User Manual

G32R502

Arm® Cortex®-M52 Core-based 32-bit Real-time MCU

Version: V1.0

© Geehy Semiconductor Co., Ltd.

Contents

1 Introduction and document description rules 11

1.1 Introduction 11

1.2 Document description rules 12

2 CPU and system architecture 14

2.1 CPU 14

2.2 Bus Architecture and Address Mapping 16

2.3 Register bank address 21

2.4 Register address mapping 21

2.5 Register functional description 22

3 Mathematics instruction extension (Zidian) 31

4 System Boot (BOOT) 32

4.1 Introduction 32

4.2 Functional description 32

4.3 Register 32

5 Write register protection (WRPRT) 33

5.1 Introduction 33

5.2 Main characteristics 33

5.3 Functional description 33

5.4 Register bank address 34

5.5 Register address mapping 34

5.6 Register functional description 35

6 UID 36

6.1 Register bank address 36

6.2 Register address mapping 36

6.3 Register functional description 36

7 Power management 38

7.2 PMU Module Description 38

8 Device identification and configuration registers 42

8.1 Introduction 42

8.2 Register bank address 42

8.3 Register address mapping 42

8.4 Register functional description 43

9 Low-power mode 53

9.1 IDLE 53
9.2 HALT 53
10 Reset 56
10.1 Functional description 56
11 Clock 58
11.1 Structure block diagram 58
11.2 Functional description 59
11.3 Register bank address 68
11.4 Register address mapping 68
11.5 Register functional description 69
12 System control 76
12.1 SPEC Functional Description 76
12.2 SPEC register bank address 77
12.3 SPEC register address mapping 77
12.4 Functional description of SPEC registers 78
12.5 NMI_INTRUPT register bank address 85
12.6 NMI_INTRUPT register address mapping 85
12.7 Functional description of NMI_INTRUPT registers 86
12.8 PERIPH_AC register bank address 91
12.9 PERIPH_AC register address mapping 91
12.10 Functional description of PERIPH_AC register 92
12.11 ROM_PREFETCH register bank address 99
12.12 ROM_PREFETCH register address mapping 99
12.13 Functional description of ROM_PREFETCH register 99
12.14 ROM_WAIT_STATE register bank address 100
12.15 ROM_WAIT_STATE register address mapping 100
12.16 Functional description of ROM_WAIT_STATE register 100
13 Nested Vector Interrupt Controller (NVIC) 101
13.1 Full Name and Abbreviation Description of Terms 101
13.2 Introduction 101
13.3 Main characteristics 101
13.4 Interrupt and exception vector table 101
14 External interrupt and event controller (EXTI) 107
14.1 Introduction 107
14.2 Main characteristics 107
14.3 Block Diagram 108

14.4 Functional description 108
14.5 Register bank address 111
14.6 Register address mapping 111
14.7 Register functional description 112
15 Dual Code Security (DCS) 117
16 Configurable static memory subsystem (CFGSMS) 118
17 DBGMCU 119
17.1 Real-time Mode 119
17.2 Register Address Mapping 120
18 Nonvolatile memory controller (NVMC) 125
18.1 Introduction 125
18.2 Block diagram 125
18.3 Main characteristics 127
18.4 Clocks 128
18.5 Reset 128
18.6 Interrupts 128
18.7 Low Power Consumption 129
18.8 Functional description 130
18.9 Register bank address 138
18.10 Register address mapping 138
18.11 Register functional description 139
19 Timer 0/1/2 (TMR0/1/2) 158
19.1 Full Name and Abbreviation Description of Terms 158
19.2 Introduction 158
19.3 Main characteristics 158
19.4 Structure block diagram 158
19.5 Functional description 159
19.6 Register bank address 159
19.7 Register address mapping 159
19.8 Register functional description 160
20 Watchdog timer (WDT) 164
20.1 Full Name and Abbreviation Description of Terms 164
20.2 Introduction 164
20.3 Main characteristics 164
20.4 Structure block diagram 165
20.5 Functional description 165

20.6 Register bank address 168
20.7 Register address mapping 168
20.8 Register functional description 168
21 NMIWDT 172
21.1 Introduction 172
21.2 Key Features 172
21.3 Function Description 172
21.4 Register Function Description 172
22 Dual-clock comparator (DCCOMP) 173
22.1 Introduction 173
22.2 Main characteristics 173
22.3 Structure block diagram 173
22.4 Functional description 174
22.5 Register bank address 177
22.6 Register address mapping 177
22.7 Register functional description 178
23 General-Purpose Input/Output Pin (GPIO) 183
23.1 Full Name and Abbreviation Description of Terms 183
23.2 Introduction 183
23.3 Main characteristics 183
23.4 Structure block diagram 183
23.5 Functional description 184
23.6 GPIO Multiplexing Function Configuration 195
23.7 Register bank address 200
23.8 Register address mapping 200
23.9 Register functional description 202
24 X-BAR 250
24.1 Introduction 250
24.2 Main characteristics 250
24.3 Functional description 251
24.4 Register bank address 260
24.5 Register address mapping 260
24.6 Register functional description 263
25 Direct memory access (DMA) 285
25.1 Full Name and Abbreviation Description of Terms 285
25.2 Introduction 285

25.3 Main characteristics 285
25.4 Structure block diagram 286
25.5 Functional description 286
25.6 Register bank address 295
25.7 Register address mapping 296
25.8 Register functional description 297
26 Analog Subsystem (AS) 310
26.1 Introduction 310
26.2 Main characteristics 310
26.3 Structure block diagram 310
26.4 Functional description 312
26.5 Register bank address 317
26.6 Register address mapping 317
26.7 Register functional description 317
27 Analog-to-digital converter (ADC) 325
27.1 Full Name and Abbreviation Description of Terms 325
27.2 Introduction 325
27.3 Main characteristics 325
27.4 Structure block diagram 326
27.5 Functional description 327
27.6 Register bank address 347
27.7 Register address mapping 347
27.8 Register functional description 350
28 Buffer digital-to-analog converter (DAC) 372
28.1 Introduction 372
28.2 Main characteristics 372
28.3 Structure block diagram 372
28.4 Functional description 372
28.5 Lock register 374
28.6 Register bank address 375
28.7 Register address mapping 375
28.8 Register functional description 375
29 Comparator (COMP) 378
29.1 Full Name and Abbreviation Description of Terms 378
29.2 Introduction 378
29.3 Main characteristics 378

29.4 Structure block diagram 379
29.5 Functional description 380
29.6 Register bank address 389
29.7 Register address mapping 389
29.8 Register functional description 390
30 Pulse width modulator (PWM) 399
30.1 Full Name and Abbreviation Description of Terms 399
30.2 Introduction 399
30.3 Main characteristics 400
30.4 Structure block diagram 404
30.5 Functional description 406
30.6 Register bank address 480
30.7 Register address mapping 481
30.8 Register functional description 484
31 High-resolution pulse width modulator (HRPWM) 553
31.1 Full Name and Abbreviation Description of Terms 553
31.2 Introduction 553
31.3 Main characteristics 553
31.4 HRPWM structure block diagram 554
31.5 Functional description 555
31.6 Register 572
32 Capture (CAP) 573
32.1 Full Name and Abbreviation Description of Terms 573
32.2 Introduction 573
32.3 Main characteristics 573
32.4 Structure block diagram 575
32.5 Function description of CAP pins 575
32.6 Capture description function description 578
32.7 Function description of APWM mode 584
32.8 CAP module application 587
32.9 Register bank address 591
32.10 Register address mapping 591
32.11 Register functional description 592
33 Quadrature encoder pulse (QEP) 600
33.1 Full Name and Abbreviation Description of Terms 600
33.2 Introduction 600

33.3 Main characteristics 600
33.4 Structure block diagram 601
33.5 Functional description 601
33.6 Function description of quadrature mode adapter (QMA) 608
33.7 Function description of quadrature coder unit (QDU) 610
33.8 Function description of position counter and control unit (PCCU) 614
33.9 Function description of edge capture unit 620
33.10 Register bank address 623
33.11 Register address mapping 623
33.12 Register functional description 624
34 Serial peripheral interface (SPI) 639
34.1 Full Name and Abbreviation Description of Terms 639
34.2 Introduction 639
34.3 Main characteristics 639
34.4 Structure block diagram 640
34.5 Functional description 641
34.6 Register bank address 662
34.7 Register address mapping 662
34.8 Register functional description 663
35 Universal asynchronous receiver/transmitter (UART) 674
35.1 Full Name and Abbreviation Description of Terms 674
35.2 Introduction 674
35.3 Main characteristics 674
35.4 Structure block diagram 675
35.5 Functional description 675
35.6 Register bank address 685
35.7 Register address mapping 685
35.8 Register functional description 686
36 Internal integrated circuit interface (I2C) 696
36.1 Full Name and Abbreviation Description of Terms 696
36.2 Introduction 696
36.3 Main characteristics 696
36.4 Structure block diagram 697
36.5 Functional description 697
36.6 Register bank address 710
36.7 Register address mapping 710

36.8 Register functional description 710
37 Power management bus (PMBUS) 722
37.1 Full Name and Abbreviation Description of Terms 722
37.2 Introduction 722
37.3 Main characteristics 722
37.4 Structure block diagram 723
37.5 Functional description 723
37.6 Register bank address 745
37.7 Register address mapping 745
37.8 Register functional description 746
38 Controller area network (CAN) 757
38.1 Full Name and Abbreviation Description of Terms 757
38.2 Introduction 757
38.3 Main characteristics 757
38.4 Structure block diagram 759
38.5 Functional description 760
38.6 Register bank address 809
38.7 Register address mapping 809
38.8 Register functional description 810
39 Controller Area Network (CAN FD) 839
39.1 Full Name and Abbreviation Description of Terms 839
39.2 Introduction 839
39.3 Main Characteristics 840
39.4 Structure Block Diagram 841
39.5 Functional Description 842
39.6 Register bank address 878
39.7 Register Address Mapping 878
39.8 Register Functional Description 880
40 Local Interconnection Network (LIN) 909
40.1 Full Name and Abbreviation Description of Terms 909
40.2 Introduction 909
40.3 Main characteristics of UART 909
40.4 Main characteristics of LIN 910
40.5 Structure block diagram 911
40.6 LIN Functional Description 914
40.7 Low-power mode 938

40.8 Simulation Mode 941
40.9 UART Functional Description 941
40.10 Register bank address 957
40.11 Register address mapping 957
40.12 Register functional description 958
41 Flexible logic block (FLB) 991
42 Revision history 992

1 Introduction and document description rules

1.1 Introduction

G32R502x is a high-performance MCU developed by Geehy for real-time control applications. It is equipped with a Arm® Cortex®-M52 microprocessor based on the Arm v8.1-M architecture and supports up to 1 cores. It is suitable for such applications as household appliances, photovoltaic inverters, digital power supplies, and on-board chargers (OBC).

The maximum operating frequency of G32R5xx MCU can be up to 200MHz, realizing custom datapath extension (CDE) for Arm Rv8-M.Which further improves the processing performance through innovative Zidian mathematical calculation extended instruction set. The Zidian mathematical calculation extended instruction set can quickly execute the algorithms that include the common trigonometric operations in transformation and torque loop calculation, and reduce the delay of common complex mathematical operation in coding application.

G32R502x supports up to 256KB Flash internally. It is divided into two 128KB independent storage bodies., supporting parallel programming and execution. Besides, through the built-in CFGSMS, efficient system partitioning can be implemented for 40KB SRAM storage on the chip. Each logical block has a size of 8KB, which can be configured for different types of usage, e.g. ITCM, DTCM, and SRAM. In addition, G32R502x also supports Flash, RAM ECC parity check, and security attribute configuration.

G32R502x chip integrates high-performance analog unit, which can further improve the system control performance. Two independent 12-bit ADC can accurately and efficiently collect and process multiple analog signals, thereby improving system throughput. 4 comparator subsystems can constantly monitor the input voltage level through the tripping function.

G32R502x includes performance-leading control peripherals (with frequency-independent PWM and CAP), which can place excellent control over the system.

Universal communication ports (e.g. SPI, UART, I2C, LIN, and CAN/CAN FD) are built in G32R502x, and multiple multiplexing options are provided to meet the communication requirements of various applications. The G32R502x device also provides the PMBus interface that fully comply with the standards. Besides, G32R502x also supports JTAG, cJTAG, and SWD debugging interfaces. The multiple debugging modes are suitable for different system environments and performance requirements.

G32R502x supports the operating temperatures ranging from -40°C to 105°C/125°C, and provides multiple packages for choice such as VQFN32, HTQFP48, LQFP64, and

LQFP80.

1.1.1 Register access method

If the registers described in the Manual are 16-bit registers, these registers can be accessed according to 16-bit operations only; if the registers described in the Manual are 32-bit registers, they can only be accessed according to 32-bit operations. CPU cannot initiate non-aligned access to the peripheral area; otherwise, it will enter hardfault (Hardware error interrupt).

1.2 Document description rules

1.2.1 Full Name and Abbreviation Description of Terms

Table 1 R/W Abbreviation and Description

R/W	Description	Abbreviations
read/write	The software can read and write this bit.	R/W
read-only	The software can only read this bit.	R
write-only	The software can only write this bit, and after reading this bit, the reset value will be returned.	W
read/clear	The software can read this bit and clear it by writing 1. Writing 0 has no effect on this bit.	RC_W1
read/clear	The software can read this bit and clear it by writing 0. Writing 1 has no effect on this bit.	RC_W0
read/clear by read	The software can read this bit, reading this bit will automatically clear it to 0, and writing this bit is invalid.	RC_R
read/set	The software can read and set this bit, and writing 0 has no effect on this bit.	R/S
read-only write trigger	The software can read this bit and writing 0 or 1 can trigger an event but has no effect on the value of this bit.	RT_W
toggle	The software can flip this bit only by writing 1, and writing 0 has no effect on this bit.	T
Read Returns 0s	Read returns 0	R-0
Write 1 to set	Write 1 to set	W1S
Write 1 to clear	Write 1 to clear	W1C
Write once	Write once	WOnce
Write Set once	Write set once	WSonce
Write once	Write this bit once	WO

Table 2 Full Name and Abbreviation of Modules

Full name in English	English abbreviation
Reset and Clock Management	RCM
Clock Recovery System	CRS
Power Management Unit	PMU
Nested Vector Interrupt Controller	NVIC
External Interrupt /Event Controller	EXTI
Direct Memory Access	DMA
Debug MCU	DBG MCU
General-Purpose Input Output Pin	GPIO
Alternate Function Input Output Pin	AFIO
Timer	TMR
Watchdog Timer	WDT
Independent Watchdog Timer	IWDT
Windows Watchdog Timer	WWDT
Real-Time Clock	RTC
Universal Synchronous Asynchronous Receiver Transmitter	UART
Inter-integrated Circuit Interface	I2C
Serial Peripheral Interface	SPI
Inter-IC Sound Interface	I2S
Controller Area Network	CAN
Analog-to-Digital Converter	ADC
Digital-to-Analog Converter	DAC
Comparator	COMP
Cyclic Redundancy Check Calculation Unit	CRC

2 CPU and system architecture

2.1 CPU

Arm Cortex-M52 is a high-performance 32-bit processor specifically designed for MCU applications and based on the Armv8-M architecture. Supports multiple features to meet the requirements of various application scenarios for processing latency, DSP processing, floating-point operations, low power consumption, etc.

2.1.1 Main characteristics

(1) A 4-stage pipelined Harvard architecture that supports parallel instruction and data read/write operations to improve system efficiency.

(2) Supports fast interrupt response to meet the low-latency requirements of mission-critical applications.

- Hardware-based context switching: registers are pushed onto and popped from the stack directly by the hardware.
- Data reads and writes occur simultaneously with instruction fetches: register push operations and ISR instruction fetches are executed concurrently.
- When an interrupt occurs, it can pre-empt multi-data read/write instructions currently being executed, such as LDM and STM.
- Supports cascading interrupt execution: upon completion of the current interrupt, the system checks whether another interrupt has occurred; if so, it is executed immediately, saving context-switching time.
- Supports nested interrupts and priority pre-emption to reduce processing latency for high-priority tasks: during the interrupt entry and execution phases, high-priority interrupts may pre-empt lower-priority interrupts.

(3) Supports tightly coupled memory (TCM): supports independent ITCM and DTCM to improve system efficiency and reduce processing latency.

(4) Supports independent I-Cache and D-Cache.

- 4 KB, 2-way set-associative I-Cache
- 4 KB, 4-way set-associative D-Cache

(5) Supports a DSP-extended instruction set to meet the requirements of digital signal processing.

- Supports a variety of bit/byte operations: pack, saturating, bit/byte reverse, sign-extend, byte select, etc.

- Supports single-cycle multiply-accumulate: 32-bit MAC, 2 × 16-bit MAC, unsigned MUL, etc.
- Supports SIMD instruction extensions: 8/16-bit SIMD instructions.

(6) Supports single-precision floating-point extensions (FPU)

(7) Supports user-defined instruction extensions: CDE and co-processor.

(8) Supports memory protection and security features: MPU, SAU, IDAU, etc.

(9) Supports a wide range of debugging features: DWT, ETM, TPIU, DAP, etc.

(10) CPU

- Cortex-M52
- CACHE
  - 8KB cache (4KB I-cache + 4KB D-cache)
  - Supports ECC check
- TCM
  - 39-bit 16KB (default size) ITCM, 8–32KB configurable by cfgsms
  - 39-bit 12KB (default size) DTCM0, 8–32KB configurable by cfgsms
  - 39-bit 12KB (default size) DTCM1, 8–32KB configurable by cfgsms
  - Supports ECC check
- MPU
- FPU (Single Precision)
- CDE (DIV/FPCDE/CRC)
- COP (for EALLOW)

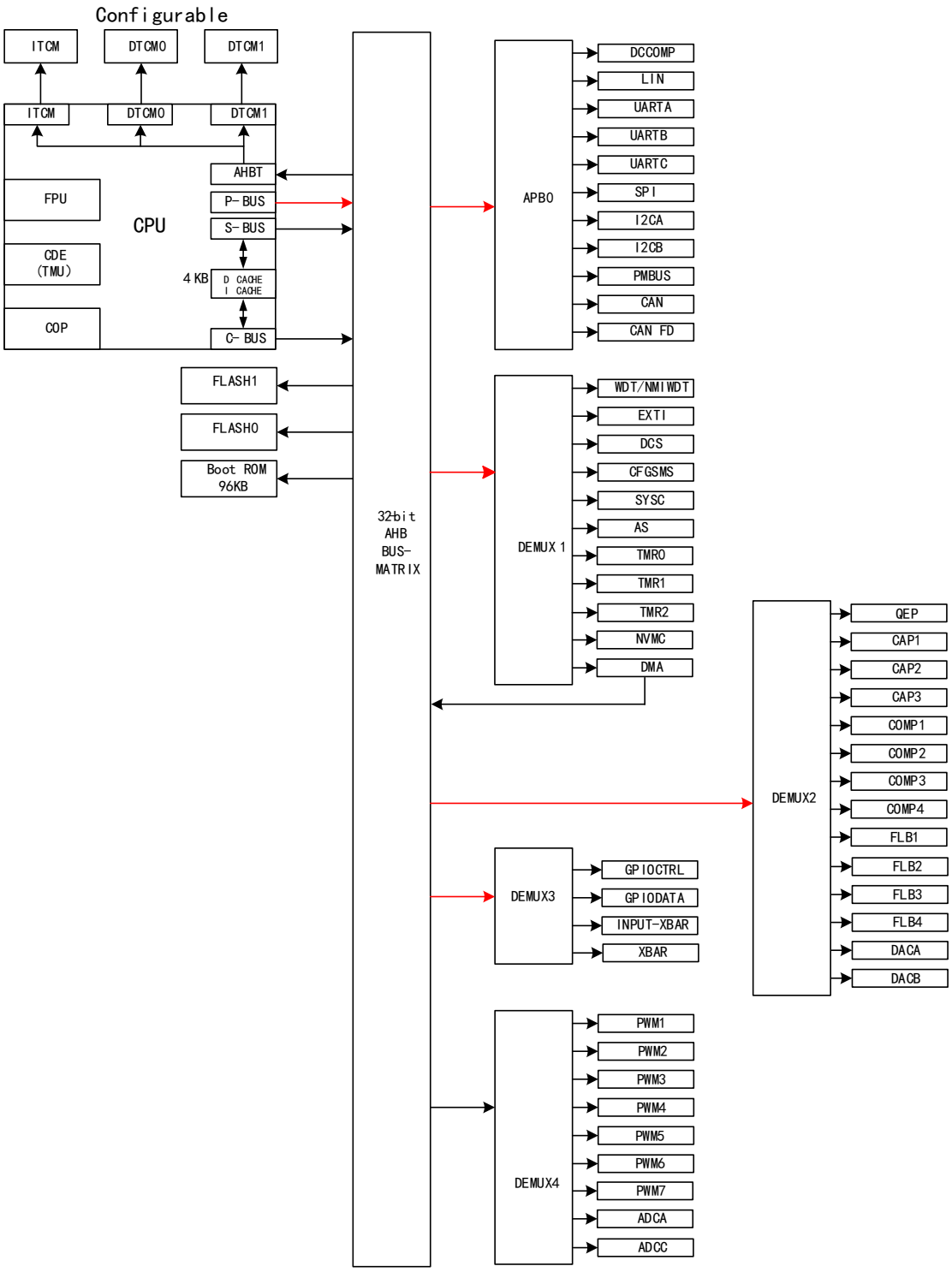
2.1.2 Helium

For specific information on Helium, please refer to the relevant CPU manuals. The R502 Helium supports the following features: scalar half-precision and single-precision floating-point calculations, and integer MVE. It does not support half-precision or single-precision MVE.

2.2 Bus Architecture and Address Mapping

2.2.1 Bus Architecture

Figure 1 R502 Bus Architecture



The R502 SoC is a system-on-chip based on the single-core Cortex-M52 architecture. The chip utilises the widely used AMBA 2.0 bus to integrate peripheral IP components: components requiring high bandwidth are connected to the AMBA AHB bus, whilst low-speed components are connected to the AMBA APB bus. The architectural block diagram of this SoC is shown above.The DMA does not have permission to access registers associated with the FLB.

Storage Section

(1) On-chip FLASH
- 144-bit bus width
- Total size: 256 KB
  - Main memory blocks (128 KB + 128 KB)
  - Data blocks (32KB + 32KB)
- Supports single/dual-bank switching

(2) On-chip Boot ROM
- 32-bit bus width, 96KB

(3) On-chip CAN-dedicated SRAM
- 1KB (CAN)
- 4KB (CAN FD)

Bus architecture

(1) AHB bus matrix
- 32-bit bus width
- 4 master devices: CPU-C-BUS, CPU-S-BUS, CPU-P-BUS, DMA
- 8 slave devices: CPU-AHBT, FLASH0, FLASH1, Boot ROM, DEMUX0, DEMUX1, DEMUX2, APB0

(2) DEMUX0
- 9 slave devices: PWM1, PWM2, PWM3, PWM4, PWM5, PWM6, PWM7, ADCA, ADCC

(3) DEMUX1
- 9 slave devices: WDT/NMIWDT, EXTI, DCS, CFGSMS, SYSC, ANALOG SUB SYSTEM, TMR0, TMR1, TMR2, NVMC, DMA

(4) DEMUX2
- 14 slave devices: QEP, CAP1, CAP2, CAP3, COMP1, COMP2, COMP3, COMP4, FLB1, FLB2, FLB3, FLB4, DACA, DACB

(5) DEMUX3
- 4 slave devices: GPIOCTRL, GPIODATA, INPUT-XBAR, XBAR

(6) APB0

- 11 slave devices: DCCOMP, LIN, CAN, CANFDA, UARTA, UARTB, UARTC, SPI, I2CA, I2CB, PMBUS

2.2.2 Address mapping

2.2.2.1 Memory address mapping

Table 3 General-purpose Memory Mapping

Address range	Size (maximum value)	Description
0x0000_0000 – 0x0000_7FFF	32KB	CPU ITCM (default 16KB)
0xA000_0000 – 0xA000_7FFF	32KB	CPU ITCM (CPU-AHBT) (default 16KB)
0x0800_0000 – 0x0803_FFFF	256KB	FLASH memory on the bus matrix
0x0810_0000 – 0x0810_BFFF	64KB	FLASH INFO on the bus matrix
0x0818_0000 – 0x0818_3FFF	16KB	FLASH INFO1 on the bus matrix
0x0900_0000 – 0x0900_7FFF	32KB	FLASH ECC (Main)
0x0902_0000 – 0x0902_1FFF	8KB	FLASH ECC (IFREN)
0x1000_0000 – 0x1001_7FFF	96KB	Boot ROM
0x2000_0000 – 0x2000_7FFF	32KB	CPU DTCM (default 24 KB)
0xA010_0000 – 0xA010_7FFF	32KB	CPU DTCM (CPU-AHBT) (default 24 KB)

Table 4 General- Bus Address Mapping

Address range	Bus	Function
0x4000_0000 – 0x4000_FFFF	APB0	APB peripherals
0x5000_0000 – 0x5000_FFFF	DEMUX4	AHB peripherals
0x4001_0000 – 0x4001_FFFF	DEMUX1	AHB peripherals
0x4002_0000 – 0x4002_FFFF	DEMUX2	AHB peripherals
0x4003_0000 – 0x4003_FFFF	DEMUX3	AHB peripherals

2.2.2.2 Peripheral address mapping

The register blocks for most on-chip peripherals are aligned to 1KB boundaries. Modules that may require additional address space will be allocated an extra 1KB block.Within a 1KB block, peripheral registers may not be fully decoded. The description for each peripheral will define the behaviour when accessing undecoded registers. For

registers where not all 32 bits are implemented, undecoded bits will return zero on read, whilst writing to undecoded bits will have no effect. Generally, to ensure future compatibility, undecoded bits should return zero.

Table 5 Peripheral Address Mapping

Bus	IP	Address
APB0	DCCOMP	0x4000_0000–0x4000_03FF
	LIN	0x4000_0400–0x4000_07FF
	UARTA	0x4000_0800–0x4000_0BFF
	UARTB	0x4000_0C00–0x4000_0FFF
	UARTC	0x4000_1000–0x4000_13FF
	SPI	0x4000_1400–0x4000_17FF
	I2CA	0x4000_1800–0x4000_1BFF
	I2CB	0x4000_1C00–0x4000_1FFF
	PMBUS	0x4000_2000–0x4000_23FF
	CAN	0x4000_2400–0x4000_2BFF
	CANFDA	0x4000_2C00–0x4000_33FF
	CANFDA-SRAM	0x4000_3400–0x4000_43FF
	Reserved	0x4000_4400–0x4000_FFFF
DEMUX4	PWM1	0x5000_0000–0x5000_03FF
	PWM2	0x5000_0400–0x5000_07FF
	PWM3	0x5000_0800–0x5000_0BFF
	PWM4	0x5000_0C00–0x5000_0FFF
	PWM5	0x5000_1000–0x5000_13FF
	PWM6	0x5000_1400–0x5000_17FF
	PWM7	0x5000_1800–0x5000_1BFF
	ADCA	0x5000_1C00–0x5000_1FFF
	ADCC	0x5000_2000–0x5000_23FF
	Reserved	0x5000_2400–0x5000_FFFF
DEMUX1	WDT NMIWDT	0x4001_0000–0x4001_00BF 0x4001_00C0–0x4001_03FF
	EXTI	0x4001_0400–0x4001_07FF
	DCS	0x4001_0800–0x4001_27FF
	CFGSMS	0x4001_2800–0x4001_2BFF

Bus	IP	Address
	SYSC	0x4001_2C00–0x4001_6BFF
	AS	0x4001_6C00–0x4001_6FFF
	TMR0	0x4001_7000–0x4001_73FF
	TMR1	0x4001_7400–0x4001_77FF
	TMR2	0x4001_7800–0x4001_7BFF
	NVMC	0x4001_7C00–0x4001_83FF
	DMA	0x4001_8400–0x4001_87FF
	Reserved	0x4001_8800–0x4001_FFFF
DEMUX2	QEP	0x4002_0000–0x4002_03FF
	CAP1	0x4002_0400–0x4002_07FF
	CAP2	0x4002_0800–0x4002_0BFF
	CAP3	0x4002_0C00–0x4002_0FFF
	COMP1	0x4002_1000–0x4002_13FF
	COMP2	0x4002_1400–0x4002_17FF
	COMP3	0x4002_1800–0x4002_1BFF
	COMP4	0x4002_1C00–0x4002_1FFF
	FLB1	0x4002_2000–0x4002_27FF
	FLB2	0x4002_2800–0x4002_FFFF
	FLB3	0x4002_3000–0x4002_37FF
	FLB4	0x4002_3800–0x4002_3FFF
	DACA	0x4002_4000–0x4002_43FF
	DACB	0x4002_4400–0x4002_47FF
	Reserved	0x4002_4800–0x4002_FFFF
DEMUX3	GPIOCTRL	0x4003_0000–0x4003_07FF
	GPIODATA	0x4003_0800–0x4003_0BFF
	INPUTXBAR SyncSocREG	0x4003_0C00–0x4003_0C7F 0x4003_0C80–0x4003_0FFF
	XBAR_REG PWM_XBAR_REG FLB_XBAR_REG OUTPUT_XBAR_REG	0x4003_1000 – 0x4003_103F 0x4003_11C0 – 0x4003_123F 0x4003_1240 – 0x4003_12BF 0x4003_12C0 – 0x4003_13FF
	Reserved	0x4003_1400–0x4003_FFFF

2.2.2.3 **Bus matrix**

Table 6 Bus Matrix Interconnection

Bus Slaves	Bus Masters			
	CPU C-BUS	CPU S-BUS	CPU P-BUS	DMA
CPU-AHBT	×	×	×	√
FLASH BANK 0	√	×	×	√
FLASH BANK1	√	×	×	√
Boot ROM	√	×	×	√
DEMUX4	×	√	×	√
DEMUX1	×	×	√	√
DEMUX2	×	×	√	√ (1)
DEMUX3	×	×	√	√
APB0	×	×	√	√

Note:

(1) DMA does not have permission to access FLB-related registers.

