bank0[247:246]=2’b11, Mask bank1write function;
- bank0[245:244]=2’b11, Mask bank1read function;

13.3.4.3 Module Interface Output Signals

Some values of efuse will be output through the module interface signals, which are:

Table 13-3: efuse interface signals

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

13.3.5 efusec Register

Table 13-4: efusec Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31:5]			RSVD	
[4]	rw	1'h0	IE	Interrupt enable
[3:2]	rw	2'h0	BANKSEL	Bank select
[1]	rw	1'h0	MODE	0 - READ, 1 - PGM
[0]	w1s	1'h0	EN	Write 1 to enable PGM/READ. Self clear
0x04			**TIMR**	Timer Register
[31:21]			RSVD	
[20:10]	rw	11'h78	TCKHP	SCLK high period for PGM. Recommended value ~10us
[9:7]	rw	3'h0	THPCK	SCLK to CSB hold time into PGM mode. Recommended value > 20ns
[6:0]	rw	7'h6	THRCK	SCLK to CSB hold time into READ mode. Recmmended value > 500ns
0x08			**SR**	Status Register
[31:1]			RSVD	
[0]	rw1c	1'h0	DONE	Indicates PGM/READ done. Write 1 to clear
0x0C			**RSVDR**	Reserved Register
0x0C			RSVDR	
[31:0]			RSVD	
0x10			**PGM_DATA0**	Program Data0
[31:0]	rw	32'h0	DATA	
0x14			**PGM_DATA1**	Program Data1
[31:0]	rw	32'h0	DATA	
0x18			**PGM_DATA2**	Program Data2
[31:0]	rw	32'h0	DATA	
0x1C			**PGM_DATA3**	Program Data3
[31:0]	rw	32'h0	DATA	
0x20			**PGM_DATA4**	Program Data4
[31:0]	rw	32'h0	DATA	
0x24			**PGM_DATA5**	Program Data5
[31:0]	rw	32'h0	DATA	
0x28			**PGM_DATA6**	Program Data6
[31:0]	rw	32'h0	DATA	
0x2C			**PGM_DATA7**	Program Data7
[31:0]	rw	32'h0	DATA	
0x30			**BANK0_DATA0**	Bank0 Data0
[31:0]	r	32'h0	DATA	
0x34			**BANK0_DATA1**	Bank0 Data1
[31:0]	r	32'h0	DATA	
0x38			**BANK0_DATA2**	Bank0 Data2
[31:0]	r	32'h0	DATA	

Continued on the next page...

Table 13-4: efusec Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x3C			**BANK0_DATA3**	Bank0 Data3
[31:0]	r	32'h0	DATA	
0x40			**BANK0_DATA4**	Bank0 Data4
[31:0]	r	32'h0	DATA	
0x44			**BANK0_DATA5**	Bank0 Data5
[31:0]	r	32'h0	DATA	
0x48			**BANK0_DATA6**	Bank0 Data6
[31:0]	r	32'h0	DATA	
0x4C			**BANK0_DATA7**	Bank0 Data7
[31:0]	r	32'h0	DATA	
0x50			**BANK1_DATA0**	Bank1 Data0
[31:0]	r	32'h0	DATA	
0x54			**BANK1_DATA1**	Bank1 Data1
[31:0]	r	32'h0	DATA	
0x58			**BANK1_DATA2**	Bank1 Data2
[31:0]	r	32'h0	DATA	
0x5C			**BANK1_DATA3**	Bank1 Data3
[31:0]	r	32'h0	DATA	
0x60			**BANK1_DATA4**	Bank1 Data4
[31:0]	r	32'h0	DATA	
0x64			**BANK1_DATA5**	Bank1 Data5
[31:0]	r	32'h0	DATA	
0x68			**BANK1_DATA6**	Bank1 Data6
[31:0]	r	32'h0	DATA	
0x6C			**BANK1_DATA7**	Bank1 Data7
[31:0]	r	32'h0	DATA	
0x70			**BANK2_DATA0**	Bank2 Data0
[31:0]	r	32'h0	DATA	
0x74			**BANK2_DATA1**	Bank2 Data1
[31:0]	r	32'h0	DATA	
0x78			**BANK2_DATA2**	Bank2 Data2
[31:0]	r	32'h0	DATA	
0x7C			**BANK2_DATA3**	Bank2 Data3
[31:0]	r	32'h0	DATA	
0x80			**BANK2_DATA4**	Bank2 Data4
[31:0]	r	32'h0	DATA	
0x84			**BANK2_DATA5**	Bank2 Data5
[31:0]	r	32'h0	DATA	
0x88			**BANK2_DATA6**	Bank2 Data6
[31:0]	r	32'h0	DATA	
0x8C			**BANK2_DATA7**	Bank2 Data7
[31:0]	r	32'h0	DATA	
0x90			**BANK3_DATA0**	Bank3 Data0
[31:0]	r	32'h0	DATA	
0x94			**BANK3_DATA1**	Bank3 Data1
[31:0]	r	32'h0	DATA	
0x98			**BANK3_DATA2**	Bank3 Data2
[31:0]	r	32'h0	DATA	
0x9C			**BANK3_DATA3**	Bank3 Data3
[31:0]	r	32'h0	DATA	
0xA0			**BANK3_DATA4**	Bank3 Data4
[31:0]	r	32'h0	DATA	

Continued on the next page...

Table 13-4: efusec Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0xA4			**BANK3_DATA5**	Bank3 Data5
[31:0]	r	32'h0	DATA	
0xA8			**BANK3_DATA6**	Bank3 Data6
[31:0]	r	32'h0	DATA	
0xAC			**BANK3_DATA7**	Bank3 Data7
[31:0]	r	32'h0	DATA	
0xB0			**ANACR**	Bank3 Data7
[31:24]	r	8'h0	RESERVE1	
[23:16]	rw	8'h0	RESERVE0	
[15:11]			RSVD	
[10:8]	rw	3'h0	LDO_DC_TR	
[7:5]			RSVD	
[4]	rw	1'h0	LDO_MODE	
[3:1]	rw	3'h4	LDO_VREF_SEL	
[0]	rw	1'h0	LDO_EN	
0xB4			**DB_SEL**	debug signal select
[31:0]	rw	32'h0	db_sel	debug signal select

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 supports up to 8 channels; each channel can select a master or slave device for monitoring, count the number of AHB read/write transmissions within a specifed address range, and trigger a PTC event upon reaching a specifed count.

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

The chip contains a total of 2 BUSMON units. BUSMON1 has 8 channels and can monitor the AHB devices of HPSYS. BUSMON2 has 4 channels capable of monitoring LPSYS AHB devices.

13.4.2 Main Features

- Up to 8independently configured 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.
- Each of the 8 channels independently generates PTC triggers

13.4.3 Monitoring Device Selection

Each channel selects the monitored master or slave device via CRx.SEL. When the master device is selected, the channel monitors the AHB transfer requests issued by the master device. When the slave device is selected, the channel monitors the AHB transfer requests received by the slave device.

Table 13-5: BUSMON1 Select to monitor the 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	LP2HP	DMAC1	EXTDMA	NNACC1_C	EZIP1	EPIC_A	EPIC_B	LCDC1
3	FACC1	NNACC1_B	FFT1	SDMMC1	NNACC1_A	USBC	AES	PTC1
4	/	/	/	/	/	/	/	/
5	/	/	/	/	/	/	/	/
6	RAM0	RAM1	RAM2	RAM3	MPI1	MPI2	HP_APB	HP_AHB
7	HP2LP	MPI3	/	/	/	/	/	/

Table 13-6: BUSMON2 Select to monitor the HPSYS device

CRx.SEL	CH1	CH2	CH3	CH4
0	LCPU_C	LCPU_C	LCPU_C	LCPU_C
1	LCPU_S	LCPU_S	LCPU_S	LCPU_S
2	DMAC2	HP2LP	PTC2	/
3	/	/	/	/
4	/	/	/	/
5	MPI5	RAM0	RAM1	RAM2
6	TCM	LP_APB	LP_AHB	PHY_DUMP
7	/	/	/	LP2HP

13.4.4 Monitoring Address Range

Each channel can independently configure the monitored address range via registers. The monitoring address range of channel x is [MINx, MAXx), where the MIN address is included and the MAX address is excluded. For example, when MIN1=0x60010000 and MAX1=0x60020000, channel 1 will monitor addresses from 0x60010000 to 0x6001FFFF of AHB transfers, but will not count AHB transfers at address 0x60020000.

13.4.5 Monitoring Transfer Types

Each channel can configure the monitored transfer type 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 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 channel compare enable CRx.CMPEN is set to 0 , when the channel counter value equals 0x7fffffff , the overflow flag CNTx. of bit is set to 1 and a PTC event trigger is generated. Upon overflow, if 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, CNTx can be queried directly to acquire the real-time count value of the channel monitoring; alternatively, 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 for starting 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 interrupt in the PTC module, 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
[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

Continued on next page...

Table 13-7: BUSMON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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
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

Continued on next page...

Table 13-7: **BUSMON Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
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
[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

Continued on next page...

Table 13-7: **BUSMON Register Map (continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[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
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

Continued on next page...

Table 13-7: BUSMON Register Map (continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[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 SDMMC1

The chip contains 2 SDMMC units, both located in HPSYS. It should be noted that the usage methods of the two SDMMC units differ, and the drivers invoked are also different. Among them, SDMMC1 has built-in DMA and supports DDR and 8-line mode; SDMMC2 does not support DDR or 8-line mode, but can send requests to DMAC1.

14.1.1 Introduction

SDMMC supports SD protocol 3.0 and eMMC standard 4.51,acting as a HOSTcontroller to interact with SD/SDIO/eMMC devices .SDMMC built-in chained DMA controller and 1Kbyte FIFO,enabling autonomous data read and write, supporting block data transfer.SDMMCsupports SDRsingle-line, 4 single-line and 8-line modes, and supports DDR 4-line and 8-line modes

14.1.2 Main Features

- Compatible with SD Host Controller Standard Specification Version 3.0
- Compatible with SD 3.0 Physical Layer Specification Version 3.01
- Compatible with SDIO Specification Version 3.0
- Compatible with JEDEC JESD84-B451 eMMC 4.51 Specification
- SupportsSDSC/SDHC/SDXC/SDHScards
- Supports UHS-I:SDR12/SDR25/SDR50/SDR104/DDR50
- Supports SDR single-line, 4-line, 8-line modes
- Supports DDR 4-line, 8-line modes
- Built-in 1K-byte FIFO, maximum support for single block of 512 bytes
- Configurable clock
- Built-in chained DMA

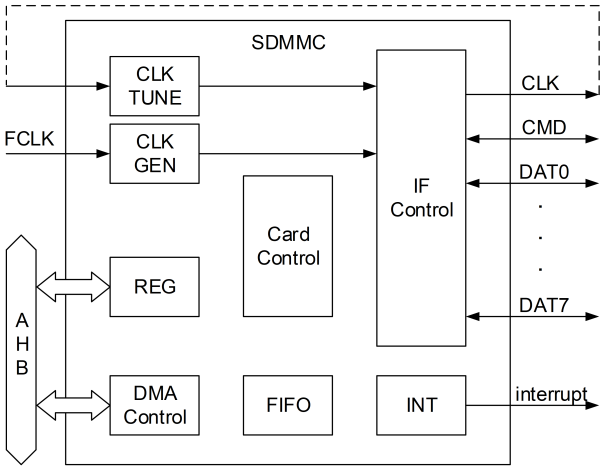


Figure 14-1: **SDMMC Block Diagram**

14.1.3 Functional Description

14.1.3.1 SD/eMMC Interface

The controller supports the SD/eMMC interface, which includes one CLK line, one CMD line, and 8 DAT lines (DAT0~DAT7). The CLK line is used to output the clock. The CMD line transmits CMD commands and response RSP packets according to the protocol. The DAT lines transmit data streams according to the protocol. The CMD line supports only single-edge (SDR) transmission, meaning CMD is valid only on the rising edge of the CLK. The DAT line supports single-edge (SDR) transmission, meaning DAT is valid only on the rising edge of the CLK; It also supports double-edge (DDR) transmission, meaning DAT is valid on both the rising and falling edges of the CLK.