2.3 Register bank address

Table 7 Register Bank Address

Device register	Register bank	Start address	End address
CpuSysRegs	CPU_SYS_REGS	0x4001_3600	0x4001_37FF

2.4 Register address mapping

Table 8 CPU_SYS_REGS Offset Address

Register name	Register description	Offset address	WRPRT
CPUSYSLOCK1	Lock CPUSYS register	0x00	√
PCLKCR0	Peripheral clock gating register 0	0x44	√
PCLKCR2	Peripheral clock gating register 2	0x4C	√
PCLKCR3	Peripheral clock gating register 3	0x50	√
PCLKCR4	Peripheral clock gating register 4	0x54	√
PCLKCR7	Peripheral clock gating register 7	0x60	√
PCLKCR8	Peripheral clock gating register 8	0x64	√
PCLKCR9	Peripheral clock gating register 9	0x68	√
PCLKCR10	Peripheral clock gating register 10	0x6C	√
PCLKCR13	Peripheral clock gating register 13	0x78	√

Register name	Register description	Offset address	WRPRT
PCLKCR14	Peripheral clock gating register 14	0x7C	√
PCLKCR16	Peripheral clock gating register 16	0x84	√
PCLKCR17	Peripheral clock gating register 17	0x88	√
PCLKCR19	Peripheral clock gating register 19	0x90	√
PCLKCR20	Peripheral clock gating register 20	0x94	√
PCLKCR21	Peripheral clock gating register 21	0x98	√
LPMCR	LPM control register	0xEC	√
GPIOLPMSEL0	GPIO LPM wake-up select register 0	0xF0	√
GPIOLPMSEL1	GPIO LPM wake-up select register 1	0xF4	√
TMR2CLKCTL	TMR2 clock measurement function control register	0xF8	√
RESCCLR	Reset cause clear register	0xFC	-
RESC	Reset cause register	0x100	-

2.5 Register functional description

2.5.1 Lock CPUSYS register (CPUSYSLOCK1)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	Reserved			0h
x+3	PCLKCRx	R/WSonce	Lock bit for PCLKCRx register (x=0-4) 1: Unlocked 1: Locked	0h
9:8	Reserved			0h
x+3	PCLKCRx	R/WSonce	Lock bit for PCLKCRx Register (x=7-10) 1: Unlocked 1: Locked	0h
15:14	Reserved			0h
x+3	PCLKCRx	R/WSonce	Lock bit for PCLKCRx Register (x=13-16) 1: Unlocked 1: Locked	0h
20	Reserved			0h
21	LPMCR	R/WSonce	Lock bit for LPMCR Register 1: Unlocked 1: Locked	0h

Field	Name	R/W	Description	Reset value
x+22	GPIOLPMSELx	R/WSonce	Lock bit for GPIOLPMSELx Register (x=0-1) 1: Unlocked 1: Locked	0h
x+3	PCLKCRx	R/WSonce	Lock bit for PCLKCRx Register (x=17-21) 1: Unlocked 1: Locked	0h
31:29	Reserved			0h

2.5.2 Peripheral clock gating register 0 (PCLKCR0)

Offset address: 0x44

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
2	DMA	R/W	DMA Clock Enable 0: Disable 1: Enable	0h
x+3	CPUTIMERx	R/W	CPUTIMER Clock Enable 0: Disable 1: Enable	0h
15:6	Reserved			0h
16	HRPWM	R/W	HRPWM Clock Enable Set the clock to the HRPWM module during setting. 0: Disable 1: Enable	0h
17	Reserved			0h
18	TBCLKSYNC	R/W	PWM Time Base Clock sync Enable After setting, synchronize all enabled PWM modules to the reference clock (TBCLK). 0: Disable 1: Enable	0h
31:19	Reserved			0h

2.5.3 Peripheral clock gating register 2 (PCLKCR2)

Offset address: 0x4C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	PWMx	R/W	PWMx Clock Enable (x=1-7) 0: Disable 1: Enable	0h
31:7	Reserved			0h

2.5.4 Peripheral clock gating register 3 (PCLKCR3)

Offset address: 0x50

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	CAPx	R/W	CAPx Clock Enable (x=1-3) 0: Disable 1: Enable	0h
31:3	Reserved			0h

2.5.5 Peripheral clock gating register 4 (PCLKCR4)

Offset address: 0x54

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	QEPx	R/W	QEPx Clock Enable (x=1) 0: Disable 1: Enable	0h
31:1	Reserved			0h

2.5.6 Peripheral clock gating register 7 (PCLKCR7)

Offset address: 0x60

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	UART_A	R/W	UARTA Clock Enable 0: Disable 1: Enable	0h
1	UART_B	R/W	UARTB Clock Enable 0: Disable 1: Enable	0h
2	UART_C	R/W	UARTC Clock Enable 0: Disable 1: Enable	0h
31:3	Reserved			0h

2.5.7 Peripheral clock gating register 8 (PCLKCR8)

Offset address: 0x64

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SPI_A	R/W	SPIA Clock Enable 0: Disable 1: Enable	0h
31:1	Reserved			0h

2.5.8 Peripheral clock gating register 9 (PCLKCR9)

Offset address: 0x68

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	I2C_A	R/W	I2CA Clock Enable 0: Disable 1: Enable	0h
1	I2C_B	R/W	I2CB Clock Enable 0: Disable 1: Enable	0h
31:2	Reserved			0h

2.5.9 Peripheral clock gating register 10 (PCLKCR10)

Offset address: 0x6C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CAN_A	R/W	CANA Clock Enable 0: Disable 1: Enable	0h
3:1	Reserved			0h
4	CANFD_A	R/W	CANFDA Clock Enable 0: Disable 1: Enable	0h
31:5	Reserved			0h

2.5.10 Peripheral clock gating register 13 (PCLKCR13)

Offset address: 0x78

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ADC_A	R/W	ADCA Clock Enable 0: Disable 1: Enable	0h
1	Reserved			0h
2	ADC_C	R/W	ADCC Clock Enable 0: Disable 1: Enable	0h
31:3	Reserved			0h

2.5.11 Peripheral clock gating register 14 (PCLKCR14)

Offset address: 0x7C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	COMPx	R/W	COMPx Clock Enable (x=1-4) 0: Disable 1: Enable	0h
31:4	Reserved			0h

2.5.12 Peripheral clock gating register 16 (PCLKCR16)

Offset address: 0x84

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	Reserved			0h
16	DAC_A	R/W	DACA Clock Enable 0: Disable 1: Enable	0h
17	DAC_B	R/W	DACB Clock Enable 0: Disable 1: Enable	0h
31:18	Reserved			0h

2.5.13 Peripheral clock gating register 17 (PCLKCR17)

Offset address: 0x88

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	FLBx	R/W	FLBx Clock Enable (x=1-4) 0: Disable 1: Enable	0h
31:4	Reserved			0h

2.5.14 Peripheral clock gating register 19 (PCLKCR19)

Offset address: 0x90

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	LIN_A	R/W	LINA Clock Enable 0: Disable 1: Enable	0h
31:1	Reserved			0h

2.5.15 Peripheral clock gating register 20 (PCLKCR20)

Offset address: 0x94

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PMBUS_A	R/W	PMBUSA Clock Enable 0: Disable 1: Enable	0h
31:1	Reserved			0h

2.5.16 Peripheral clock gating register 21 (PCLKCR21)

Offset address: 0x98

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DCCOMP_0	R/W	DCCOMP0 Clock Enable 0: Disable 1: Enable	0h
31:1	Reserved			0h

2.5.17 LPM control register (LPMCR)

Offset address: 0xEC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	LPM	R/W	Low power mode configure Effective when CPU executes IDLE instruction (when IDLE instruction is not in the EXE stage of the assembly line) 00: Idle mode 01: Reserved 1x: Stop mode	0h
7:2	Reserved			3Fh
31:8	Reserved			0h

2.5.18 GPIO LPM wake-up select register 0 (GPIOLPMSEL0)

Offset address: 0xF0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x	GPIOx	R/W	(pin is connected to LPM circuit) (x=0-31) 0: Disconnect 1: Connect	0h

2.5.19 GPIO LPM wake-up select register 1 (GPIOLPMSEL1)

Offset address: 0xF4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-32	GPIOx	R/W	(pin is connected to LPM circuit) (x=32-63) 0: Disconnect 1: Connect	0h

2.5.20 TMR2 clock measurement function control register (TMR2CLKCTL)

Offset address: 0xF8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	TMR2CLKSRCSEL	R/W	TMR2 Clock Source Select 000: SYSCLK selection (default value when reset, bypassing the prescaler) 001: INTOSC1 010: INTOSC2 011: XTAL 100: FLPUMPOSC 101: FOSCCLK (reserved) 110: AUXPLLCLK (reserved) 111: Reserved	0h
5:3	TMR2CLKPRESCALE	R/W	TMR2 Clock Prescale Select 000: /1: (default value during reset) 001: /2 010: /4 011: /8 100: /16 Others: Backup (default value is /16) Note: TMR clock synchronization logic detects the input clock edge and generates appropriate clock pulses to TMR 2 when configured as any clock source other than SYSCLK. If SYSCLK is approximately the same as or smaller than the input clock source, it is necessary to configure a prescaler value, ensuring the speed of SYSCLK is at least twice the prescaler value.	0h
31:6	Reserved			0h

2.5.21 Reset cause clear register (RESCCLR)

Offset address: 0xFC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
2	WDRSn	W1S	WDRSn Flag Clear Reading RESCCLR always returns 0. Write 1 to this bit to clear the status bit in RESC to 0. Writing 0 is invalid.	0h
3	NMIWDRSn	W1S	NMIWDRSn Flag Clear	0h

Field	Name	R/W	Description	Reset value
			Reading RESCCLR always returns 0. Write 1 to this bit to clear the status bit in RESC to 0. Writing 0 is invalid.	
7:4	Reserved			0h
8	SCCRESETn	W1S	SCCRESETn Flag Clear Reading RESCCLR always returns 0. Write 1 to this bit to clear the status bit in RESC to 0. Writing 0 is invalid.	0h
31:9	Reserved			0h

2.5.22 Reset cause register (RESC)

Offset address: 0x100

Reset type: Except for [31:30] which is N/A, others are POR

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
2	WDRSn	R	WDRSn Flag In order to maintain consistency, the bits within the WDT module also provide the same information. This is because this register is a one-stop service for software to understand the reason for Arm® Cortex®-M52 core reset. 0: No effect 1: Force this bit to be 0 Note: Upon initial power-up, the reset value of this register may be 1; it should be cleared to zero during the initialisation phase.	0h
3	NMIWDRSn	R	NMIWDRSn Flag The exact reason for NMI reset requires software to read the NMISHDFLG register. 0: No effect 1: Force this bit to be 0	0h
7:4	Reserved			0h
8	SCCRESETn	R	SCCRESETn Flag The device is reset by SCCRESETn (triggered by DCS). 0: No effect 1: Force this bit to be 0	0h
29:9	Reserved			0h
30	XRSn_pin_status	R	XRSn pin status Read this bit to provide the current status of the XRSn pin. The reset value reflects the pin status.	0h

Field	Name	R/W	Description	Reset value
31	DCON	R	Debugger connection to the Arm® Cortex®-M52 CPU Flag Read this bit to provide the status of the debugger connection to the CPU. 0: Not connected 1: Connect Note: This bit is connected to the DCON o/p signal of the Arm® Cortex®-M52 CPU.	0h

3 Mathematics instruction extension (Zidian)

Zidian accelerates certain specific integer and floating-point arithmetic by adding new instructions, so as to extend the functions of Arm processor. This is achieved by using the ACI functions provided by Arm. Please refer to *Zidian User Manual* for description of this module.

4 System Boot (BOOT)

4.1 Introduction

CPU of master core needs to boot from BOOTROM.

4.2 Functional description

4.2.1 Boot Modes

For Boot Modes configuration and pin selection, please refer to Boot ROM and Peripheral Boot in *G32R502 Datasheet*.

4.3 Register

For BOOT registers, please refer to System Control Registers.

5 Write register protection (WRPRT)

5.1 Introduction

The WRPRT mechanism is designed to prevent CPU from performing erroneous write operations on some registers in the system. This mechanism uses special co-processor instructions ("WRPRT_DISABLE" and "WRPRT_ENABLE") to disable and enable write protection.

5.2 Main characteristics

(1) To use WRPRT, the CP1 bit in the CPACR register must be enabled (set to 0b11) to obtain access rights to Coprocessor 1 (CP1); these access rights are determined by interface number 12 within CP1 (i.e. the WRPRT bit).

(2) Executing the “WRPRT_ENABLE” instruction can enable write protection (clear the WRPRT bit), CPU couldn’t write the protected registers

(3) Execute the “WRPRT_DISABLE” instruction to disable write protection (clear the WRPRT bit), CPU can write the protected registers

(4) The WRPRT bits of CPU0 and CPU1 are independent

(5) At the beginning of ISR, the WRPRT bit is cleared to zero

- The current WRPRT value will be saved in the shadow register

5.3 Functional description

5.3.1 WRPRT protection mechanism

Table 9 Access to WRPRT Protected Registers

WRPRT bit	Access methods	DEBUG	DMA	CPU0/1
0	Read	Enable	Enable	Enable
	Write	Enable	Ignore	Ignore
1	Read	Enable	Enable	Enable
	Write	Enable	Enable	Enable

Note: The JTAG/SWD port covers WRPRT bit, allowing full access to protected registers in debug mode.

When booting G32R502, the register protection function is enabled by default:

- In protected state, CPU read and DEBUG read/write are enabled, and CPU's write operations to protected registers are ignored.

- Executing the “WRPRT_ENABLE” instruction can disable the protection (CPU is allowed to freely write to protected registers)
- After the register is modified, executing the "WRPRT_DISABLE" instruction can clear the WRPRT bit (enable protection again)

5.3.2 WRPRT protection in ISR

Table 10 Write Operation of CPU0/1 to WRPRT Protected Registers

Protection status	CPU0 write operation
WRPRT_ENABLE	Ignored
WRPRT_DISABLE	Allowed

When the CPU service is interrupted, the current value of WRPRT is saved in the shadow register and WRPRT is cleared to zero. Therefore, access to protected registers is disabled at the beginning of ISR. If ISR must access protected registers, it must contain a “WRPRT_DISABLE” instruction. At the end of ISR, WRPRT can be restored from the shadow register.

5.3.3 Instructions for use

As WRPRT_ENABLE/WRPRT_DISABLE instruction is implemented through the COP interface, their effective time may differ from the application purpose. This error can be avoided by adding the DSB/ISB instruction between register access and WRPRT_ENABLE/WRPRT_DISABLE instruction, and be solved by adding a read check after writing to the protected register.

5.4 Register bank address

Table 11 WRPRT Register Bank Address

Device register	Register bank	Start address	End address
WrprtRegs	WRPRT_REGS	-	-

Note: The register can only be accessed through coprocessor instructions and cannot be accessed through the bus.

5.5 Register address mapping

Table 12 WRPRT Register Address Mapping

Register name	Register description	Offset address	WRPRT
WRPRT_WRPRT	Write protection register	-	-

5.6 Register functional description

5.6.1 Write protection register (WRPRT_WRPRT)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	WRPRT0	R/W	Protecting registers accessed by CPU	0h
31:1	Reserved			0h

6 UID

6.1 Register bank address

Table 13 Register Bank Address

Device register	Register bank	Start address	End address
UIDRegs	UID_REGS	0x0810_AF70	0x0810_AF8F

6.2 Register address mapping

Table 14 UID Register Address Mapping

Register name	Description	Offset address	WRPRT
UID_PSRAND0	Pseudo random portion 0 of the UID	0x00	-
UID_PSRAND1	Pseudo random portion 1 of the UID	0x04	-
UID_PSRAND2	Pseudo random portion 2 of the UID	0x08	-
UID_PSRAND3	Pseudo random portion 3 of the UID	0x0C	-
UID_PSRAND4	Pseudo random portion 4 of the UID	0x10	-
UID_PSRAND5	Pseudo random portion 5 of the UID	0x14	-
UID_UNIQUE	Unique 32-bit number register of UID	0x18	-
UID_CHECKSUM	Checksum register of UID_PSRANDx and UID_UNIQUE	0x1C	-

6.3 Register functional description

6.3.1 Pseudo random portion register x of UID (UID_PSRANDx)

Offset address: 0x00+0x04*x (x= 0-5)

Reset type: N/A

Field	Name	R/W	Description	Reset value
31:0	RandomID	R	Psuedorandom portion of the UID UID [191:128]: Random value UID [127:40]: LOT ID, ASCLL code (data is aligned to higher bits; if the lower bit lacks data, it will be filled with zero, and stored in hexadecimal) UID [39:32]: Wafer ID, 8-bit unsigned number (stored in hexadecimal) UID [31:16]: X coordinate, BCD code (stored in decimal) UID [15:0]: Y coordinate, BCD code (stored in decimal)	0h

6.3.2 Unique 32-bit number register of UID (UID_UNIQUE)

Offset address: 0x18

Reset type: N/A

Field	Name	R/W	Description	Reset value
31:0	UniqueID	R	Unique portion of the UID This identifier is unique among all devices with the same PARTIDH.	0h

6.3.3 Checksum register of UID_PSRANDx and UID_UNIQUE (UID_CHECKSUM)

Offset address: 0x1C

Reset type: N/A

Field	Name	R/W	Description	Reset value
31:0	Checksum	R	Fletcher checksum of UID_PSRANDx and UID_UINIQUE Check value of UID CRC, obtained by checking UID_PSRAND0-5CRC.	0h

7 Power management

7.1.1 Chip power supply pins

Table15 Power Supply Names and Descriptions

Pin Name	Minimum value/V	Typical value/V	Maximum / V	Description
VDD	0.99	1.1	1.21	1.1 V digital logic power supply pin. It is recommended that a decoupling capacitor of at least 20 μF be placed near each VDD pin. When the internal regulator is not used, the value of the decoupling capacitor is determined by the system’s voltage regulation solution.
VDDA	3	3.3	3.63	3.3V analogue power supply pins. A decoupling capacitor of at least 2.2 μF must be placed on each pin to VSSA.
VDDIO	3	3.3	3.63	3.3V digital I/O power supply pins. A decoupling capacitor of at least 0.1 μF must be placed on each pin.
VSS		0		Digital ground
VSSA		0		Analogue ground

7.1.2 FLASH Power Supply

FLASH has two power supplies, VDDIO (3.3V) and VDD (1.1V). The VDDIO pin of the chip supplies power to the VDDIO of FLASH through a PAD.

7.1.3 Digital Core Power Supply

It can be configured via software to operate in one of the following modes to supply the required 1.1 V voltage to the digital core (VDD):

- External power supply
- Internal 1.1 V LDO voltage regulator (VREG)

The LDO is powered by VDDIO and generates the 1.1 V required to power the VDD pin. This function is enabled by pulling the VREGENZ pin low to VSS. As VREGENZ controls the LDO, it must be kept low after conversion. Setting VREGENZ high will disable the LDO.

7.2 PMU Module Description

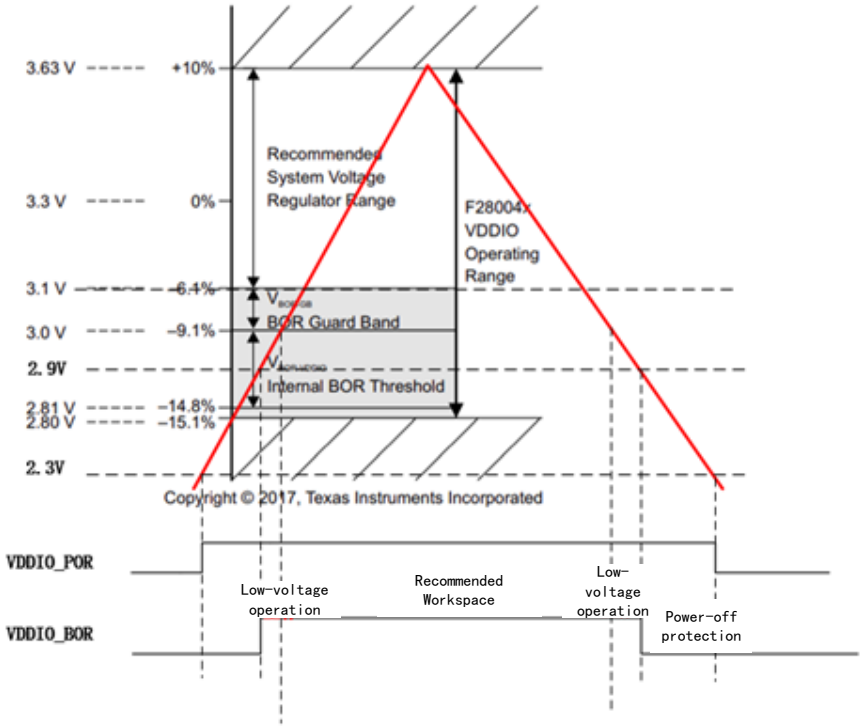
7.2.1 Overview

The Power Management Unit (PMU) handles all power management functions required

for operating the device.

7.2.2 System Design Specifications

Figure 2 I/O BOR operating region



Note:

(1) VDDIO ⩽ 2.3 V: the BG, LDO and other modules do not operate;

(2) 2.3V < VDDIO < 2.81V: the BG and 0.7V buffer begin to operate, whilst the LDO does not operate;

(3) 3V ⩽ VDDIO, all modules operate.

Table: Truth table for the16 at different operating voltages

Voltage Operating State	Voltage Range	VDDIO_POR	VDDIO_BOR	Chip Function
VDDIO	Operating voltage: 3.1 V–3.6 V	H	H	Chip operating normally (BG operating normally)
	Low-voltage specifications: 3.0V–3.1V	H	H	Chip operating at low voltage (transition region) (BG operating normally)

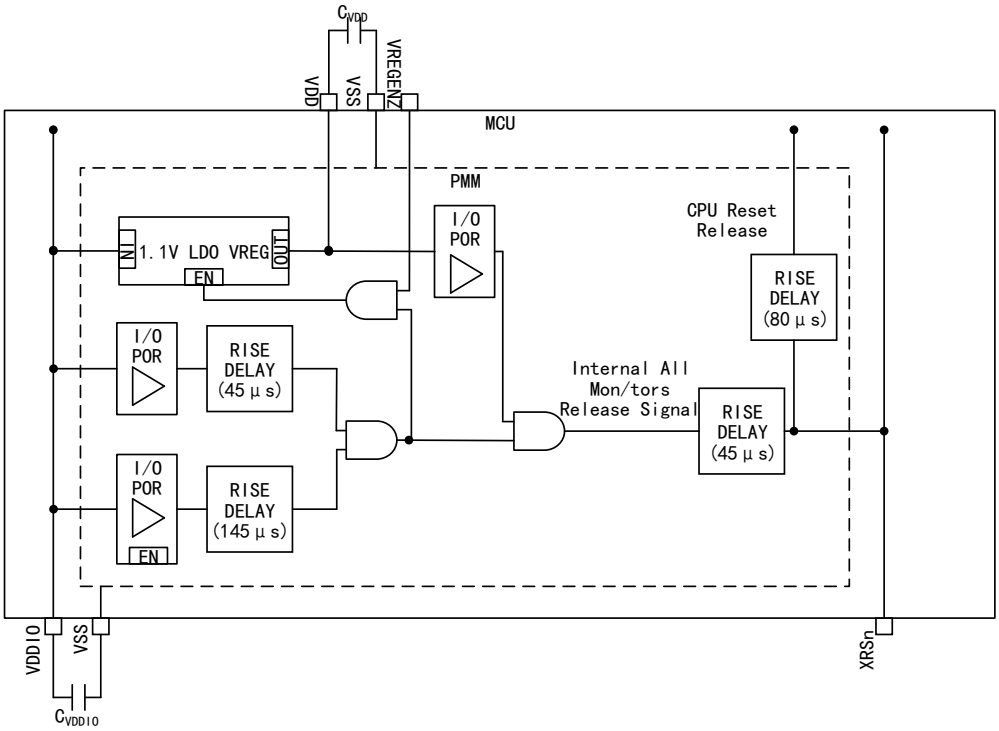
	Power-down protection: 2.81 V to 3V	H	L	Chip is in an under-voltage state, cannot operate normally BG operating normally
	Reset status indicator: ≤2.3V	L	L	Chip reset state; chip not operating BG is inactive

7.2.3 Power Management Module (PMM)

The Power Management Module (PMM) handles all power management functions required for the operation of the device.

The figure below shows a block diagram of the PMM. As can be seen, the PMM comprises several sub-components, which will be described in subsequent sections.

Figure3 Block diagram of the PMM structure



7.2.3.1 Power Rail Monitor

The PMM features voltage monitors on the power rails; if the voltage exceeds a set threshold during power-up, the XRSn signal is pulled high. These voltage monitors can also cause the XRSn signal to be pulled low if any voltage falls below a set threshold. The various voltage monitors will be described in subsequent sections.

Note: Where voltage monitors are not supported, it is recommended that an external monitor be used if the device requires monitoring of the supply voltage whilst the application is in operation. The three voltage monitors

I/O POR, I/O BOR and VDD POR must all release their respective outputs before the device begins operation (i.e. before XRSn goes high). If any voltage monitor trips, XRSn will be driven low.When any voltage monitor trips, the I/O remains in a high-impedance state.

I/O POR Monitoring

Power-on reset (I/O POR) is used to monitor the VDDIO supply rail. During chip power-up, this is the first monitor on VDDIO to release its trip.

I/O BOR monitoring

Undervoltage reset (I/O BOR) is also used to monitor the VDDIO power rail. During power-up, this is the second monitor to trip on VDDIO. I/O BOR has stricter tolerances than I/O POR.

If the voltage drops below the recommended operating voltage, the I/O BOR will trip and reset the chip; however, this function can be disabled by setting VMONCTL [BORLVMONDIS] to 1. The I/O BOR can only be disabled once the chip has fully booted. If the I/O BOR is disabled, the I/O POR will reset the chip should the voltage drop.

Note: The threshold for the I/O POR to trip is significantly lower than the minimum recommended voltage for VDDIO; therefore, it should not be used for device monitoring.

VDD POR Monitoring

VDD POR is used to monitor the VDD power rail. During chip power-up, if the voltage exceeds the set VDD trip level, the monitoring circuit will trip.

Note: VDD POR is set to a level below the recommended minimum voltage for VDD. If VDD needs to be monitored in the application, one should not rely on VDD POR monitoring; an external monitor should be used instead.

8 Device identification and configuration registers

8.1 Introduction

The device identification register and configuration register provide information about part number, product series, revision, pin number, qualification status, and device function availability. All device information is in DEV_CFG_REGS. The identification registers are PARTIDL, PARTIDH, and REVID.

8.2 Register bank address

Table 17 Register Bank Address

Device register	Register bank	Start address	End address
DevRegs	DEV_REGS	0x4001_3100	0x4001_33FF

8.3 Register address mapping

Table 18 Register Address Mapping

Register name	Register description	Offset address	WRPRT
PARTIDL	Device part identification number low register	0x10	-
PARTIDH	Device part identification number high register	0x14	-
REVID	Device version number register	0x18	-
DC3	Device function register 3	0x2C	-
DC4	Device function register 4	0x30	-
DC5	Device function register 5	0x34	-
DC8	Device function register 8	0x40	-
DC9	Device function register 9	0x44	-
DC10	Device function register 10	0x48	-
DC11	Device function register 11	0x4C	-
DC14	Device function register 14	0x58	-
DC15	Device function register 15	0x5C	-
DC17	Device function register 17	0x64	-
DC21	Device function register 21	0x74	-
DC23	Device function register 23	0x7C	-

Register name	Register description	Offset address	WRPRT
DC24	Device function register 24	0x80	-
DC25	Device function register 25	0x84	-
SOFTPRES2	Processing block software reset register 2	0x10C	√
SOFTPRES3	Processing block software reset register 3	0x110	√
SOFTPRES4	Processing block software reset register 4	0x114	√
SOFTPRES7	Processing block software reset register 7	0x120	√
SOFTPRES8	Processing block software reset register 8	0x124	√
SOFTPRES9	Processing block software reset register 9	0x128	√
SOFTPRES10	Processing block software reset register 10	0x12C	√
SOFTPRES13	Processing block software reset register 13	0x138	√
SOFTPRES14	Processing block software reset register 14	0x13C	√
SOFTPRES16	Processing block software reset register 16	0x144	√
SOFTPRES17	Processing block software reset register 17	0x148	√
SOFTPRES19	Processing block software reset register 19	0x150	√
SOFTPRES20	Processing block software reset register 20	0x154	√
PACKSEL	Select chip package register	0x25C	√

8.4 Register functional description

8.4.1 Device part identification number low register (PARTIDL)

Offset address: 0x10

Reset type: XRSn

Field	Name	R/W	Description	Reset value
5:0	Reserved			0h
7:6	QUAL	R	Criteria Select 00: Engineering sample (TMX) 01: Pilot product (TMP) 10: Fully qualified (TMS)	0h
10:8	PIN_COUNT	R	Pin Number Select 000: 32 pins 001: 48 pins 010: 64 pins 011: 80 pins 100-111: Reserved	X
12:11	Reserved			0h

Field	Name	R/W	Description	Reset value
14:13	INSTASPIN	R	00: InstaSPIN-MOTION + InstaSPIN-FOC 01: InstaPIN-FOC 10–11: NONE	0h
15	Reserved			0h
23:16	FLASH_SIZE	R	FLASH Size Select 00000000-00000011: Reserved 00000100: 256KB 00000101: 640KB 00000110-11111111: Reserved	4h
31:24	Reserved			0h

8.4.2 Device part identification number high register (PARTIDH)

Offset address: 0x14

Reset type: XRSn

Field	Name	R/W	Description	Reset value
7:0	Reserved			0h
15:8	FAMILY	R	Device series 00000000-00000011: Reserved 00000100: Single core 00000101-11111111: Reserved	4h
23:16	PARTNO	R	Device Part Number Refer to the data table	X
31:24	DEVICE_CLASS_ID	R	Device class ID	X

8.4.3 Device version number register (REVID)

Offset address: 0x18

Reset type: XRSn

Field	Name	R/W	Description	Reset value
15:0	REVID	R	Device Revision Load when OPT_KEY is 0x5A.	0h
31:16	Reserved			0h

8.4.4 Device function (DC3)

Offset address: 0x2C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	PWMx	R	PWMx function (x=1-7) 0: The device does not have this feature 1: The device has this feature	X

Field	Name	R/W	Description	Reset value
31:7	Reserved			X

8.4.5 Device function (DC4)

Offset address: 0x30

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	CAPx	R	CAPx function (x=1-3) 0: The device does not have this feature 1: The device has this feature	X
31:3	Reserved			X

8.4.6 Device function (DC5)

Offset address: 0x34

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	QEPx	R	QEPx function (x=1) 0: The device does not have this feature 1: The device has this feature	X
31:1	Reserved			X

8.4.7 Device function (DC8)

Offset address: 0x40

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	UART_A	R	UARTA function 0: The device does not have this feature 1: The device has this feature	X
1	UART_B	R	UARTB function 0: The device does not have this feature 1: The device has this feature	X
2	UART_C	R	UARTC function 0: The device does not have this feature 1: The device has this feature	X
31:3	Reserved			X

8.4.8 Device function (DC9)

Offset address: 0x44

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	SPI_A	R	SPIA function 0: The device does not have this feature 1: The device has this feature	X
31:1	Reserved			X

8.4.9 Device function (DC10)

Offset address: 0x48

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	I2C_A	R	I2CA function 0: The device does not have this feature 1: The device has this feature	X
1	I2C_B	R	I2CB function 0: The device does not have this feature 1: The device has this feature	X
31:2	Reserved			X

8.4.10 Device function (DC11)

Offset address: 0x4C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	CAN_A	R	CANA function 0: The device does not have this feature 1: The device has this feature	X
3:1	Reserved			X
4	CANFD_A	R	CANFDA function 0: The device does not have this feature 1: The device has this feature	X
31:5	Reserved			X

8.4.11 Device function (DC14)

Offset address: 0x58

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	ADC_A	R	ADCA function 0: The device does not have this feature 1: The device has this feature	X
1	Reserved			X

Field	Name	R/W	Description	Reset value
2	ADC_C	R	ADCC function 0: The device does not have this feature 1: The device has this feature	X
31:3	Reserved			X

8.4.12 Device function (DC15)

Offset address: 0x5C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	COMPx	R	COMPx function (x=1–4) 0: The device does not have this feature 1: The device has this feature	X
31:4	Reserved			X

8.4.13 Device function (DC17)

Offset address: 0x64

Reset type: XRSn

Field	Name	R/W	Description	Reset value
15:0	Reserved			X
16	DAC_A	R	DACA function 0: The device does not have this feature 1: The device has this feature	X
17	DAC_B	R	DACB function 0: The device does not have this feature 1: The device has this feature	X
31:18	Reserved			X

8.4.14 Device function (DC21)

Offset address: 0x74

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	FLBx	R	FLBx function (x=1-4) 0: The device does not have this feature 1: The device has this feature	X
31:4	Reserved			X

8.4.15 Device function (DC23)

Offset address: 0x7C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	LIN_A	R	LINA function 0: The device does not have this feature 1: The device has this feature	X
31:1	Reserved			X

8.4.16 Device function (DC24)

Offset address: 0x80

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	PMBUS_A	R	PMBUSA function 0: The device does not have this feature 1: The device has this feature	X
31:1	Reserved			X

8.4.17 Device function (DC25)

Offset address: 0x84

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	DCCOMP	R	DCCOMP function(x=1) 0: The device does not have this feature 1: The device has this feature	X
31:1	Reserved			X

8.4.18 Processing block software reset register 2 (SOFTPRES2)

Offset address: 0x10C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	PWMx	R/W	PWMx reset Configuration (x=1-7) 0: Reset is determined by normal device reset structure 1: Reset	0h
31:7	Reserved			0h

8.4.19 Processing block software reset register 3 (SOFTPRES3)

Offset address: 0x110

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	CAPx	R/W	CAPx reset Configuration (x=1-3) 0: Reset is determined by normal device reset structure 1: Reset	0h

Field	Name	R/W	Description	Reset value
31:3	Reserved			0h

8.4.20 Processing block software reset register 4 (SOFTPRES4)

Offset address: 0x114

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	QEPx	R/W	QEPx reset Configuration (x=1) 0: Reset is determined by normal device reset structure 1: Reset	0h
31:1	Reserved			0h

8.4.21 Processing block software reset register 7 (SOFTPRES7)

Offset address: 0x120

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	UART_A	R/W	UARTA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
1	UART_B	R/W	UARTB reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
2	UART_C	R/W	UARTC reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:3	Reserved			0h

8.4.22 Processing block software reset register 8 (SOFTPRES8)

Offset address: 0x124

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SPI_A	R/W	SPIA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:1	Reserved			0h

8.4.23 Processing block software reset register 9 (SOFTPRES9)

Offset address: 0x128

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	I2C_A	R/W	I2CA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
1	I2C_B	R/W	I2CB reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:2	Reserved			0h

8.4.24 Processing block software reset register 10 (SOFTPRES10)

Offset address: 0x12C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CAN_A	R/W	CANA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
3:1	Reserved			0h
4	CANFD_A	R/W	CANFDA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:5	Reserved			0h

8.4.25 Processing block software reset register 13 (SOFTPRES13)

Offset address: 0x138

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ADC_A	R/W	ADCA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
1	Reserved			0h
2	ADC_C	R/W	ADCC reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:3	Reserved			0h

8.4.26 Processing block software reset register 14 (SOFTPRES14)

Offset address: 0x13C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	COMPx	R/W	COMPx reset Configuration (x=1-4) 0: Reset is determined by normal device reset structure 1: Reset	0h
31:4	Reserved			0h

8.4.27 Processing block software reset register 16 (SOFTPRES16)

Offset address: 0x144

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	Reserved			0h
16	DACA	R/W	DACA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
17	DACB	R/W	DACB reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:18	Reserved			0h

8.4.28 Processing block software reset register 17 (SOFTPRES17)

Offset address: 0x148

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	FLBx	R/W	FLBx reset Configuration (x=1-4) 0: Reset is determined by normal device reset structure 1: Reset	0h
31:4	Reserved			0h

8.4.29 Processing block software reset register 19 (SOFTPRES19)

Offset address: 0x150

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	LIN_A	R/W	LINA reset Configuration 0: Reset is determined by normal device reset structure 1: Reset	0h
31:1	Reserved			0h

8.4.30 Processing block software reset register 20 (SOFTPRES20)

Offset address: 0x154

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1:0	PMBUS_A	R/W	PMBUSA reset Configuration(x=1-7) 0: Reset is determined by normal device reset structure 1: Reset	0h
31:7	Reserved			0h

8.4.31 Select chip package register (PACKSEL)

Offset address: 0x25C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	PACKSEL	R/W	Chip Package Select 0x1: 80pin 0x2: 64pin 0x4: 48pin 0x5: 32pin Other: Reserved	0h
31:4	Reserved			0h

9 Low-power mode

G32R502 supports the two low-power modes of IDLE and HALT. These two low-power modes are entered by configuring the LPMCR register and executing WFI or WFE instructions. Besides, the SLEEPDEEP bit of the CPU SCB register needs to be configured according to the low-power mode. The low-power modes cannot be entered while FLASH is performing erase or write operations. When entering HALT mode, all CPU and peripheral activities will be stopped, including current ongoing communication transmission and control algorithms.