14.1.3.2 Clock configuration

SDMMC has an independent functional clock FCLK, which can be selected from either the system clock or the DLLclock. The selection register isCSR_SEL_SDMMCin the RCCmodule. note that during SDMMC operation, the system clock frequency HCLK must not be lower than FCLK , and CR3_INTCLKEN must be set to 1.

The CLK output to the SD/eMMC interface is generated by dividing the FCLK frequency. The division ratio CLKDIV is a 10-bit value, 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, the CLK frequency = FCLK frequency / (2 × CLKDIV). For example, when FCLK is 192M, to output a 400KHz CLK, the CLKCR_DIV must be configured to 240. During DDR transmission, the CLKDIV should be configured as an even number.

CLK output is controlled by CR3_CLKEN. Setting it to 1 enables continuous clock output, while setting it to 0 disables the output. When the clock frequency is adjusted, the CLK output should 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.

Table 14-1: Command Packet Format

Bit position	47	46	[45:0]	[39:8]	[7:1]	0
Width(bits)	1	1	6	32	7	1
Value	0	1	x	x	x	1
Description	Start bit	Transimission bit	Command index	Argument	CRC7	End bit

Each command is distinguished by a 6bit command index representing different functions and carries a 32bit parameter. The device uses a 7bit CRC to verify the correctness of the received command and sends a response when required.

Based on the 6bit command index, commands are typically named CMDn, where n is the command index configured by the CR1_CMDIDX register.

The 32-bit parameter (argument) of the command 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 48-bit response; Certain commands, such as CMD2, require receiving a 136-bit response. Before

sending a command, the expected response type must be selected by configuring the CR1_RSPTYPE register.

Some responses contain the command index and CRC checksum. To verify the correctness of the response content, CR1_CHECKIDX and CR1_CHECKCRC can be configured as needed.

Certain commands, such as CMD18 and CMD24, require data read/write after transmission, and CR1_DATAPRESENT must be set to 1.

Command transmission is triggered by writing to CR1_CMDIDX; therefore, it is recommended to prepare all command parameters and write them to the CR1 register in a single operation to 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.

At the end 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 an interrupt can be generated when IER_TCIE is 1.

Recommended software procedure for command transmission:

1. Configure the index, argument, and expected response type once, then initiate command transmission.
2. Wait for the command complete interrupt and the transfer complete interrupt (if the response includes a busy signal).
3. Check the returned response and obtain the argument

14.1.3.4 Data Transmission

Data transmission includes reading data from the device and writing data to the device, initiated by the controller sending specific commands (such as CMD 17, CMD24, etc.) . Data transmission ends after the preset length or is terminated by the controller sending a specific command (such as CMD12).

Data transmission defined by the SD and SDIO protocols can use single-line or 4-line modes. Data transmission defined by the eMMC protocol can use single-line, 4-line, or 8-line modes. The controller supports single-line, 4-line, or 8-line transmission, configured via the CR2_DATWIDTH and CR2_EXTWIDTH registers. In single-line mode, data is transmitted via DAT0, which also indicates 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 or 8-line DDR transmission. configure the CR4_UHSMODE register to 4 for DDR transmission; otherwise, SDR transmission is used.

Table 14-2: Data Transmission Mode configuration

Transfer 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
	8 line	x	1	4

Data is transmitted in blocks, with block size varying according to the device type. For SD and eMMC devices, the single-block data size is typically for 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 must be configured according to the device's single-block data byte count. 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 transfers, CR1_MULTIBLK should be set to 0; in this case, BSR_BCNT is ineffective. For multi-block transfers, CR1_MULTIBLK and CR1_BCNTEN should be set to 1, and BSR_BCNT should be configured to the total number of blocks to be transferred.

The direction of data transmission is configured by CR1_DIR; 1 indicates read, and 0 indicates write.

After data transmission configuration is completed, when the controller sends a read/write command (command configured by CR1_DATA PRESENT set to 1),data transmission will automatically start.

Data transmission 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 device is also temporarily stored in the FIFO, then retrieved by the CPU or 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 end command (for example, CMD12). The controller supports automatic transmission of the end 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 set to 1.
2. Transmission errors include data timeout (ISR_DTOERR) , data CRC error (ISR_DCRCERR) , and data end bit error (DE-BERR). Each error type can generate a corresponding interrupt. The data timeout duration is configured byCR3_DTOCNT, with the calculation formula: Timeout =FCLK cycle*4096*2^{CR3_DTOCNT}, whereCR3_DTOCNT is less than 15. For example, when FCLK is 48 MHz,CR3_DTOCNT is 13, the timeout duration is approximately 699 ms. When a transmission error occurs, the normal completion interrupt ISR_TC may not be generated; therefore, both the normal completion interrupt and the error interrupt must be monitored.

Recommended software procedure for data transmission:

1. Configure transmission direction, bus 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 ISER can be recorded, and whether the recorded events generate interrupts is configured via the IER register. Therefore, when enabling interrupts, the corresponding event bits in both ISER and IER must be set to 1 .

14.1.3.6 FIFO Management

The SDMMC features a built-in 1KB FIFO for buffering read and write data. The FIFO operates in conjunction with the built-in DMA to perform data transfers. When writing data to the device, the controller initiates a single-block write operation only when the amount of data stored in the FIFO reaches or exceeds the size of one block; otherwise, it waits for the CPU or DMA to transfer data into the FIFO. During this waiting period, the interface CLK does not pause output. During the process of writing multiple blocks of data, each block write operation follows the aforementioned mechanism

When reading data from the device, the controller initiates a single-block read operation only when the remaining space in the FIFO is greater than or equal to the size of one block; otherwise, it waits for the CPU or DMA to transfer data out of the FIFO. During the waiting period, the interface CLK suspends output to prevent the device from transmitting data. During the process of 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. DMA mode is selected via CR2_DMAMODE, 0 for SDMA, 2 for ADMA.

In SDMA mode, the starting address for data transfer SAR must be configured, typically pointing to memory 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 contiguous address data and automatically pauses at specific memory address boundaries.

For convenient 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 equals 1), and can generate an interrupt (when IER_DMAIE equals 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 a 2KB data transfer starts from 0x20019c00, then after the transfer of 4 bytes of data at 0x20019ffc is completed,SDMA will automatically pause. At this point, the value

of the NSAR register is 0x2001a000. If a new ad dress 0x2001b000 is written to the SAR register at this time,SDMA will automatically resume transferring the remaining 1KB of data starting from 0x2001b000.

The ADMA mode requires the creation of a linked descriptor table before use. The starting address of the linked descriptor table shall 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 two types: transfer entries and jump entries. The next entry of a transfer entry must be stored sequentially with the transfer entry itself; that is, the starting address of the next entry equals the starting address of the transfer entry plus 8 bytes. The entry pointed to by a jump entry can be stored at any address aligned to 4 bytes, regardless of the storage location of the jump entry itself. The entry format is as follows.

Table 14-3: 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 word felds are as follows:：

VALID	For 1 indicates the current entry is valid; a value of 0will generate the error flag-ISR_ADMAERRand may trigger interrupt.
END	For 1 indicates that the current table entry is the fnal entry, with no subsequent entry. 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. ACT2 being 0indicates that the current table entry does not require execution and proceeds directly to the next entry. ACT2 being 1andACT1being 0indicates that the current table entry is a transfer entry, andADMArequires data transfer. ADDR indicates the starting address of the data. ACT2 being 1 andACT1 being 1 indicates that the current table entry is a jump entry, whereADMA does not require data transfer, and ADDR indicates the storage address of the next table entry.
LENGTH	When the current table entry is a transfer entry, it indicates the number of bytes to transfer, where 0 represents 65536 bytes. The number of transfer bytes shall be an integer multiple of the single block data byte count.
ADDR	When the current table entry is a transfer entry, it indicates the starting address of the data. When the current table entry is a jump entry, it indicates the storage address of the next table entry.

An example of an ADMA chain descriptor table is shown in the following fgure. Table entries ABCD are sequentially stored in the address space starting at address A , and table entries EF are sequentially stored in the address space starting at address E . Table entries ABCD are executed sequentially; when execution reaches table entry D , it jumps to table entry E for execution. Table entries EF are executed sequentially; when execution of table entry F is completed, the ADMA transfer ends

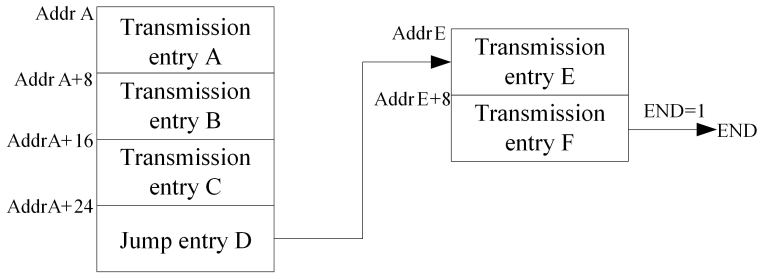


Figure 14-2: Example of an ADMA chain descriptor table

14.1.3.8 eMMC Open-drain Mode

When the eMMC device is in certain states (such as inactive , idle , or identifcation states; see the eMMC protocol for details), the CMD line needs to 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 CMDline must be configured as push-pull mode, i.e., CR2_PPMODE configured 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 3methods

1. Detection via a 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 the pin level. When no SD card is inserted, the pin level remains high. When the SD card is inserted into the slot, this pin is connected to ground, resulting in a low level. By detecting the GPIO input level of this IO, insertion and ejection of the SD card can be detected.
2. Detection is performed via the DAT3 pin. When the SD card is powered on, there is a 50K ohm pull-up resistor from the internal DAT3 line 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 IO level connected to DAT3. When no SD card is inserted, this pin level remains low. When the SD card is inserted into the slot, this pin is pulled up to a high level. Insertion and removal detection of the SD card can be achieved by monitoring the GPIO input level of this IO. When the SD card initiates data transmission, this detection should be disabled, 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 communicating with the device via SDMMC, the CMD and DAT signals from the slave device must be sampled on the edge of the CLK. Due to signal transmission delay, it may be necessary to adjust the sampling clock edge to ensure correct sampling.

SCR_SEL selects whether the controller uses an external loopback clock or an internally generated clock for sampling. The external loopback clock requires the IO to output. of CLK is looped back externally to the chip through another IO, enabling more accurate compensation for IOdelay at the cost of occupying an additional IO.

The SCR_DLY register can adjust the sampling clock edge, with a total of 64 levels; the larger the configured value, the greater the delay. When the CLK frequency exceeds 50M , it is often necessary to adjust the sampling clock to ensure correct interface functionality. When interacting with the eMMC device, the controller can configure different delays and send CMD21 along with the tuning block to identify the optimal sampling point.

Recommended software procedure for eMMC tuning:

1. Set the device to HS200 mode (CMD6).
2. The controller configures the CLK frequency to the target frequency and sets the target bus width.
3. SCR_DLY is configured to the initial value (0 or another empirical value).
4. The controller sends CMD21 to request the device to send the tuning block and receives the response.
5. The controller receives the tuning block data sent by the device.
6. Compare the tuning block data with the tuning block template and record the correctness of the data.

7. Set Increase SCR_DLY by 1 (maximum supported value is 63).
8. Repeat steps4 and 7several times
9. In the data correctness record after scanning, select the SCR_DLYat the midpoint of the continuous correctly received window as the optimal sampling value
10. Set configure SCR_DLY as the optimal sampling value and initiate subsequent data transmission

The above tuning process should be repeated periodically or whenever the operating state changes to track temperature or environmental variations.

14.1.4 SDMMC Register

Table 14-4: SDMMC Register map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**SAR**	System Address/Argument2 Register
[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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: **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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 14-4: 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 SDMMC2

14.2.1 Introduction

SDMMC supports SD protocol 3.0 and eMMC standard 4.51 and can function as a HOST controller to interact with SD/SDIO/eMMC devices, working in conjunction with DMAC for data read and write operations. SDMMC supports SDR single line and 4 line modes, but does not support DDR or SPI mode.

14.2.2 Main Features

- Compatible with SD Host Controller Standard Specification Version 3.0
- Compatible with SD 3.0 Physical Layer Specification Version 3.01
- Compatible with SDIO Specification Version 3.0
- Compatible with JEDEC JESD84-B451 eMMC 4.51 Specification
- SupportsSDSC/SDHC/SDXC/SDHScards
- SupportsSDR12/SDR25/SDR50
- Supports SDRsingle-line and4line modes
- Built-in 2K bytes FIFO, with a maximum support for a single block of 512 bytes
- Configurable clock
- Operates with DMACfor data transfer

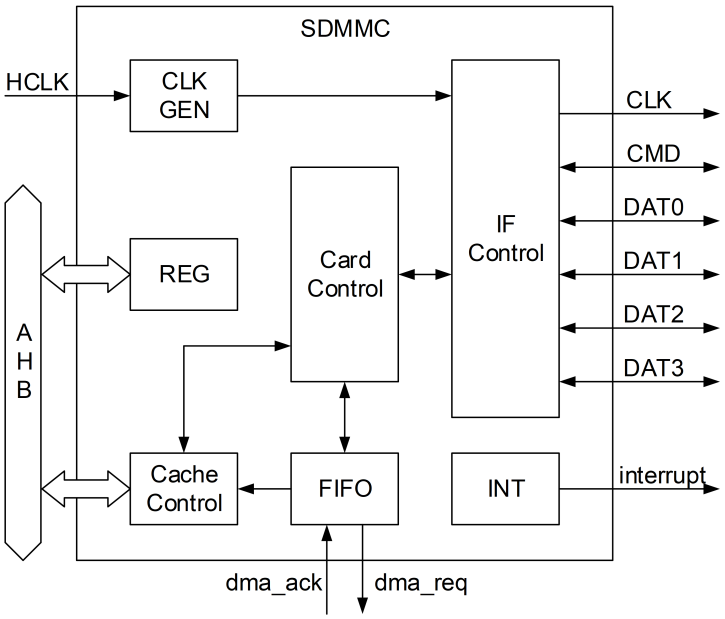


Figure 14-3: SDMMC Block Diagram

14.2.3 Function Description

14.2.3.1 SD/eMMC Interface

The controller supports a SD/eMMC interface that includes one CLK line, one CMD line, and 4 DAT lines (DAT0-DAT3). In single-line mode, only one DAT line is utilized(DAT0). The CLK line is used for clock output. The CMD line transmits CMD commands and responses RSP according to the protocol. The DAT lines transmit data streams in accordance with the protocol. This module only supports single-edge (SDR) transmission, with CMD and DAT being valid only on the rising edge of CLK.

14.2.3.2 Clock Settings

The controller operates at the system clock frequency, referred to as HCLK. The output to the SD/eMMC interface's CLK is generated by dividing the HCLK frequency, with the division ratio specifed by CLKCR_DIV. The division formula is CLK frequency = HCLK frequency / (CLKCR_DIV + 1). For example, when HCLK is 48M, to output a CLK of 400KHz, it is necessary to configure CLKCR_DIV to 119. The minimum supported setting for CLKCR_DIV is 1, which means the division ratio is at least 2.

The CLK output is controlled by CLKCR_STOP_CLK. When set to 0, the clock will continue to output; when set to 1, the output will be disabled.

14.2.3.3 Sending Commands

Commands are transmitted by the controller via the CMD line in a fxed-format packet received by SD/eMMC/SDIO devices, which is used to configure the status of SD/eMMC/SDIO devices and control data transmission. The length of the command packet is fxed at 48 bits.

Command Packet Format						
Bit position	47	46	[45:0]	[39:8]	[7:1]	0
Width(bits)	1	1	6	32	7	1
Value	0	1	x	x	x	1
Description	Start bit	Transimission bit	Command index	Argument	CRC7	End bit

Each command is distinguished by a 6-bit command number, accompanied by a 32-bit parameter. The device uses a 7-bit CRC to verify the correctness of the received command and sends a response when necessary.

Based on the 6-bit command number, commands are typically designated as CMDn, where nrepresents the command number configured by the CCR_CMD_INDEX Register.

The 32-bitparameter of the command is configured by the CARRegister.

Certain commands, such as CMD0 and CMD4 , do not require a response; others, such as CMD8 and CMD11 , require a 48-bit response; while commands like CMD2 require a 136-bit response. Before sending a command, it is essential to configure the CCR_CMD_HAS_RSP and CCR_CMD_LONG_RSP registers to select the expected response for the command.

Response Configuration				
Response	CMD Example	RSP Example	CCR_CMD_HAS_RSP	CCR_CMD_LONG_RSP
None	CMD0，CMD4	None	0	0
48bit	CMD8，CMD11	R1,R7	1	0
136bit	CMD2，CMD9	R2	1	1

Before the controller sends a command, it must complete the configuration of the command and set the CCR_CMD_TX_EN bit. Subsequently, it should set the CMD_START bit, and the controller will initiate the command sending process.

There are two ways to conclude the command sending process:：

1.Command completed; SR_CMD_DONEis asserted high, and an interrupt can be generated whenIER_CMD_DONE_MASKis set to 1.When the command is completed and does not require to receive a response, or when a response is received within the timeout period, regardless of whether the response's CRC is correct, the command will be completed. If a CRC error occurs during response reception, SR_CMD_RSP_CRC will be asserted high, and an interrupt can be generated when IER_CMD_RSP_CRC_MASK is set to 1.

2.Command timeout; SR_CMD_TIMEOUT is asserted high, and an interrupt can be generated when IER_CMD_TIMEOUT_MASK is set to 1. When the command is completed and requires a response, but no response is received within the timeout period configured in the TOR register (no response received for the start bit) , a timeout will occur. Once a command timeout occurs, the controller must be reset (via the RCC module) before sending the next command, and the configuration must be restored (only the SDMMC controller needs to be reconfigured; the state of interaction with the device does not need to change).

The command sending process can be checked via SR_CMD_BUSY to determine if it has completed. When the current command sending process is not fnished, the controller will not respond to new command sending requests. During debugging, if SR_CMD_BUSY remains low for an extended period, please verify whether the TOR settings are appropriate and check if a reset and reconfiguration of the controller were performed after the previous command timed out.

For certain commands, when expecting a response (such as R1b) , it is also necessary to check the busy status feedback from the device through the DAT line, which can be determined by querying the DSR register to see if busy has been

cleared.

The command index of the received response is stored in the RIR register, while the argument is stored in RAR1 and RAR4 , with the 48-bit response argument stored in RAR1.

Suggested process for command transmission:

1. Configure index, argument, and expected response type, then initiate the command transmission
2. Wait for either the command completion interrupt or the command timeout interrupt.
3. If the command completes, check the returned response and retrieve the argument.
4. If the command times out, reset the controller and reconfigure (primarily including clock settings and timeout duration, etc.).

14.2.3.4 Data Transmission

Data transmission encompasses reading data from the device and writing data to the device, initiated by the controller sending specific commands (such as CMD17, CMD24, etc.), concluding after a preset length of data transmission, or terminated by the controller sending a specific command (such as CMD12).

Data transmission defined by the SD and SDIO protocols can utilize either single-line or 4 line modes. The data transmission defined by the eMMC protocol can employ single-line, 4 line, or 8 line modes. This controller exclusively supports single-line or 4 line transmission, which is configured through the DCR_WIRE_MODE Register. In single-line mode, data is transmitted via DAT0, which also indicates the busy status of the data transmission. In 4 line mode, data is transmitted through DAT0 DAT3, with DAT0 also indicating the busy status of the data transmission.

Data is transmitted in blocks, with the amount of data per block varying according to the device type. For SD and eMMC devices, the typical block size is 512 bytes , but there are exceptions. Some devices allow the command (such as CMD16) to change the amount of data per block. Before initiating data transmission, the controller should configure the DCR_BLOCK_SIZE Register to the appropriate value (the number of bytes per block equals DCR_BLOCK_SIZE plus 1) . The controller must also configure the DLR_DATA_LEN Register to determine the total amount of data transmitted in a single operation (the total number of bytes equals DLR_DATA_LEN plus 1) . For single block transmission, DLR_DATA_LEN should equal DCR_BLOCK_SIZE . For multi-block transmission, (DLR_DATA_LEN+1) should be an integer multiple of (DCR_BLOCK_SIZE+1) .

The direction of data transmission is configured by DCR_R_WN. The DCR_TRAN_DATA_EN register enables data transmission and should remain high during the transmission process. DCR_DATA_START is used to initiate data transmission.

Data transmission must pass through the internal FIFO cache of the controller. The data to be sent is written to the FIFO by the CPU or DMA, after which the controller transfers the data from the FIFO to the device. Data read from the device is also temporarily stored in the FIFO and is then retrieved by the CPU or DMA. The FIFO register address space occupies a total of 512 bytes, and accessing any aligned address within it yields the same effect. If the FIFO overflows or underflows, it will generate SR_FIFO_OVERRUN or SR_FIFO_UNDERRUN records, which can trigger an interrupt

Data transmission may either complete normally or time out if data or CRC is not received within the specifed time. In either case, SR_DATA_DONE will be asserted high and can generate an interrupt. If a CRC error is detected while reading the device, or if a CRC error is returned by the device during a write operation, an SR_DATA_CRC record will be generated. If the data start bit is not detected within the timeout period configured by TOR while reading the device, or if the CRC response from the device is not received within the timeout period configured by TOR while writing, an SR_DATA_TIMEOUT record will be generated.

The status of data transmission completion can be queried through SR_DATA_BUSY. When the current data transmission is not complete, the controller will not respond to new data transmission requests.

Recommended process for data reading:

1.Configure the single block data amount and total data amount, then initiate data reading:
2. Configure DMACfor reading FIFO, and then start DMAC.
3. Send the read command (e.g. CMD17).
4. Wait for the command interrupt, then process the device response (e.g. R1).
5. Wait for the data transfer completion interrupt.
6. If it is a multi-block read, send the transfer end command (e.g. CMD12).
7. Wait for the DMAcompletion interrupt.

Recommended process for data writing:

1. Send the write command (e.g. CMD24).
2. Wait for the command interrupt, then process the device response (e.g. R1).
3. Configure the single block data amount and total data amount, then initiate data writing.
4. Configure DMACfor writing FIFO, and then start DMAC.
5. Waiting for the data transfer complete interrupt
6. If it is a multi-block write, send the transfer end command (such as CMD12)

14.2.3.5 Interrupt Generation

SDMMC can generate interrupts to notify the CPU of specific events. These events include command completion, command timeout, data completion, data timeout, and various errors. Events that can generate interrupts are configured through the IER register, and occurring events can be queried through the SR register.

14.2.3.6 FIFO Management

SDMMC features a built-in 2KB FIFO for caching read and write data. The FIFO can generate DMA requests based on the fullness of its contents, working with DMAC to complete data transfers. During multi-block data transfers, the bandwidth of DMAC typically meets the device's data transfer bandwidth, allowing for continuous data transfer. However, if the storage space at the other end of DMAC becomes blocked and does not respond to FIFO requests in a timely manner, it may cause FIFO to experience overflow or underflow, resulting in data transfer errors. To prevent this situation, the CLKCR_VOID_ FIFO_ERROR register can be configured so that when the contents of FIFO cannot meet the data transfer requirements, the interface CLK is automatically halted, and data transfer is stopped until the contents of FIFO meet the requirements. At this point, a pause in the CLK can be observed on the interface.

14.2.3.7 eMMC Open Drain Mode

When the eMMC device is in certain states (such as inactive , idle , or identification state; refer to the eMMC protocol for details), the CMD line must be configured as open drain mode by setting CDR_CMD_ODto 1.

When accessing SD/SDIO devices, and when the eMMC enters other states, the CMD line must be configured as push-pull mode, meaning CDR_CMD_OD should be set to 0 .

14.2.3.8 SDIO Interrupt

SDIO devices can generate SDIO interrupts to notify the controller by pulling down DAT1.