Before getting ready to enter HALT mode, the application program needs to ensure that the system has been ready to enter the low-power mode. Before entering HALT mode, it is also necessary to check the value of the GPIODAT register to ensure that the HALT wake-up pin selected through the GPIOLPMSEL0/1 register is currently in a non-valid state, so that the system can smoothly enter HALT mode.

9.1 IDLE

In IDLE mode, CPU will enter SLEEPING mode. At this point, the CPU clock is turned off, but the system clock and peripheral clock are still reserved. Therefore, IDLE mode can be used to save power while the CPU is waiting for peripheral events or interrupts. The wake-up method varies depending on the instruction executed when entering IDLE.

The way to enter IDLE mode:

- Set LPMCR.LPM to 0x0
- Execute WFI/WFE instructions

Note: There are two options to enter IDLE mode according to the configuration of the SLEEPINEXIT bit.

(1) If the SLEEPINEXIT bit is cleared, the MCU will immediately enter IDLE mode when executing WFI or WFE instructions.

(2) If the SLEEPINEXIT bit is set, the MCU will immediately enter IDLE mode after exiting the lowest priority ISR.

9.2 HALT

HALT mode is a global low-power mode, which can turn off the majority of system clocks and allows some analog clock sources and analog IP to be turned off. Entering HALT mode will not automatically power down XTAL, and waking up from HALT mode will not automatically power on XTAL. XTAL can be powered down through software implementation by configuring the XTALCR.OSCOFF bit.

For applications that need to maintain the minimum power consumption in HALT mode, the software needs to turn off XTAL before entering HALT mode. It is important to note

that if the current system clock source is XTAL, the system clock source needs to be switched to INTOSC1 or INTOSC2 first.

Each GPIO can be configured as wake-up pins of HALT mode. However, the watchdog can still be configured to work state in HALT mode. And it can be configured to generate watchdog reset when a timeout event occurs.

The way to enter HALT mode:

(1) Enable WDT_WAKE in NVIC.

(1) Set LPMCR.LPM to 0x2, GPIOLPMSEL0 and GPIOLPMSEL1, and connect the selected GPIO to the LPM module.

(2) Set CPU SCB.SLEEPDEEP.

(3) Set CLKSRCCTL1.WDHALTI to enable the watchdog timer, and power on INTOSC1 and INTOSC2 in HALT.

(4) Reset CLKSRCCTL1.WDHALTI, turn off the watchdog timer, and power down INTOSC1 and INTOSC2 in HALT.

(5) Execute WFI/WFE instructions to enter HALT.

Note: If the current FLASH memory is performing a write operation, entering HALT mode will be delayed until the write operation of FLASH memory is completed. In addition, if the processor is accessing peripherals on the APB, entering HALT mode will be delayed until APB peripheral access is completed.

Table 19 Low-power Mode

Module name	Enter	Wake-up	Impact on the clock
IDLE	The LPM bit of the LPMCR register is 2'b00 + WFI	Arbitrary interrupt	CPU CLK is turned off, with no impact on other clocks or analog clock sources
	The LPM bit of the LPMCR register is 2'b00 + WFE	Wake-up event	
HALT	The LPM bit of the LPMCR register is 2'b10 + CPU's SCB->SLEEPDEEP bit set to 1 + WFE/WFI	GPIOLPMSEL selects GPIO	Turn off the majority of system clocks, including CPU CLK and peripheral clocks, except the analog clock source. According to power consumption requirements, analog clock sources such as XTAL, PLL, INTOSC1, and INTOSC2 are allowed to be further turned off.

Table 20 Low-power Mode Clock State

Clock domain/module	IDLE	HALT
SYSCLK	Active	Gating
CPU_CLK	Gating	Gating
APBCLK	Active	Gating
WDTCLK	Active	Gating if CLKSRCTL1.WDHALTI = 0
PLL	Power-on	The software must power off the phase-locked loop before entering HALT.
INTOSC1	Power-on	Power off if CLKSRCTL1.WDHALTI = 0
INTOSC2	Power-on	Power off if CLKSRCTL1.WDHALTI = 0
XTAL(1)	Power-on	Power-on
FLASH(2)	Power-on	Power-on

Note:

(1) XTAL will not be powered off by hardware under any LPM. It can be powered off by setting XTALCR.OSCOFF bit to 1 through software. If XTAL is not required, this operation can be completed at any time in the application program.

(2) Flash module will not be powered off by hardware under any LPM. If the application needs it, it can be powered off by software.

10 Reset

Table 21 Reset Source

Name	CPU core reset	JTAG/Debugging logic reset	Peripheral reset	IOs	XRSn output
POR	√	√	√	Hi-Z	√
WDRSn	√	×	√	Hi-Z	√
NMIWDRSn	√	×	√	Hi-Z	√
SCCRESETn	√	×	√	Hi-Z	×
SYSRS (debugger reset)	√	×	√	Hi-Z	×
XRSn pin	√	×	√	Hi-Z	-

Note: "n" in the table indicates that the low level is valid.

Reset can be divided into chip-level reset and system reset. The chip-level reset is used to reset all or almost all devices, while system reset is used to reset a large subset of devices, but reserve some system-level configurations. After reset, the RESC register is updated, and the bits in this register can only be cleared by power-on reset or by the relevant bits in the RESCCLR register. Individual bits are cleared by the boot ROM as part of the boot routine.

10.1 Functional description

10.1.1 Power-on reset (POR)

In order to suppress the faults on GPIO, the power-on reset circuit needs to establish a clean reset during the power-on period. The XRSn pin remains low during POR period. XRSn generally remains low for a sufficient period of time to reset other system IC. If longer pulses are required, the XRSn pin can be externally driven to a low level to provide correct reset duration. POR can reset all actions of XRSn, debugging logic used by JTAG port, RESC registers, NMISHDFLG register and X1CNT register. After performing the POR operation, the POR bit and XRSn bit are set. Then they will be cleared by boot ROM.

10.1.2 External reset (XRSn)

External reset is main chip-level reset of the device. It is used to reset the CPU, all peripherals, I/O pin configuration, and most system control registers. XRSn can be driven by an external source, but is driven internally during watchdog, NMI, and power-on reset. There is an open-drain pin that can be used to drive the reset pins of other IC in the application. When XRSn is driven to low, the XRSn bit will be set and then cleared by boot ROM.

10.1.3 Debugger reset (SYSRSn)

Sometimes it is necessary to reset the CPU and peripherals without disconnecting the debugger connection or interrupting the system-level configuration. The debuggers of IDE such as Keil can be used to trigger a separate subsystem reset of the CPU. SYSRSn can reset CPU, peripherals, I/O pin configurations, and many system control registers.

10.1.4 Watchdog reset (WDRSn)

If the watchdog timer is not served by the CPU within the specified time, it can selectively trigger a reset. WDRSn generates an XRSn that lasts for 810 INTOSC1 cycles. After the watchdog is reset, the WDRSn bit in RESC is set. If both XRSn and WDRSn bits are set simultaneously in RESC, the watchdog initiates a reset. If only the XRSn bit is set in RESC, the watchdog is reset from XRSn.

10.1.5 NMI watchdog reset (NMIWDRS)

NMI is used to detect hardware errors in the system. It has a watchdog timer, which will trigger a reset if the CPU does not respond to an error within the specified time. NMIWDRSn generates an XRSn that lasts for 810 INTOSC1 cycles. After the NMI watchdog is reset, the NMIWDRSn bit is set.

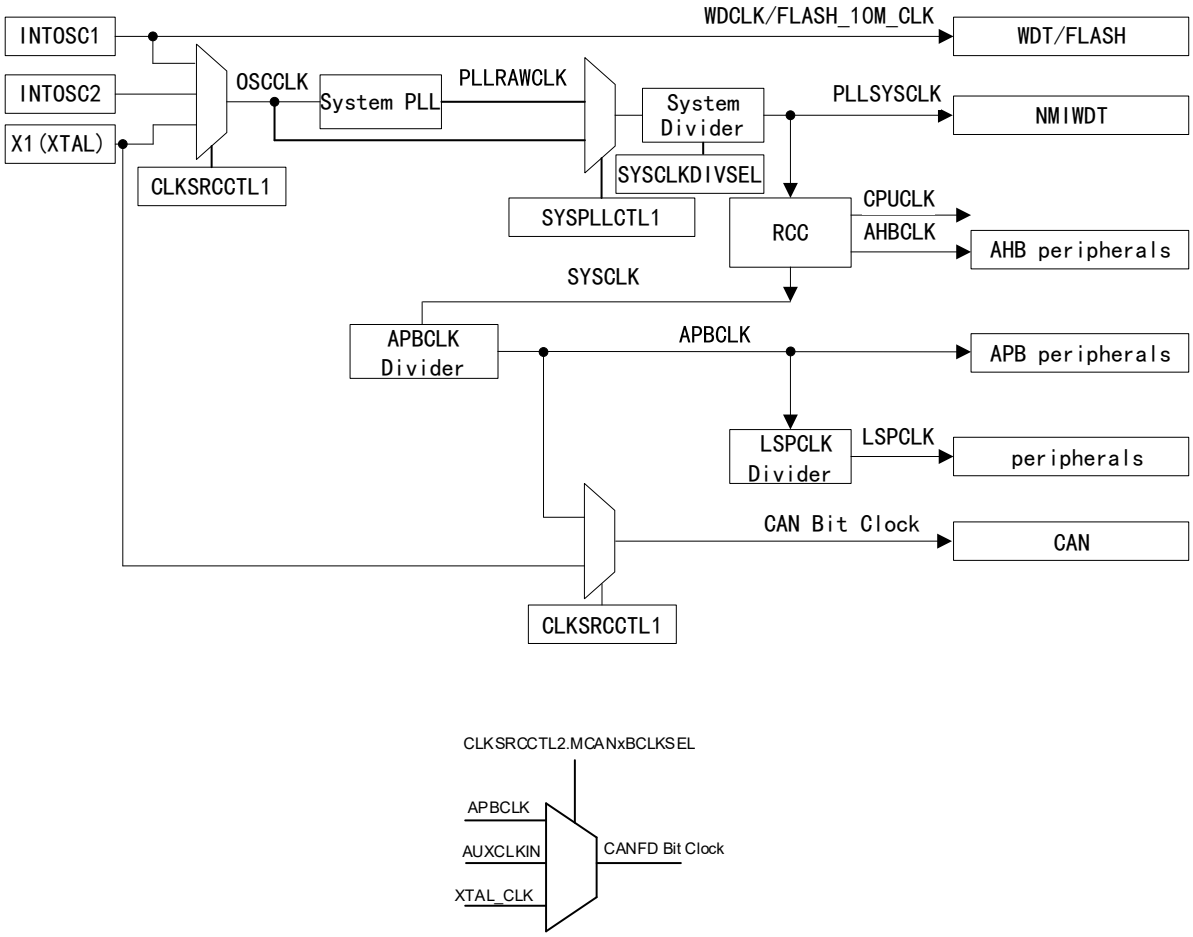
10.1.6 DCS security code copy reset (SCCRESETn)

DCS can prevent read access to certain areas of flash memory. It provides ROM function to securely access these memory areas, which can make CRC check more convenient. If vector acquisition occurs in secure copy or CRC function, DCS will trigger a reset. SCCRESETn is similar to SYSRS. In order to deny access from potential attackers, the debug logic will be reset simultaneously. Disabling interrupts before calling a function can prevent security vulnerabilities. After security reset, the SCCRESETn bit will be set.

11 Clock

11.1 Structure block diagram

Figure 4 Clock Source



The PLL output frequency can support up to 250MHz clock output (PLLRAWCLK). The clock control unit module adds APB clock division and dual-core system clock control in addition to the original control logic. The calculation formula is as follows:

$$f_{PLLRAWCLK}=(f_{osccllk})*NF/ODIV$$

$$NF=IMULT+FMULT/4$$

“IMULT” represents integer part of PLL frequency multiplication; “FMULT” represents decimal part of PLL frequency multiplication; “ODIV” represents PLL output divider. For detailed functional description, please refer to the "SYSPLL Frequency Multiplier Register".

11.2 Functional description

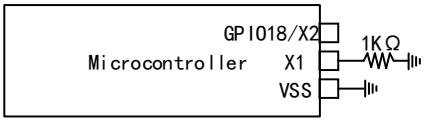
11.2.1 Clock source

Main internal oscillator (INTOSC2)

The main internal clock source INTOSC2 is the default system clock during reset. When powered on, the clock of the device is controlled by the on-chip 10MHz INTOSC2. Used to run boot ROM. When INTOSC2 is used as the system clock source, GPIO18(X2) can be used as GPIO. At this point, a 1k pull-down resistor needs to be connected to X1. Due to its interaction with the oscillator circuit, GPIO18 has different electrical characteristics from those of other GPIO. It should be noted that the frequency tolerance of INTOSC2 is relatively loose, and using a CAN module requires an external oscillator to meet the timing requirements of CAN.

Note: As GPIO19 shares an I/O pin with X1, it cannot be used as a GPIO even when XTAL is not in operation. GPIO18 can be used as a GPIO when XTAL is disabled and an external 1 kΩ pull-down resistor is connected to X1.

Figure 5 Use GPIO18 When INTOSC2 is SYSCLK Source



Backup internal oscillator (INTOSC1)

The 10MHz INTOSC1 is a backup clock source, which generally only controls the clocks of watchdog timers and missing clock detection circuits. If MCD is enabled and a missing system clock is detected, the system phase-locked loop will be bypassed, and all system clocks will automatically connect to INTOSC1, or INTOSC1 can be manually selected as the system clock source.

External oscillator (XTAL)

XTAL can serve as the master system clock source and CAN bit clock source. The external clock source uses X1 and X2/GPIO18 pins.

According to the ALT setting mode, three external clock sources can be supported:

Table 22 ALT Mode

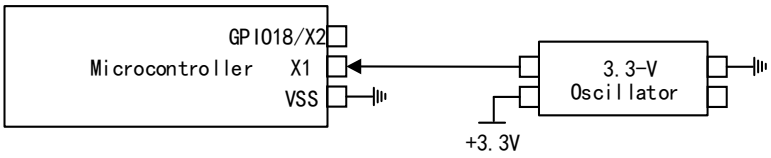
XTALCR[OSCOFF]	SE	Mode	Whether GPIO can be used on X2
0	0	Crystal method: Quartz crystal connects to X1/X2	×
0	1	Single-ended mode: X1 is an external clock	×

XTALCR[OSCOFF]	SE	Mode	Whether GPIO can be used on X2
1	0	The oscillator is disabled	√ (When GPIO18 is used, a 1KΩ pull-down is required on X1)
1	1	Single-ended mode: X1 is an external clock (there is approximately 1KΩ pull-down on X1, and the external single-ended clock should be able to drive this load)	×

According to the above table, the following three external clock sources can be supported:

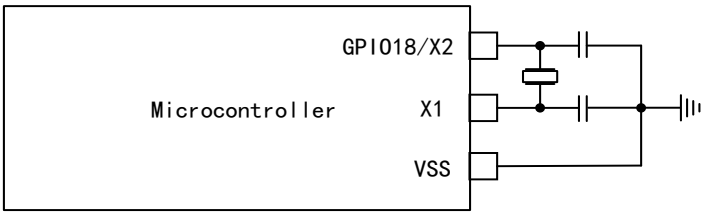
(1) Single-ended 3.3V external clock. The clock signal is connected to X1, and X2/GPIO 18 cannot be used as GPIO.

Figure 6 Single-ended 3.3V External Clock



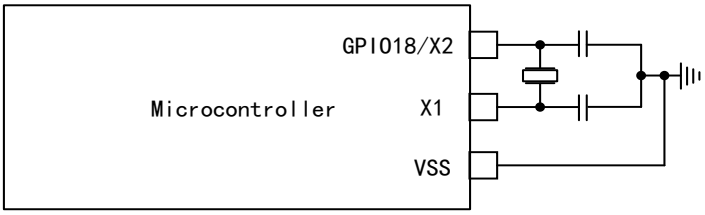
(2) External crystal. The crystal is connected to X2 through X1, and the load capacitor is connected to VSS.

Figure 7 External Crystal



(3) External resonator. The resonator is connected to X2 across X1, and the ground is connected to VSS.

Figure 8 External Resonator



11.2.2 Generated clock

The clock sources can be multiplied and decomposed using a phase-locked loop to

generate the desired clock frequency. In this process, a set of generated clocks will be generated.

Oscillator clock (OSCCLK)

One of the three clock sources must be selected as the main reference clock (OSCCLK) for the CPU and most peripherals. OSCCLK can be used directly or fed through the system PLL to achieve higher frequencies. OSCCLK is the default system clock during reset and is connected to INTOSC2.

System PLL output clock (PLLRAWCLK)

PLLRAWCLK is the output of PLL voltage controlled oscillator (VCO) The device is allowed to run at the maximum rated operating frequency and generate the master system clock in most applications. This PLL takes OSCCLK as the reference and has a fraction multiplier.

11.2.3 Equipment clock domain

Provide clock input for various modules in the device. It can be connected to the generated clock directly or through an additional voltage divider.

System clock (PLLSYSCLK)

PLLSYSCLK is a separate clock domain of NMI watchdog timers, and it supports up to 200MHz clock output (PLLRAWCLK). Gate control is performed in HALT mode. Although the names are different, PLLSYSCLK can be connected to system PLL (PLLRAWCLK) or OSCCLK. The selected clock source runs through a frequency divider and is configured by the SYSCLKDIVSEL register.

CPU clock (CPUCLK)

CPUCLK and SYSCLK are both generated by PLLSYSCLK and are synchronous clocks. The switch of CPU0CLK is controlled by the clock control unit. When CPU is in SLEEPING (corresponding to IDLE mode) or SLEEPDEEP (corresponding to HALT mode), the clock control unit turns off CPU0CLK. When exiting IDLE or HALT mode, the clock control unit turns on CPUCLK.

CPU subsystem clock (SYSCLK)

SYSCLK is generated by the SYSCLK divider. It is used for the operation of all peripherals, buses, and SRAM except for WDT. The switch of SYCLK is controlled by the clock control unit. When the system enters HALT mode, the clock control unit will automatically turn off SYSCLK to reduce power consumption. At this point, all peripheral clocks are turned off (not affected by the current PCLKCR configuration).

APBCLK

APBCLK is obtained by dividing SYSCLK frequency and is the synchronous clock of the system clock. Since peripherals are unable to reach the same operating frequency (for example, 160MHz) as the CPU, namely AHB bus, SYSCLK frequency needs to be divided to ensure that the peripherals can function properly. APBCLK supports frequency division of 1 and 2. After power on, it is 2 frequency division by default. When the system needs to switch to a frequency greater than 100MHz, it is necessary to first configure APBCLK to at least 2 frequency division to ensure that APBCLK does not exceed the operating frequency of the peripherals.

Note:

(1) Before configuring APB frequency division, ensure that DMA is in IDLE state.

(2) After configuring the frequency division, it is necessary to read the frequency division configuration register and insert 10 nop instructions, so as to ensure that read and write access to APB peripherals is performed after the frequency division switch is completed.

Low-speed peripheral clock (LSPCLK)

The UART and SPI modules are connected to a shared clock distributor, which generates a low-speed peripheral clock (LSPCLK) from APBCLK. LSPCLK uses a 1/4 frequency divider by default, and this ratio can also be changed using the LOSPCP register. The module can communicate at a much slower bit rate than CPU frequency. Each clock of UART and SPI modules can be independently gated using PCLKCRx registers.

TMR2 clock (TMR2CLK)

The timer is connected to SYSCLK by default. But TMR2 can be configured to connect to INTOSC1, INTOSC2, or XTAL using the TMR2CLKCTL register. At the same time, a separate pre-range divider is provided for TMR2.

CAN bit clock

The frequency tolerance required for bit timing settings and network configuration of CAN bit clock can be accurate to 0.1%. The frequency of the CAN bit clock cannot be higher than the frequency of the APBCLK. Due to the inaccuracy of the master system clock in the form of APBCLK, the bit clock can be connected to XTAL using the CLKSRCCTL2 register. Each CAN module can be selected independently.

CAN FD bit clock

The CAN FD bit clock frequency is dependent on the communication rate; at the maximum communication rate, 32 MHz is required. Therefore, CAN FD also supports

using GPIO29 (AUXCLK) as the bit clock source for CAN FD.

11.2.4 Clock connection

Table 23 Clock Connection

Clock Domains	Module
PLLSYSCLK	NMIWDT
CAN Bit Clock	CAN
CAN FD Bit Clock	CAN FD
WDTCLK(INTOSC1)	WDT
LSPCLK	UART
	SPI
CPUCLK	CPU
	Flash
	FPU
	TMU
	VCU
	TCM (i.e. ITCM and DTCM)
	Boot ROM
SYSCLK	GPIO input synchronisation
	CFGSMS
	NVMC
	DCS
	PWM
	EXTI
	SYSC
	TMR
	DMA
	Analogue subsystem
	ADC
	DAC
	COMP
	QEP
	CAP

Clock Domains	Module
	FLB
APBCLK	DCCOMP
	SPI
	I2C
	UART
	LIN
	CAN
	PMBUS
	CAN FD

11.2.5 Clock source and PLL

The application program requirements determine the clock configuration. The requirements for application performance, power consumption, total system cost, and EMC determine the desired CPU frequency, whether CAN is required, and what type of external oscillator or clock source will be used. If CAN is required, an external clock source with precise frequency shall be used as the reference clock. Otherwise, only INTOSC2 will be used to eliminate the need for more external components.

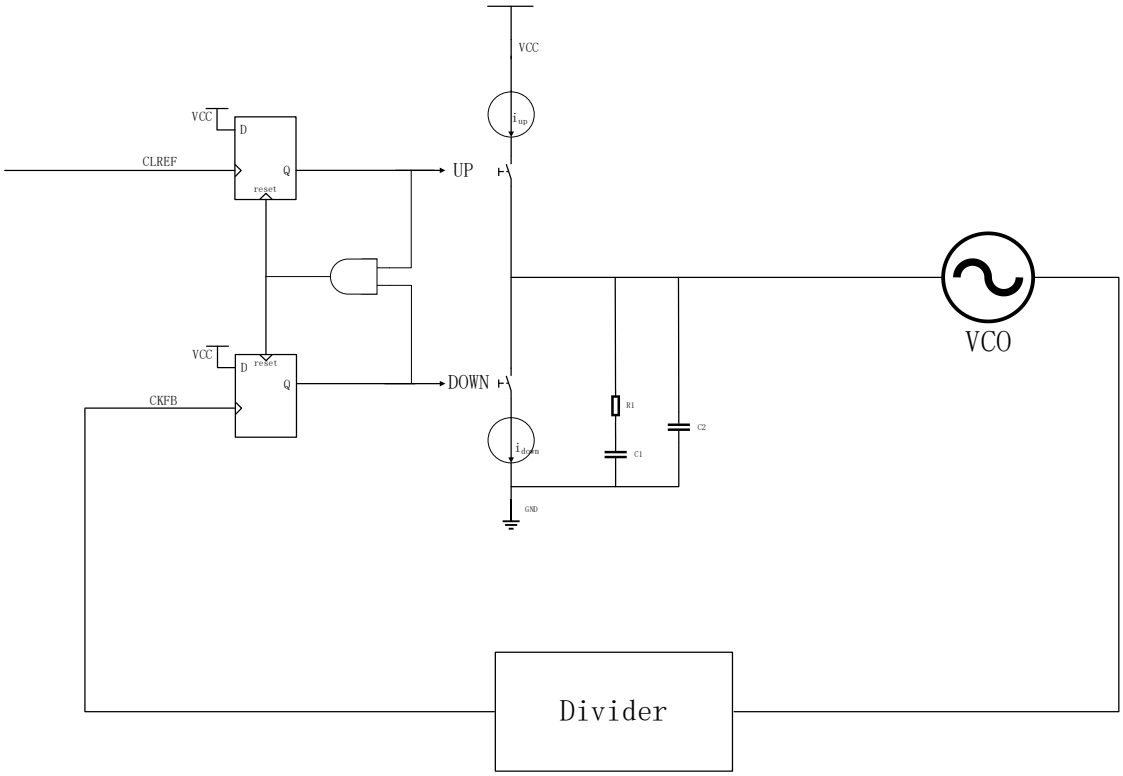
11.2.5.1 Description of the analog PLL module

Overview

A PLL circuit comprises a current bias circuit, a PFD (phase-frequency detector) circuit, a CP (charge pump) circuit, an equivalent LPF (low-pass filter) circuit, a VCO (voltage-controlled oscillator) circuit, a divider circuit, and so on.This phase-locked loop provides a clock for the system, with a maximum frequency of 200 MHz and a minimum frequency of 15 MHz; the feedback division ratio ranges from 6 to 127.75, with a minimum feedback division step of 0.25; the output division ratio ranges from 2 to 8, with a minimum output division step of 1.

Module Block Diagram

Figure9 PLL Structure



11.2.6 System clock setting

By default, INTOSC2 is used as the system clock (PLLSYSCLK). The steps to configure the required clock for the application program are as follows:

(1) Write OSCCLKSRCSEL bit to select the reference clock source. Clock source modification needs to wait for at least 300 NOP instructions to take effect. To enable XTAL, please operate according to the instructions in 11.2.9.

(2) If PLL is required, use the SysCtl_setClock() function to configure the PLL clock.

(3) Write LSPCLK bit to select LSPCLK frequency division.

(4) To switch CAN BIT CLK, write CLKSRCCTL2.

(5) Write the peripheral clock required to enable PCLKCRx register.

The system clock configuration can be changed during running. Changing the clock source of OSCCLK will automatically bypass the PLL and set the multiplication factor to zero. Changing the multiplication factor of PLL from a non-zero value to other value will temporarily bypass the PLL until PLL is relocked. After the clock source of OSSCLK is changed, it is necessary to wait for at least 300 CPU clock cycles.

When the system clock is greater than 100MHz, the APB needs to be configured with 2 divisions.

11.2.7 Select PLL settings

The PLL is divided into frequency multiplication and frequency division, and the specific parameters and their relationship are as follows:

Table 24 Parameter Description

Parameter	Description
f_{OSCCLK}	System oscillator clock frequency
IMULT	Integer part of multiplication
FMULT	Decimal part of multiplication
ODIV	PLL output frequency divider
SYSCLKDIVSEL	System clock frequency divider

The formula is as follows:

$f_{PLLSYSCLK} = f_{OSCCLK} * (IMULT + FMULT) / (ODIV * PLLSYSCLKDIV)$

Multiple combinations of multiplication and division can produce the same output frequency. However, the product of the reference clock frequency and the VCO frequency cannot exceed the range of f_{VCO}. The quotient of dividing the frequency of the voltage controlled oscillator by the output frequency of the PLL cannot exceed the range of $f_{PLLRAWCLK}$. The system clock frequency cannot exceed the range of f_{SYSCLK}. Oscillation tolerance is not allowed.

11.2.8 External clock output (XCLKOUT)

The output characteristics of external clock allows for direct observation of the clock by connecting the clock to an external pin. The clock sources include PLLSYSCLK, PLLRAWCLK, SYSCLK, APBCLK, FLPUMPOSC, INTOSC1, INTOSC2 and XTAL. To use XCLKOUT, it is necessary to first configure the CLKSRCCTL3 register to select the clock source, then use the XCLKOUTDIVSEL register to select the output frequency divider, and then connect GPIO16 or GPIO18 to multiplex channel 11.

11.2.9 Use of external crystal or resonator

GPIO18 is on X2 pin. When powered on, the X2 pin is in GPIO mode, and the on-chip crystal oscillator is powered off.

The steps to switch the pin to X2 mode and enable the oscillator are as follows:

(1) Clear the XTALCR[OSCOFF] bit.

(2) Wait for the crystal to be powered on. (Usually 1 millisecond needs to be waited for, depending on the crystal)

(3) Keep writing 1 to the CLR bit of X1CNT until the count value in the X1CNT register is not equal to 1023, and then wait for the count value to become 1023. Repeat this process three times.

(4) Write the OSCCLKSRCSEL bit and select XTAL as the OSCCLK source.

(5) Check the MCLKSTS bit of the MCDCR register. If MCLKSTS is 1, more time needs to be waited for, for the oscillator to power on:
- Write MCLKCLR bit to clear MCLKSTS
- Repeat steps 2-5 (the device cannot be reset at this point; otherwise the oscillator will be powered off, and the operation needs to be performed again from Step 1)
- If the oscillator is not powered on within 10 milliseconds, there is a real clock fault

(6) If MCLKSTS is 0, it indicates that the oscillator has started successfully and the system clock is from XTAL.

11.2.10 Use of external oscillator

The process of using an external oscillator connected to the X1 pin is similar to the process of using a crystal or resonator:

(1) Clear the XTALCR[OSCOFF] bit.

(2) Set the SE bit to enable single-ended mode.

(3) Keep writing 1 to the CLR bit of X1CNT until the count value in the X1CNT register is not equal to 1023, and then wait for the count value to become 1023. Repeat this process three times.

(4) Write the OSCCLKSRCSEL bit and select XTAL as the OSCCLK source.

(5) If the MCLKSTS bit is 1, it indicates a fault of external oscillator or device.

(6) If MCLKSTS is 0, it indicates that the oscillator started successfully, and the system clock is generated from XTAL.

(7) Missing clock detection (MCD)

The MCD subsystem uses INTOSC1 as a reference clock to detect OSCCLK faults, only detects loss of OSCCLK and does not detect frequency drift. OSCCLK drives a 7-bit counter MCDPCNT. INTOSC1 drives a 13-bit counter MCDSCNT. When MCDPCNT overflows, MCDSCNT is reset. Therefore, if OSCCLK exists and its frequency is greater than 1/64 of INTOSC1 frequency, MCDSCNT will never overflow. If OSCCLK stops, MCDSCNT will overflow and clock loss will be detected at this point.

When the MCD subsystem detects loss of OSCCLK:
- MCLKSTS is set, and OSCCLKSRCSEL bit does not work

- CLOCKFAIL signal is pulled up, a trip event is generated to the PWM module and NMI is triggered
- To prevent further detection of clock loss, the MCDSCNT counter will be frozen
- The system clock no longer comes from PLL, and the OSCCLK clock source is switched to INTOSC1. PLLMULT is automatically reset to zero

Writing 1 to MCLKCLR can clear the detected clock loss. At the same time, the function of OSCCLKSRCSEL bit will be restored and the MCD counter will be reset. To lock the PLL in case of clock loss, configure OSCCLKSRCSEL, select INTOSC1 as the OSCCLK clock source, clear the detected clock loss, and write to the PLL register.