If the controller needs to respond to SDIO interrupts, in single-line mode, the CEATA_ENABLE_SDIO_IRQ Register must be set to 1 . In four-line mode, it is additionally required to set CEATA_SDIO_4WIRES_IRQ to 1 .

In four-line mode, interrupt requests during multi-block data transfer intervals for CMD53 are not supported (Block Gap Interrupt) . However, interrupt requests before a single CMD data transfer or after the transfer is completed can be responded to normally.

14.2.3.9 Card Detection

SD cards support hot swapping. The detection of card insertion and ejection is typically implemented in the following 3methods.

1.Through a dedicated CD (card detect) pin for detection. SD card slots that support insertion detection usually provide a dedicated card detection pin (CD) , which has a pull-up resistor connected to the power supply. The chip needs to allocate an independent IO to detect the level of this pin. When no SD card is inserted, the pin level is high. When the SD card is inserted into the slot, the pin connects to ground, and the level is low. By determining the GPIO input level of this IO , the insertion and ejection of the SD card can be detected.

2.Detection is performed through the DAT3 pin. When the SD card is powered on, there is a 50K ohm pull-up resistor from the internal DAT3 of the card to the power supply. The external circuit of the chip must include a weak pull-down resistor to ground on the DAT3 pin (typically greater than 470K ohm). The chip determines whether the card is inserted by detecting the IO level connected to DAT3. When the SD card is not inserted, the pin level is low. When the SD card is inserted into the slot, the pin is pulled up to a high level. By checking the GPIO input level of this IO, the insertion and ejection of the SD card can be detected. When the SD card begins data transmission, this detection should be halted, and DAT3 should be utilized as a normal data line.

3.Detection through command polling. The chip initiates command interactions with the SDcard at regular intervals, determining the card's presence based on whether a response is received.

SDMMC only supports card detection via DAT3. SR_CARD_EXIST reports the card's presence based on the level of DAT3, while card insertion and removal events are reported by SR_CARD_INSERT and SR_CARD_REMOVE, which can generate interrupts.

14.2.3.10 Bus Direct Read Mode

SDMMC allows the data space of the SD card to be mapped onto the AHB bus, enabling other master control modules to directly access the address to read data from the SD card via the AHB bus. Before utilizing this feature, the SDMMC must first configure the SD card into data transfer mode. Subsequently, when there is a read access to the mapped space on the bus, the SDMMC automatically issues commands to the SD card and reads multiple blocks of data, storing the data in the built-in cache (shared storage space with FIFO) and returning the data via the bus. If the data to be read from the bus is located in the cache, the SDMMC will not initiate a new read command but will instead return the results directly from the cache.

14.2.3.11 Sampling Clock Adjustment

When communicating with the device via SDMMC , it is essential to sample the CMD and DAT signals from the device on the rising edge of the CLK . Due to signal transmission delays, it may be necessary to adjust the sampling clock edge to ensure accurate sampling. The CLKCR_CLK_TUNE_SEL register can be utilized to adjust the sampling clock edge backward, offering a total of 4 levels for adjustment based on the actual conditions of the product. When the CLK frequency exceeds 50M , it is often necessary to adjust the sampling clock to ensure the proper functionality of the interface.

14.2.4 SDMMC Register

Table 14-5: SDMMC Register Mapping Table

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**SR**	command and data status register
[31:18]			RSVD	
[17]	rw	1'h0	cache_err	Detect cache error Read 1: cache error occur Read 0: no cache error Write 1: clear the bit Write 0: no any influence to the bit
[16]	rw	1'h0	sdio	Detect SDIO Card Interrupt Read 1: detect sdio card generating interrupt Read 0: no interrupt Write 1: clear the bit Write 0: no any influence to the bit
[15]	r	1'h0	card_exist	Card exist status Read 1: card exist Read 0: no card exist This bit will be valid after enable detect card.
[14]	rw	1'h0	card_remove	Detect card removed Read 1: detect card removed. When detect card inserted bit is set, the bit will also be back to 0 Read 0: no meaning Write 1: clear the bit Write 0: no any influence to the bit
[13]	rw	1'h0	card_insert	Detect card inserted Read 1: detect card inserted. When detect card removed bit is set, the bit will also be back to 0 Read 0: no meaning Write 1: clear the bit Write 0: no any influence to the bit
[12]	rw	1'h0	cmd_sent	Command sent (perhaps no response back yet) Read 1: command sent. When command start bit is set, the bit will also be back to 0 Read 0: command transferring or others Write 1: clear the bit Write 0: no any influence to the bit
[11]			RSVD	
[10]	rw	1'h0	fifo_overrun	FIFO overrun Read 1: FIFO overrun error Read 0: no FIFO overrun error Write 1: clear the bit Write 0: no any influence to the bit

Continued on the next page...

Table 14-5: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'h0	fifo_underrun	FIFO underrun Read 1: FIFO underrun error Read 0: no FIFO underrun error Write 1: clear the bit Write 0: no any influence to the bit
[8]	rw	1'h0	startbit_error	Wide bus start bits error Didn't detect all start bits in data bus Read 1: start bits error Read 0: no start bits error Write 1: clear the bit Write 0: no any influence to the bit
[7]	rw	1'h0	data_timeout	Data timeout Read 1: timeout Read 0: no timeout Write 1: clear the bit Write 0: no any influence to the bit
[6]	rw	1'h0	data_crc	Data CRC error Read 1: data CRC error Read 0: data CRC right Write 1: clear the bit Write 0: no any influence to the bit
[5]	rw	1'h0	data_done	Data transfer done Read 1: transfer data done, and start a new transfer will take the bit into 0 Read 0: data transferring or idle Write 1: clear the bit Write 0: no any influence to the bit
[4]	r	1'h0	data_busy	Transfer Data busy 1: busy, and when busy, start transfer data bit is no usage and you should not modify the relative register. If want to do this, first disable transfer data enable bit, then the busy bit will be back to 0, and this transfer will also be cancelled. 0: data idle
[3]	rw	1'h0	cmd_timeout	Command timeout (response timeout) Read 1: timeout Read 0: no timeout Write 1: clear the bit Write 0: no any influence to the bit
[2]	rw	1'h0	cmd_rsp_crc	Command response CRC error status Read 1: response CRC error Read 0: response CRC right Write 1: clear the bit Write 0: no any influence to the bit
[1]	rw	1'h0	cmd_done	Command done Read 1: transfer command done, and start a new transfer will take the bit into 0 Read 0: command transferring or idle Write 1: clear the bit Write 0: no any influence to the bit
[0]	r	1'h0	cmd_busy	Command busy 1: busy, and when busy, start TX command bit is no usage and should not modify the relative register 0: command idle
0x04			**CCR**	command control register
[31:24]			RSVD	
[23:18]	rw	6'h0	cmd_index	Command index

Continued on the next page...

Table 14-5: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[17]	rw	1'h0	cmd_long_rsp	1: Response will be 136-bit, long response 0: Response will be 48-bit, normal response
[16]	rw	1'h1	cmd_has_rsp	1: Response expected after command 0: No response expected after command
[15:10]			RSVD	
[9]	rw	1'h0	cmd_pend	Command pending enable When prepare to send stop command, this bit should be set. Controller will calculate a proper time point to send out the command to guarantee all the data have been transferred. And this is mainly used in stream mode. Recommend using set_block_count (SD/MMC basis command) to control transferring data for block mode. If send stop command for canceling this transfer (such as CRC error in multi-block), no need to set the bit.
[8]	rw	1'h0	cmd_tx_en	TX command enable 1: enable TX command 0: disable TX command
[7:1]			RSVD	
[0]	rw	1'h0	cmd_start	Command start write 1 to start command TX, and when begin to TX command, the bit will return into 0.
0x08			**CAR**	command argument register
[31:0]	rw	32'h0	cmd_arg	Command argument
0x0C			**RIR**	response command index register
[31:6]			RSVD	
[5:0]	r	6'h0	rsp_index	Response command index
0x10			**RAR1**	response command argument1 register
[31:0]	rw	32'h0	rsp_arg1	Response command content If long response, it is rsp_arg[39:8]
0x14			**RAR2**	response command argument2 register
[31:0]	rw	32'h0	rsp_arg2	Long response, it is rsp_arg[71:40]
0x18			**RAR3**	response command argument3 register
[31:0]	rw	32'h0	rsp_arg3	Long response, it is rsp_arg[103:72]
0x1C			**RAR4**	response command argument4 register
[31:24]			RSVD	
[23:0]	rw	24'h0	rsp_arg4	Long response, it is rsp_arg[127:104]
0x20			**TOR**	timeout count register
[31:0]	rw	32'h1000	timeout_cnt	Used to determine how much time waiting response or data bus busy is timeout, and decreased under card clock. Set to 400000 for 1s timeout if interface clock is 400KHz.
0x24			**DCR**	data control register
[31:27]			RSVD	
[26:16]	rw	11'h1ff	block_size	Data block size is block_size+1 (max 2048 bytes) 0: 1 byte 0x1ff: 512 bytes
[15:13]			RSVD	
[12:11]	rw	2'h0	wire_mode	Wide data bus mode 00: 1 wire bus 01: 4 wires wide bus 1X: reserved
[10]	rw	1'h0	stream_mode	Data transfer mode 0: block 1: stream

Continued on the next page...

Table 14-5: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[9]	rw	1'h0	r_wn	Write or read 0: write data into card 1: read data from card
[8]	rw	1'h0	tran_data_en	Transfer data enable 0: disable transfer data. After disable data transfer, stop command should be sent to card 1: enable data transfer
[7:1]			RSVD	
[0]	rw	1'h0	data_start	Start transfer data set 1 to let the controller begin to transfer data (in fact, go into wait write or wait read state). After begin to transfer, this bit will be back to 0.
0x28			**DLR**	data length register
[31:16]	r	16'h0	block_tran_num	The number of blocks which have been transferred successfully 1 = 1 block transferred It is cleared when start transfer data bit is set.
[15:0]	rw	16'h1ff	data_len	Data length value. The number of data bytes is data_len+1. The number of data bytes should be a multiple of data block size. 0 is 1 byte. 0x1ff is 512 bytes. Max is 63.5KB.
0x2C			**IER**	command and data interrupt mask register
[31:18]			RSVD	
[17]	rw	1'h1	cache_err_mask	cache error mask for interrupt
[16]	rw	1'h0	sdio_mask	Detect SDIO interrupt(data[1]) mask for interrupt
[15]			RSVD	
[14]	rw	1'h0	card_remove_mask	Detect card remove mask for interrupt
[13]	rw	1'h0	card_insert_mask	Detect card insert mask for interrupt
[12]	rw	1'h0	cmd_sent_mask	Command sent mask for interrupt
[11]			RSVD	
[10]	rw	1'h1	fifo_overrun_mask	FIFO overrun bit mask for interrupt
[9]	rw	1'h1	fifo_underrun_mask	FIFO underrun bit mask for interrupt
[8]	rw	1'h0	startbit_error_mask	Wide bus start bits error bit mask for interrupt
[7]	rw	1'h1	data_timeout_mask	Data timeout bit mask for interrupt
[6]	rw	1'h1	data_crc_mask	Data CRC error bit mask for interrupt
[5]	rw	1'h1	data_done_mask	Data transfer done bit mask for interrupt
[4]			RSVD	
[3]	rw	1'h1	cmd_timeout_mask	Command timeout bit mask for interrupt
[2]	rw	1'h1	cmd_rsp_crc_mask	Command CRC error bit mask for interrupt
[1]	rw	1'h1	cmd_done_mask	Command done bit mask for interrupt
[0]			RSVD	
0x30			**CLKCR**	clock control register
[31:21]			RSVD	
[20:8]	rw	13'h80	div	Divide card clock counter. 0 is illegal. sd_clock = hclk/(div + 1) If hclk is 240M and div is 599, 400KHz SD clock will be generated.
[7:4]			RSVD	
[3:2]	rw	2'h0	clk_tune_sel	select clock delay for rx sample 0: no delay 1: delay level 1 (~1.5ns typical) 2: delay level 2 (~3ns typical) 3: delay level 3 (~5ns typical)

Continued on the next page...

Table 14-5: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[1]	rw	1'h0	void_fifo_error	Void FIFO error 0: close the function 1: open the function If open it, when FIFO will be overrun or underrun soon, the SD_CLK and the clock enable of this module will be closed, and wait to host to read or write FIFO. Note: this function needs to be supported by card.
[0]	rw	1'h1	stop_clk	Disable SD card clock 1: stop SD card clock 0: SD card clock generated
0x3C			**CDR**	card interface control and card detect register
[31:19]	rw	13'h0	otiming	define output timing
[18:6]	rw	13'h0	itiming	define input timing
[5]	rw	1'h0	cmd_od	Open Drain mode for cmd line (for eMMC identification mode) 0: cmd line is push-pull 1: cmd line is open-drain
[4]	rw	1'h1	cd_hvalid	Card detect high level valid 0: detect low level means card exist 1: detect high level means card exist (default)
[3]	rw	1'h1	en_cd	Enable card detect Only when the bit is valid, controller does card detect. If use sd_data[3] to do card detect, the bit should be cleared when transfer valid data.
[2]	rw	1'h0	otiming_sel	select output timing (according to otiming config)
[1]	rw	1'h0	itiming_sel	select input sample timing (according to itiming config)
[0]	rw	1'h1	sd_data3_cd	Use sd_data[3] to do card detect 0: use special pin to do card detect / write protect. (Currently not supported) 1: use sd_data[3] to do card detect (default)
0x40			**DBGR1**	card debug port1 register
[31]			RSVD	
[30:16]	r	15'h1	data_st	data state for debug only
[15:0]	r	16'h1	cmd_st	command state for debug only
0x44			**DBGR2**	card debug port2 register
[31:30]	rw	2'h0	dbg_sel	for debug only
[29:26]			RSVD	
[25:16]	r	10'h0	valid_data_cou	for debug only
[15:14]			RSVD	
[13:0]	r	14'h0	host_word_counter	for debug only
0x48			**CEATA**	CE-ATA/SDIO mode register
[31:4]			RSVD	
[3]	rw	1'h0	sdio_4wires_multi_irq	Select the sdio host 4 wires interrupt on multi-block support 0: host not support 4 wires interrupt on multi-block data transfers 1: host support 4 wires interrupt on multi-block data transfers
[2]	rw	1'h0	sdio_4wires_irq	Select the sdio host 4 wires interrupt support 0: host not support 4 wires interrupt on single-block data transfers 1: host support 4 wires interrupt on single-block data transfers
[1]	rw	1'h0	enable_sdio_irq	Select the sdio card mode, default is sd card 0: sd card mode , no sdio card interrupt 1: sdio card mode , enable sdio card interrupt
[0]	rw	1'h0	ata_mode	Select the card type, default is sd card 0: sd card mode 1: CE-ATA device mode
0x54			**DSR**	data status register
[31:8]			RSVD	

Continued on the next page...

Table 14-5: SDMMC Register Mapping Table (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[7:0]	r	8'h0	sd_data_i_ll	The status of each sd data pad status
0x58			**CDCR**	clock duty cycle register
[31:1]			RSVD	
[0]	rw	1'h1	clk_config	1: the sd clock is 50% duty cycle 0: the high level of the sd clock is 1 hclk cycle
0x5C			**CASR**	cache status register
[31:4]			RSVD	
[3]	rw1s	1'h0	cache_flush	Set 1 to flush cache. Should set when cache not busy.
[2]	r	1'h0	cache_busy	Indicates cache is working
[1]	rw1c	1'h0	sd_busy	Read 1 indicates sd is ready for normal access. Ahb access will be hold during sd_busy asserted. After sd normal access done, write 1 to clear, and ahb access will continue
[0]	rw1s	1'h0	sd_req	Set 1 to request sd normal access. sd_req will be cleared automatically after sd_busy asserted
0x60			**CACR**	cache control register
[31]	rw	1'h1	cache_en	enable cache 1: ahb read will return cached data 0: ahb read always return dummy data with no error response
[30]	rw	1'h1	cache_to_en	enable ahb read timeout recover
[29]	rw	1'h0	cache_force_read	force cache read done 1: start new fetch for miss access only after cache read done 0: start new fetch for miss access even when cache is still filling (read will be breaked by cmd12)
[28]	rw	1'h1	cache_sdsc	select card version 1: card size <=2GB, address of cmd18 is in byte 0: card size >2GB, address of cmd18 is in block
[27]	rw	1'h0	cache_nocrc	1: return ahb data without crc check 0: return ahb data after block crc pass
[26]	rw	1'h0	cache_hresp	1: generate ahb error response when error occur 0: no ahb error response generated. Could check cache_err interrupt
[25:24]			RSVD	
[23:20]	rw	4'h8	cache_pref_block	cache prefetch depth is cache_pref_block blocks. Should be no less than cache_block
[19]			RSVD	
[18:16]	rw	3'h4	cache_block	cache depth is cache_block blocks
[15]	rw	1'h0	stop_long_rsp	Stop response is 136-bit, long response
[14]	rw	1'h1	stop_has_rsp	Stop command have a response
[13:8]	rw	6'h0c	stop_index	Command index for stop. CMD12 by default
[7]	rw	1'h0	read_long_rsp	Read response is 136-bit, long response
[6]	rw	1'h1	read_has_rsp	Read command have a response
[5:0]	rw	6'h12	read_index	Command index for cache read. CMD18 by default
0x64			**CACNT**	cache counter register
[31:16]	rw	16'hffff	cache_tor	timeout count register for ahb read
[15:8]	rw	8'h0	cache_ndc	data-cmd interval counter in hclk cycles
[7:0]	rw	8'h20	cache_ncc	cmd-cmd interval counter in hclk cycles
0x68			**CAOFF**	cache offset register
[31:0]	rw	32'h0	cache_offset	offset to map ahb address to sd address for ahb access
0x200			**FIFO**	FIFO entry
[31:0]	rw	32'h0	data	Entry to access internal FIFO. Access should be word-aligned, ranging from 0x200 to 0x3fc. Inside the range, write to any address will push the data into the FIFO, and read any address will pop a word from the FIFO.

14.3 MPI

The chip contains 4 MPIs, among which MPI1, MPI2, and MPI3 are located in HPSYS and can send requests to DMAC1; MPI5 is located at LPSYS and can send requests to DMAC2.

The input/output of MPI1 is connected to IO(SA) for accessing the chip-integrated (SiP) 8-bit pSRAM.

The input/output of MPI2 is connected to IO(SB) for accessing another chip-integrated (SiP) 8-bit pSRAM.

The input/output of MPI3 is connected to IO(PA) for accessing external NOR/NAND Flash.

The input/output of MPI5 is connected to IO(SC) for accessing the chip-integrated (SiP) 4-bit NOR Flash.

The MPI (Memory Peripheral Interface)controller is a dedicated memorycommunication interface supporting multiple types of off-chip memory devices, including:

- SPI NOR Flash , supports 1line/2lines/4lines, supports DTRmode
- SPI NAND Flash , supports 1line/2lines/4lines
- pSRAM , supports x8and x16data bus widths, supports Xccelastandard interface, compatible with Legacyinterface
- HyperRAM , supports x8 and x16 data bus widths, supports HyperBus standard interface

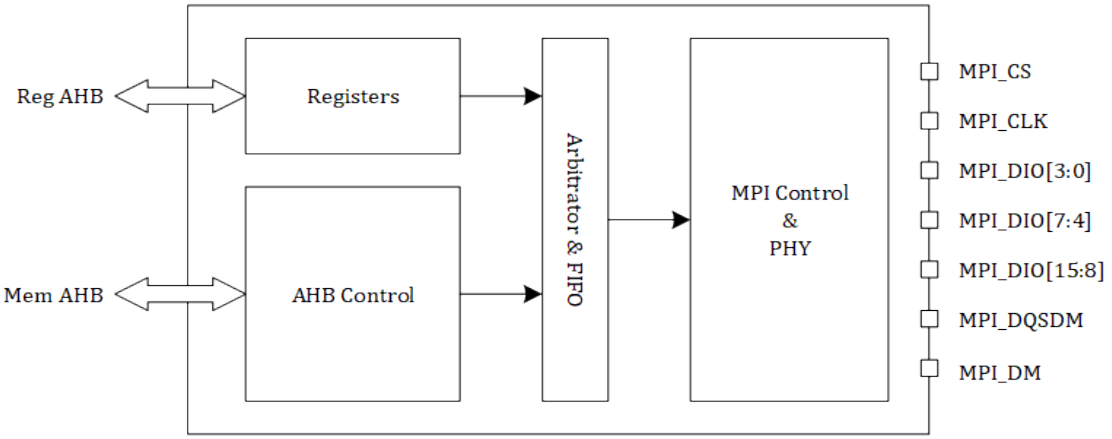


Figure 14-4: MPI controller block diagram

The MPI controller supports two operating modes: (1) Register mode and (2) Address mapping mode. Switching between these modes is automatically managed by hardware, enabling dynamic interleaved execution. Regardless of the mode, highly customizable interface timing is supported to ensure compatibility with various memory devices. **Register Mode**

- Transmit a command sequence via register operations. The command can also be configured as a status query command, repeatedly sent until the read-back data satisfes a predefined status.
- Supports sending a sequence containing two command sequences, where the second command can be configured as a status query command, repeatedly sent until the read-back data satisfes a predefined status.
- Supports DMAchannels, performing data transfer through the register FIFOinterface.

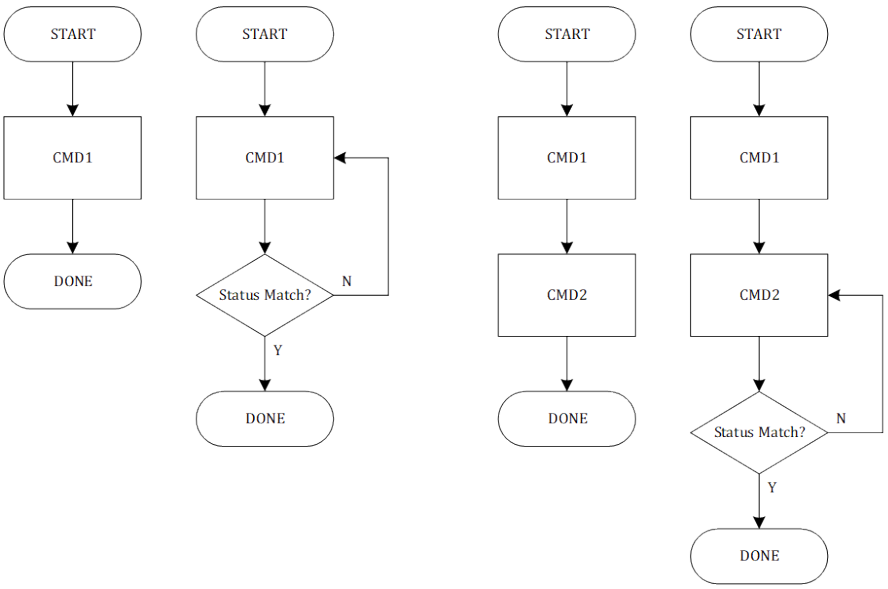


Figure 14-5: Sequence of single and multiple command timings in register mode

Address mapping mode

- External memory is mapped to the AHB address space, automatically converting AHB bus read and write operations to preset Memory interface timing, enabling XIP functionality
- Supports Byte (8-bit), Half-word (16-bit), and Word (32-bit) AHBaccess
- effcient conversion ofAHB Wrapoperations, independent of device support for Wrap
- Supports XIP real-time (On-The-Fly) decryption in AES128-CTR or AES256-CTR modes
- Supports continuous read and write operations; if the current AHB read/write address is sequential to the previous one, data transfer begins immediately, omitting command and address phases. This feature signifcantly improves effective bandwidth during large data transfers
- Regarding the internal dynamic refresh characteristics of pSRAM and HyperRAM , automatic management of the longest CS low time per chip, the most recent CS access interval, the maximum burst data length, and other constraints is performed without requiring software intervention.