11.3 Register bank address

Table 25 Register Bank Address

Device register	Register bank	Start address	End address
ClkRegs	CLK_REGS	0x4001_3400	0x4001_35FF

11.4 Register address mapping

Table 26 Register Address Mapping

Register name	Register description	Offset address	WRPRT
CLKCFGLOCK1	CLK lock register	0x04	√
CLKSRCCTL1	Clock source control register 1	0x10	√
CLKSRCCTL2	Clock source control register 2	0x14	√
CLKSRCCTL3	Clock source control register 3	0x18	√
SYSPLLCTL1	SYSPLL control register 1	0x1C	√
SYSPLLMULT	SYSPLL multiplier register	0x28	√
SYSPLLSTS	SYSPLL flag register	0x2C	-
SYSCLKDIVSEL	System clock divider register	0x44	√
XCLKOUTDIVSEL	XCLKOUT divider register	0x50	√
LOSPCP	Low-speed clock source prescaler register	0x58	√
MCDCR	Missing clock detection control register	0x5C	√
X1CNT	10-bit counter on X1 clock	0x60	-
XTALCR	XTAL control register	0x64	√

11.5 Register functional description

11.5.1 CLK lock register (CLKCFGLOCK1)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CLKSRCCTL1	R/WSonce	CLKSRCCTL1 register Lock 1: Unlocked 1: Locked	0h
1	CLKSRCCTL2	R/WSonce	CLKSRCCTL2 register Lock 1: Unlocked 1: Locked	0h
2	CLKSRCCTL3	R/WSonce	CLKSRCCTL3 register Lock 1: Unlocked 1: Locked	0h
3	SYSPLLCTL1	R/WSonce	SYSPLLCTL1 register Lock 1: Unlocked 1: Locked	0h
5:4	Reserved			0h
6	SYSPLLMULT	R/WSonce	SYSPLLMULT register Lock 1: Unlocked 1: Locked	0h
10:7	Reserved			0h
11	SYSCLKDIVSEL	R/WSonce	SYSCLKDIVSEL register Lock 1: Unlocked 1: Locked	0h
14:12	Reserved			0h
15	LOSPCP	R/WSonce	LOSPCP register Lock 1: Unlocked 1: Locked	0h
16	XTALCR	R/WSonce	XTALCR register Lock 1: Unlocked 1: Locked	0h
31:17	Reserved			0h

11.5.2 Clock source control register 1 (CLKSRCCTL1)

Offset address: 0x10

Reset type: XRSn

Note: As write configuration (write 1, clear 0) is performed to CLKSRCCTL1[INTOSC2OFF] or CLKSRCCTL1[WDHALTI] bit, and the SYSPLLMULT

register will be cleared, it is recommended to configure the CLKSRCCTL1 register first in the software and then configure the SYSPLLMULT register.

Field	Name	R/W	Description	Reset value
1:0	OSCCLKSRCSEL	R/W	Oscillator Clock Source Select 00: INTOSC2 01: External oscillator (XTAL) 10: INTOSC1 (recommended to use in the absence of clock detection) 11: Reserved (INTOSC1 by default) After power-on or XRSn, INTOSC2 is selected by default. When this field is used to change the clock source, SYSPLLMULT will be forced to zero, and PLL will be bypassed and powered off. This can prevent potential PLL overshoot. Appropriate multipliers must be configured in SYSPLLMULT. It takes 10 OSCCLK cycles to configure the appropriate multiplier in SYSPLLMULT or the previous one clock source needs to be disabled to allow the change to be completed.	0h
2	Reserved			0h
3	INTOSC2OFF	R/W	Internal Oscillator 2 Off 0: Enable 1: Disable The clock source can be disabled only when it is not in use.	0h
4	Reserved			0h
5	WDHALTI	R/W	Watchdog HALT Mode Ignore 0: The watchdog does not work in HALT mode. When the system enters HALT mode, the clock gates the watchdog and INTOSC1/2 is powered off 1: The watchdog works in HALT mode. When the system enters HALT mode, the clock does not gate the watchdog and INTOSC1/2 is not powered off	0h
31:6	Reserved			0h

11.5.3 Clock source control register 2 (CLKSRCCTL2)

Offset address: 0x14

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
3:2	CANABCLKSEL	R/W	CANA Bit-Clock Source Select 00: APBCLK 01: External oscillator (XTAL) 1x: Reserved The lost clock detection circuit has no impact on this field.	0h

Field	Name	R/W	Description	Reset value
5:4	CANFDABCLKSEL	R/W	CANFDA Bit-Clock Source Select 00: APBCLK 01: External oscillator (XTAL) 10: GPIO29/AUXCLK 1x: Reserved The lost clock detection circuit has no impact on this field.	0h
31:6	Reserved			0h

11.5.4 Clock source control register 3 (CLKSRCCTL3)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	XCLKOUTSEL	R/W	XCLKOUT Source Select 000: PLLSYSCLK 001: PLLRAWCLK 010: SYSCLK 011：APBCLK 100: FLPUMPOSC 101: INTOSC1 110: INTOSC 2 111: XTAL OSC o/p clock	0h
31:3	Reserved			0h

11.5.5 SYSPLL control register 1 (SYSPLLCTL1)

Offset address: 0x1C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	PLLEN	R/W	SYSPLL Enabled 0: Disable; the clock system transmits directly from OSCCLK 1: Enable	0h
1	PLLCLKEN	R/W	SYSPLLCLK Enabled Decide whether to bypass SYSPLL when PLLSYSCLK is generated. 0: Bypass SYSPLL. The clock system transmits directly from OSCCLK 1: Provide PLLSYSCLK from SYSPLL clock output. Ensure that the PLL is locked before this clock is enabled to the system.	0h
31:2	Reserved			0h

11.5.6 SYSPLL multiplier register (SYSPLLMULT)

Offset address: 0x28

Reset type: XRSn

Note:

(1) As write configuration (write 1, clear 0) is performed on CLKSRCCTL1[INTOSC2OFF] or CLKSRCCTL1[WDHALTI] bit, the SYSPLLMULT register will be cleared. Therefore, it is recommended to first configure the CLKSRCCTL1 register on the software and then configure the SYSPLLMULT register.

(2) The FMULT and IMULT fields must be written simultaneously to ensure the PLL functions properly.

Field	Name	R/W	Description	Reset value
6:0	IMULT	R/W	SYSPLL Integer Multiplier Select 0000000- 0000001: 1 0000010: 2 0000011: 3 ... 1111111: 127	0h
7	Reserved			0h
9:8	FMULT	R/W	SYSPLL Fractional Multiplier Select 00: 0 01: 025 10: 0.5 11: 0.75	0h
15:10	Reserved			0h
18:16	ODIV	R/W	SYSPLL Output Clock Divider PLL output voltage divider=ODIV+1	0h
31:19	Reserved			0h

11.5.7 SYSPLL flag register (SYSPLLSTS)

Offset address: 0x2C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	LOCKS	R	SYSPLL Lock Flag 1: Unlocked 1: Locked	0h
1	SLIPS	R	SYSPLL is out of lock range Flag 0: Not exceeding the lock range 1: Out of lock range	0h
31:2	Reserved			0h

11.5.8 System clock divider register (SYSCLKDIVSEL)

Offset address: 0x44

Reset type: XRSn

Field	Name	R/W	Description	Reset value
5:0	PLLSYSCLKDIV	R/W	SYSCLK Divide Select 00000001: 1 divided frequency 000001: 2 divided frequency 000010: 4 divided frequency (default when reset) 000011: 6 divided frequency 000100: 8 divided frequency ... 111111: 126 divided frequency	2h
31:6	Reserved			0h

11.5.9 XCLKOUT divider register (XCLKOUTDIVSEL)

Offset address: 0x50

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	XCLKOUTDIV	R/W	XCLKOUT Divide Select 00: 1 divided frequency 01: 2 divided frequency 10: 4 divided frequency 11: 8 divided frequency (default when reset)	3h
31:2	Reserved			0h

11.5.10 Low-speed clock source prescaler register (LOSPCP)

Offset address: 0x58

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	LSPCLKDIV	R/W	Low-speed peripheral clock Divide Select This clock is used as a strobe for UART and SPI modules. 000: 1 divided frequency 001: 2 divided frequency 010: 4 divided frequency (default when reset) 011: 6 divided frequency ... 111: 14 divided frequency	2h
31:3	Reserved			0h

11.5.11 Missing clock detection control register (MCDCR)

Offset address: 0x5C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	MCLKSTS	R	Missing Clock Flag 0: Flag of no missing clock 1: Flag of missing clock	0h
1	MCLKCLR	R-0/W1S	Missing Clock Flag Clear Setting this bit can clear the MCLKSTS bit and reset the lost clock detection circuit.	0h
2	MCLKOFF	R/W	Missing Clock Detect Disable 0: Enable 1: Disable	0h
3	OSCOFF	R/W	Disconnect the OSCCLK from the MCD 0: Connect 1: Disconnect	0h
31:4	Reserved			0h

11.5.12 10-bit counter on X1 clock (X1CNT)

Offset address: 0x60

Reset type: POR (only [9:0] has reset type)

Field	Name	R/W	Description	Reset value
9:0	X1CNT	R	X1 Counter This counter increases progressively on the positive side of each X1 clock, and freezes once it reaches the value of 0x3FF. This counter must be saturated to switch from INTOSC2 to X1 so as to ensure the oscillation of the crystal connected to X1/X2.	0h
15:10	Reserved			0h
16	CLR	R-0/W1S	X1 Counter clear Writing 0 to this field will be ignored, writing 1 will clear the X1CNT bit, and it will start counting again from 0x0.	0h
31:17	Reserved			0h

11.5.13 XTAL control register (XTALCR)

Offset address: 0x64

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	OSCOFF	R/W	XTAL oscillator macro Disable 0: Enable 1: Disable If the crystal is connected to X1/X2, this bit needs to be cleared first, and the clock source needs to be switched to X1/X2 after the oscillator is powered on.	1h

Field	Name	R/W	Description	Reset value
1	SE	R/W	XTAL oscillator mode Configures Configure the mode when the XTAL oscillator is powered on. 0: Crystal mode 1: Single-ended mode	0h
2	Reserved			1h
31:3	Reserved			0h

12 System control

12.1 SPEC Functional Description

System control consists of a set of sub-modules, collectively referred to as the system control module. The system control module provides the following functions:

- System-level reset, including power-on and power-off reset
- Clock source selection and phase-locked loop (PLL) configuration
- Lack of clock detection
- Clock gating Low-power modes
- Peripheral interrupt handling
- Unmaskable interrupts under certain fault conditions
- 32-bit Timer
- Window watchdog timer, capable of generating an interrupt or a reset
- RAM initialisation, write protection and control

The WRPRT protection mechanism is designed to prevent the CPU from performing erroneous write operations to certain registers in the system; this mechanism uses special coprocessor instructions (“WRPRT” and “WRPRTDIS”) to enable and disable access to protected registers.

Table27 Access to WRPRT-Protected Registers

Protection status	CPU Write Operation
CPU WRPRT	Allowed
CPU WRPRTDIS	Ignore

Note: The JTAG/SWD port overrides the WRPRT bit, allowing full access to protected registers in debug mode.

When the G32R5xx is started up, the register protection function is enabled by default:

- In protected mode, the CPU is permitted to read, and DEBUG is permitted to read/write; write operations by the CPU to protected registers are ignored.
- Executing the ‘WRPRT’ instruction disables protection (the CPU is permitted to freely write to protected registers)
- After modifying a register, executing the “WRPRTDIS” instruction clears the WRPRT bit (re-enabling protection)

12.1.1 Restrictions on system control register configuration

The memory mapping registers in system control are operated on the INTOSC1 clock domain. To avoid possible loss during the second write, CPU needs to have a delay between subsequent writes to these registers. The application program needs to take this into consideration and add delay based on the number of NOP instructions after each write to these registers as mentioned below. The formula for calculating the delay

between subsequent writes:

$$\text{Delay (SYSCLK cycle)} = 3 \times (F_{SYSCLK} \div F_{INTOSC1}) + 9$$

The registers that need to be delayed after each write are:

- FLBCLKCTL
- PERCLKDIVSEL
- SYSCLKDIVSEL
- SYSPLLCTL1
- SYSPLLMULT
- WDCR
- XCLKOUTDIVSEL
- XTALCR
- CLKSRCCTL1
- CLKSRCCTL2
- CLKSRCCTL3

12.2 SPEC register bank address

Table 28 Register Bank Address

Device register	Register bank	Start address	End address
SpecRegs	SPEC_REG	0x5002 0000	0x5002 03FF

12.3 SPEC register address mapping

Table 29 SPEC_REG Register Bank Address Mapping

Register name	Register description	Offset address	WRPRT
SPEC_EMUBOOTPINCFG	Emulate BOOT configuration register	0x00	-
SPEC_EMUBOOTDEFL	Emulate BOOT definition register low	0x04	-
SPEC_EMUBOOTDEFH	Emulate BOOT definition register high	0x08	-
SPEC_CFGSMSSEL	Select CFGSMS static configuration register	0x64	-
SPEC_APBCLKDIVCFG	APB clock division configuration register	0x74	-
SPEC_VOS	Voltage regulating register	0x78	-
CANFDAWAKESTATUS	CANFDA Wake-up Status Register	0x90	-
CANFDA Wake-up Status Clear Register	CANFDA Wake-up Status Clear Register	0x94	-
CANFDA Clock Stop Request	CANFDA Low-Power Request Register	0x98	-

Register name	Register description	Offset address	WRPRT
CANFDACLKSTOPACK	CANFDA Low-Power Request Acknowledgement Register	0x9C	-
ROM_BIST_CR	ROM BIST Control Register	0xA0	-
CACHE_BIST_CR	CACHE BIST Control Register	0xA4	-
CANA_BIST_CR	CANA BIST control register	0xA8	-
CANFDA_BIST_CR	CANFDA BIST Control Register	0xAC	-
TCM_BIST_CR	TCM BIST Control Register	0xB0	-
TCM_BIST_SR	TCM BIST Status Register	0xB4	-

12.4 Functional description of SPEC registers

12.4.1 Emulate BOOT configuration register (SPEC_EMUBOOTPINCFG)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	BOOTSEL0	R/W	Boot Mode Select Pin 0 Set as GPIO pin used at boot (up to 255). 0x00 = GPIO0 0x01 = GPIO1 ….. If 0xFF is invalid and other BOOTSEL are set to 0xFF, select the default BOOTSEL0. If other BOOTSEL are not set to 0xFF, setting BOOTSEL to 0xFF will disable this specific BOOTSEL.	0h
15:8	BOOTSEL1	R/W	Boot Mode Select Pin 1 Refer to BOOTSEL0 Description.	0h
23:16	BOOTSEL2	R/W	Boot Mode Select Pin 2 Refer to BOOTSEL0 Description.	0h
31:24	KEY	R/W	Key Word Write 0x5A to these 8 bits to inform the boot ROM code that the bits in this register are valid.	0h

12.4.2 Emulate BOOT definition register low (SPEC_EMUBOOTDEFL)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	BOOTDEF0	R/W	BOOT_DEF0 Mode/Options It includes changing the GPIO of specific boot peripherals or specifying different Flash entry points. Any unsupported boot mode will cause the device to reset.	0h
15:8	BOOTDEF1	R/W	BOOT_DEF1 Mode/Options Refer to BOOTDEF0 Description.	0h
23:16	BOOTDEF2	R/W	BOOT_DEF2 Mode/Options Refer to BOOTDEF0 Description.	0h
31:24	BOOTDEF3	R/W	BOOT_DEF3 Mode/Options Refer to BOOTDEF0 Description.	0h

12.4.3 Emulate BOOT definition register high (SPEC_EMUBOOTDEFH)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	BOOTDEF4	R/W	BOOT_DEF4 Mode/Options Refer to BOOTDEF0 Description.	0h
15:8	BOOTDEF5	R/W	BOOT_DEF5 Mode/Options Refer to BOOTDEF0 Description.	0h
23:16	BOOTDEF6	R/W	BOOT_DEF6 Mode/Options Refer to BOOTDEF0 Description.	0h
31:24	BOOTDEF7	R/W	BOOT_DEF7 Mode/Options Refer to BOOTDEF0 Description.	0h

12.4.4 Select CFGSMS static configuration register (SPEC_CFGSMSSEL)

Offset address: 0x64

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	SCAHRPWMCFG	R/W	CFGSMS Static Configuration select 0000: Static configuration 0 0001: Static configuration 1 …… 1111: Static configuration 16	0h
31:4	Reserved			0h

12.4.5 APB clock division configuration register (SPEC_APBCKDIVCFG)

Offset address: 0x74

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	APBCLKDIVCFG	R/W	APB clock divide configure 00000: 1:1 00001: 2:1 Others: Reserved	1h
31:5	Reserved			0h

12.4.6 Voltage regulating register (SPEC_VOS)

Offset address: 0x78

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	VOSEN	R/W	VOS Enable 0: Disable 1: Enable	0h
3:1	LDOVSCACFG	R/W	LDO voltage scaling configure 000: VDD voltage 1.1V 001: Reserved 010: Reserved 011: VDD voltage 1.0V 100: VDD voltage 1.05V 101: VDD voltage 1.1V 11x: VDD voltage 1.15V	0h
31:4	Reserved			0h

12.4.7 CANFDA Wake-up Status Register (CANFDAWAKESTATUS)

Offset address: 0x90

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	FLAG	R/W	Wake-up event flag (WAKE UP FLAG) 0: No wake-up event 1: Wake-up event present	0h
31:1	Reserved			0h

12.4.8 CANFDA Wake-up Status Clear Register (CANFDAWAKESTATUSCLR)

Offset address: 0x94

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CLEAR	RC_W1	Wake-up event flag clear (WAKE UP CLEAR) Write 1 to clear; read returns 0	0h
31:1	Reserved			0h

12.4.9 CANFDA Low-Power Request Register (CANFDACLKSTOPREQ)

Offset address: 0x98

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	REQ	R/W	CANFDA clock stop request; this bit must be set to 1 when the CANFDA enters low-power mode, and is cleared to 0 by the hardware upon wake-up.	0h
15:1	Reserved			0h
31:16	KEY	W	Only when 0x5634 is written to the KEY field will a write operation to any bit in this register be successful.	0h

12.4.10 CANFDA Low-Power Request Acknowledgement Register (CANFDACLKSTOPACK)

Offset address: 0x9C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ACK	R/W	CANFDA Low-Power Request Acknowledgement Flag	0h
31:1	Reserved			0h

12.4.11 ROM BIST Control Register (ROM_BIST_CR)

Offset address: 0xA0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ROM_BIST_START	R/W	BootROM BIST enabled This bit must be cleared after BIST has been completed. 0: BIST mode disabled 1: BIST mode enabled	0h
1	Reserved			0h
2	ROM_BIST_DONE	R	BootROM BIST completion flag	0h
3	ROM_BIST_FAIL	R	BootROM BIST failure flag	0h
15:4	Reserved			0h
31:16	PASSWORD	W	The key for bit[0] When writing to bit[0], 0xA05F must be written.	0h

12.4.12 CACHE BIST Control Register (CACHE_BIST_CR)

Offset address: 0xA4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CACHE_BIST_START	R/W	Cache BIST enabled This bit must be cleared after BIST has been completed. 0: BIST mode disabled 1: BIST mode enabled	0h
1	Reserved			0h
2	IDATA_BIST_DONE	R	IDATA BIST completion flag	0h
3	IDATA_BIST_FAIL	R	IDATA BIST Failure Flag	0h
4	ITAG_BIST_DONE	R	I-cache tag RAM BIST completion flag	0h
5	ITAG_BIST_FAIL	R	I-cache tag RAM BIST failure flag	0h
6	DDATA_BIST_DONE	R	D-cache data RAM BIST completion flag	0h
7	DDATA_BIST_FAIL	R	D-cache data RAM BIST failure flag	0h
8	DTAG_BIST_DONE	R	D-cache tag RAM BIST completion flag	0h
9	DTAG_BIST_FAIL	R	D-cache tag RAM BIST failure flag	0h
15:10	Reserved			0h
31:16	PASSWORD	W	Key for Bit[0] When writing to bit[0], 0xA05F must be written.	0h

12.4.13 CANA BIST Control Register (CANA_BIST_CR)

Offset address: 0xA8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CANA_BIST_START	R/W	CANA FIFO RAM BIST : Enabled (The corresponding IP clock must be enabled to perform BIST) This bit must be cleared after BIST has been completed. 0: BIST mode disabled 1: BIST mode enabled	0h
1	Reserved			0h
2	CANA_BIST_DONE	R	CANA FIFO RAM BIST completion flag	0h
3	CANA_BIST_FAIL	R	CANA FIFO RAM BIST Failure Flag	0h
15:4	Reserved			0h
31:16	PASSWORD	W	The key for Bit[0] When writing to bit[0], you must write 0xA05F.	0h

12.4.14 CANFDA BIST Control Register (CANFDA_BIST_CR)

Offset address: 0xAC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CANFDA_BIST_START	R/W	CANFDA FIFO RAM BIST : Enabled (The corresponding IP clock must be enabled to perform BIST) This bit must be cleared after BIST has been completed. 0: BIST mode disabled 1: BIST mode enabled	0h
1	Reserved			0h
2	CANFDA_BIST_DONE	R	CANFDA FIFO RAM BIST completion flag	0h
3	CANFDA_BIST_FAIL	R	CANFDA FIFO RAM BIST Failure Flag	0h
15:4	Reserved			0h
31:16	PASSWORD	W	The key for Bit[0] When writing to bit[0], you must write 0xA05F.	0h

12.4.15 TCM_BIST Control Register (TCM_BIST_CR)

Offset address: 0xB0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	BANK0_BIST_START	R/W	Enables BIST for physical block bank 0 within logic block bank 11 of Taosms (The corresponding IP clock must be enabled to perform BIST) This bit must be cleared after BIST has been completed. 0: BIST mode disabled 1: BIST mode enabled	0h
1	BANK1_BIST_START	R/W	Enables BIST for physical block 1 within logic block bank 11 of Taosms See BANK0_BIST_START.	0h
2	BANK2_BIST_START	R/W	Enables BIST for physical block 0 within logic block bank 12 of Taosms See BANK0_BIST_START.	0h
3	BANK3_BIST_START	R/W	Taosms logic block bank 12, physical block bank 1 BIST enabled See BANK0_BIST_START.	0h
4	BANK4_BIST_START	R/W	Taosms logic block bank 13, physical block bank 0, BIST enabled See BANK0_BIST_START.	0h

5	BANK5_BIST_START	R/W	Taosms logic block bank 13, physical block bank 1 BIST enabled See BANK0_BIST_START.	0h
6	BANK6_BIST_START	R/W	Enables BIST for physical block bank 0 within logic block bank 14 of Taosms. See BANK0_BIST_START.	0h
7	BANK7_BIST_START	R/W	Enables BIST for physical block bank 1 within Taosms logic block bank 14. See BANK0_BIST_START.	0h
8	BANK8_BIST_START	R/W	BIST enabled for physical block bank 0 within Taosms logic block bank 15. See BANK0_BIST_START.	0h
9	BANK9_BIST_START	R/W	Physical block BIST enabled in logic block bank 15 of Taosms; refer to BANK0_BIST_START.	0h
15:10	Reserved			0h
31:16	PASSWORD	W	The key for bit[9:0] When writing to bit[9:0], the value must be 0xA05F.	0h

12.4.16 TCM_BIST status register (TCM_BIST_SR)

Offset address: 0xB4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	BANK0_BIST_DONE	R	BIST completion flag for physical block bank 0 within logic block bank 11 of Taosms	0h
1	BANK1_BIST_DONE	R	Taosms logical block bank 11, physical block bank 1 BIST completion flag	0h
2	BANK2_BIST_DONE	R	Taosms logical block bank 12, physical block bank 0 BIST completion flag	0h
3	BANK3_BIST_DONE	R	Taosms logic block bank 12 physical block bank 1 BIST completion flag	0h
4	BANK4_BIST_DONE	R	Taosms logical block bank 13, physical block bank 0 BIST completion flag	0h
5	BANK5_BIST_DONE	R	Taosms logic block bank 13 physical block bank 1 BIST completion flag	0h
6	BANK6_BIST_DONE	R	Taosms logical block bank 14, physical block bank 0 BIST completion flag	0h
7	BANK7_BIST_DONE	R	Taosms logic block bank 14 physical block bank 1 BIST completion flag	0h
8	BANK8_BIST_DONE	R	Taosms logic block bank 15, physical block bank 0 BIST completion flag	0h
9	BANK9_BIST_DONE	R	Taosms logic block bank 15 physical block bank 1 BIST completion flag	0h
10	BANK0_BIST_FAIL	R	Taosms logic block bank 11, physical block bank 0 BIST failure flag	0h

11	BANK1_BIST_FAIL	R	BIST failure flag for physical block bank 0 within Taosms logic block bank 11	0h
12	BANK2_BIST_FAIL	R	Taosms logical block bank 12, physical block bank 0 BIST failure flag	0h
13	BANK3_BIST_FAIL	R	Taosms logical block bank 12 physical block bank 1 BIST failure flag	0h
14	BANK4_BIST_FAIL	R	Taosms logical block bank 13, physical block bank 0 BIST failure flag	0h
15	BANK5_BIST_FAIL	R	Taosms logical block bank 13, physical block bank 1 BIST failure flag	0h
16	BANK6_BIST_FAIL	R	Taosms logical block bank 14, physical block bank 0 BIST failure flag	0h
17	BANK7_BIST_FAIL	R	Taosms logical block bank 14, physical block bank 1 BIST failure flag	0h
18	BANK8_BIST_FAIL	R	Taosms logical block bank 15, physical block bank 0 BIST failure flag	0h
19	BANK9_BIST_FAIL	R	Taosms logical block bank 15, physical block bank 1 BIST failure flag	0h
31:20	Reserved			0h

12.5 NMI_INTRUPT register bank address

Table 30 NMI_INTRUPT Register Bank Address

Device register	Register bank	Start address	End address
NmiIntruptRegs	NMI_INTRUPT_REGS	0x400100C0	0x400103FF

12.6 NMI_INTRUPT register address mapping

Table 31 NMI_INTRUPT Register Address Mapping

Register name	Register description	Offset address	WRPRT
NMICFG	Configuration register	0x00	√
NMIFLG	Flag register	0x02	-
NMIFLGCLR	Flag clear register	0x04	√
NMIFLGFRC	Flag forced register	0x06	√
NMIWDCNT	Watchdog counter register	0x08	-
NMIWDPRD	Watchdog cycle register	0x0A	√
NMISHDFLG	Shadow flag register	0x0C	-
ERRORSTS	Error pin status register	0x0E	-
ERRORSTSCLR	ERRORSTS clear register	0x10	√

Register name	Register description	Offset address	WRPRT
ERRORSTSFRC	ERRORSTS set register	0x12	√
ERRORCTL	Error pin control register	0x14	√
ERRORLOCK	Latch register for the error pin register	0x16	√

12.7 Functional description of NMI_INTRUPT registers

12.7.1 Configuration register (NMICFG)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	NMIE	R/W1S	Global NMI Enable This bit indicates that the NMI module has been enabled, allowing system error to trigger the NMI of the CPU. This bit is set during startup of the boot ROM. It can only be cleared through system reset.	0h
15:1	Reserved			0h

12.7.2 Flag register (NMIFLG)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	NMIINT	R	NMI Flag 0: No flag 1: Generate a flag Indicate whether NMI is generated. It can be cleared through system reset or by writing the corresponding bit in the NMIFLGCLR register. NMI will not be generated again before this flag is cleared.	0h
1	CLOCKFAIL	R	Clock Error Flag 0: No flag 1: Generate a flag Indicate whether a clock error is detected. It can be cleared through system reset or by writing the corresponding bit in the NMIFLGCLR register.	0h
2	RAMUNCERR	R	RAM Uncorrectable Error Flag 0: No flag 1: Generate a flag Indicates whether an uncorrectable ECC error has been detected during a RAM access operation. It can be cleared by a system reset or by writing to the corresponding bit in the NMIFLGCLR register.	0h

Field	Name	R/W	Description	Reset value
3	FLUNCERR	R	Flash Uncorrectable Error Flag 0: No flag 1: Generate a flag Indicate whether an uncorrectable ECC error occurred in the flash access process. It can be cleared through system reset or by writing the corresponding bit in the NMIFLGCLR register.	0h
7:4	Reserved			0h
8	FLBNMI	R	FLB Generates NMI flag 0: No flag 1: Generate a flag Indicate whether NMI is generated by FLB. It can be cleared through system reset or by writing the corresponding bit in the NMIFLGCLR register.	0h
12:9	Reserved			0h
13	SWERR	R	Software-Forced NMI Flag 0: No flag 1: Generate a flag This bit can only be set by writing the corresponding bit in the NMIFLGFRC register. It can be cleared through system reset or by writing the corresponding bit in the NMIFLGCLR register. NMI will not be generated again before this flag is cleared.	0h
15:14	Reserved			0h

12.7.3 Flag clear register (NMIFLGCLR)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	NMIINT	R-0/W1S	NMI Flag Clear Clear the corresponding bit in the NMIFLG register. This flag should be cleared only after all other activity flags have been cleared.	0h
1	CLOCKFAIL	R-0/W1S	Clock Error Flag Clear Clear the corresponding bit in the NMIFLG register.	0h
2	RAMUNCERR	R-0/W1S	RAM Uncorrectable Error Flag Clear Clear the corresponding bit in the NMIFLG register.	0h
3	FLUNCERR	R-0/W1S	Flash Uncorrectable Error Flag Clear Clear the corresponding bit in the NMIFLG register.	0h
7:4	Reserved			0h
8	FLBNMI	R-0/W1S	FLB Generates NMI flag Clear Clear the corresponding bit in the NMIFLG register.	0h
12:9	Reserved			0h

Field	Name	R/W	Description	Reset value
13	SWERR	R-0/W1S	Software-Forced NMI Flag Clear Clear the corresponding bit in the NMIFLG register.	0h
15:14	Reserved			0h

12.7.4 Flag forced register (NMIFLGFRC)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	CLOCKFAIL	R-0/W1S	Clock Error Flag Set Set the corresponding bit in the NMIFLG register.	0h
2	RAMUNCERR	R-0/W1S	RAM Uncorrectable Error Flag Set Set the corresponding bit in the NMIFLG register.	0h
3	FLUNCERR	R-0/W1S	Flash Uncorrectable Error Flag Set Set the corresponding bit in the NMIFLG register.	0h
7:4	Reserved			0h
8	FLBNMI	R-0/W1S	FLB Generates NMI flag Set Set the corresponding bit in the NMIFLG register.	0h
12:9	Reserved			0h
13	SWERR	R-0/W1S	Software-Forced NMI Flag Set Set the corresponding bit in the NMIFLG register.	0h
15:14	Reserved			0h

12.7.5 Watchdog counter register (NMIWDCNT)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	NMIWDCNT	R	NMI Watchdog Counter Set When setting the bit of the NMIFLG register, the watchdog counter increases once every SYSCLK cycle. If the counter reaches the cycle value in the NMIWDPRD register, the NMI module will generate an NMIWDRS reset. After reset, the counter is reset to zero and stops counting. If all NMI flags are cleared, the counter will reset to zero and stop counting until another NMI flag is set.	0h

12.7.6 Watchdog period register (NMIWDPRD)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	NMIWDPRD	R/W	NMI Watchdog Period Set Specify the period of the NMI watchdog timer in the SYSCLK cycle. Writing a period value smaller than the current counter value will immediately force a reset (NMIWDRS).	FFFFh

12.7.7 Shadow flag register (NMISHDFLG)

Offset address: 0x0C

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	CLOCKFAIL	R-0/W1S	Shadow Clock Error Flag	0h
2	RAMUNCERR	R-0/W1S	Shadow RAM Uncorrectable Error Flag	0h
3	FLUNCERR	R-0/W1S	Shadow Flash Uncorrectable Error Flag	0h
7:4	Reserved			0h
8	FLBNMI	R-0/W1S	Shadow FLB Generates NMI flag	0h
12:9	Reserved			0h
13	SWERR	R-0/W1S	Shadow Software-Forced NMI Flag	0h
15:14	Reserved			0h

12.7.8 Error PIN Status Register (ERRORSTS)

Offset address: 0x0E

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	PINSTS	R	0: error pin is 0 1: error pin is 1	0h
1	ERROR	R	0: No error source has been generated; 1: One or more error sources have occurred, or ERRPORSTS.ERROR has been set by ERRORSTSFRC.ERROR bit 1. Once set, the ERROR flag can only be cleared by writing a 1 to the ERRORSTSCLR.ERROR bit. The following events/triggers can set the flag: NMI interrupt; WDT reset; EFFUSE error	0h
15:2	Reserved			0h

12.7.9 ERRORSTS Clear Register (ERRORSTSCLR)

Offset address: 0x10

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	ERROR	R-0/W1S	0: No effect 1: ERROR STS.ERROR is cleared to 0	0h
15:1	Reserved			0h

12.7.10 ERRORSTS set register (ERRORSTSFRC)

Offset address: 0x12

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	ERROR	R-0/W1S	0: No effect 1: ERRORSTS.ERROR is set to 1	0h
15:1	Reserved			0h