14.3.1 MPI Register

Table 14-6: MPI Register Map

Offset	Attribute	Reset Value	Register Name	Register Description
0x00			**CR**	Control Register
[31]	w1t	1'h0	ABORT	Write 1 to abort internal state machine. For debug purpose only
[30:26]			RSVD	
[25]	rw	1'h0	AHBDIS	Hold hreadyout low if AHB access
[24]	rw	1'h0	DFM	Dual Flash Mode Reserved-Do not modify
[23]	rw	1'b0	MX16	Mode X16 Reserved-Do not modify
[22]	rw	1'b0	PREFE	Prefetch enable. If enabled, MPI will prefetch at consequtive address following a read transaction. Recommend to use when reading large data in a burst manner. 0: prefetch disabled 1: prefetch enabled

Continued on next page...

Table 14-6: MPI Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[21]	rw	1'b0	OPIE	OPI interface enable 0: x8 mode disabled 1: x8 mode enabled
[20]	rw	1'b0	HWIFE	Hardware interface enable Reserved-Do not modify
[19]	rw	1'b0	SMM	Status match mode 0: AND mode 1: OR mode
[18]	rw	1'b0	SME2	Status match enable. If enabled, CMD2 will be issued repeatedly until the data match the value in SMR and SMKR 0: disabled 1: enabled
[17]	rw	1'b0	SME1	Status match enable. If enabled, CMD1 will be issued repeatedly until the data match the value in SMR and SMKR 0: disabled 1: enabled (either SME1 or SME2 can be enabled, and SME1 has high priority)
[16]	rw	1'b0	CMD2E	Enable CMD2 0: disabled 1: CMD2 is enabled and will be issued after CMD1 with an interval of TI2
[15:14]			RSVD	
[13]	rw	1'h0	RBXIE	Row boundary crossing interrupt enable
[12]	rw	1'h0	CSVIE	CS max violation interrupt enable
[11]	rw	1'h0	SMIE	Status match interrupt enable
[10]			RSVD	
[9]			RSVD	
[8]	rw	1'h0	TCIE	Transfer complete interrupt enable
[7]	rw	1'h0	CTRM	AES-CTR mode 0: AES-128 1: AES-256
[6]	rw	1'h0	CTRE	AES-CTR on-the-fly decryption enable 0: disabled 1: enabled, data read from memory will be decrypted on the fly by MPI controller
[5]	rw	1'h0	DMAE	DMA enable 0: disabled 1: enable DMA to read or write DR register
[4]	rw	1'h0	HOLD	The value of HOLD when HOLDE is set
[3]	rw	1'h0	HOLDE	Enable HOLD function on IO3. Use this only in SPI or Dual SPI mode
[2]	rw	1'h0	WP	The value of WP when WPE is set
[1]	rw	1'h0	WPE	Enable WP function on IO2. Use this only in SPI or Dual SPI mode
[0]	rw	1'h0	EN	Enable MPI
0x04			**DR**	Data Register
[31:0]	rw	32'h0	DATA	The entry of internal data FIFO
0x08			**DCR**	Device Control Register
[31]	rw	1'h0	FIXLAT	Indicate PSRAM is fixed latency or variable latency. It must be compatible to the configuration in PSRAM registers. Recommend always set to 1. 0: variable latency 1: fixed latency

Continued on next page...

Table 14-6: **MPI Register Map (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[30:26]	rw	5'h0	TRCMIN	Write/Read cycle minimum time in internal MCLK cycles. Please see MCLK frequency in PSCLR description. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and TRCMIN = n, then tRC time = (n+1) * 1000/240 ns which must meet minimum tRC requirement for PSRAM
[25:22]	rw	4'h2	CSHMIN	Minimum CS high deselect time in MCLK cycles. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSHMIN = n, then CS High time = (n+1) * 1000/240 ns which must meet minimum tCPH requirement for PSRAM
[21:18]	rw	4'h0	CSLMIN	Minimum CS low active time in MCLK cycles. For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSLMIN = n, then CS Low time = (n+1) * 1000/240 ns which must meet the minimum tCEM requirement for PSRAM
[17:6]	rw	12'h0	CSLMAX	Maximum CS low active time in MCLK cycles For example, if PSRAM clock is 120MHz (i.e. internal MCLK is 240MHz) and CSLMAX = n, then CS Low time = (n+1) * 1000/240 ns which must meet the maximum tCEM requirement for PSRAM
[5]	rw	1'h0	XLEGACY	Xccela legacy protocol. Set to 1 for AP 32Mb PSRAM only, othersize always set to 0.
[4]	rw	1'h0	HYPER	HyperBus protocol. Set to 1 for HyperRAM.
[3]	rw	1'h0	DQSE	DQS enable. Setting to 1 indicates device provides DQS signal for Rx data latching
[2:0]	rw	3'h0	RBSIZE	Row boundary size. 0: no row boundary 1: $2^{(1+3)}$ = 16 bytes 2: $2^{(2+3)}$ = 32 bytes ... n: $2^{(n+3)}$ bytes
0x0C			**PSCLR**	Prescaler Register
[31:8]			RSVD	
[7:0]	rw	8'h4	DIV	Prescaler divider. 0: MCLK = FCLK/1 1: MCLK = FCLK/1 2: MCLK = FCLK/2 n: MCLK = FCLK/n Note: FLASH clock = MCLK. E.g. FCLK=192M and DIV=2, then FLASH clock = MCLK = 192/2 = 96MHz PSRAM clock = MCLK/2. E.g. FCLK=240M and DIV=1, then PSRAM clock = MCLK/2 = 240/2 = 120MHz
0x10			**SR**	Status Register
[31]	r	1'h0	BUSY	For debug purpose only
[30:6]			RSVD	
[5]	r	1'h0	RBXF	Row boundary crossing flag
[4]	r	1'h0	CSVF	CS max violation flag
[3]	r	1'h0	SMF	Status match flag in Polling Mode
[2]			RSVD	
[1]			RSVD	
[0]	r	1'h0	TCF	Transfer complete flag
0x14			**SCR**	Status Clear Register
[31:6]			RSVD	

Continued on next page...

Table 14-6: MPI Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5]	w1c	1'h0	RBXFC	Write 1 to clear RBXF
[4]	w1c	1'h0	CSVFC	Write 1 to clear CSVF
[3]	w1c	1'h0	SMFC	Write 1 to clear SMF
[2]			RSVD	
[1]			RSVD	
[0]	w1c	1'h0	TCFC	Write 1 to clear TCF
0x18			**CMDR1**	Command Register
[31:8]			RSVD	
[7:0]	rw	8'h0	CMD	Command. Write to this register will trigger the sequence specified in CCR1
0x1C			**AR1**	Address Register
[31:0]	rw	32'h0	ADDR	Address
0x20			**ABR1**	Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	Alternate byte
0x24			**DLR1**	Data Length Register
[31:20]			RSVD	
[19:0]	rw	20'h0	DLEN	Data length 0: one byte 1: two bytes ... n: (n+1) bytes
0x28			**CCR1**	Communication Configuration Register
[31:22]			RSVD	
[21]	rw	1'b0	FMODE	Function Mode 0: read mode 1: write mode
[20:18]	rw	3'h0	DMODE	Data Mode 0: no data phase 1: single line 2: dual lines 3: quad lines 4/5/6: reserved 7: quad lines DDR
[17:13]	rw	5'h0	DCYC	Number of dummy cycles 0: no dummy cycle 1: one dummy cycle 2: two dummy cycles
[12:11]	rw	2'h0	ABSIZE	Alternate byte size 0: one byte 1: two bytes 2: three bytes 3: four bytes
[10:8]	rw	3'h0	ABMODE	Alternate byte mode 0: no alternate byte 1: single line 2: dual lines 3: quad lines 4/5/6: reserved 7: quad lines DDR
[7:6]	rw	2'h0	ADSIZE	Address size 0: one byte 1: two bytes 2: three bytes 3: four bytes

Continued on next page...

Table 14-6: MPI Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
[5:3]	rw	3'h0	ADMODE	Address mode 0: no address phase 1: single line 2: dual line 3: quad line 4/5/6: reserved 7: quad line DDR
[2:0]	rw	3'h0	IMODE	Instruction mode 0: no instruction phase 1: single line 2: dual lines 3: quad lines 4/5/6 - reserved 7 - quad lines DDR
0x2C			**CMDR2**	Command Register
[31:8]			RSVD	
[7:0]	rw	8'h0	CMD	Command 2. If CMD2E is enabled, the CMD2 sequence will be issued after CMD1 as specified in CCR2 Note: CMD2 sequence cannot be issue individually
0x30			**AR2**	Address Register
[31:0]	rw	32'h0	ADDR	Address byte in CMD2 sequence
0x34			**ABR2**	Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	Alternate byte in CMD2 sequence
0x38			**DLR2**	Data Length Register
[31:20]			RSVD	
[19:0]	rw	20'h0	DLEN	Data length in CMD2 sequence
0x3C			**CCR2**	Communication Configuration Register
[31:22]			RSVD	
[21]	rw	1'b0	FMODE	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	
[2:0]	rw	3'h0	IMODE	This register specifies the format of CMD2 sequence. Refer to CCR1 description
0x40			**HCMDR**	AHB Command Register
[31:16]			RSVD	
[15:8]	rw	8'h02	WCMD	AHB write command. During XIP, the AHB write transaction will be translated into this Write Command on memory interface
[7:0]	rw	8'h0b	RCMD	AHB read command. During XIP, the AHB read transaction will be translated into this Read Command on memory interface
0x44			**HRABR**	AHB Read Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	
0x48			**HRCCR**	AHB Read Communication Configuration Register
[31:21]			RSVD	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	

Continued on next page...

Table 14-6: **MPI Register Map (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[2:0]	rw	3'h0	IMODE	This register specifies the format of AHB read command sequence. Refer to CCR1 description
0x4C			**HWABR**	AHB Write Alternate Byte Register
[31:0]	rw	32'h0	ABYTE	
0x50			**HWCCR**	AHB Write Communication Configuration Register
[31:21]			RSVD	
[20:18]	rw	3'h0	DMODE	
[17:13]	rw	5'h0	DCYC	
[12:11]	rw	2'h0	ABSIZE	
[10:8]	rw	3'h0	ABMODE	
[7:6]	rw	2'h0	ADSIZE	
[5:3]	rw	3'h0	ADMODE	
[2:0]	rw	3'h0	IMODE	This register specifies the format of AHB write command sequence. Refer to CCR1 description
0x54			**FIFOCR**	FIFO Control Register
[31:15]			RSVD	
[14:10]	rw	5'h8	TXSLOTS	When DMA enabled, asserts DMA reqeust if TXFIFO vacant slots is greater than or equal to TXSLOTS. Note: this field should be set in accordance to the burst length in DMA. For example, if DMA employs BURST8 transction, then this filed is set to 8
[9]	r	1'h0	TXF	Tx FIFO full flag
[8]	w1c	1'h0	TXCLR	write 1 to clear Tx FIFO
[7:2]			RSVD	
[1]	r	1'h1	RXE	Rx FIFO empty
[0]	w1c	1'h0	RXCLR	write 1 to clear Rx FIFO
0x58			**MISCR**	Miscelaneous Register
[31:28]	rw	4'h0	DBGSEL	
[27]			RSVD	
[26]	rw	1'h0	DTRPRE	Enable pre-sampling for DTR Reserved-Do not modify
[25]	rw	1'h1	SCKINV	Invert output clock. This bit is used to align (coarse tune) the output clock to the center of output data.
[24]	rw	1'h0	RXCLKINV	Invert internal Rx clock to add half-cycle delay (coarse tune) when sampling data. It is usually used for FLASH device w/ higher frequency.
[23:16]	rw	8'h0	DQSDLY	Delay the input DQS signal to the appropriate sampling position. For device w/ DQS signal only. Note: effective 7-bit
[15:8]	rw	8'h0	SCKDLY	Add delay on output clock to fine tune the clock position. Note: effective 7-bit
[7:0]	rw	8'h0	RXCLKDLY	Add delay on internal Rx clock to fine tune the sampling position. Note: effective 5-bit
0x5C			**CTRSAR**	CTR Starting Address Register
[31:10]	rw	22'h0	SA	Starting address of the AES decryption area. Since the lowest 10 bits are zero, the address is always 1KB aligned. Together with CTREAR, the total area is [CTRSAR, CTREAR) For example, CTRSAR = 32'h0, CTREAR = 32'h200000, then the on-the-fly decryption area is 0x0 - 0x1FFFFF
[9:0]			RSVD	
0x60			**CTREAR**	CTR Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the AES decryption area
[9:0]			RSVD	
0x64			**NONCEA**	Nonce A Register
[31:0]	rw	32'h0	NONCEA	Used for on-the-fly decryption
0x68			**NONCEB**	Nonce B Register
[31:0]	rw	32'h0	NONCEB	Used for on-the-fly decryption

Continued on next page...