12.7.11 ERRORSTS Control Register (ERRORSTSCTL)

Offset address: 0x14

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	ERRORPOLSEL	R/W	0: If ERRORSTS.ERROR is 1, the error pin is set to 0; otherwise, it is set to 1 1: If ERRORSTS.ERROR is 1, the error pin is set to 0; otherwise, it is set to 1	0h
15:1	Reserved			0h

12.7.12 Error latch register (ERRORLOCK)

Offset address: 0x16

Reset type: PORESETn

Field	Name	R/W	Description	Reset value
0	ERRORCTL	R/WSonce	0: Writing to the ERRORCTL register is permitted 1: ERRORCTL register is locked; writing is not permitted Writing 0 has no effect; writing 1 sets the bit, which can only be cleared by a reset	0h
15:1	Reserved			0h

12.8 PERIPH_AC register bank address

Table 32 PERIPH_AC Register Bank Address

Device register	Register bank	Start address	End address
PeriphAcRegs	PERIPH_AC_REGS	0x40013800	0x40013FFF

12.9 PERIPH_AC register address mapping

Table 33 PERIPH_AC Register Address Mapping

Register name	Register description	Offset address	WRPRT
ADCA_AC	ADCA main access control register	0x00	√
ADCC_AC	ADCC main access control register	0x08	√
COMP1_AC	COMP1 main access control register	0x20	√
COMP2_AC	COMP2 main access control register	0x24	√
COMP3_AC	COMP3 main access control register	0x28	√
COMP4_AC	COMP4 main access control register	0x2C	√
DACA_AC	DACA main access control register	0x50	√
DACB_AC	DACB main access control register	0x54	√
PWM1_AC	PWM1 main access control register	0x90	√
PWM2_AC	PWM2 main access control register	0x94	√
PWM3_AC	PWM3 main access control register	0x98	√
PWM4_AC	PWM4 main access control register	0x9C	√
PWM5_AC	PWM5 main access control register	0xA0	√
PWM6_AC	PWM6 main access control register	0xA4	√
PWM7_AC	PWM7 main access control register	0xA8	√
QEP1_AC	QEP1 main access control register	0xE0	√
CAP1_AC	CAP1 main access control register	0x100	√
CAP2_AC	CAP2 main access control register	0x104	√
CAP3_AC	CAP3 main access control register	0x108	√
FLB1_AC	FLB1 main access control register	0x160	√
FLB2_AC	FLB2 main access control register	0x164	√
FLB3_AC	FLB3 main access control register	0x168	√
FLB4_AC	FLB4 main access control register	0x16C	√

Register name	Register description	Offset address	WRPRT
SPIA_AC	SPIA main access control register	0x220	√
PMBUS_A_AC	PMBUSA main access control register	0x260	√
LIN_A_AC	LINA main access control register	0x270	√
CANA_AC	CANA main access control register	0x280	√
CANFDA_AC	CANB main access control register	0x290	√
HRPWM_A_AC	HRPWMA main access control register	0x354	√
PERIPH_AC_LOCK	Lock register	0x3FC	√

12.10 Functional description of PERIPH_AC register

12.10.1 ADCA main access control register (ADCA_AC)

Offset address: 0x00

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.2 ADCC main access control register (ADCC_AC)

Offset address: 0x08

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h

Field	Name	R/W	Description	Reset value
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.3 COMPx main access control register (COMPx_AC) (x=1-4)

Offset address: 0x20+(a-1)*0x04 (a=1-4)

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.4 DACA main access control register (DACA_AC)

Offset address: 0x50

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h

Field	Name	R/W	Description	Reset value
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.5 DACB main access control register (DACB_AC)

Offset address: 0x54

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.6 PWMx main access control register (PWMx_AC) (x=1-7)

Offset address: 0x90+(a-1)*0x04 (a=1-7)

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved	3h

Field	Name	R/W	Description	Reset value
			10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	
31:6	Reserved			0h

12.10.7 QEPx main access control register (QEPx_AC) (x=1)

Offset address: 0xE0+(a-1)*0x04 (a=1)

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.8 CAPx main access control register (CAPx_AC) (x=1-3)

Offset address: 0x100+(a-1)*0x04 (a=1-3)

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h

Field	Name	R/W	Description	Reset value
31:6	Reserved			0h

12.10.9 FLBx main access control register (FLBx_AC) (x=1-4)

Offset address: 0x160+(a-1)*0x04 (a=1-4)

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
31:4	Reserved			0h

12.10.10 SPIA main access control register (SPIA_AC)

Offset address: 0x220

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.11 PMBUSA main access control register (PMBUS_A_AC)

Offset address: 0x260

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals	3h

Field	Name	R/W	Description	Reset value
			01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.12 LINA main access control register (LIN_A_AC)

Offset address: 0x270

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.13 CANA main access control register (CANA_AC)

Offset address: 0x280

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h

Field	Name	R/W	Description	Reset value
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.14 CANFDA main access control register (CANFDA_AC)

Offset address: 0x284

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU0_ACC	R/W	CPU0 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
3:2	CPU1_ACC	R/W	CPU1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.15 HRPWMA main access control register (HRPWM_A_AC)

Offset address: 0x354

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	CPU_ACC	R/W	CPU access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access	3h

Field	Name	R/W	Description	Reset value
			11: Read/write full access	
3:2	Reserved			3h
5:4	DMA1_ACC	R/W	DMA1 access permission Select 00: Read/write access is not performed on peripherals 01: Reserved 10: Protect RD from changing the FIFO of the read register, Clear access, without write access 11: Read/write full access	3h
31:6	Reserved			0h

12.10.16 Lock register (PERIPH_AC_LOCK)

Offset address: 0x3FC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	LOCK_AC_WR0	R/WSonce	CPU access permission Select 0: The access control register allows read/write access 1: The access control register is read-only Write “1” to set the bit, but writing “0” does not work.	0h
31:1	Reserved			0h

12.11 ROM_PREFETCH register bank address

Table 34 ROM_PREFETCH Register Bank Address

Device register	Register bank	Start address	End address
RomprefetchRegs	ROM_PREFETCH_REGS	0x40013C00	0x40013CFF

12.12 ROM_PREFETCH register address mapping

Table 35 ROM_PREFETCH Register Address Mapping

Register name	Register description	Offset address	WRPRT
BOOT_ROM_PREFETCH	Enable BOOT ROM prefetch register	0x00	√

12.13 Functional description of ROM_PREFETCH register

12.13.1 Enable BOOT ROM prefetch register (BOOT_ROM_PREFETCH)

Offset address: 0x00

Reset type: XRSn

Field	Name	R/W	Description	Reset value
0	BOOT_ROM_PREFETCH	R/W	BOOT ROM prefetch function enable 0: Disable 1: Enable	0h
31:1	Reserved			0h

12.14 ROM_WAIT_STATE register bank address

Table 36 ROM_WAIT_STATE Register Bank Address

Device register	Register bank	Start address	End address
RomwaitstateRegs	ROM_WAIT_STATE_REGS	0x40013C00	0x40013CFF

12.15 ROM_WAIT_STATE register address mapping

Table 37 ROM_WAIT_STATE Register Address Mapping

Register name	Register description	Offset address	WRPRT
BOOT_ROM_LATENCY	BOOT ROM delay configuration register	0x00	√

12.16 Functional description of ROM_WAIT_STATE register

12.16.1 BOOT ROM delay configuration register (BOOT_ROM_LATENCY)

Offset address: 0x00

Reset type: XRSn

Field	Name	R/W	Description	Reset value
2:0	BOOT_ROM_LATENCY	R/W	BOOT ROM delay configuration 000: No delay 001: 1 HCLK delay … 111: 7 HCLK delay	1h
31:3	Reserved			0h

13 Nested Vector Interrupt Controller (NVIC)

13.1 Full Name and Abbreviation Description of Terms

Table 38 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Non Maskable Interrupt	NMI
Nested Vectored Interrupt Controller	NVIC

13.2 Introduction

The core integrates nested vectored interrupt controller (NVIC), which is closely coupled with the core, and can handle exceptions and interrupts and power management control efficiently and with low delay.

13.3 Main characteristics

(1) 227 maskable interrupt channels (excluding 16 core interrupt lines)

(2) Low-delay exception and interrupt processing

(3) Power management control

(4) Realization of system control register

13.4 Interrupt and exception vector table

Table 39 Interrupt and Exception Vector Table

Exception type	Vector No.	Priority	Vector address	Description
-	-	-	0x0000_0000	Reserved
Reset	-	-15	0x0000_0004	Reset
NMI	-	-14	0x0000_0008	Non-maskable interrupt
HardFault	-	-13	0x0000_000C	Various hardware faults
MemManage	-	-12	0x0000_0010	Memory fault
BusFault	-	-11	0x0000_0014	Bus fault
UsageFault	-	-10	0x0000_0018	Instruction execution fault
SecureFault	~	-9	0x0000_001C	Data security fault
SVCall	-	-5	0x0000_002C	System service called by general SWIU instruction
Debug Monitor	-	-4	0x0000_0030	Debugging function exception

Exception type	Vector No.	Priority	Vector address	Description
PendSV	-	-2	0x0000_0038	Pending system service
SysTick	-	-1	0x0000_003C	System tick timer
-	0~10	-	0x0000_0040 ~0x0000_0068	Reserved
DCCOMP	11	Can set	0x0000_006C	DCCOMP interrupt
-	12	Can set	0x0000_0070	Reserved
TMR1	13	Can set	0x0000_0074	TMR1 interrupt
TMR2	14	Can set	0x0000_0078	TMR2 interrupt
-	15	Can set	0x0000_007C	Reserved
RAM	16	Can set	0x0000_0080	System memory interrupt
-	17~31	-	0x0000_0084 ~0x0000_00BC	Reserved
ADCA1	32	Can set	0x0000_00C0	ADCA1 interrupt
-	33	-	0x0000_00C4	Reserved
ADCC1	34	Can set	0x0000_00C8	ADCC1 interrupt
-	35	Can set	0x0000_00CC	Reserved
-	36	Can set	0x0000_00D0	Reserved
-	37	Can set	0x0000_00D4	Reserved
TMR0	38	Can set	0x0000_00D8	TMR0 interrupt
WDT_WAKE	39	Can set	0x0000_00DC	WDT interrupt/WAKE interrupt
PWM1 TRIP ZONE	40	Can set	0x0000_00E0	PWM1 TRIP interrupt
PWM2 TRIP ZONE	41	Can set	0x0000_00E4	PWM2 TRIP interrupt
PWM3 TRIP ZONE	42	Can set	0x0000_00E8	PWM3 TRIP interrupt
PWM4 TRIP ZONE	43	Can set	0x0000_00EC	PWM4 TRIP interrupt
PWM5 TRIP ZONE	44	Can set	0x0000_00F0	PWM5 TRIP interrupt
PWM6 TRIP ZONE	45	Can set	0x0000_00F4	PWM6 TRIP interrupt
PWM7 TRIP ZONE	46	Can set	0x0000_00F8	PWM7 TRIP interrupt
-	47	-	0x0000_00FC	Reserved
PWM1	48	Can set	0x0000_0100	PWM1 interrupt
PWM2	49	Can set	0x0000_0104	PWM2 interrupt
PWM3	50	Can set	0x0000_0108	PWM3 interrupt
PWM4	51	Can set	0x0000_010C	PWM4 interrupt
PWM5	52	Can set	0x0000_0110	PWM5 interrupt

Exception type	Vector No.	Priority	Vector address	Description
PWM6	53	Can set	0x0000_0114	PWM6 interrupt
PWM7	54	Can set	0x0000_0118	PWM7 interrupt
-	55	-	0x0000_011C	Reserved
CAP1	56	Can set	0x0000_0120	CAP1 interrupt
CAP2	57	Can set	0x0000_0124	CAP2 interrupt
CAP3	58	Can set	0x0000_0128	CAP3 interrupt
CAP4	59~63	Can set	0x0000_012C ~0x0000_013C	Reserved
QEP1	64	Can set	0x0000_0140	QEP1 interrupt
-	65	-	0x0000_0144	Reserved
-	66	Can set	0x0000_0148	Reserved
-	67	Can set	0x0000_014C	Reserved
FLB1	68	Can set	0x0000_0150	FLB1 interrupt
FLB2	69	Can set	0x0000_0154	FLB2 interrupt
FLB3	70	Can set	0x0000_0158	FLB3 interrupt
FLB4	71	Can set	0x0000_015C	FLB4 interrupt
SPIA_RX	72	Can set	0x0000_0160	SPIA receive interrupt
SPIA_TX	73	Can set	0x0000_0164	SPIA transmit interrupt
-	74~79	-	0x0000_0168 ~0x0000_017C	Reserved
DMA_CH1	80	Can set	0x0000_0180	DMA channel 1 interrupt
DMA_CH2	81	Can set	0x0000_0184	DMA channel 2 interrupt
DMA_CH3	82	Can set	0x0000_0188	DMA channel 3 interrupt
DMA_CH4	83	Can set	0x0000_018C	DMA channel 4 interrupt
DMA_CH5	84	Can set	0x0000_0190	DMA channel 5 interrupt
DMA_CH6	85	Can set	0x0000_0194	DMA channel 6 interrupt
-	86	-	0x0000_0198	Reserved
-	87	-	0x0000_019C	Reserved
I2CA	88	Can set	0x0000_01A0	I2CA interrupt
I2CA FIFO	89	Can set	0x0000_01A4	I2CA FIFO interrupt
-	90	-	0x0000_01A8	Reserved
I2CB	91	Can set	0x0000_01AC	I2CB interrupt
I2CB FIFO	92	Can set	0x0000_01B0	I2CB FIFO interrupt

Exception type	Vector No.	Priority	Vector address	Description
-	93~95	-	0x0000_01B4 ~0x0000_01BC	Reserved
UARTA_RX	96	Can set	0x0000_01C0	UARTA receive interrupt
UARTA_TX	97	Can set	0x0000_01C4	UARTA transmit interrupt
UARTB_RX	98	Can set	0x0000_01C8	UARTB receive interrupt
UARTB_TX	99	Can set	0x0000_01CC	UARTB transmit interrupt
CANA INT0	100	Can set	0x0000_01D0	CANA interrupt 0
CANA INT1	101	Can set	0x0000_01D4	CANA interrupt 1
-	102	-	0x0000_01D8	Reserved
-	103	-	0x0000_01DC	Reserved
ADCA EVENT	104	Can set	0x0000_01E0	ADCA EVENT interrupt
ADCA2	105	Can set	0x0000_01E4	ADCA2 interrupt
ADCA3	106	Can set	0x0000_01E8	ADCA3 interrupt
ADCA4	107	Can set	0x0000_01EC	ADCA4 interrupt
-	108	-	0x0000_01F0	Reserved
-	109	-	0x0000_01F4	Reserved
-	110	-	0x0000_01F8	Reserved
-	111	-	0x0000_01FC	Reserved
EXTI_LINE0	112	Can set	0x0000_0200	EXTI LINE0 interrupt
EXTI_LINE1	113	Can set	0x0000_0204	EXTI LINE1 interrupt
EXTI_LINE2	114	Can set	0x0000_0208	EXTI LINE2 interrupt
EXTI_LINE3	115	Can set	0x0000_020C	EXTI LINE3 interrupt
EXTI_LINE4	116	Can set	0x0000_0210	EXTI LINE4 interrupt
EXTI_LINE5	117	Can set	0x0000_0214	EXTI LINE5 interrupt
EXTI_LINE6	118	Can set	0x0000_0218	EXTI LINE6 interrupt
EXTI_LINE7	119	Can set	0x0000_021C	EXTI LINE7 interrupt
EXTI_LINE8	120	Can set	0x0000_0220	EXTI LINE8 interrupt
EXTI_LINE9	121	Can set	0x0000_0224	EXTI LINE9 interrupt
EXTI_LINE10	122	Can set	0x0000_0228	EXTI LINE10 interrupt
EXTI_LINE11	123	Can set	0x0000_022C	EXTI LINE11 interrupt
EXTI_LINE12	124	Can set	0x0000_0230	EXTI LINE12 interrupt
-	125	-	0x0000_0234	Reserved
-	126	-	0x0000_0238	Reserved

Exception type	Vector No.	Priority	Vector address	Description
-	127	-	0x0000_023C	Reserved
-	128	-	0x0000_0240	Reserved
FPU DZC	129	Can set	0x0000_0244	FPU DZC interrupt
FPU IDC	130	Can set	0x0000_0248	FPU IDC interrupt
FPU IOC	131	Can set	0x0000_024C	FPU IOC interrupt
FPU OFC	132	Can set	0x0000_0250	FPU OFC interrupt
FPU UFC	133	Can set	0x0000_0254	FPU UFC interrupt
FPU IXC	134	Can set	0x0000_0258	FPU IXC interrupt
-	135~168	-	0x0000_025C ~0x0000_02E0	Reserved
CANFDA INT0	169	Can set	0x0000_02E4	CANFDA Interrupt 0
CANFDA INT1	170	Can set	0x0000_02E8	CANFDA interrupt 1
CANFDA BEC INT	171	Can set	0x0000_02EC	CANFDA ECC single-bit error
CANFDA BEU INT	172	Can set	0x0000_02F0	CANFDA ECC multi-bit error
CANFDA WAKE UP INT	173	Can set	0x0000_02F4	CANFDA Low-Power Wake-up
-	174~183	-	0x0000_02F8 ~0x0000_031C	Reserved
LINA INT1	184	Can set	0x0000_0320	LINA interrupt 1
LINA INT2	185	Can set	0x0000_0324	LINA interrupt 2
UARTC_RX	186	Can set	0x0000_0328	UARTC RX interrupt
UARTC_TX	187	Can set	0x0000_032C	UARTC TX interrupt
PMBUSA	188	Can set	0x0000_0330	PMBUSA interrupt
-	189~199	-	0x0000_0334 ~0x0000_035C	Reserved
ADCC EVENT	200	Can set	0x0000_0360	ADCC EVENT interrupt
ADCC2	201	Can set	0x0000_0364	ADCC2 interrupt
ADCC3	202	Can set	0x0000_0368	ADCC3 interrupt
ADCC4	203	Can set	0x0000_036C	ADCC4 interrupt
-	204~217	-	0x0000_0370 ~0x0000_03A4	Reserved
FLASH_ECC_ERR	218	Can set	0x0000_03A8	FLASH ECC interrupt
-	219	-	0x0000_03AC	Reserved
SYS_PLL_SLIP	220	Can set	0x0000_03B0	PLL SLIP interrupt
-	221~224	-	0x0000_03B4	Reserved

Exception type	Vector No.	Priority	Vector address	Description
			~0x0000_03C0	
CTI INT0	225	Can set	0x0000_03C4	CTI interrupt 0
CTI INT1	226	Can set	0x0000_03C8	CTI interrupt 1

14 External interrupt and event controller (EXTI)

14.1 Introduction

The Extended Interrupt and Event Controller (EXTI) is used to manage internal and external asynchronous events/interrupts and to generate event requests to the CPU/interrupt controller. In this manual, external interrupts refer to interrupts/events triggered by input signals from I/O pins, denoted as EXTIx in the interrupt vector table; other interrupts refer to internal interrupts/events, the event lines of which can wake the device from IDLE mode.The EXTI controller also allows software-simulated events or interrupts to be generated on the corresponding hardware event lines by programming dedicated registers.

14.2 Main characteristics

(1) Support 13 event/interrupt requests

(2) Can be configured independently as the line of external/internal event request

(3) Each event/interrupt line can be masked independently

(4) Capable of independently configuring effective edges

(5) Each external event/interrupt line can be triggered independently

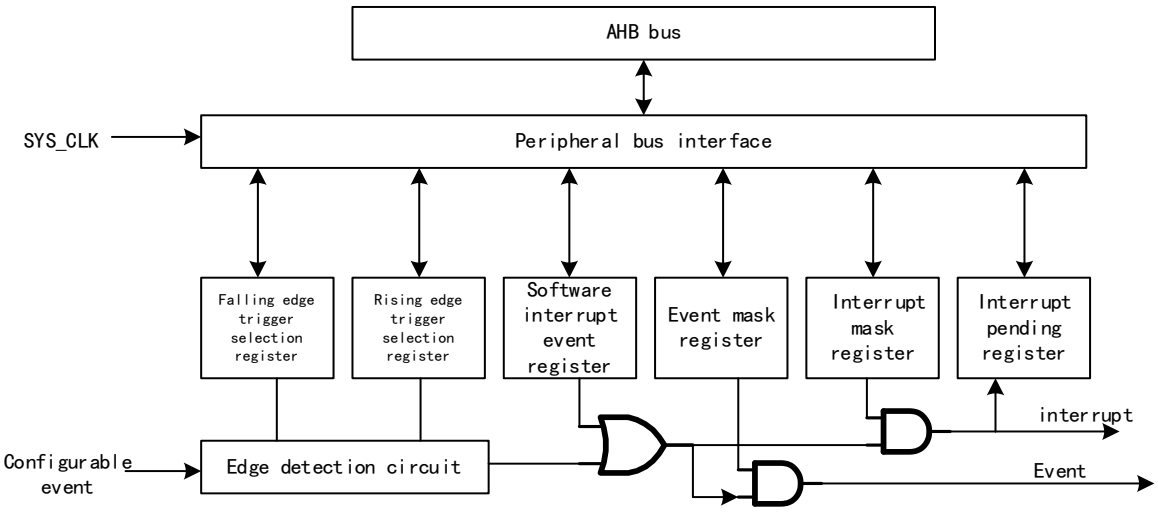
(6) Each external interrupt line has dedicated status bit

(7) Capable of configuring IO trigger selection

(8) Simulate all external event interrupts

14.3 Block Diagram

Figure10 Extended Interrupt and Event Controller (EXTI) Block Diagram



14.4 Functional description

To use the EXTI interrupt line, it must be configured and enabled in order to generate an interrupt. Set the two trigger registers according to the required edge detection, whilst writing a ‘1’ to the corresponding bit in the interrupt mask register to enable the interrupt request. When a selected signal edge occurs on the interrupt line, an interrupt request is generated and the corresponding suspend bit is set to 1. Writing a ‘1’ to the corresponding bit in the suspend register will clear the interrupt flag.

To generate an event, the event line must first be configured and enabled. Set the two trigger registers according to the required edge detection, whilst writing a ‘1’ to the corresponding bit in the event mask register to permit the event request. When a selected signal edge occurs on the event line, an event pulse is generated, but the corresponding suspend bit is not set to 1.

All EXTI lines can generate interrupt/event requests by writing a ‘1’ to the software interrupt/event register via software.

14.4.1 Classification and difference of "external interrupt and event"

"External interrupt and event" can be classified into external hardware interrupt, external hardware event, external software event and external software interrupt according to trigger source, configuration and execution process. The differences are shown in the table below:

Table 40 Classification and Differences of "External Interrupts and Events"

Name	Trigger source	Configuration and execution process
External hardware interrupt	External signal	(1) Set the trigger mode, allow the interrupt request, and enable corresponding peripheral interrupt line (enable in NVIC); (2) When an edge consistent with the configuration is generated on the external interrupt line, an interrupt request will be generated, and the corresponding pending bit will be set to 1; write 1 to the corresponding bit of the pending register and the interrupt request will be cleared.
External hardware event	External signal	(1) Set the trigger mode and enable the event line; (2) When an edge consistent with the configuration is generated on the external event line, an event request pulse will be generated, and the corresponding pending bit will not be set to 1.
External software event	Software interrupt register/transmit event (SEV) instruction	(1) Enable the event line; (2) Write 1 to the software interrupt event register of the corresponding event line to generate an event request pulse, and the corresponding pending bit will not be set to 1.
External software interrupt	Software interrupt register	(1) Allow interrupt request, and enable the corresponding peripheral interrupt line (enable in NVIC); (2) Write 1 to the software interrupt event register of the corresponding interrupt line to generate an interrupt request, the corresponding pending bit will be set to 1; write 1 to the corresponding bit of the pending register and the interrupt request will be cleared.

14.4.1.1 Hardware Interrupt Selection

To configure a line as an interrupt source, carry out the following steps:

(1) Configure the interrupt line’s mask bit (EXTI_IMSK)

(2) Configure the interrupt line’s trigger selection bits (EXTI_RTSR and EXTI_FTSR)

(3) Configure the enable and mask bits for the NVIC IRQ channels mapped to the Extended Interrupt Controller (EXTI), so that requests from any interrupt line can be responded to correctly.

14.4.1.2 Hardware Event Selection

To configure a line as an event source, carry out the following steps:

(1) Configure the event line’s mask bit (EXTI_EMSK)

(2) Configure the event line’s trigger selection bits (EXTI_RTSR and EXTI_FTSR)

14.4.1.3 Software Interrupt/Event Selection

Any configurable line can be configured as a software interrupt/event line. To generate a software interrupt, the following steps must be carried out

(1) Configure the interrupt/event line’s mask bits (EXTI_IMSK and EXTI_EMSK)

(2) Clear the corresponding request bit (EXTI_SWINTE) in the software interrupt register

(3) Set the corresponding request bit (EXTI_SWINTE) in the software interrupt register

(4) Clear the corresponding request bit (EXTI_SWINTE) in the software interrupt register

14.4.2 Core wake-up

Using WFI and WFE instructions can stop the core. When WFI instruction is used, any interrupt can wake up the core; when WFE instruction is used, the core can be woken up by an event.

When interrupt is used for wake-up, the interrupt handler function will be triggered, and normal interrupt configuration can wake up the core. When an event is used to wake up the core, the interrupt handler function will not be triggered, which will reduce the wake-up time, and the configuration method is:

(1) Trigger an internal interrupt (internal hardware event) but do not trigger the interrupt handler function for wake-up
- Enable an internal interrupt in the peripheral, but do not enable the corresponding interrupt in NVIC to avoid triggering the interrupt handler function
- Enable SEVONPEND bit in the system controller of the core, and execute WFE instruction to make the core enter sleep mode
- Generate an interrupt to wake up the core; when the core recovers from WFE, it is required to clear the pending bit of corresponding peripheral interrupt and the pending bit of peripheral NVIC interrupt channel (clear the pending register in the NVIC interrupt)

(2) Wake up by EXTI line events (external hardware event)
- Configure EXTI line as the event mode
- Execute WFE instruction to make the core enter the sleep mode
- Generate an interrupt to wake up the core; after the CPU recovers from WFE, since the pending bit of corresponding event line is not set, it is unnecessary to clear the interrupt pending bit of corresponding peripheral or the pending bit NVIC interrupt channel

14.4.3 External interrupt and event line mapping

Table 41 External Interrupt and Event Line Mapping

External Interrupt and Event Channel Name	External Interrupt and Event Line No.	Whether DMA can be triggered	Description
GPIO0-GPIO14	EXTI 0	No	IO trigger of GPIO0~GPIO14
GPIO15-GPIO29	EXTI 1	No	IO trigger of GPIO15~GPIO29
GPIO30-GPIO44	EXTI 2	No	IO trigger of GPIO30~GPIO44
GPIO45-GPIO59	EXTI 3	No	IO trigger of GPIO45~GPIO59
INPUTXBAR4	EXTI 4	Yes	Signal trigger of INPUTXBAR4
INPUTXBAR5	EXTI 5	Yes	Signal trigger of INPUTXBAR5
INPUTXBAR6	EXTI 6	Yes	Signal trigger of INPUTXBAR6
INPUTXBAR13	EXTI 7	Yes	Signal trigger of INPUTXBAR13
INPUTXBAR14	EXTI 8	Yes	Signal trigger of INPUTXBAR14
COMP1	EXTI 9	No	Signal trigger of COMP1.CTRIPH_OR_CTRIPL
COMP2	EXTI 10	No	Signal trigger of COMP2.CTRIPH_OR_CTRIPL
COMP3	EXTI 11	No	Signal trigger of COMP3.CTRIPH_OR_CTRIPL
COMP4	EXTI 12	No	Signal trigger of COMP4.CTRIPH_OR_CTRIPL

14.5 Register bank address

Table 42 Register Bank Address

Device register	Register bank	Start address	End address
EXTIRegs	EXTI_REGS	0x4001_0400	0x4001_07FF

14.6 Register address mapping

Table 43 External Interrupt/Event Controller Register Address Mapping

Register name	Description	Offset address	WRPRT
EXTI_RTEN	Rising edge trigger selection register	0x00	-
EXTI_FTEN	Falling edge trigger selection register	0x04	-
EXTI_SWINTE	Software interrupt event register	0x08	-

Register name	Description	Offset address	WRPRT
EXTI_GPIOSEL	Select GPIO trigger signal	0x0C	-
EXTI_IMASK	CPU interrupt mask register	0x10	-
EXTI_EMASK	CPU event mask register	0x14	-
EXTI_IPEND	CPU interrupt pending register	0x18	-
EXTI_INT4CNT	Interrupt line 4 counter	0x30	-
EXTI_INT5CNT	Interrupt line 5 counter	0x34	-
EXTI_INT6CNT	Interrupt line 6 counter	0x38	-

14.7 Register functional description

14.7.1 Enable rising edge trigger selection register (EXTI_RTEN)

Offset address: 0x00

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	RTENx	R/W	Rising Trigger Event Enable and Interrupt of Line x (x=0-12) 0: Disable 1: Enable	0h
31:13	Reserved			0h

Note: Since the external wake-up lines are edge-triggered, there should be no glitch signal on these lines; when writing EXTI_RTEN register, if the rising edge signal is on the external interrupt line, it will not be recognized and the pending bit will not be set; on the same interrupt line, the rising edge trigger and falling edge trigger can be set at the same time.

14.7.2 Enable falling edge trigger selection register (EXTI_FTEN)

Offset address: 0x04

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	FTENx	R/W	Falling Trigger Event Enable and Interrupt of Line x (x=0-12) 0: Disable 1: Enable	0h
31:13	Reserved			0h

Note: Since the external wake-up lines are edge-triggered, there should be no glitch signal on these lines; when writing EXTI_FTEN register, if the rising edge signal is on the external interrupt line, it will not be recognized and the pending bit will not be set; on the same interrupt line, the rising edge trigger and falling edge trigger can be set at the same time.

14.7.3 Software interrupt event register (EXTI_SWINTE)

Offset address: 0x08

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	SWINTEx	R/W	Software Interrupt Event on Line x (x=0-12) When the bit is 1, writing 1 has no effect. When this bit is 0, the pending bit of EXTI_IPEND can be set by writing 1. If EXTI_IMASK (EXTI_EMASK) is set to open the interrupt (event) request, an interrupt (event) will be generated. 0: Clear a software interrupt (event) 1: Generate a software interrupt (event)	0h
31:13	Reserved			0h

14.7.4 Select GPIO trigger signal register (EXTI_GPIOSEL)

Offset address: 0x0C

Reset type: System reset

Field	Name	R/W	Description	Reset value
3:0	GPIOSEL0	R/W	GPIO Trigger signal select 0 0000: GPIO0 0001: GPIO1 … 1110: GPIO14 1111: Reserved	0h
7:4	GPIOSEL1	R/W	GPIO Trigger signal select 1 0000: GPIO15 0001: GPIO16 … 0110: GPIO29 1111: Reserved	0h
11:8	GPIOSEL2	R/W	GPIO Trigger signal select 2 0000：GPIO30 0001：GPIO31 … 1110：GPIO44 1111: Reserved	0h
15:12	GPIOSEL3	R/W	GPIO Trigger signal select 3 0000：GPIO45 0001：GPIO46 … 1110：GPIO59 1111: Reserved	0h
31:16	Reserved			0h

14.7.5 CPU interrupt mask register (EXTI_IMASK)

Offset address: 0x10

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	IMASKx	R/W	Interrupt Request Mask on Line x (x=0~12) 0: Mask 1: Open	0h
31:13	Reserved			0h

14.7.6 CPU event mask register (EXTI_IMASK)

Offset address: 0x14

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	EMASKx	R/W	Event Request Mask on Line x (x=0-12) 0: Mask 1: Open	0h
31:13	Reserved			0h

14.7.7 CPU interrupt pending register (EXTI_IPEND)

Offset address: 0x18

Reset type: System reset

Field	Name	R/W	Description	Reset value
12:0	IPENDx	RC_W1	Interrupt Pending Occur of Line x Flag (x=0-12) Whether the selectable trigger request occurs 0: No 1: Occurred When a software trigger interrupt occurs or a request is triggered by the corresponding edge of EXTI_RTEN/EXTI_FTEN on the external interrupt line, set 1 by hardware; clear 0 by writing 1 to this bit. Failure to clear this bit does not affect the triggering of the external interrupt line on a subsequent occasion.	0h
31:13	Reserved			0h

14.7.8 Interrupt line 4 counter (EXTI_INT4CNT)

Offset address: 0x30

Reset type: System reset

Field	Name	R/W	Description	Reset value
15:0	INT4CNT	R	Interrupt Line 4 Counter Generating an effective interrupt or system reset can reset the counter. When CPU enables EXTI 4 interrupt, it starts counting up at the clock frequency of SYSCLK, with a maximum count of 0xFFFF. When the interrupt is enabled and the count reaches 0xFFFF, the counter will restart counting, and when EXTI 4 generates an interrupt, the counter will be cleared to 0. When both CPU disable the EXTI 4 interrupt, the counter will stop counting. The counter is a read-only register; it can only be reset by generating a valid interrupt or by a system reset.	0h
31:16	Reserved			0h

14.7.9 Interrupt line 5 counter (EXTI_INT5CNT)

Offset address: 0x34

Reset type: System reset

Field	Name	R/W	Description	Reset value
15:0	INT5CNT	R	Interrupt Line 5 Counter Generating an effective interrupt or system reset can reset the counter. When CPU enables EXTI 5 interrupt, it starts counting up at the clock frequency of SYSCLK, with a maximum count of 0xFFFF. When the interrupt is enabled and the count reaches 0xFFFF, the counter will restart counting, and when EXTI 5 generates an interrupt, the counter will be cleared to 0. When both CPU disable the EXTI 5 interrupt, the counter will stop counting. The counter is a read-only register; it can only be reset by generating a valid interrupt or by a system reset.	0h
31:16	Reserved			0h

14.7.10 Interrupt line 6 counter (EXTI_INT6CNT)

Offset address: 0x38

Reset type: System reset

Field	Name	R/W	Description	Reset value
15:0	INT6CNT	R	Interrupt Line 6 Counter Generating an effective interrupt or system reset can reset the counter. When CPU enables EXTI 6 interrupt, it starts counting up at the clock frequency of SYSCLK, with a maximum count of 0xFFFF. When the interrupt is enabled and the count reaches 0xFFFF, the counter will restart counting, and when EXTI 6 generates an interrupt, the counter will be cleared to 0. When both CPU disable the EXTI 6 interrupt, the counter will stop counting. The counter is a read-only register; it can only be reset by generating a valid interrupt or by a system reset.	0h
31:16	Reserved			0h

15 Dual Code Security (DCS)

For description of this module, please refer to *G32R5xx DCS User Manual V1.0*.