Table 14-6: MPI Register Map (Continued)

Offset	Attribute	Reset Value	Register Name	Register Description
0x6C			**AASAR**	Address Aliasing Start Address Register
[31:10]	rw	22'h0	SA	Starting address of the address aliasing area. Always 1KB aligned.Together with AAEAR, the aliasing area is [AASAR, AAEAR). If the address falls into this area, an offset AAOAR is added and the aliased address will be used to access external memory
[9:0]			RSVD	
0x70			**AAEAR**	Address Aliasing Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the address aliasing area
[9:0]			RSVD	
0x74			**AAOAR**	Address Aliasing Offset Address Register
[31:10]	rw	22'h0	OA	The offset to be added to the original address
[9:0]			RSVD	
0x78			**CIR**	Command Interval Register
[31:16]	rw	16'h0	INTERVAL2	The interval between CMD1 and CMD2 (or between CMD2 itself) if CMD2E is enabled. The unit is in MCLK cycles
[15:0]	rw	16'h0	INTERVAL1	The interval between CMD1 itself. The unit is in MCLK cycles
0x7C			**SMR**	Status Match Register
[31:0]	rw	32'h0	STATUS	If status match is enabled, this register is compared with the data read from external memory. Together with SMKR, only the bits with mask=1 will be considered to compare in AND or OR mode as configured in SMM field.
0x80			**SMKR**	Status Mask Register
[31:0]	rw	32'h0	MASK	Status mask 0: the corresponding bit is not considered to compare 1: the corresponding bit is considered to compare
0x84			**TIMR**	Timer Register
[31:16]			RSVD	
[15:0]	rw	16'h0	TIMEOUT	After the transaction is complete, CS remains low for multiple cycles of MCLK as specified by this register. For example if TIMEOUT=n, CS remains active for n cycles, during which if a new transaction occurs and the address is consecutive, the memory access can be resumed w/o sending the command and address again.
0x88			**WDTR**	WDT Register
[31]	r	1'h0	TOF	Timeout flag. Self cleared when HREADYOUT becomes ready
[30:17]			RSVD	
[16]	rw	1'b0	EN	WDT enable. This watchdog is on AHB side such that bus access will not hang in exceptional cases
[15:0]	rw	16'hffff	TIMEOUT	Set timeout value in number of clk_wdt cycles
0x8C			**PRSAR**	Prefetch Starting Address Register
[31:10]	rw	22'h0	SA	Starting address of the prefetch area If prefetch is enabled and the read address falls into [PRSAR, PREAR), controller will prefetch the following data
[9:0]			RSVD	
0x90			**PREAR**	Prefetch Ending Address Register
[31:10]	rw	22'h0	EA	Ending address of the prefetch area
[9:0]			RSVD	
0x94			**CALCR**	Calibration Clock Register
[31]	rw	1'h0	EN	calibration enable
[30:9]			RSVD	
[8]	r	1'h0	DONE	calibration done flag
[7:0]	r	8'h0	DELAY	calibration delay result
0x98			**APM32CR**	APM32 Control Register

Continued on next page...

Table 14-6: **MPI Register Map (Continued)**

Offset	Attribute	Reset Value	Register Name	Register Description
[31:8]			RSVD	
[7:4]	rw	4'h4	TCPHW	For special use by AP 32Mb PSRAM. Reserved-Do not modify
[3:0]	rw	4'h2	TCPHR	For special use by AP 32Mb PSRAM. Reserved-Do not modify
0x9C			**CR2**	Control Register 2
[31:8]			RSVD	
[7:0]	rw	8'h0	LOOP	Repeat CMD1->CMD2 sequence for n times. This filed is only valid when CMD2E=1 and SME2=0. For example if LOOP=0, then the sequence is CMD1 -> CMD2. If LOOP=2, then the sequence is (CMD1->CMD2) -> (CMD1->CMD2) -> (CMD1->CMD2)

15 Bluetooth

15.1 Introduction

Support Bluetooth protocol v5.3 and is compatible with v4.2, v4.1, and v4.0.

The main functions are as follows:

- BLE mode:
  - Support rate (1M/2M);
  - Support all packet formats (broadcast packet/expanded broadcast packet/data packet, etc.);
  - Support data encryption and decryption;
  - Support data stream processing (redundancy checking, whitening);
  - Support two frequency hopping modes;
  - Maximum transmission power of BLE is 19dBm;
- classic Bluetooth mode：
  - Support BR, EDR2, EDR3;
  - Support all packet types of ACL, CSB, SCO and eSCO;
  - Support data encryption and decryption (E0 encryption and AES-CCM encryption);
  - Support data stream processing (HEC, CRC, Whitening, FEC2/3, FEC1/3);
  - Support coding and decoding of audio data (CVSD and a/μ-Law);
  - Support adaptive frequency hopping;
  - Maximum transmission power of BR is 19dBm and maximum transmission power of EDR2 and EDR3 is 19dBm; 13dBm
- and：
  - Support AMBA AHB bus access;
  - Support WIFI coexistence mechanism.
  - Under only BLE existing scence，the maximum number of links is 4. Under only BT existing scence，the maximum number of links is 7.

15.2 Coexistence Interface

15.2.1 Introduction

The coexistence interface can be used to implement WIFI coexistence. The coexistence interface includes a total of 4 signal lines, comprising 3 output signals: BT_ACTIVE, BT_PRIORITY, BT_COLLISION, and 1 input signal: WLAN_ACTIVE. It supports multiple configuration modes compatible with common single-line, 2 -line, 3 -line, and 4 -line coexistence interfaces, and supports both external and internal arbitration schemes. The output signal indicates the Bluetooth operating status, and the input signal controls the Bluetooth transmission and reception behavior.

Input and output signals can be directly connected to the chip IOor can trigger PTC events.

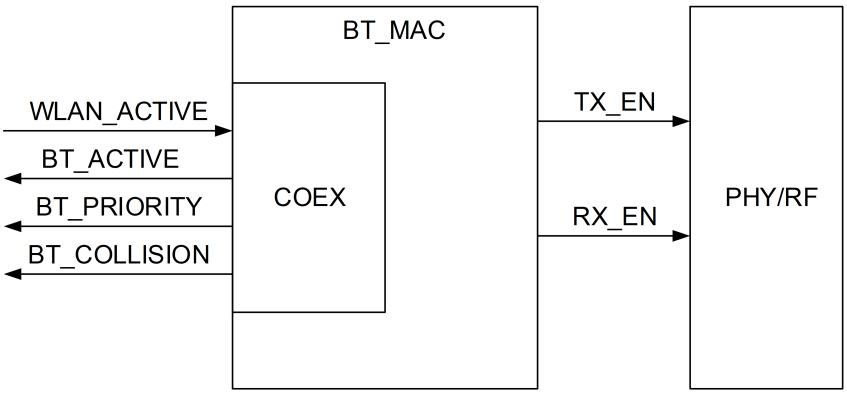


Figure 15-1: **Bluetooth Coexistence Diagram**

15.2.2 Main Features

- 4-wire interface, compatible with single-wire, 2-wire, 3-wire, and 4-wire schemes
- Bluetooth operating status output with configurable polarity and merged priority output
- Bluetooth operating priority output with configurable priorities for various scenarios and adjustable thresholds
- Bluetooth frequency conflict output with configurable conflict frequencies
- WIFI operating status input with configurable polarity, capable of masking Bluetooth transmission and reception

15.2.3 Output Signal BT_ACTIVE

BT_ACTIVE indicates whether Bluetooth is active, with configurable polarity. This signal can be generated by the state machine of BT_MAC or configured to a specifed level via registers.BT_ACTIVE can represent a complete Bluetooth event unit, such as a BLE broadcast polling of three channels, or a single BT sniff interaction; it can also represent a single Bluetooth transmission or reception. The interval TRF from the activation of BT_ACTIVE to the start of Bluetooth packet transmission and reception is typically several tens of us. BT_ACTIVE can also incorporate the function of BT_PRIORITY, outputting the active status only when BT_PRIORITY is set to high priority. BT_ACTIVE signal is typically used to send Bluetooth transmission requests to an external arbiter and can also be directly used to control an external RF switch.

The procedure to configure BT_ACTIVE is as follows:

1. configure the output polarity of BT_ACTIVE via the register BTCOEXIFCNTL2.PTA_ACTPOL in BT_MAC . The default value is 0 , indicating Blue tooth activity with a high level. When polarity is inverted, a low level indicates Bluetooth activity;
2. Select the coverage scope of BT_ACTIVE via the register BTCOEXIFCNTL2.PTA_ACTSEL . The default value is 0 , covering one complete Bluetooth event unit. If set to 1 , it covers only Bluetooth transmission; if set to 2 , it covers only Bluetooth reception;
3. If the functions of BT_ACTIVE and BT_PRIORITY need to be combined, set the BTCOEXIFCNTL2.PTA_ACTMODE register to 1 . In the default polarity, a high level in dicates that Bluetooth is active and the priority is high. When the polarity is inverted, a low level indicates that Bluetooth is active and the priority is high.

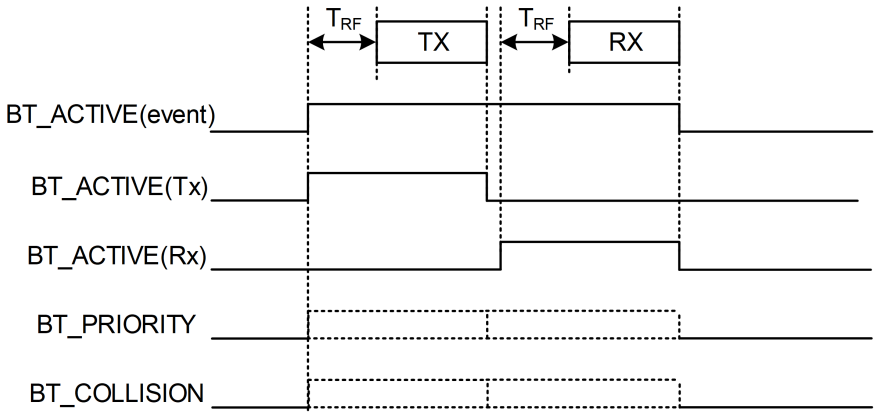


Figure 15-2: Output Signal Diagram

15.2.4 Output Signal BT_PRIORITY

BT_PRIORITY indicates the priority of Bluetooth operation, with high / low levels representing high / low priority, respectively. This signal can be generated by the state machine of BT_MAC or configured to a specifed level via registers. The Bluetooth MAC can set the priority corresponding to transmit and receive behaviors in different scenarios (such as broadcasting, connection establishment, connection maintenance, call, retransmission, etc.) and can configure a unifed priority threshold. When the priority corresponding to the Bluetooth transmit and receive behavior is not lower than the priority threshold, BT_PRIORITY is driven high; otherwise, BT_PRIORITY is low. The time interval from BT_PRIORITY activation to Bluetooth RF transmission and reception enabling equals TRF . BT_PRIORITY can also be combined with the BT_COLLISION function, whereby a high priority output is generated only when coexistence frequency points conflict.