16 Configurable static memory subsystem (CFGSMS)

For description of this module, please refer to *G32R5xx CFGSMS User Manual V1.0*.

17 DBGMCU

17.1 Real-time Mode

17.1.1 Function description

Real-time mode supports the CPU responding to interrupts whilst in the Debug-halt state.

When real-time mode is enabled, the interrupt response process is as follows:

(1) The CPU executes until it reaches a breakpoint and enters the Debug-halt state;

(2) Upon detection of an interrupt request, the CPU exits the Debug-halt state, enters the RUN state, and responds to the interrupt service;

(3) Once all interrupt service routines have been completed and the interrupt has been cleared, the CPU returns to the original breakpoint and re-enters the Debug-halt state;

17.1.2 Important Notes

(1) Do not set breakpoints within interrupt service routines, as this will cause the CPU to halt at the breakpoint;

(2) When enabling real-time mode, it is recommended to enable both the CPU_REAL_TIME_MODE and CPU_INT_RUN_MODE bits;

(3) For peripherals that need to function normally in debug mode, please refer to the DBG_CTRLx registers for configuration;

(4) Real-time mode debugging does not currently support software breakpoints; it only supports software breakpoints created by inserting the BKPT instruction into the code or hardware breakpoints;

(5) Real-time mode debugging does not currently support single-step execution; debugging can only be performed using a combination of breakpoints and full-speed execution;

(6) If interrupt requests occur too frequently, debugging may become impossible. When interrupt requests occur at a high frequency, it is recommended to follow the procedure below for debugging.

17.1.3 Recommended debugging workflow for real-time mode

Using KEIL debugging as an example (it is recommended to use only one hardware breakpoint for debugging):

(1) Run at full speed; the CPU will halt at the breakpoint;

(2) Clear the previous breakpoint and set the next one;

(3) Repeat steps 1 – 2 to carry out debugging.

17.2 Register Address Mapping

Base address: 0xE005 9000

Table44 DBGMCU Register Address Mapping

Register Name	Register Description	Offset Address	WRPRT
DBG_IDCODE	ID register	0x00	-
DBG_CFG	Configuration register	0x04	-
DBG_CTRL1	Control Register 1	0x30	-
DBG_CTRL2	Control Register 2	0x38	-
DBG_CTRL3	Control Register 3	0x40	-

17.2.1 ID Register (DBG_IDCODE)

Offset address: 0x00

Reset value: 0x0021 A008

Field	Name	R/W	Description
11:0	DEV_ID	R	Device identifier
15:12	WAFERVER	R	Wafer version
31:16	REV_ID	R	Version identifier

17.2.2 Configuration register (DBG_CFG)

Offset address: 0x04

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	DBG_STOP_EN	R/W	Debug Stop Mode Enable 0: (Clock Disabled) In stop mode, the clock disable status is controlled by the CLKSRCCTL1 register. Upon exiting stop mode, the clock configuration is the same as that following a RESET. Therefore, the software must reprogram the clock controller to enable the PLL, crystal oscillator and other components. 1: (Clock enabled) In this case, when entering stop mode, the system clock remains active during stop mode. Upon exiting stop mode, the software must reprogram the clock controller to enable the PLL, crystal oscillator, etc. (the procedure is the same as when DBG_STOP_EN = 0)
5:1	Reserved		

Field	Name	R/W	Description
6	CPU_TSG_HALT	R/W	0: The Global Timestamp is not paused when the CPU enters HALT mode 1: The Global Timestamp is suspended when the CPU enters HALT mode
15:7	Reserved		
16	CPU_REAL_TIME_MODE	R/W	0: The CPU does not respond to interrupts whilst in HALT mode 1: When the CPU is running in a non-interrupt service routine and enters HALT mode via a breakpoint, if an interrupt occurs, the interrupt service routine will be processed; upon completion, the CPU will return to the breakpoint and re-enter HALT mode. When configured for real-time mode, the CPU does not support entering HALT mode via a breakpoint whilst in an interrupt service routine.
17	Reserved		
18	CPU_INT_RUN_MODE	R/W	0: When the CPU is configured to operate in real-time mode, execution within an interrupt service routine will cause the CPU to enter HALT mode 1: When the CPU is configured to operate in real-time mode, the CPU cannot enter HALT mode whilst executing an interrupt service routine, thereby preventing errors caused by an unexpected HALT occurring within the interrupt service routine.
27:19	Reserved		
28	TRGOEN	R/W	Configure the direction of the bidirectional trigger pin TRGIO 0: Input – TRGIO is connected to TRGIN 1: Output – TRGIO is connected to TRGOUT
31:29	Reserved		

17.2.3 Control Register 1 (DBG_CTRL1)

Offset address: 0x30

Reset value: 0x0000 0000

Field	Name	R/W	Description
6:0	DBG_CPU_PWMx	R/W	The state of PWMx (x=1–7) when the CPU enters HALT mode 0: When the CPU enters HALT mode, PWMx operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of PWMx is controlled by the peripheral’s internal debug-related registers
7	Reserved		

Field	Name	R/W	Description
10:8	DBG_CPU_CAPx	R/W	The state of CAPx (x=1–3) when the CPU enters HALT mode 0: When the CPU enters HALT mode, CAPx operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of CAPx is controlled by the peripheral’s internal debug-related registers
15:11	Reserved		
19:16	DBG_CPU_COMPx	R/W	The state of COMPx (x=1–4) when the CPU enters HALT mode 0: When the CPU enters HALT mode, COMPx operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of COMPx is controlled by the peripheral’s internal debug-related registers
23:20	Reserved		
24	DBG_CPU_QEP	R/W	The state of QEP when the CPU enters HALT mode 0: When the CPU enters HALT mode, QEP operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, QEP operation is controlled by the peripheral’s internal debug-related registers
31:25	Reserved		

17.2.4 Control Register 2 (DBG_CTRL2)

Offset address: 0x38

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	DBG_CPU_LIN	R/W	The state of LIN when the CPU enters HALT mode 0: When the CPU enters HALT mode, LIN operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, LIN operation is controlled by the peripheral’s internal debug-related registers
1	DBG_CPU_UARTA	R/W	The state of UARTA when the CPU enters HALT mode 0: When the CPU enters HALT mode, UARTA operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of the UARTA is controlled by the peripheral’s internal debug-related registers
2	DBG_CPU_SPI	R/W	The state of SPI when the CPU enters HALT mode 0: When the CPU enters HALT mode, SPI operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, SPI operation is controlled by the peripheral’s internal debug-related registers

Field	Name	R/W	Description
3	DBG_CPU_I2CA	R/W	The state of I2CA when the CPU enters HALT mode 0: When the CPU enters HALT mode, I2CA operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, I2CA operation is controlled by the peripheral’s internal debug-related registers
4	DBG_CPU_CAN	R/W	The state of CAN when the CPU enters HALT mode 0: When the CPU enters HALT mode, CAN operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, CAN operation is controlled by the peripheral’s internal debug-related registers
8:5	Reserved		
9	DBG_CPU_UARTB	R/W	The state of UARTB when the CPU enters HALT mode 0: When the CPU enters HALT mode, UARTB operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of UARTB is controlled by the peripheral’s internal debug-related registers
10	Reserved		
11	DBG_CPU_I2CB	R/W	The state of I2CB when the CPU enters HALT mode 0: When the CPU enters HALT mode, I2CB operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of I2CB is controlled by the peripheral’s internal debug-related registers
12	Reserved		
13	DBG_CPU_UARTC	R/W	The state of UARTC when the CPU enters HALT mode 0: When the CPU enters HALT mode, UARTC operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of the UARTC is controlled by the peripheral’s internal debug-related registers
15:14	Reserved		
19:16	DBG_CPU_FLBx	R/W	The state of FLBx (x=1–4) when the CPU enters HALT mode 0: After the CPU enters HALT mode, FLBx operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of FLBx is controlled by the peripheral’s internal debug-related registers
31:20	Reserved		

17.2.5 Control Register 3 (DBG_CTRL3)

Offset address: 0x40

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	DBG_CPU_WDT	R/W	The state of WDT when the CPU enters HALT mode 0: When the CPU enters HALT mode, the WDT/NMIWDT operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of the WDT/NMIWDT is controlled by the peripheral’s internal debug-related registers
7:1	Reserved		
10:8	DBG_CPU_TMR	R/W	The state of CPU Timer x (x=0–2) when the CPU enters HALT mode, 0: When the CPU enters HALT mode, CPU Timer x operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, the operation of CPU Timer x is controlled by the peripheral’s internal debug-related registers
11	DBG_CPU_DMA	R/W	The state of DMA when the CPU enters HALT mode 0: When the CPU enters HALT mode, DMA operates normally and is not affected by debug signals 1: When the CPU enters HALT mode, DMA operation is controlled by the peripheral’s internal debug-related registers
31:12	Reserved		

18 Nonvolatile memory controller (NVMC)

18.1 Introduction

CPU can read the content of Flash through paths of CPU CACHE-> C-Bus -> busmatrix -> Flash; while CPU CACHE is used to accelerate C-BUS access.

All FLASH erase/write operations are performed by calling the Flash API. The FLASH MEM area address is read-only for CPU0/1 and DMA, and all write operations of the bus will be ignored. When erase/write and read requests are initiated simultaneously, the erase/erase priority is higher than read priority.

When all interfaces initiate read access simultaneously, the priority of Flash response to read access is:

(1) BUS Interface0 access

(2) BUS Interface1 access

BUS Interface0/1 can be accessed simultaneously, and users can flexibly allocate which CPU to access which Flash through which interface according to their actual situation, in order to avoid resource competition.

When accessing through CBUS:

The Flash address space of 0x08XX_XXXX is accessed through BUS Interface 0.

The Flash address space of 0x09XX_XXXX is accessed through BUS Interface 1.

CPU CBUS access is arbitrated through busmtrix and given to BUS Interfaces0 and 1.

18.2 Block diagram

The specific implementation of the NVMC is shown in the figure below.

The NVMC comprises five sub-modules: bus0 serves as the read interface for external masters; flash0 and flash1 comprise the FLASH memory and associated controllers; OBL is the hardware boot module that operates between a hard reset and a soft reset; and reg_biu is the NVMC control register interface.

Figure11 NVMC Block Diagram

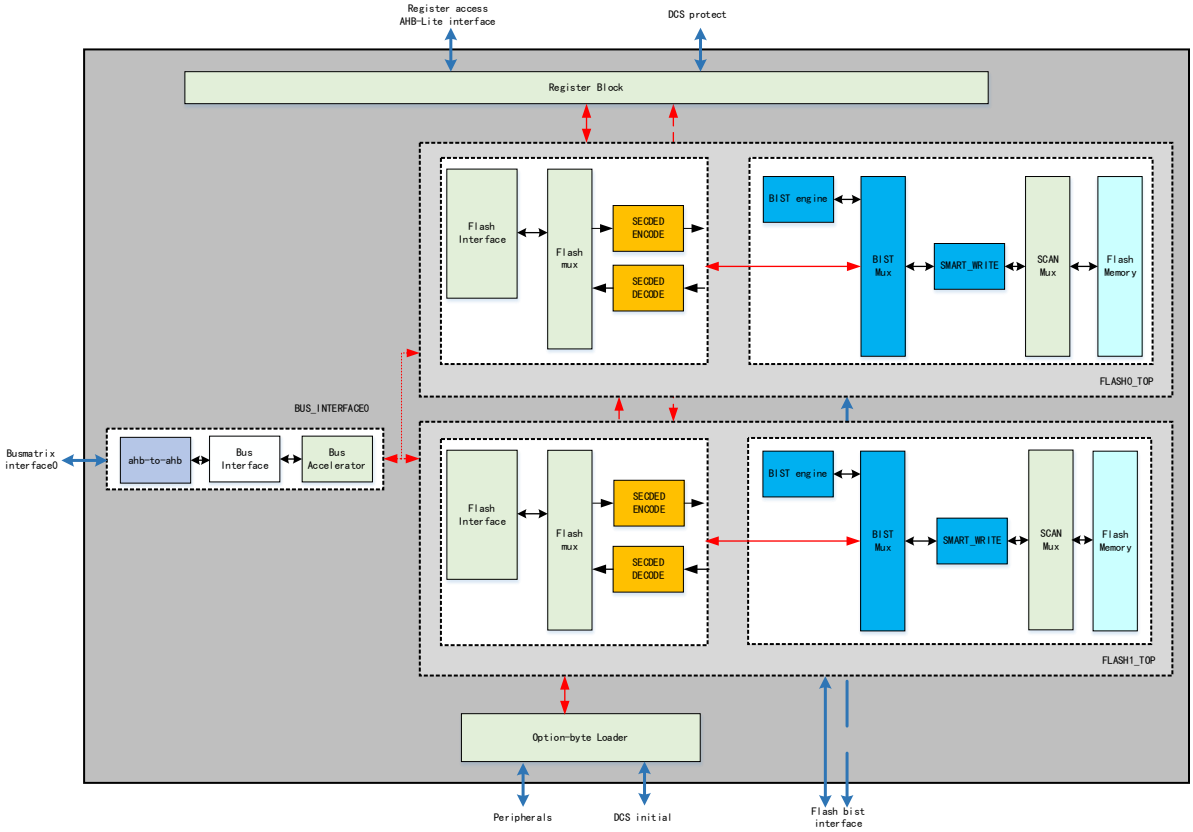
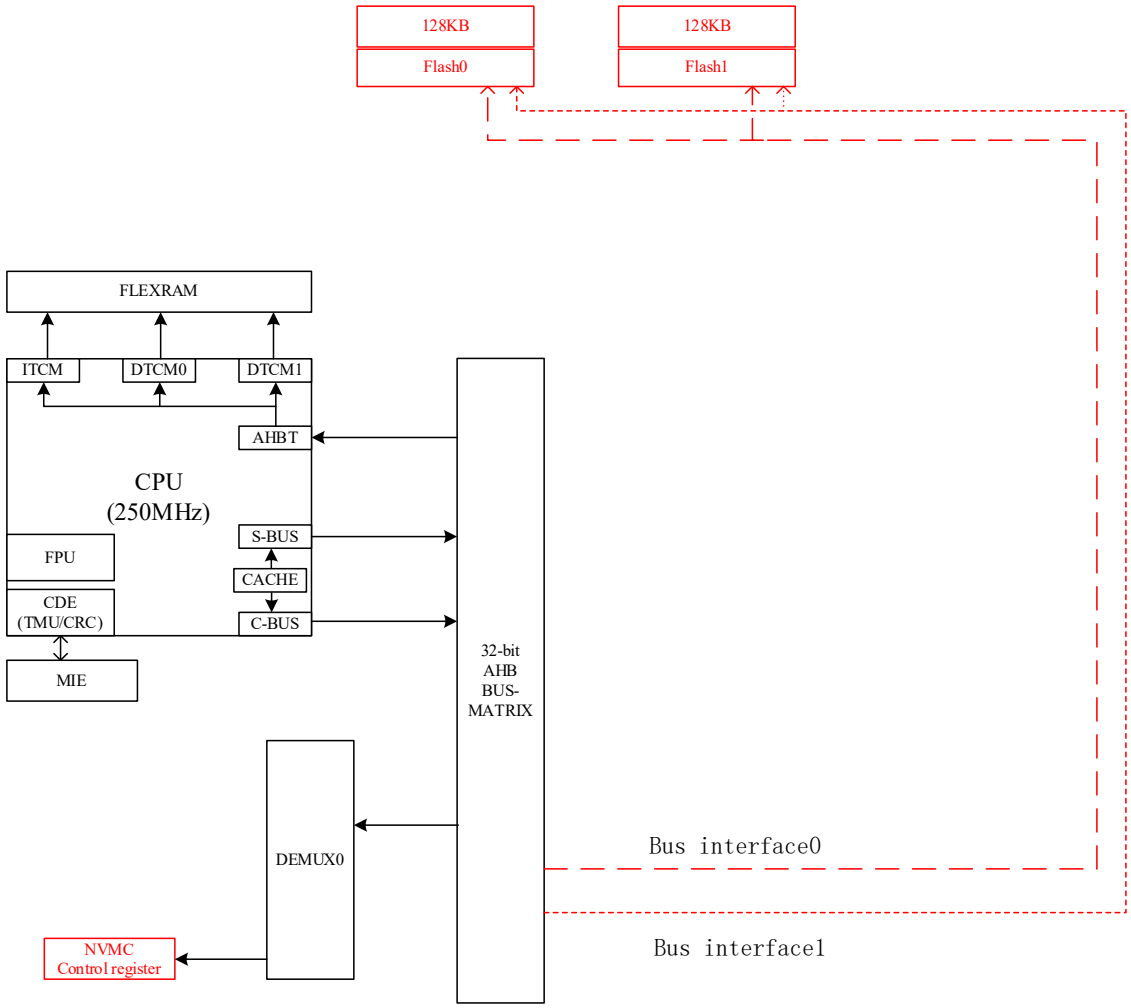


Figure12 Block Diagram of the System Architecture



18.3 Main characteristics

(1) Flash supports switching between single and double banks, with variable bit width

- In single-bank mode: nDBANK=1; Flash bit width is 256 bits
- In dual-bank mode: nDBANK=0; Flash bit width is 128 bits

(2) Read/write parallel (dual-bank mode only)

(3) Read parallel (dual-bank mode only)

(4) Busmatrix prefetch: Used to accelerate CPU to access Flash through busmatrix. Two prefetch buffers are used for ping-pong prefetching, with a buffer bit width of 128/256 bits (depending on the bit width).

(5) Support 64/128-bit programs with ECC

(6) Support 256-bit programs with ECC (single-bank mode bank0&1 mix area)

(7) Support Flash sector erase, with a sector size of 16KB (single-bank mode bank0&1 mix area)

(8) Support Flash sector erase, with a sector size of 8KB (dual-bank mode; or single-bank mode bank1 exclusive area)

(9) Support OTP area, DCS related OTP configuration can be programmed once; OTP sectors cannot be erased.

(10) Support user option byte, can configure Flash single/dual-bank mode; CPU boot mode and boot address; CFGSMS space allocation.

(11) Support ECC
- SECDED FMC supports single and double error correction
- The address bits are included in ECC
- Test mode is used to check the health status of ECC logic

(12) Flash supports low-power mode

(13) Flash erase/write is completed by calling relevant API functions

(14) AB SWAP is supported

18.4 Clocks

The NVMC contains three clock domains: the system clock domain (up to 200 MHz), the HSI10 clock domain (10 MHz) and the BIST clock domain (up to 20 MHz). The BIST clock domain is only active during CP/FT FLASH BIST testing.

18.5 Reset

There are two reset sources in the NVMC: system reset (soft reset) and hard reset. The OBL submodule is controlled only by hard resets (power-on reset, WDT reset, XRSn reset), whilst all other modules are subject to all reset signals.

18.6 Interrupts

There are two types of interrupts:

FLASH_CORRECTABLE_ERR, generated when a certain number of ECC single-bit errors occur; this is connected to the CPU's NVIC.

FLASH_UNCORRECTABLE_ERR, which is generated when a multi-bit ECC error occurs; this is connected to the watchdog (WDT) as one of the NMI sources.

18.7 Low Power Consumption

The NVMC's low-power design comprises two parts:

(1) Low power consumption for FLASH memory

(2) Low power consumption for digital logic

18.7.1 Low power consumption for FLASH memory

A counter is used; if the FLASH memory receives no access requests within the counting window, it requests the FIU to put the FLASH memory into sleep mode.

The low-power modes for FLASH are as follows:

(1) Bank Power Mode: Supports standby and active.

Table45 Bank Power Mode Comparison Table

Flash Status	Description
standby	Flash in sleep mode, minimum power consumption
active	Normal mode

To enter the low-power mode:

- Configure the FBAC register to set the number of cycles for which the ACTIVE state can be maintained.
- Configure the FBFALLBACK register to specify the state the Flash bank is to enter.
- Cease access to Flash (stop all read, write and erase operations).
- When the duration of inactivity in accessing Flash reaches the number of cycles configured in the FBAC register, Flash automatically enters the low-power state configured in the FBFALLBACK register.
- Once in low-power mode, any access to the Flash will wake it up.

(2) Charge Pump Power Mode: Only the 'active' mode is supported (the Flash will not lose power).

(3) Time from bank standby to active = 65 INTOSC1 clock (10 MHz) cycles.

18.7.2 Digital Logic Low Power

The method for achieving low power consumption in digital logic is to manually insert ICGs at the submodule level and disable the clock when the submodule is not in use. These ICGs can be configured to bypass via OTP.

18.8 Functional description

18.8.1 Flash memory organization

Flash memory has two organisational configurations: single-bank mode and dual-bank mode. Whilst the total capacity of the flash memory is the same in both modes, the following differences exist:

(1) In single-bank mode, the two 128KB main regions of the flash memory, totalling 256KB of space, are concatenated in terms of bit width, increasing the DOUT bit width for reading a single flash memory address to 288 bits (2 × 144). After concatenation, parallel read or read/write operations cannot be performed across different banks.

(2) In dual-bank mode, the read bit width of the FLASH memory is not concatenated; the DOUT bit width for reading a single FLASH memory address is 144 bits. Different banks can perform parallel read or parallel read/write operations.

The two FLASH banks are identical, with each FLASH bank divided into the following three sections:

(3) Main block
- Total 8Kx144bit(128KByte)

(4) IFREN block
- Total 2Kx128bit(32KByte)
- 4 sectors, 8KByte per sector

Table 46 Memory Single-bank Organization (256-bit Read Width)

IP	Name	Sector based on C-BUS interface	Sector size	Bank number
Main memory	Sector0	0x0800_0000~0x0800_3FFF	16KByte	Bank0 & Bank1 mix
	Sector1	0x0800_4000~0x0800_7FFF	16KByte	Bank0 & Bank1 mix
	Sector2	0x0800_8000~0x0800_BFFF	16KByte	Bank0 & Bank1 mix
	…	…	…	Bank0 & Bank1 mix
	Sector14	0x0803_8000~0x0803_BFFF	16KByte	Bank0 & Bank1 mix
	Sector15	0x0803_C000~0x0803_FFFF	16KByte	Bank0 & Bank1 mix
IFREN	Bank0 back-up sector0	0x0810_0000~0x0810_1FFF	8Kbyte	Bank0
	Bank0 back-up sector1	0x0810_2000~0x0810_3FFF	8Kbyte	Bank0
	Bank0 back-up sector2	0x0810_4000~0x0810_5FFF	8KByte	Bank0
	Bank0 back-up sector3	0x0810_6000~0x0810_7FFF	8KByte	Bank0
	Bank1 User Option byte	0x0810_8000~0x0810_9FFF	8Kbyte	Bank1
	Bank1 OTP	0x0810_A000~0x0810_BFFF	8Kbyte	Bank1
	Bank1 back-up sector2	0x0810_C000~0x0810_DFFF	8KByte	Bank1

IP	Name	Sector based on C-BUS interface	Sector size	Bank number
	Bank1 back-up sector3	0x0810_E000~0x0810_FFFF	8KByte	Bank1
IFREN1	Bank0 FLASH trimming	0x0818_0000~0x0818_1FFF	8KByte	Bank0
	Bank1 FLASH trimming	0x0818_2000~0x0818_3FFF	8KByte	Bank1
ECC	Main ECC	0x0900_0000~0x0900_7FFF	32Kbyte	Bank0 & Bank1 mix
	Bank0 IFREN ECC	0x0902_0000~0x0902_0FFF	4Kbyte	Bank0
	Bank1 IFREN ECC	0x0902_1000~0x0902_1FFF	4Kbyte	Bank1

Table 47 Memory Double-bank Organization (128-bit Read Width)

IP	Name	Sector based on C-BUS interface	Sector size	Bank number
Main memory bank0	Sector0	0x0800_0000~0x0800_1FFF	8KByte	Bank0
	Sector1	0x0800_2000~0x0800_3FFF	8KByte	Bank0
	Sector2	0x0800_4000~0x0800_5FFF	8KByte	Bank0
	…	…	…	Bank0
	Sector14	0x0801_C000~0x0801_DFFF	8KByte	Bank0
	Sector15	0x0801_E000~0x0801_FFFF	8KByte	Bank0
Main memory bank1	Sector16	0x0802_0000~0x0802_1FFF	8KByte	Bank1
	Sector17	0x0802_2000~0x0802_3FFF	8KByte	Bank1
	Sector18	0x0802_4000~0x0802_5FFF	8Kbyte	Bank1
	…	…	…	Bank1
	Sector30	0x0803_C000~0x0803_DFFF	8KByte	Bank1
	Sector31	0x0803_E000~0x0803_FFFF	8KByte	Bank1
IFREN	Bank0 back-up sector0	0x0810_0000~0x0810_1FFF	8Kbyte	Bank0
IFREN	Bank0 back-up sector1	0x0810_2000~0x0810_3FFF	8Kbyte	Bank0
	Bank0 back-up sector2	0x0810_4000~0x0810_5FFF	8KByte	Bank0
	Bank0 back-up sector3	0x0810_6000~0x0810_7FFF	8KByte	Bank0
	Bank1 User Option byte	0x0810_8000~0x0810_9FFF	8Kbyte	Bank1
	Bank1 OTP	0x0810_A000~0x0810_BFFF	8Kbyte	Bank1
	Bank1 back-up sector2	0x0810_C000~0x0810_DFFF	8KByte	Bank1
	Bank1 back-up sector3	0x0810_E000~0x0810_FFFF	8KByte	Bank1
IFREN1	Bank0 FLASH trimming	0x0818_0000~0x0818_1FFF	8KByte	Bank0
	Bank1 FLASH trimming	0x0818_2000~0x0818_3FFF	8KByte	Bank1
ECC	Bank0 Main ECC	0x0900_0000~0x0900_3FFF	16Kbyte	Bank0
	Bank1 Main ECC	0x0900_4000~0x0900_7FFF	16Kbyte	Bank1
	Bank0 IFREN ECC	0x0902_0000~0x0902_0FFF	4Kbyte	Bank0
	Bank1 IFREN ECC	0x0902_1000~0x0902_1FFF	4Kbyte	Bank1

18.8.1.1 BANK SWAP mode

Bank swap mode is only enabled in dual-bank mode; once bank swap mode is enabled, the address mappings of the main regions of Flash banks 0 and 1 are swapped.

Changes to the address mapping caused by the bank swap function may lead to the leakage of security code protected by DCS; therefore, the bank swap function can only be switched when DCS Z1 and Z2 are both unlocked; alternatively, if DCS Z1 and Z2 are not both unlocked, the bank swap function can be switched provided that the DCS OTP security configurations for the sectors corresponding to FLASH BANK 0 and 1 are identical.

Table48 Dual-bank memory structure with bank switching (128-bit read word width)

IP	Name	Sector based on the C-BUS interface	Sector size	Bank number
Main memory bank 0	Sector 0	0x0800_0000–0x0800_1FFF	8 KByte	Bank 0
	Sector 1	0x0800_2000–0x0800_3FFF	8 KByte	Bank 0
	Sector 2	0x0800_4000–0x0800_5FFF	8 KByte	Bank 0
	…	…	…	Bank 0
	Sector14	0x0801_C000–0x0801_DFFF	8 KByte	Bank 0
	Sector 15	0x0801_E000–0x0801_FFFF	8 KByte	Bank 0
Main memory bank 1	Sector 16	0x0802_0000–0x0802_1FFF	8 KByte	Bank 1
	Sector 17	0x0802_2000–0x0802_3FFF	8 KByte	Bank 1
	Sector 18	0x0802_4000–0x0802_5FFF	8 Kbyte 1	Bank 1
	…	…	…	Bank 1
	Sector 30	0x0803_C000–0x0803_DFFF	8 KByte	Bank 1
	Sector 31	0x0803_E000–0x0803_FFFF	8 KByte	Bank 1
IFREN	Bank 0 backup sector 0	0x0810_0000–0x0810_1FFF	8 Kbyte	Bank 0
	Bank 0 back-up sector 1	0x0810_2000–0x0810_3FFF	8 Kbyte	Bank 0
	Bank0 backup sector 2	0x0810_4000–0x0810_5FFF	8 KByte	Bank 0
	Bank0 backup sector 3	0x0810_6000–0x0810_7FFF	8 KByte	Bank 0
	Bank1 User Option byte	0x0810_8000–0x0810_9FFF	8 Kbyte	Bank 1
	Bank1 User OTP	0x0810_A000–0x0810_BFFF	8 Kbyte	Bank 1
	Bank1 backup sector 2	0x0810_C000–0x0810_DFFF	8 KByte	Bank 1
	Bank 1 backup sector 3	0x0810_E000–0x0810_FFFF	8 KByte	Bank 1
IFREN1	Bank 0 FLASH trimming	0x0818_0000–0x0818_1FFF	8 KByte	Bank 0

	Bank1 FLASH trimming	0x0818_2000–0x0818_3FFF	8 KByte	Bank 1
ECC	Bank 0 Main ECC	0x0900_0000–0x0900_3FFF	16 Kbyte	Bank0
	Bank1 Main ECC	0x0900_4000–0x0900_7FFF	16 Kbyte	Bank 1
	Bank 0 IFREN ECC	0x0902_0000–0x0902_0FFF	4 Kbytes	Bank 0
	Bank1 IFREN ECC	0x0902_1000–0x0902_1FFF	4 Kbytes	Bank 1

18.8.2 DCS OTP

0x0810_A000~0x0810_A7FF and 0x0810_B000~0x0810_B7FF are DCS OTP area, and this area is programmable.

LINKPOINTER area of DCS:

0x0810_A000~0x0810_A017, 0x0810_A400~0x0810_A417;

0x0810_B000~0x0810_B017, 0x0810_B400~0x0810_B417;

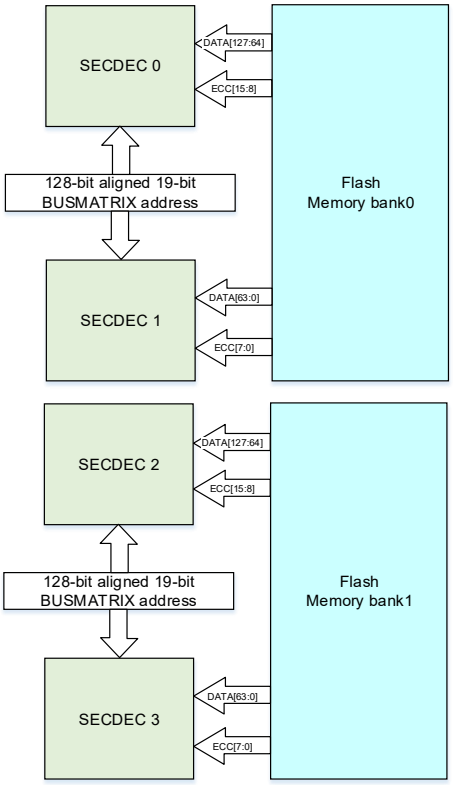
This area can be programmed multiple times (programming 1 to 0) and is not protected by ECC. It can be programmed by calling API.

The remaining OTP of DCS is protected by ECC and can only be programmed once, and can be programmed by calling API.

18.8.3 Flash ECC

Four SECDED ECC hardware verification modules are used in this chip, because two Flashes are needed to read data in parallel.

Figure 13 Flash ECC Decoding Block Diagram

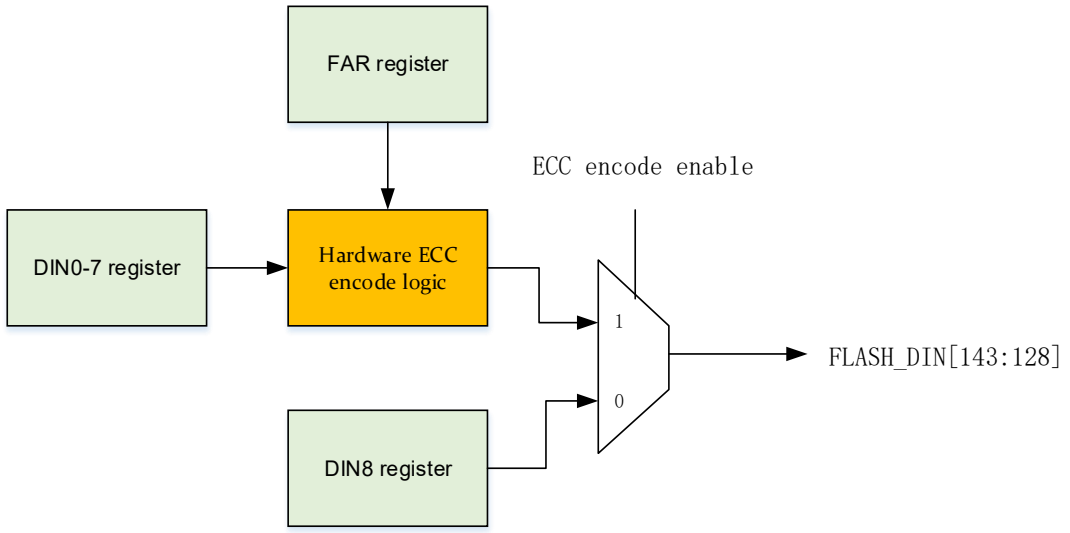


ECC_ENABLE.ENCODE_ENABLE is used to select to use hardware to generate ECC or define ECC by users. The hardware structure is shown in the following figure.

When the ENCODE_ENABLE value is 0xA, the hardware ECC encoding module calculates the corresponding ECC value based on the values in DIN0-7 and FAR registers, and stores it in Flash during programming.

When the ENCODE_ENABLE value is not 0xA, the ECC value that needs to be written to Flash can be set by hiding the register DIN8. When programming, DIN8[15:0] will be written to the ECC area of Flash bank0, and DIN8[31:16] will be written to the ECC area of Flash bank1.

Figure 14 Flash ECC Encoding Block Diagram



Directly access the ECC value stored in the Flash main memory area through the address. It can be used to directly access the address range of ECC.

Through the Flash ECC area, namely 0x0900_0000~0x0901_FFFF address, the ECC value stored in the main memory can be directly read in a half-word manner.

The data read each time is 16 bits, corresponding to the ECC value of the offset *8 of the read address relative to 0x0900_0000 in the main memory. That is, the value read at (0x0900_000 + offset) is the ECC value of the data at (0x0800_0000 + offset*8).

The specific mapping relationship between the main area, back-up area, user option byte area, and user OTP area is as follows:

Table 49 Flash ECC Address Mapping Relationship Table

IP	Name	Address range on the bus	Address range on ITCM	Corresponding ECC address (Half-word access)
Main memory	Main Sectors	0x0800_0000~0x080A_0000	0x0010_0000~0x001A_0000	0x0900_0000~0x0901_4000
IFREN	Bank0 back-up sector0	0x0810_0000~0x0810_1FFF	0x0008_0000~0x0008_1FFF	0x0902_0000~0x0902_0400
IFREN	Bank0 back-up sector1	0x0810_2000~0x0810_3FFF	0x0008_2000~0x0008_3FFF	0x0902_0400~0x0902_0800
IFREN	Bank0 back-up sector2	0x0810_4000~0x0810_5FFF	0x0008_4000~0x0008_5FFF	0x0902_0800~0x0902_0c00
IFREN	Bank0 back-up sector3	0x0810_6000~0x0810_7FFF	0x0008_6000~0x0008_7FFF	0x0902_0c00~0x0902_1000
IFREN	Bank1 User Option byte	0x0810_8000~0x0810_9FFF	0x0008_8000~0x0008_9FFF	0x0902_1000~0x0902_1400

IP	Name	Address range on the bus	Address range on ITCM	Corresponding ECC address (Half-word access)
IFREN	Bank1 User OTP	0x0810_A000~0x0810_BFFF	0x0008_A000~0x0008_BFFF	0x0902_1400~0x0902_1800

When there is a Flash EC error, two interrupts will be generated: Flash_ecc_correctable_int and Flash_ecc_uncorrectable_int. Flash_ecc_uncorrectable_int connects to CPU NMI, and Flash_ecc_correctable_int connects to CPU NVIC.

Note: Regardless of whether the BANK SWAP function is enabled, the ECC check logic for FLASH bank 0 calculates the ECC using the fixed base address 0x0800_0000, whilst FLASH bank 1 calculates the ECC using the fixed base address 0x0802_0000.

18.8.4 Low power consumption of Flash

(1) Bank power mode supports standby and active modes

(2) Charge Pump power mode only supports active mode (Flash will not power down)

(3) Bank standby to active time=65 INTOSC1 clock (10MHz) cycles

Table 50 Bank Power Mode

Flash state	Description
standby	Flash standby state, minimum power consumption
active	Normal mode

Enter the low-power process:

- Configure FBAC register, and the number of cycles during which an active state can be maintained.
- Configure the FBFALLBACK register, and the state to be entered by the Flash bank.
- Stop accessing Flash (stop any read/write/erase).
- When the duration of not accessing Flash reaches the number of cycles configured in the FBAC register, Flash will automatically enter the low-power state configured in FBFALLBACK.
- After entering the low-power state, any access to Flash will wake it up.

18.8.5 Flash read operation

Relationship between system clock frequency and Flash read time

In order to accurately read Flash instructions and data, RWAIT in the FRDCNTL register must be configured according to the frequency of the system clock SYSCLK to meet the requirements of read wait delay of Flash. The zero wait read supports a maximum

frequency of Flash is 36MHz.

Table 51 Read Time of System Frequency

System frequency	RWAIT
0<f≤36MHz	0
36MHz<f≤72MHz	1
72MHz<f≤108MHz	2
108MHz<f≤144MHz	3
144MHz<f≤180MHz	4
180MHz<f≤200MHz	5

Note:

(1) To increase the system frequency, first configure RWAIT to the target frequency range and then increase the system frequency.

(2) To reduce the system frequency, first configure the system frequency to lower the clock, and then configure RWAIT to the target frequency range.

CBUS access interface

Because there is already cache inside the CPU, CBUS access is only accelerated by prefetching on the Flash controller interface. There are a total of two buffers for alternate ping-pong prefetching, with each buffer having a bit width of 128 bits (256 bits for single-bank mode).

18.8.6 Flash power-on initialization process

The power-on initialization process is completed by hardware modules, including the following steps:

(1) Power-on reset release, stable clock

(2) Flash starts loading its own trim to put Flash in a reliable state

(3) Flash starts loading DCS linkpointer, searching for the start address of DCS configuration

(4) Flash starts loading DCS configuration

(5) Flash starts loading user option bytes

(6) Flash starts loading efuse and chip system configuration

(7) After all configurations are loaded, release the system reset

18.9 Register bank address

Table 52 Register Bank Address

Device register	Register bank	Start address	End address
NVMCRegs	NVMC_REGS	0x 4001_7C00	0x 4001_83FF

18.10 Register address mapping

Table 53 Register Address Mapping

Register name	Register description	Offset address	WRPRT
FRDCNTL	Flash read control register	0x00	√
BANK_SWAP_SOFT	Flash Bank Swap Register	0x04	√
FBAC	Flash Bank access control register	0x3C	√
FBFALLBACK	Flash Bank fallback power register	0x40	√
FBPRDY	Flash Bank pump ready register	0x44	√
FMSTAT	Flash module state register	0x54	√
FRD_INTF_CTRL	Flash read interface control register	0x300	√
CR	Control Register	0x478	√
ECC_ENABLE	ECC_ENABLE register	0x600	√
SINGLE_ERR_ADDR_LOW_0	Bank0 single error address low register	0x604	√
SINGLE_ERR_ADDR_HIGH_0	Bank0 single error address high register	0x608	√
UNC_ERR_ADDR_LOW_0	Bank0 uncorrectable error address low register	0x60C	√
UNC_ERR_ADDR_HIGH_0	Bank0 uncorrectable error address high register	0x610	√
ERR_STATUS	Error status register	0x614	√
ERR_POS	Error location register	0x618	√
ERR_STATUS_CLR	Error status clear register	0x61C	√
ERR_CNT	Error counter register	0x620	√
ERR_THRESHOLD	Error threshold register	0x624	√
ERR_INTFLG	Error interrupt flag register	0x628	√
ERR_INTCLR	Clear error interrupt register	0x62C	√
FDATAH_TEST	Data high test register	0x630	√
FDATAL_TEST	Data low test register	0x634	√

Register name	Register description	Offset address	WRPRT
FADDR_TEST	ECC test address register	0x638	√
FECC_TEST	ECC test ECC register	0x63C	√
FECC_CTRL	ECC control register	0x640	√
FOUH_TEST	Test data output high register	0x644	√
FOUL_TEST	Test data output low register	0x648	√
FECC_STATUS	ECC status register	0x64C	√
SINGLE_ERR_ADDR_LOW_1	Bank1 single error address low register	0x650	√
SINGLE_ERR_ADDR_HIGH_1	Bank1 single error address high register	0x654	√
UNC_ERR_ADDR_LOW_1	Bank1 uncorrectable error address low register	0x658	√
UNC_ERR_ADDR_HIGH_1	Bank1 uncorrectable error address high register	0x65C	√

18.11 Register functional description

18.11.1 Flash read control register (FRDCNTL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	Reserved			0h
15:8	RWAIT	R/W	Flash read wait period It can be set to any value between 0 and 0xF. For flash access, the data is returned at RWAIT+1 SYSCLK cycles. 0: 0<f≤36MHz 1: 36MHz<f≤72MHz 2: 72MHz<f≤108MHz 3: 108MHz<f≤144MHz 4: 144MHz<f≤180MHz 5: 180MHz<f≤200MHz	Fh
31:16	Reserved			0h

18.11.2 Flash bank swap register (BANK_SWAP_SOFT)

Offset address: 0x04

Reset types: Power-on, WDT, XRSn

Note:

（1） This register is writable when DCS Z1 and Z2 are both unlocked.

（2） When DCS Z1/Z2 are not both unlocked, this register can be written to if the currently active configuration is dual-bank mode (DBANK = 1'b1) and the DCS OTP security configurations for FLASH BANK 0 and 1 are identical.

Field	Name	R/W	Description	Reset Value
0	SWAP_ENN	R/W	Disable BANK SWAP (BANK SWAP disable) 0: Enable the BANK SWAP function; the specific mapping method is determined by the SWAP setting 1: Disables the BANK SWAP function; default address mapping applies	1h
1	SWAP	R/W	BANK SWAP configuration bit 0: BANK0/1 uses the default address mapping BANK0 main address is 0x0800_0000–0x0801_FFFF The main address of BANK1 is 0x0802_0000 to 0x0803_FFFF 1: BANK0 and BANK1 address mappings are swapped The main address range for BANK0 is 0x0802_0000 to 0x0803_FFFF The main address range for BANK1 is 0x0800_0000 to 0x0801_FFFF	1h
2	SWAP_SEL	R/W	BANK SWAP function selection 0: The BANK SWAP function is controlled by the value loaded into the option byte 1: The BANK SWAP function is controlled by register software	1h
3	SWAP_STATUS	R	BANK SWAP status bit (BANK SWAP status) 0: BANK 0/1 currently has the default address mapping 1: The address mapping for BANK 0/1 is currently swapped	1h
31:4	Reserved			0h

18.11.3 Flash Bank access control register (FBAC)

Offset address: 0x3C

Reset type: SYSRSn

Field	**Name**	**R/W**	**Description**	**Reset value**
7:0	Reserved			Fh
15:8	BAGP	R/W	Bank Active Grace Period These bits contain the starting count value of the BAGP countdown counter. Any access to a certain bank will cause its BAGP counter to reload to the bank's BAGP value. After the last access to the flash storage area, the countdown counter will delay by 0-255 prescale SYSCLK clock cycles, and then set this storage area to one of the fallback power modes determined by the FBFALLBACK register. When the fallback mode is not in Active state, this value must be greater than 1.	0h

Field	Name	R/W	Description	Reset value
			Note: The prescale clock used for the BAGP down counter is the 16 division clock input to SYSCLK.	
31:16	Reserved			0h

18.11.4 Flash Bank fallback power register (FBFALLBACK)

Offset address: 0x40

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	BNKPWR0	R/W	Fallback power mode 0 00: Standby 01: Standby 10: Reserved 11: Active	0h
3:2	BNKPWR1	R/W	Fallback power mode 1 00: Standby 01: Standby 10: Reserved 11: Active	0h
31:4	Reserved			0h

18.11.5 Flash Bank pump ready register (FBPRDY)

Offset address: 0x44

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	BANK0RDY	R	Bank 0 Ready Flag Allow software to confirm that Bank 0 is ready for Flash access before attempting to access. It needs to attempt to access before the pump and Bank are ready. 0: Bank0 not ready 1: Bank0 is in active power mode and ready to access.	0h
1	BANK1RDY	R	Bank 1 Ready Flag Allow software to confirm whether Bank 1 is ready for Flash access before attempting to access. It needs to attempt to access before the pump and Bank are ready. 0: Bank1 not ready 1: Bank1 is in an active status and ready to access	0h
14:2	Reserved			0h

Field	Name	R/W	Description	Reset value
15	PUMPRDY	R	Pump Ready Flag Allow software to confirm whether the pump is ready for Flash access before attempting to access. If the pump is not ready to access, a waiting state will be ascertained until the pump is ready. 0: The pump is not ready 1: The pump is ready and in active state	0h
31:16	Reserved			0h

18.11.6 Flash state register (FMSTAT)

Offset address: 0x54

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	PSUSP	R	When this bit is set, it indicates that the Flash module has received and processed a program suspension operation. This bit remains set before issuing a program recovery instruction or running the Clear_More instruction.	0h
2	ESUSP	R	When this bit is set, it indicates that the Flash module has received and processed a program erase suspension operation. This bit remains set before issuing an erase recovery instruction or running the Clear_More instruction.	0h
3	Reserved			0h
4	CSTST	R	Command Status. Once FSM stFACCs is wrong, CSTST will be set. When CSTST is set, this bit will notify the host that the programming, erasing, or verifying sector instruction has failed, and the instruction has been stopped. Clear this bit through the clear status command. For certain errors, as they do not fall into other error bit categories, this bit will serve as the unique indication of FSM errors.	0h
5	INVDAT	R	Invalid Data. When this bit is set, it indicates that the user is attempting to write a “1” to an existing “0”.	0h
6	PGM	R	Program Active.	0h

Field	Name	R/W	Description	Reset value
			When this bit is set, it indicates that the Flash module is currently performing programming operation. This bit is set to 1 when programming stFACCs, and cleared to 0 when programming is completed. When programming pauses, this bit will be cleared; when programming is restored, this bit will be set.	
7	ERS	R	Erase Active. When this bit is set, it indicates that the Flash module is currently performing erase operation. This bit is set to 1 when erasing stFACCs, and cleared to 0 when erasing is completed. When erasing pauses, this bit will be cleared; when erasing is restored, this bit will be set.	0h
8	BUSY	R	When this bit is set, it indicates that programming, erasing, or suspension operations are being performed.	0h
9	Reserved			0h
10	EV	R	Erase Verify. When this bit is set, it indicates that the sector is not successfully erased after the maximum allowed number of erase pulses is given for the erase operation.	0h
11	Reserved			0h
12	PGV	R	Program verify. When this bit is set, it indicates that the field is not successfully programmed after the maximum allowed number of programming pulses is given for the programming operation.	0h
13	REG_NOT_WRITABLE	R	Limit NVMC register write 0: NVMC register can write 1: NVMC register cannot write	0h
31:14	Reserved			0h

18.11.7 Flash read interface control register (FRD_INTF_CTRL)

Offset address: 0x300

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	Reserved			0h

Field	Name	R/W	Description	Reset value
10	Bus_prften	R/W	busmatrix interface prefetch enable 0: Disable 1: Enable	0h
11	Bus_buf_clr	R/W	busmatrix interface buffer clear 0: No effect 1: Clear	0h
31:12	Reserved			0h

18.11.8 CR Control Register (CR)

Offset address: 0x478

Reset types: Power-on, XRSn, WDT and NMIWDT

Fields	Name	R/W	Description	Reset value
0	DBANK	R	After a reset, this bit is loaded from the option byte. Changing this bit will result in the loss of data in the Flash main memory. 0: Single-bank mode: 256-bit read word width 1: Dual-bank mode: 128-bit read word width	X
31:1	Reserved			0h

Note: The Flash option byte area and the OTP area always remain in dual-bank mode, regardless of the DBANK bit.

18.11.9 ECC_ENABLE register (ECC_ENABLE)

Offset address: 0x600

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	DECODE ENABLE	R/W	ECC decode enable. The value of 0xA will enable hardware ECC decoding, and any other value will disable ECC	Ah
7:4	ENCODE ENABLE	R/W	ECC encode enable. The value of 0xA will enable hardware ECC encoding, and any other value will disable ECC	Ah

Note: Control the ECC enable to be turned off for all Flash areas except for DCSM LINKPOINTER OTP.

18.11.10 Bank0 single error address register (SINGLE_ERR_ADDR_LOW_0)

Offset address: 0x604

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ERR_ADDR_L_0	R/W	A 64-bit aligned address with a single bit error occurs in the lower 64 bits of the 128-bit aligned Flash bank0 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

(3) This register stores the address used for ECC calculations. Regardless of whether the BANK SWAP function is enabled, the ECC verification logic for FLASH Bank 0 calculates ECC using the fixed base address 0x0800_0000, whilst FLASH Bank 1 calculates ECC using the fixed base address 0x0802_0000.

18.11.11 Bank0 single error address register (SINGLE_ERR_ADDR_HIGH_0)

Offset address: 0x608

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ERR_ADDR_H_0	R/W	A 64-bit aligned address with a single bit error occurs in the higher 64 bits of the 128-bit aligned Flash bank0 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

(3) This register stores the address used for ECC calculations. Regardless of whether the BANK SWAP function is enabled, the ECC verification logic for FLASH Bank 0 calculates ECC using the fixed base address 0x0800_0000, whilst FLASH Bank 1 calculates ECC using the fixed base address 0x0802_0000.

18.11.12 Bank0 uncorrectable error address low register (UNC_ERR_ADDR_LOW_0)

Offset address: 0x60C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	UNC_ADDR_L_0	R/W	A 64-bit aligned address with uncorrectable error occurs in the lower 64 bits of the 128-bit aligned Flash bank0 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

(3) This register stores the address used for ECC calculations. Regardless of whether the BANK SWAP function is enabled, the ECC verification logic for FLASH Bank 0 calculates ECC using the fixed base address 0x0800_0000, whilst FLASH Bank 1 calculates ECC using the fixed base address 0x0802_0000.

18.11.13 Bank0 uncorrectable error address high register (UNC_ERR_ADDR_HIGH_0)

Offset address: 0x610

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	UNC_ADDR_H_0	R/W	A 64-bit aligned address with uncorrectable error occurs in the higher 64 bits of the 128-bit aligned Flash bank0 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

(3) This register stores the address used for ECC calculations. Regardless of whether the BANK SWAP function is enabled, the ECC verification logic for FLASH Bank 0 calculates ECC using the fixed base address 0x0800_0000, whilst FLASH Bank 1 calculates ECC using the fixed base address 0x0802_0000.

18.11.14 Error status Register (ERR_STATUS)

Offset address: 0x614

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Flash0_FAIL_0_L	R	Fail on 0. 0: No single bit error of Fail on 0 occurs in the lower 64 bits of the Flash bank0 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the lower 64 bits of the 128-bit aligned address, and the corrected value is 0. It can be cleared to zero by writing 1 to the Flash0_FAIL_0_L_CLR bit of ERR_STATUS_CLR register.	0h
1	Flash0_FAIL_1_L	R	Fail on 1. 0: No single bit error of Fail on 1 occurs in the lower 64 bits of Flash bank0 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the lower 64 bits of the 128-bit aligned address, and the corrected value is 1. It can be cleared to zero by writing 1 to the Flash0_FAIL_1_L_CLR bit of ERR_STATUS_CLR register.	0h
2	Flash0_UNC_ERR_L	R	Uncorrectable error. 1: The value of 1 indicates that an uncorrectable error occurs in the lower 64 bits of Flash bank0 of a 128-bit aligned address. It can be cleared to zero by writing 1 to the Flash0_UNC_ERR_L_CLR bit of ERR_STATUS_CLR register.	0h
7:3	Reserved			0h
8	Flash0_FAIL_0_H	R	Fail on 0. 0: No single bit error of Fail on 0 occurs in the higher 64 bits of Flash bank0 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the higher 64 bits of a 128-bit aligned address, and the corrected value is 0. It can be cleared to zero by writing 1 to the Flash0_FAIL_0_H_CLR bit of ERR_STATUS_CLR register.	0h
9	Flash0_FAIL_1_H	R	Fail on 1. 0: No single bit error of Fail on 1 occurs in the higher 64 bits of the Flash bank0 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the higher 64 bits of the 128-bit aligned address, and the corrected value is 1. It can be cleared to zero by writing 1 to the Flash0_FAIL_1_H_CLR bit of ERR_STATUS_CLR register.	0h

Field	Name	R/W	Description	Reset value
10	Flash0_UNC_ERR_H	R	Uncorrectable error. 1: The value of 1 indicates that an uncorrectable error occurs in the higher 64 bits of the Flash bank0 of a 128-bit aligned address. It can be cleared to zero by writing 1 to the Flash0_UNC_ERR_H_CLR bit of ERR_STATUS_CLR register.	0h
15:11	Reserved			0h
16	Flash1_FAIL_0_L	R	Fail on 0. 0: No single bit error of Fail on 0 occurs in the lower 64 bits of the Flash bank1 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the lower 64 bits of the 128-bit aligned address, and the corrected value is 0. It can be cleared to zero by writing 1 to the Flash1_FAIL_0_L_CLR bit of ERR_STATUS_CLR register.	0h
17	Flash1_FAIL_1_L	R	Fail on 1. 0: No single bit error of Fail on 1 occurs in the lower 64 bits of the Flash bank1 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the lower 64 bits of the 128-bit aligned address, and the corrected value is 1. It can be cleared to zero by writing 1 to the Flash1_FAIL_1_L_CLR bit of ERR_STATUS_CLR register.	0h
18	Flash1_UNC_ERR_L	R	Uncorrectable error. 1: The value of 1 indicates that an uncorrectable error occurs in the lower 64 bits of the Flash bank1 of a 128-bit aligned address. It can be cleared to zero by writing 1 to the Flash1_UNC_ERR_L_CLR bit of ERR_STATUS_CLR register.	0h
23:19	Reserved			0h
24	Flash1_FAIL_0_H	R	Fail on 0. 0: No single bit error of Fail on 0 occurs in the higher 64 bits of the Flash bank1 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the higher 64 bits of a 128-bit aligned address, and the corrected value is 0. It can be cleared to zero by writing 1 to the Flash1_FAIL_0_H_CLR bit of ERR_STATUS_CLR register.	0h

Field	Name	R/W	Description	Reset value
25	Flash1_FAIL_1_H	R	Fail on 1. 0: No single bit error of Fail on 1 occurs in the higher 64 bits of the Flash bank1 with a 128-bit aligned address. 1: The value of 1 indicates that a single bit error occurs in the higher 64 bits of the 128-bit aligned address, and the corrected value is 1. It can be cleared to zero by writing 1 to the Flash1_FAIL_1_H_CLR bit of ERR_STATUS_CLR register.	0h
26	Flash1_UNC_ERR_H	R	Uncorrectable error. 1: The value of 1 indicates that an uncorrectable error occurs in the higher 64 bits of Flash bank1 of a 128-bit aligned address. It can be cleared to zero by writing 1 to the Flash1_UNC_ERR_H_CLR bit of ERR_STATUS_CLR register.	0h
31:27	Reserved			0h

18.11.15 Error address register (ERR_POS)

Offset address: 0x618

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
5:0	Flash0_ERR_POS_L	R/W	Error position. The bit of the single bit error in Flash bank0 is located in the lower 64 bits of the 128-bit aligned address. Explain the position based on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the range of error position can be from 0 to 7; otherwise it can be from 0 to 63.	0h
6	Reserved			0h
7	Flash0_ERR_TYPE_L	R/W	Error type. 0: A single bit error occurs in the lower 64 data bits of the Flash bank0 with a 128-bit aligned address. 1: A single bit error occurs in the lower 64-bit ECC bit of the Flash bank0 with a 128-bit aligned address.	0h
13:8	Flash0_ERR_POS_H	R/W	Error position. The bit of the single bit error in Flash bank0 is located in the higher 64 bits of the 128-bit aligned address. Explain the position based on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the range of error position can be from 0 to 7; otherwise it can be from 0 to 63.	0h
14	Reserved			0h

Field	Name	R/W	Description	Reset value
15	Flash0_ERR_TYPE_H	R/W	Error type. 0: A single bit error occurs in the higher 64 data bits of the Flash bank0 with a 128-bit aligned address. 1: A single bit error occurs in the higher 64-bit ECC bit of the Flash bank0 with a 128-bit aligned address.	0h
21:16	Flash1_ERR_POS_L	R/W	Error position. The bit of the single bit error in Flash bank1 is located in the lower 64 bits of the 128-bit aligned address. Explain the position based on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the range of error position can be from 0 to 7; otherwise it can be from 0 to 63.	0h
22	Reserved			0h
23	Flash1_ERR_TYPE_L	R/W	Error type. 0: A single bit error occurs in the lower 64 data bits of the Flash bank1 with a 128-bit aligned address. 1: A single bit error occurs in the lower 64-bit ECC bit of the Flash bank1 with a 128-bit aligned address.	0h
29:24	Flash1_ERR_POS_H	R/W	Error position. The bit of the single bit error in Flash bank1 is located in the lower 64 bits of the 128-bit aligned address. Explain the position based on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the range of error position can be from 0 to 7; otherwise it can be from 0 to 63.	0h
30	Reserved			0h
31	Flash1_ERR_TYPE_H	R/W	Error type. 0: A single bit error occurs in the higher 64 data bits of the Flash bank1 with a 128-bit aligned address. 1: A single bit error occurs in the higher 64-bit ECC bit of the Flash bank1 with a 128-bit aligned address.	0h

18.11.16 Error status clear register (ERR_STATUS_CLR)

Offset address: 0x61C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Flash0_FAIL_0_L_CLR	R-0/W1S	Clear Fail on 0 Writing 1 to this bit will clear the Flash0_FAIL_0_L bit of the ERR_STATUS register.	0h
1	Flash0_FAIL_1_L_CLR	R-0/W1S	Clear Fail on 1 Writing 1 to this bit will clear the Flash0_FAIL_1_L bit of the ERR_STATUS register.	0h

Field	Name	R/W	Description	Reset value
2	Flash0_UNC_ERR_L_CLR	R-0/W1S	Clear the uncorrectable error. Writing 1 to this bit will clear the Flash0_UCN_ERR_L bit of the ERR_STATUS register.	0h
7:3	Reserved			
8	Flash0_FAIL_0_H_CLR	R-0/W1S	Clear Fail on 0 Writing 1 to this bit will clear the Flash0_FAIL_0_H bit of the ERR_STATUS register.	0h
9	Flash0_FAIL_1_H_CLR	R-0/W1S	Clear Fail on 1 Writing 1 to this bit will clear the Flash0_FAIL_1_H bit of the ERR_STATUS register.	0h
10	Flash0_UNC_ERR_H_CLR	R-0/W1S	Clear the uncorrectable error. Writing 1 to this bit will clear the Flash0_UCN_ERR_H bit of the ERR_STATUS register.	0h
15:11	Reserved			
16	Flash1_FAIL_0_L_CLR	R-0/W1S	Clear Fail on 0 Writing 1 to this bit will clear the Flash1_FAIL_0_L bit of the ERR_STATUS register.	0h
17	Flash1_FAIL_1_L_CLR	R-0/W1S	Clear Fail on 1 Writing 1 to this bit will clear the Flash1_FAIL_1_L bit of the ERR_STATUS register.	0h
18	Flash1_UNC_ERR_L_CLR	R-0/W1S	Clear the uncorrectable error. Writing 1 to this bit will clear the Flash1_UCN_ERR_L bit of the ERR_STATUS register.	0h
23:19	Reserved			
24	Flash1_FAIL_0_H_CLR	R-0/W1S	Clear Fail on 0 Writing 1 to this bit will clear the Flash1_FAIL_0_H bit of the ERR_STATUS register.	0h
25	Flash1_FAIL_1_H_CLR	R-0/W1S	Clear Fail on 1 Writing 1 to this bit will clear the Flash1_FAIL_1_H bit of the ERR_STATUS register.	0h
26	Flash1_UNC_ERR_H_CLR	R-0/W1S	Clear the uncorrectable error. Writing 1 to this bit will clear the Flash1_UCN_ERR_H bit of the ERR_STATUS register.	0h

Field	Name	R/W	Description	Reset value
31:27	Reserved			0h

18.11.17 Error counter register (ERR_CNT)

Offset address: 0x620

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	ERR_CNT	R/W	Single bit error count ECC errors occur every time a bit increases. When the threshold is reached, the counter will stop counting the single bit errors. No matter whether the threshold is reached, the SINGLE_ERR_INTCLR bit can clear the ERR_CNT bit. Suitable for ECC logic test mode and normal operating mode.	0h
31:16	Reserved			0h

18.11.18 Error threshold register (ERR_THREHOLD)

Offset address: 0x624

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	ERR_THRESHOLD	R/W	Single bit error threshold When the ERR_CNT value is equal to the THRESHOLD value and a single bit error occurs, the SINGLE_ERR_INT flag will be set, and an interrupt will be triggered.	0h
31:16	Reserved			0h

18.11.19 Error interrupt flag register (ERR_INTFLG)

Offset address: 0x628

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SINGLE_ERR_INTFLG	R	Single bit error interrupt flag When the ERR_CNT value is equal to the ERR_THRESHOLD value and a single bit error occurs, the SINGLE_ERR_INT flag will be set, and a SINGLE_ERR_INT interrupt will be triggered. The SINGLE_ERR_INTCLR bit can clear this bit.	0h
1	UNC_ERR_INTFLG	R	Uncorrectable error interrupt flag When an uncorrectable error occurs, this bit will be set, and a UNC_ERR_INT interrupt will be triggered. The UNC_ERR_INTCLR bit can clear this bit.	0h
31:2	Reserved			0h

18.11.20 Clear error interrupt register (ERR_INTCLR)

Offset address: 0x62C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SINGLE_ERR_INTCLR	R/W	Single bit error interrupt flag clear This bit can be set to 1 to clear SINGLE_ERR_INTFLG. Writing 0 has no effect.	0h
1	UNC_ERR_INTCLR	R/W	Uncorrectable error interrupt flag clear This bit can be set to 1 to clear UNC_ERR_INTFLG. Writing 0 has no effect.	0h
31:2	Reserved			0h

18.11.21 Data high test register (FDATAH_TEST)

Offset address: 0x630

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	FDATAH	R/W	Selected Data High The 63:32 bit user of the selected data block in ECC test mode can configure it.	0h

18.11.22 Data low test register (FDATAL_TEST)

Offset address: 0x634

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	FDATAL	R/W	Selected Data Low The 31:0 bit user of the selected data block in ECC test mode can configure it.	0h

18.11.23 ECC test address register (FADDR_TEST)

Offset address: 0x638

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	Reserved			0h
15:3	ADDRL	R/W	Address for selected 64-bit data The selected data user in ECC test mode can configure the address bit. The three least significant bits of the address are ignored and the bit 15:3 is written to the remaining address bits of the field.	0h
21:16	ADDRH	R/W	Address for selected 64-bit data The selected data user in ECC test mode can configure the address bit. The three least significant bits of the address are ignored and the bit 21:16 is written to the remaining address bits of the field.	0h
31:22	Reserved			0h

18.11.24 ECC test ECC register (FECC_TEST)

Offset address: 0x63C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	ECC	R/W	Selected data ECC In ECC test mode, users can configure the ECC bit of the selected 64-bit data block.	0h
31:8	Reserved			0h

18.11.25 ECC control register (FECC_CTRL)

Offset address: 0x640

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ECC_TEST_EN	R/W	ECC test mode enable 0: Disable 1: Enable	0h
2:1	ECC_SELECT	R/W	ECC block select 2'b00 selects the bank data of ECC block on Flash bank0 bit [63:0]. 2'b01 selects the bank data of ECC block on Flash bank0 bit [127:64]. 2'b10 selects the bank data of ECC block on Flash bank1 bit [63:0]. 2'b11 selects the bank data of ECC block on Flash bank1 bit [127:64].	0h
3	DO_ECC_CALC	R/W	Enable ECC calculation When ECC test logic is enabled by setting ECC_TEST_EN, the ECC logic will perform ECC calculation for the data and addresses written to the ECC test register within one cycle.	0h
31:4	Reserved			0h

18.11.26 Test data output high register (FOUTH_TEST)

Offset address: 0x644

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	DATAOUTH	R	Test data High. Save the data of the ECC block selected for the 63:32 bit.	0h

18.11.27 Test data output low register (FOUTL_TEST)

Offset address: 0x648

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	DATAOUTL	R	Test data Low Save the data of the ECC block selected for the 31:0 bit.	0h

18.11.28 ECC status register (FECC_STATUS)

Offset address: 0x64C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SINGLE_ERR	R	Test mode ECC single bit error Flag 0: No flag 1: Test mode ECC single bit error flag	0h
1	UNC_ERR	R	Test mode Uncorrectable Error Flag 0: No flag 1: Test mode uncorrectable error high flag	0h
7:2	DATA_ERR_POS	R	Test mode single bit error position Save the position where a single bit error occurs. It depends on whether the CHK_ERR bit is a check bit or a data bit. If CHK_ERR is a check bit error, the value range for the error position is 0~7 or 0~63.	0h
8	ERR_TYPE	R	Test mode single bit error type 0: Data bit (if SINGLE_ERR bit is set) 1: Check bit	0h
31:9	Reserved			0h

18.11.29 Bank1 single error address low register (SINGLE_ERR_ADDR_LOW_1)

Offset address: 0x650

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ERR_ADDR_L_1	R/W	64-bit aligned address at which a single bit error occurred in the lower 64-bits of a 128-bit aligned Flash bank1 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

18.11.30 Bank1 single error address high register (SINGLE_ERR_ADDR_HIGH_1)

Offset address: 0x654

Reset type: 0x0000 0000

Field	Name	R/W	Description	Reset value
31:0	ERR_ADDR_H_1	R/W	A 64-bit aligned address with a single bit error occurs in the higher 64 bits of the 128-bit aligned Flash bank1 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

18.11.31 Bank1 uncorrectable error address low register (UNC_ERR_ADDR_LOW_1)

Offset address: 0x658

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	UNC_ADDR_L_1	R/W	A 64-bit aligned address with uncorrectable error occurs in the lower 64 bits of the 128-bit aligned Flash bank1 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

18.11.32 Bank1 uncorrectable error address high register (UNC_ERR_ADDR_HIGH_0)1

Offset address: 0x65C

Reset type: SYSRSn

Bits	Name	R/W	Description	Reset value
31:0	UNC_ADDR_H_1	R/W	A 64-bit aligned address with uncorrectable error occurs in the higher 64 bits of the 128-bit aligned Flash bank1 memory.	0h

Note:

(1) For detailed information on FLASH address blocking, please refer to the table in the ‘18.8.1’ section;

(2) In the Bank 0 and Bank 1 mixed regions, 128-bit aligned addresses ending in 0x0, 0x20, 0x40, etc. belong to Bank 0, whilst 128-bit aligned addresses ending in 0x10, 0x30, 0x50, etc. belong to Bank 1.