The procedure to configure the output signalBT_PRIORITYis as follows:

1. Set the priorities of various Bluetooth scenarios for BT and BLE via the BTMPRIO0/1/2 and BLEMPRIO0/1/2 registers. After chip power-on, default initial priority values are assigned to each scenario and can be further adjusted as required;
2. configure the priority threshold via the BTCOEXIFCNTL1.WLCPTXTHR register; effective output is generated only when the priorities of BT and BLE are greater than or equal to the threshold;
3. If the function of mergingBT_PRIORITYandBT_COLLISIONis required, i.e., generating a valid output only when frequency point conflicts occur, the BTCOEXIFCNTL2.PTA_PRIOMODE register must be set to 1.

15.2.5 Output Signal BT_COLLISION

BT_COLLISION indicates whether a coexistence frequency conflict exists in Bluetooth; a high level indicates a conflict, and a low level indicates no conflict. This signal can be generated by the state machine of BT_MAC or configured to a specifed level via registers. To use this function, potentially conflicting frequency points must be pre-marked in the BT_MAC register. When Bluetooth hops to a marked frequency point for transmission and reception, the BT_COLLISION outputs a high level; otherwise, it outputs a low level. The time interval from BT_COLLISION activation to Bluetooth RF enabling transmission and reception equals T RF .

When using this function, the COEXCHN0/1/2registers must be configured to specify whether each frequency point from 2402MHzto 2480MHzcauses coexistence conflicts.

Register bits corresponding to conflicting frequency points shall be set to 1; otherwise, set to 0.

15.2.6 Input Signal WLAN_ACTIVE

WLAN_ACTIVE indicates the operating status of external WIFI and can be used as an enable signal for Bluetooth operation; the active level is configurable. The effect of WLAN_ACTIVE can be configured as one of the following behaviors:

1. No impact on Bluetooth transmission and reception;
2. Prohibit Bluetooth transmissions below the priority threshold;
3. Prohibit Bluetooth transmissions below the priority threshold;
4. Prohibit Bluetooth receptions below the priority threshold;
5. Prohibit Bluetooth receptions and transmissions below the priority threshold;
6. Prohibit all Bluetooth transmissions;
7. Prohibit all Bluetooth receptions;
8. Prohibit all Bluetooth receptions and transmissions;

The following figure illustrates an example of WLAN_ACTIVE affecting Bluetooth transceiving. When WLAN_ACTIVE is asserted, both Bluetooth transmission and reception are immediately prohibited, except when the priority is high.

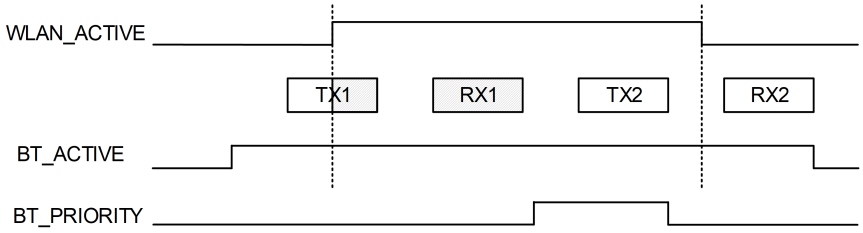


Figure 15-3: Example of WLAN_ACTIVE impact

The procedure for configuring the input signal WLAN_ACTIVE is as follows:

1. configure the active level of WLAN_ACTIVE via the BTCOEXIFCNTL2.PTA_WLANPOL register; the default value 0 indicates active high.
2. To disable Bluetooth transmission when WLAN_ACTIVE is at the active level, write 1 to the BTCOEXIFCNTL2.PTA_MASKTX and ABORTTX registers. To disable Bluetooth reception when WLAN_ACTIVE is at the active level, write 1 to the BTCOEXIFCNTL2.PTA_MASKRX and ABORTRX registers.
3. To disable only Bluetooth low-priority behavior, set the BTCOEXIFCNTL2.PTA_PRIOMODE register to 1.

15.2.7 Procedure for Configuring the Coexistence Interface

1. configure the required output signals and input signals;
2. configure PINMUX to map the required input and output signals to the specifed IO;
3. Enable the coexistence interface by setting the registers BTCOEXIFCNTL0.WLANCOEX_ENandBLECOEXIFCNTL0.WLANCOEX_EN of the BT_MAC bit 1.

15.2.8 IO Mapping

Input and output signals can be mapped to the specifed IO(PA) or IO(PB) ; refer to the IO function selection table for details. Coexistence signals are generated by the BT_MAC within LPSYS and can be directly mapped to IO(PB) ; however, if mapping to IO(PA) is required, the HPSYS must also be in an active state; otherwise, IO(PA) cannot toggle. Therefore,

when selecting IO mapping, it is necessary to comprehensively consider the low-power scenarios of the chip operating in conjunction with external WIFI.

Using PTCenables the output of coexistence signals to be mapped to any IO(PB). The coexistence output signals can generate PTCevent triggers. By using the rising and falling edges of the output signal as trigger conditions, changing the output level of any IO can realize the mapping of coexistence output signals to any IO.

15.2.9 Example of a 3-wire external arbitration scheme

In this example, Bluetooth and WIFI share a single antenna, switching via an RF switch, with control signals originating from the WIFI chip. Arbitration is performed by the WIFI chip, which determines whether to grant antenna usage rights to Bluetooth based on the Bluetooth output signals BT_ACTIVE and BT_PRIORITY , and can disable Bluetooth transceivers via the WLAN_ACTIVE signal during WIFI operation.

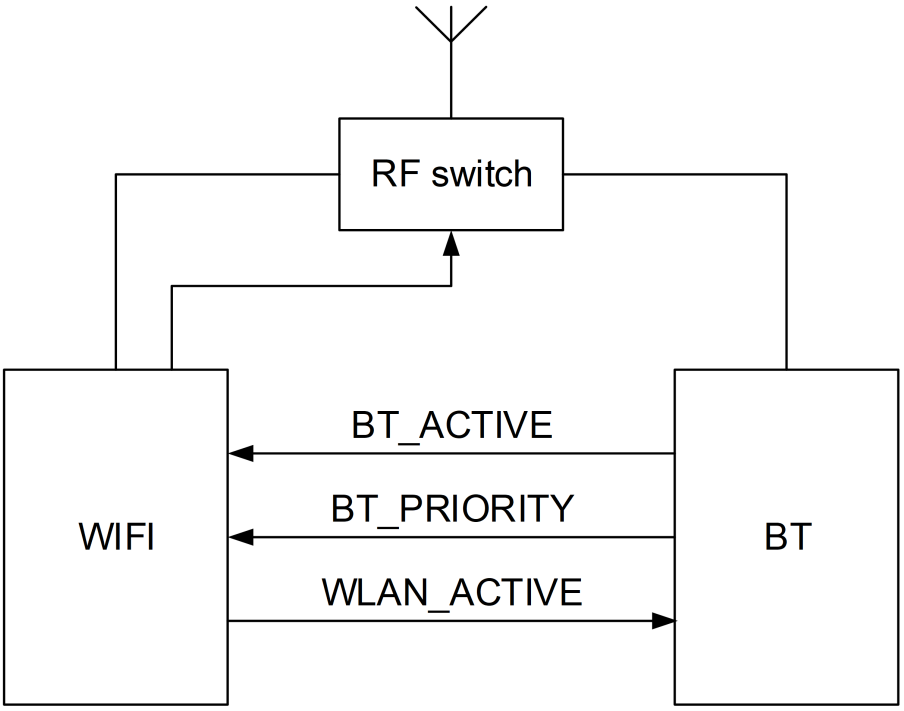


Figure 15-4: Example of a 3-wire external arbitration scheme

15.2.10 Example of a single-wire internal arbitration scheme

In this example, Bluetooth and WIFI share a single antenna, switched via an RF switch controlled by Bluetooth signals. Arbitration is performed by Bluetooth, controlling antenna usage rights with the BT_ACTIVE signal and notifying the WIFI chip. If it is necessary to allocate part of the time slots to WIFI, the functions BT_ACTIVEand BT_PRIORITY can be enabled, and the priority threshold adjusted according to the operational scenario to free some low-priority time slots for WIFI usage.

If WIFI can output a signal indicating its operational status, this scheme can be upgraded to a 2-wire mode. Connect the output of WIFI to the WLAN_ACTIVE input; Bluetooth can suppress low-priority Bluetooth transceivers when WIFI requires operation, and remains unaffected when WIFI is inactive.

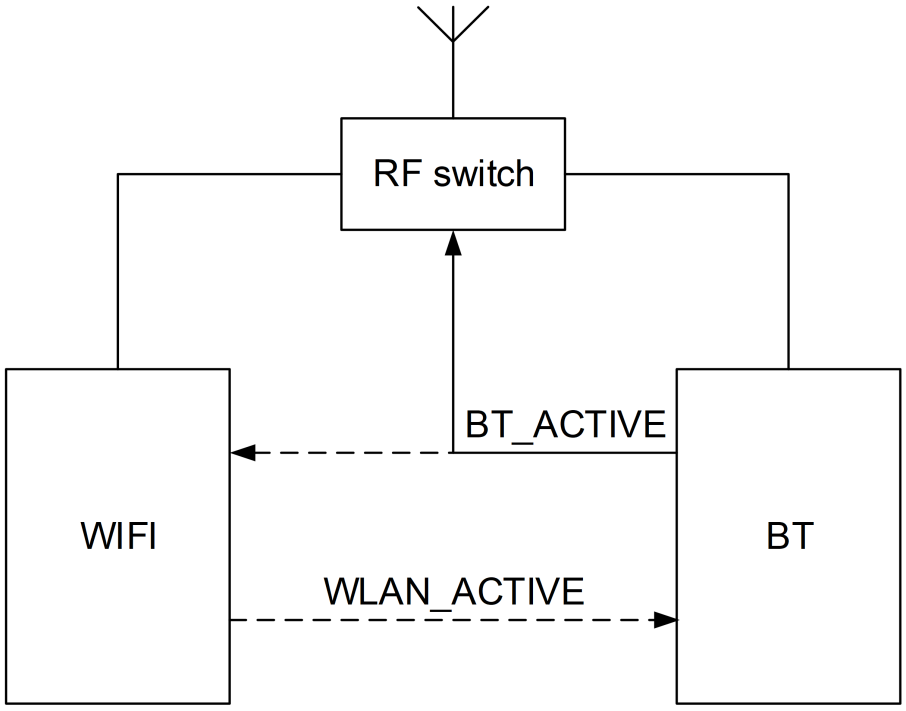


Figure 15-5: Examples of internal arbitration single-wire or 2-wire schemes

15.3 External LNA and PA

Using external PA and LNA modules to enhance Bluetooth performance can further increase the transmit power and sensitivity of Bluetooth, enabling Bluetooth transmission at a range of hundreds of meters to several kilometers.

The configuration process for external PA is follows:

1. The PADs that can be connected to an external PA are PB35 or PB27, and configure the corresponding PAD_PB27/PB35.FSEL register to 0x6. When selecting PB35 to connect an external PA, also need to configure LPSYS_CFG->SYSCR.DBG_SWAP register to 1;
2. When selecting PB37 to connect an external PA, configure LPSYS_CFG->DBGR.SEL_L register to 0x7, configure LPSYS_CFG->DBGR.BITEN_L register to 0xf0; When selecting PB35 to connect an external PA, configure LPSYS_CFG->DBGR.SEL_H register to 0x7, configure LPSYS_CFG->DBGR.BITEN_H register to 0xf0;

The configuration process for external LNA is follows:

1. The PADs that can be connected to an external LNA are PB34 or PB26, and configure the corresponding PAD_PB26/PB34.FSEL register to 0x6. When selecting PB34 to connect an external LNA, also need to configure LPSYS_CFG->SYSCR.DBG_SWAP register to 1;
2. When selecting PB26 to connect an external LNA, configure LPSYS_CFG->DBGR.SEL_L register to 0x7, configure LPSYS_CFG->DBGR.BITEN_L register to 0xf0; When selecting PB34 to connect an external LNA, configure LPSYS_CFG->DBGR.SEL_H register to 0x7, configure LPSYS_CFG->DBGR.BITEN_H register to 0xf0;

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