19 Timer 0/1/2 (TMR0/1/2)

19.1 Full Name and Abbreviation Description of Terms

Table 54 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Timer	TMR

19.2 Introduction

Arm® Cortex®-M52 integrates three timers, namely TMR0/1/2. If TMR2 is not used in the real-time operating system (RTOS), then all three timers can be used in the user application; otherwise, only TMR0 and TMR1 can be used in the application.

19.3 Main characteristics

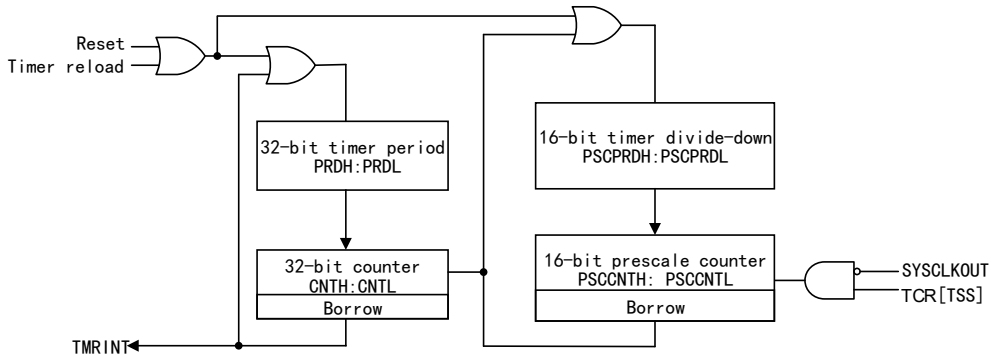
(1) Timer interrupt signal
- TMRINT0, TMRINT1, TMRINT2

(2) It can set a 32-bit timer period and store the current count value

(3) The 16-bit timer can prescale the clock signal and store the prescale count value

19.4 Structure block diagram

Figure 15 Timer Structure Block Diagram



19.5 Functional description

19.5.1 Operation steps of timers

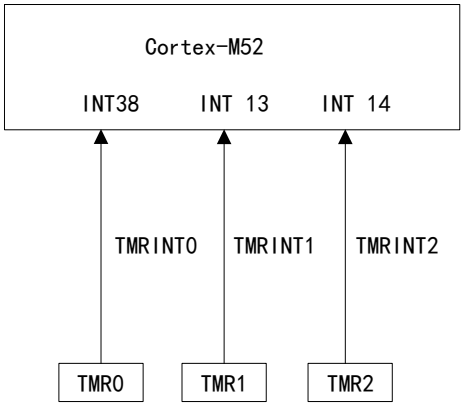
(1) Use the 32-bit counter register TIM[MSW]:TIM [LSW] to load it as the value in the periodic register PRD[MSW]: PRD[LSW]

(2) During each of the (TPR [TDDRH:TDDR]+1) SYSCLK cycles, the counter decreases once, where TDDRH: TDDR is the divider of the timer

(3) When the counter reaches 0, it will trigger the timer interrupt output signal and generate an interrupt pulse

19.5.2 Timer interrupt signal and output signal

Figure 16 Timer Interrupt Signal and Output Signal



Note:

(1) The timer register is connected to the AHB bus.

(2) The timer is synchronized with SYSCLKOUT.

19.6 Register bank address

Table 55 Register Bank Address

Device register	Register bank	Start address	End address
Tmr0Regs	TMR0_REGS	0x4001_7000	0x4001_73FF
Tmr1Regs	TMR1_REGS	0x4001_7400	0x4001_77FF
Tmr2Regs	TMR2_REGS	0x4001_7800	0x4001_7BFF

19.7 Register address mapping

Note: In TMRx, x=0…2

Table 56 TMR Offset Address

Name	Register description	Offset address	WRPRT
TIM	Counter register	0x00	-
PRD	Period register	0x04	-
TCR	Control register	0x08	-
TPR	Prescaler low-bit register	0x0C	-
TPRH	Prescaler high-bit register	0x0E	-

19.8 Register functional description

19.8.1 Counter register (TIM)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	LSW	R/W	Counter Value Low This field stores the lower 16 bits of the current 32-bit count of the counter. When TIM[MSW]: TIM[LSW] decreases to zero, TMRINT signal will be generated, and the cycle value of PRD[MSW]:PRD[LSW] will be reloaded. The value of TIM[MSW]: TIM[LSW] can be determined by setting the prescaler value TDDRH:TDDR, and during each of (TDDRH:TDDR + 1) clock cycles, the value of TIM[MSW]:TIM[LSW] will decrease by 1.	FFFFh
31:16	MSW	R/W	Counter Value High This field stores the higher 16 bits of the current 32-bit count of the counter. When TIM[MSW]: TIM[LSW] decreases to zero, TMRINT signal will be generated, and the cycle value of PRD[MSW]:PRD[LSW] will be reloaded. The value of TIM[MSW]: TIM[LSW] can be determined by setting the prescaler value TDDRH:TDDR, and during each of (TDDRH:TDDR + 1) clock cycles, the value of TIM[MSW]:TIM[LSW] will decrease by 1.	0h

19.8.2 Periodic register (PRD)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	LSW	R/W	Period Value Low This field stores the lower 16 bits of a 32-bit period. At the beginning of the next timer input clock cycle (i.e. the output of the prescaler), if TIM[MSW]:TIM [LSW] decreases to zero, TIM [MSW]:TIM[LSW] will be reloaded as the cycle value contained in PRD[MSW]:PRD[LSW]. In addition, by setting the TRB bit in the TCR register, the contents of PRD[MSW]:PRD[LSW] can also be loaded into TIM[MSW]:TIM[LSW].	FFFFh

Field	Name	R/W	Description	Reset value
31:16	MSW	R/W	Period Value High This field stores the higher 16 bits of a 32-bit period. At the beginning of the next timer input clock cycle (i.e. the output of the prescaler), if TIM[MSW]:TIM [LSW] decreases to zero, TIM [MSW]:TIM[LSW] will be reloaded as the cycle value contained in PRD[MSW]:PRD[LSW]. In addition, by setting the TRB bit in the TCR register, the contents of PRD[MSW]:PRD[LSW] can also be loaded into TIM[MSW]:TIM[LSW].	1h

19.8.3 Control register (TCR)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	Reserved			1h
4	TSS	R/W	Timer Stop Enable This flag bit indicates the start or stop status of the timer. 0: Indicates that the timer is running 1: Indicates that the timer has stopped	0h
5	TRB	R/W	Timer Reload 0: This bit is always read as zero, and is ignored when writing 0 1: When 1 is written to this bit, the value of PRD[MSW]:PRD[LSW] can be loaded into TIM[MSW]:TIM[LSW], and PSCH:PSC will be loaded as the value of TDDRH:TDDR	0h
9:6	Reserved			0h
10	SOFT	R/W	Counter Stop Time Select 0: Stop after decreasing by a few cycles; the specific number of cycles is not fixed. From the perspective of the assembly program, the cycle is related to the program where the program breakpoint is located; from the perspective of the C program, the cycle is related to the program on the breakpoint. (Valid only when FREE=0; if FREE=1, this bit is invalid) 1: Soft stop; when TIM[MSW]:TIM [LSW] decreases to 0, it will stop, and the timer will generate an interrupt before it is turned off (valid only when FREE=0; if FREE=1, this bit is invalid)	0h
11	FREE	R/W	Counter Run Mode Select 0: Hard stop; it will stop when TIM[MSW]:TIM[LSW] decreases next time (SOFT bit controls the simulation behavior) 1: Free running (SOFT bit is invalid)	0h
13:12	Reserved			0h

Field	Name	R/W	Description	Reset value
14	TIE	R/W	Timer Interrupt Enable 0: Disable the timer interrupt 1: Enable the timer interrupt. At this point, if the timer decrements to zero, the timer will assert its interrupt request	0h
15	TIF	R/W1C	Timer Overflow Flag This bit indicates that a timer overflow interrupt has occurred. If a timer overflow has occurred since TIF was last cleared, this bit is set.TIF will not be automatically cleared and there is no need to clear TIF to enable the next timer interrupt 0: Ignore when the timer does not decrease to zero, and 0 is written to it 1: When the timer decreases to zero, this flag bit will be set to 1. Writing 1 to this bit can clear the flag bit.	0h

19.8.4 Prescaler low bit register (TPR)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TDDR	R/W	Timer Prescaler Period Low During each of (TDDRH: TDDR+1) timer clock source cycles, TIM[MSW]:TIM[LSW] will decrease by one. When reset, The TDDRH: TDDR bit is cleared to zero. By setting the TDDRH:TDDR bit (subtract 1 from the required multiple and write the result to the TDDRH: TDDR bit), the counting speed of the timer can be adjusted. When the PSCH:PSC value is 0, the next timer clock source cycle will trigger the content of TDDRH:TDDR to be reloaded into PSCH: PSC, and TIM[MSW]:TIM[LSW] will decrease by one. In addition, if the software sets the TRB bit, TDDRH:TDDR will also be reloaded to PSCH: PSC.	0h
15:8	PSC	R	Timer Prescaler Counter Low This field maintains the current prescaler count of the timer. In each timer clock source cycle, as long as the PSCH:PSC value is greater than 0, the PSCH:PSC will decrease by one. When PSCH:PSC decreases to 0, within the next timer clock cycle (output of the timer prescaler), PSCH:PSC will load the content of TDDRH:TDDR, and the timer counter register (TIM[MSW]:TIM[LSW]) will decrease by one. When the software sets the TRB bit, PSCH:PSC will also be reloaded. The PSCH:PSC value cannot be directly set, but it can be checked by reading the register. This value must be obtained from TDDRH:TDDR. When reset, PSCH:PSC is set to 0.	0h

19.8.5 Prescaler high bit register (TPRH)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TDDRH	R/W	Timer Prescaler Period High Please refer to TPR Register Description	0h
15:8	PSCH	R	Timer Prescaler Counter High Please refer to TPR Register Description	0h

20 Watchdog timer (WDT)

20.1 Full Name and Abbreviation Description of Terms

Table 57 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Watchdog Timer	WDT

20.2 Introduction

The watchdog is a timer intended for systems. It consists of 8-bit counters, and is used to monitor system failures caused by software errors in order to ensure normal operation of the system.

20.3 Main characteristics

(1) Two prescalers:
- Prescaler 1: The division ratio ranges from /1 to /64 (/1 by default)
- Prescaler 2: The division ratio ranges from /2 to /4096 (/512 by default)

(2) 4 debugging modes
- Suspension mode
- Real-time single-step mode
- Free run mode
- Real-time free run mode

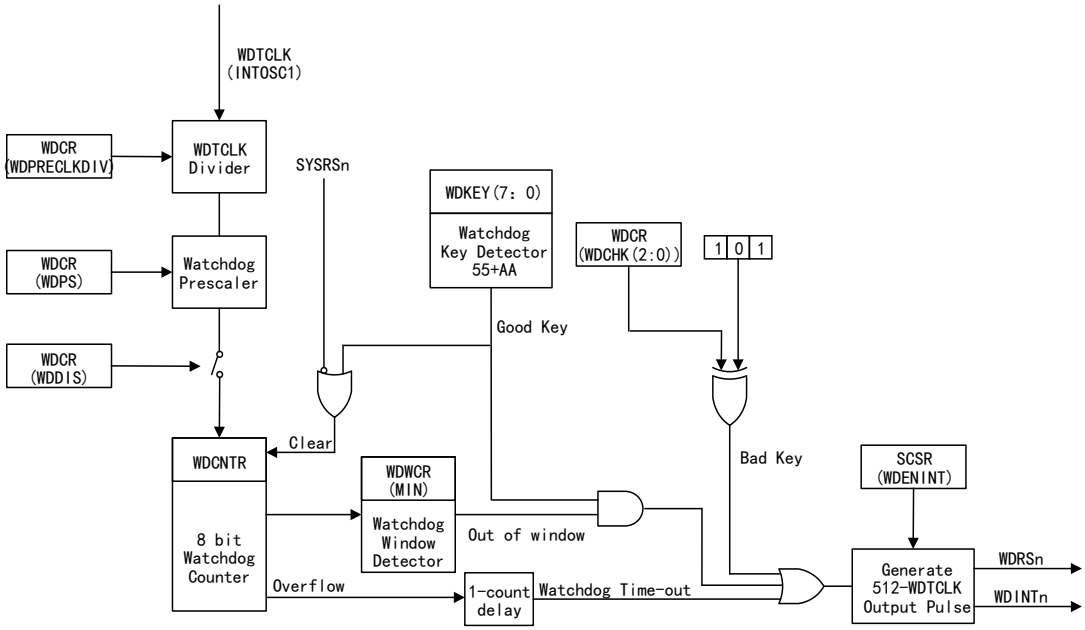
(3) Can be configured as interrupt or reset mode

(4) The low-power mode is divided into IDLE mode and HALT mode

(5) Minimum window value can be set

20.4 Structure block diagram

Figure 17 Watchdog Timer Structure Block Diagram



20.5 Functional description

20.5.1 Counter principle

The clock source of the watchdog counter is connected to the prescale clock WDTCLK of INTOSC1, and its frequency is divided by prescaler 1 and prescaler 2. The division ratio range of prescaler 1 is from /1 to /64, and the division value is a power of 2; the division ratio range of prescaler 2 is from /2 to /4096, the division value is a power of 2, and its default value is set to /512 (backward compatible).

When the watchdog counter reaches the maximum value, it will generate a pulse (with a width of 512 WDTCLK) to trigger an interrupt or reset. To prevent the watchdog timer from misjudging that the system has stopped and triggered a reset, the CPU needs to periodically write the 0x55 and 0xAA sequences to the watchdog key register to reset the watchdog counter. If the watchdog function is not used, it can be disabled

20.5.2 Debug mode

- Suspend mode: In this mode, the watchdog clock is suspended
- Real-time single-step mode: In this mode, the watchdog clock is suspended (even in real-time interrupt)
- Free run mode: In this mode, the watchdog will continue to run normally
- Real-time free run mode: In this mode, the watchdog module will run normally

20.5.3 Configure the watchdog interrupt/reset mode

In the SCSR register, the watchdog can be configured to assert a watchdog interrupt (WDINTn) or set a watchdog reset (WDRSn) when the watchdog counter reaches the maximum value.

20.5.3.1 Interrupt mode

When the watchdog counter finishes counting, it will perform the following operations:

- Pull down WDINTn signal: This operation asserts an interrupt request, and this low-level state will last for 512 OSCCLK cycles;
- Trigger WAKE interrupt: The falling edge of WDINTn will trigger this interrupt, but this trigger only occurs when the interrupt has been enabled.

Re-enabling WAKEINT when WDINTn is active does not cause the interrupt to be triggered multiple times.

Reading the SCSR[WDINTS] bit can determine the current state of WDINTn. When WDINTn is active, the software configuration of the watchdog should not be changed. The following consequences may be caused:

- If the interrupt mode is changed to reset mode, the device will reset immediately;
- If the watchdog is changed from disabled state to enabled state, repeated interrupt will result;
- If debugging reset is performed, the reason why the RESC register displays reset is watchdog reset.

20.5.3.2 Reset mode

If the watchdog is set to reset the device, when the watchdog counter reaches the maximum value, the WDRSn signal will reset the device within 512 OSCCLK cycles (by pulling down the device reset pin XRSn).

20.5.4 Watchdog operation in low-power mode

20.5.4.1 IDLE

When the watchdog interrupt (WDINTn) is triggered, the signal will remain low (for 512 OSCCLK cycles). Before entering IDLE mode, if the WDINTn signal returns to a high level, it can transmit an interrupt signal to the CPU to make it exit IDLE mode. In IDLE mode, the watchdog interrupt has the same capability as other peripherals, and it can trigger the WAKE interrupt in the NVIC register, and the peripheral that trigger the interrupt can be determined by software.

Reading the SCSR[WDINTS] bit can determine the current state of WDINTn, and the WDINTS bit will be updated within two SYSCLKOUT cycles after the WDINTn signal changes.

20.5.4.2 HALT

If the CLKSRCCTL1[WDHALTI] bit is set to 1, the internal oscillator and watchdog timer will remain active. Watchdog reset can wake up the system from HALT mode, but watchdog interrupt cannot wake up the system.

20.5.5 Minimum window check

The watchdog has an optional "window" function to enhance the timeout mechanism, which requires a minimum delay between counter resets. This function can help prevent the error of bypassing normal program flow (except for watchdog processing).

Write the required minimum watchdog count into the WDWCR register to set the minimum window value. This value will take effect after the next WDKEY sequence. Thereafter, when WDCNTR is not less than WDWCR, the watchdog can be served normally; when WDCNTR is less than WDWCR, any attempt to serve the watchdog will trigger a watchdog interrupt or reset. The window function is disabled during resetting (the minimum window value is zero).

20.5.6 Maintain the watchdog timer

Before the WDCNTR overflows, if 0x55 and 0xAA are continuously written, the WDCNTR can be reset:

- When writing 0x55 to the WDKEY register, enable reset
- When the next written value is 0xAA, WDCNTR will be reset

The operation will only be triggered if 0x55 and 0xAA are written to WDKEY. WDCNTR will reset only when 0x55 is written to the WDKEY register and then 0xAA is written. The following table shows watchdog key sequence examples.

Table 58 Watchdog Key Sequence Examples

Step	Value written to WDKEY	Result
1	0xAA	No response
2	0xaa	Write a value other than 0x55 and 0xAA to WDKEY No response
3	0x55	Enable reset (if the next value is 0xAA, WDCNTR will be reset)
4	0x55	Enable reset (if the next value is 0xAA, WDCNTR will be reset)
5	0xAA	Reset WDCNTR
6	0x20	Write a value other than 0x55 and 0xAA to WDKEY No response
7	0xAA	No response (the previous value is not 0x55)
8	0x55	Enable reset (if the next value is 0xAA, WDCNTR will be reset)
9	0xAA	Reset WDCNTR

Step	Value written to WDKEY	Result
10	0x55	Enable reset (if the next value is 0xAA, WDCNTR will be reset)
11	0xaa	Reset WDCNTR
12	0x55	Enable reset (if the next value is 0xAA, WDCNTR will be reset)
13	0xAA	Reset WDCNTR

When the watchdog is configured to reset the device, if the WDCR register overflows or writes values other than 1, 0, and 1 to the WDCR[WDCHK] bit, it will trigger the device reset and set the RESC[WDRSn] bit. After reset, the program can read the status of the bit to determine If a reset caused by the watchdog has occurred, and then clear the bit so as to detect subsequent watchdog resets. Even if the WDRSn flag bit is set, it will not prevent the watchdog from resetting.

20.6 Register bank address

Table 59 Register Bank Address

Device register	Register bank	Start address	End address
WdtRegs	WDT_REGS	0x4001_0000	0x4001_00BF

20.7 Register address mapping

Table 60 WDT address mapping

Register name	Register description	Offset address	WRPRT
SCSR	Control state register	0x44	√
WDCNTR	Counter register	0x46	√
WDKEY	Counter reset register	0x4A	√
WDCR	Control register	0x52	√
WDWCR	Window register	0x54	√

20.8 Register functional description

20.8.1 Control status register (SCSR)

Offset address: 0x44

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	WDOVERRIDE	R/W1S	Lock Disable Watchdog When writing 1 to this bit, it will be cleared and the WDDIS bit in the WDCR register will be locked. This bit	1h

Field	Name	R/W	Description	Reset value
			will remain in this state until the system is reset next time. Reading this bit will return its current value, and writing 0 is invalid.	
1	WDENINT	R/W	Watchdog Interrupt reset configure 0: The counter expires, and triggers reset (default state after power on and any system reset) 1: The counter expires, and triggers interrupt	0h
2	WDINTS	R-0	Watchdog Interrupt Status This bit is synchronized with SYSCLK. To disable and restart the watchdog, this bit needs to be at a high level; at this time, the watchdog interrupt can enter a low-power mode to wake up the system. 0: The watchdog interrupt is active 1: The watchdog interrupt is inactive	1h
15:3	Reserved			0h

20.8.2 Counter register (WDCNTR)

Offset address: 0x46

Reset type: IORSn

Field	Name	R/W	Description	Reset value
7:0	WDCNTR	R	Watchdog Counter This bit contains the current value of the watchdog counter. This counter increments in each WDTCLK (INTOSC1) cycle. If the correct value is written to the WDKEY register, the counter will be reset to 0; if the counter overflows, an interrupt or reset will be generated (based on the value of the WDENINT bit in the SCSR register).	0h
15:8	Reserved			0h

20.8.3 Counter reset register (WDKEY)

Offset address: 0x4A

Reset type: IORSn

Field	Name	R/W	Description	Reset value
7:0	WDKEY	R/W	Watchdog Counter Reset To prevent the watchdog counter from overflowing, it needs to be reset (write 0x55 and then write 0xAA; writing other values is invalid). Reading this register will return the value of the WDCR register.	0h
15:8	Reserved			0h

20.8.4 Control register (WDCR)

Offset address: 0x52

Reset type: IORSn

Field	Name	R/W	Description	Reset value
2:0	WDPS	R/W	Watchdog Clock prescaler1 This field determines WDTCLK. The frequency of WDTCLK is given by the following formula: PSCCLK = INTOSC1 / WDPRECLKDIV WDTCLK = PSCCLK / WDPS The length of the watchdog reset or interrupt pulse is 810 INTOSC1 cycles, so the counter cycle must be longer than it (the product of prescaler 1 and prescaler 2 must be greater than or equal to four). 0: WDTCLK = PSCCLK / 1(default) 1: WDTCLK = PSCCLK / 1 2: WDTCLK = PSCCLK / 2 3: WDTCLK = PSCCLK / 4 4: WDTCLK = PSCCLK / 8 5: WDTCLK = PSCCLK / 16 6: WDTCLK = PSCCLK / 32 7: WDTCLK = PSCCLK / 64	0h
5:3	WDCHK	R-0/W	Watchdog Check When performing a write operation on this register, the user must write 1, 0, 1. If the watchdog is enabled, writing any other value will immediately reset the core.	0h
6	WDDIS	R/W	Watchdog Disable This bit can be locked by the WDOVERRIDE bit in the SCSR register, and reset will enable the watchdog.	0h
7	Reserved			0h
11:8	WDPRECLKDIV	R/W	Watchdog Clock prescaler2 This field determines INTOSC1. The frequency of WDTCLK is determined by the following formula: PSCCLK = INTOSC1 / WDPRECLKDIV WDTCLK = PSCCLK / WDPS The length of the watchdog reset or interrupt pulse is 810 INTOSC1 cycles, so the counter cycle must be longer than it (the product of prescaler 1 and prescaler 2 must be greater than or equal to four). The default value for prescaler 2 is 512. 0: PSCCLK = INTOSC1 / 512 1: PSCCLK = INTOSC1 / 1024 2: PSCCLK = INTOSC1 / 2048 3: PSCCLK = INTOSC1 / 4096 4-7: Reserved 8: PSCCLK = INTOSC1 / 2 9: PSCCLK = INTOSC1 / 4 A: PSCCLK = INTOSC1 / 8 B: PSCCLK = INTOSC1 / 16 C: PSCCLK = INTOSC1 / 32 D: PSCCLK = INTOSC1 / 64	0h

Field	Name	R/W	Description	Reset value
			E: PSCCLK = INTOSC1 / 128 F: PSCCLK = INTOSC1 / 256	
15:12	Reserved			0h

20.8.5 Window Register (WDWCR)

Offset address: 0x54

Reset type: IORSn

Field	Name	R/W	Description	Reset value
7:0	MIN	R/W	Watchdog Window Threshold The counter is set through the WDKEY register (before the counter reaches the value in the register), and the watchdog will immediately trigger a reset or interrupt.	0h
8	FIRSTKEY	R	First valid WDKEY detected The first valid WDKEY is detected after MIN was configured to a non-zero value. 0: Not detected 1: Detected Note: (1) If MIN=0, this bit will not be set (2) If MIN changes back to 0x0 from a non-zero value, this bit will be automatically cleared (3) This position is only used for debugging purpose	0h
15:9	Reserved			0h

21 NMIWDT

21.1 Introduction

An NMI (Non-Maskable Interrupt) watchdog is a hardware module specifically designed to monitor system errors and abnormal conditions. It helps applications respond promptly to system-level errors by triggering a non-maskable interrupt, thereby ensuring system stability and reliability.

21.2 Key Features

(1) The NMI watchdog is capable of detecting anomalies and errors within the system and triggering an NMI in a timely manner. When an NMI occurs, a status bit is set in the NMIFLG register to indicate that the interrupt has been triggered.

(2) The NMI watchdog is equipped with a counter driven by the SYSCLK clock. If the counter value reaches the set threshold (NMIWDPRD register), the NMI watchdog reset (NMIWDRS) is triggered.

(3) The NMI handler must clear the flag (via the NMIFLGCLR register) to prevent unintended resets and ensure that the system can respond to future NMIs.

(4) In the event of a clock loss error, a Flash Uncorrectable ECC Error or a RAM Uncorrectable ECC Error, an NMI interrupt is triggered; when any bit in the NMI Shadow Flag Register (NMISHDFLG) is set, the ERRORSTS signal changes from high to low. This state persists until the CPU clears these flags or a power-on reset occurs (the NMIFLGCLR register can clear the flag bits).ERRORSTS can be routed to GPIO24, GPIO28 or GPIO29.

(5) Hardware errors that trigger an NMI on the R502 include clock loss, FLB NMI, Flash Uncorrectable ECC errors, RAM Uncorrectable ECC errors and Software-Forced Error (self-test mode of the NMI subsystem).

21.3 Function Description

Please refer to the section on resets.

21.4 Register Function Description

Please refer to the System Control section.

22 Dual-clock comparator (DCCOMP)

22.1 Introduction

The dual-clock comparator module is based on another more accurate and reliable clock, and is used to evaluate and monitor clock inputs. This instrument is used to check for faults in clock sources or clock structures, so as to enhance the system security.

22.2 Main characteristics

(1) Support point measurement in single sequence mode

(2) Allow selection of clock source for each counter

(3) Support continuous monitoring without the need for application intervention

(4) Support programmable tolerance window (defined by the number of reference clock cycles)

(5) Allow applications to ensure that the frequency of two clock signals remains in a fixed ratio

22.3 Structure block diagram

Figure 18 DCCOMP and System Connection

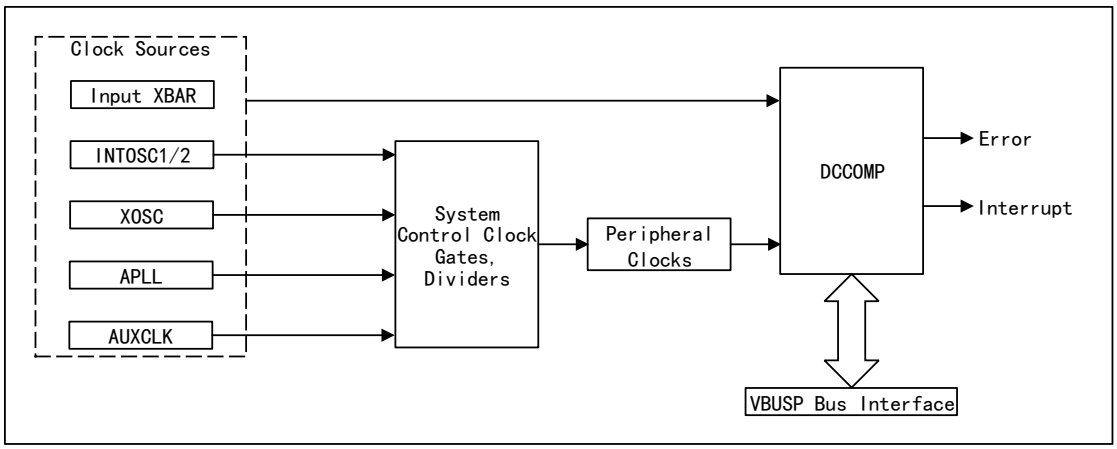
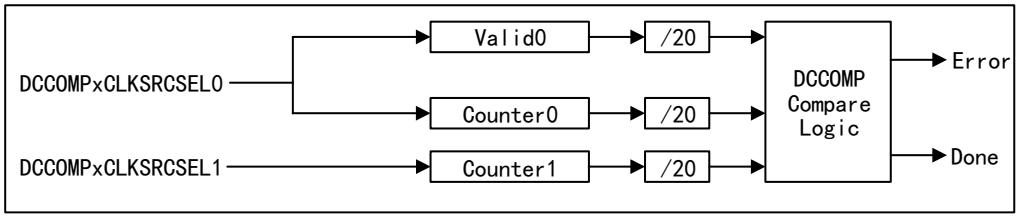


Figure 19 DCCOMP Structure Block Diagram



22.4 Functional description

The DCCOMP module includes three counters, namely Counter0, Counter1, and Valid0. Preloaded values for all counter loads need to be defined first.. When DCCOMP is enabled, Counter0 and Counter1 begin to decrease, and their rates are determined by the frequency of Clock0 and Clock1, respectively. If Counter0 equals 0, that's when the time goes out, the Valid0 counter decreases at the rate of Clock0. When Counter1 decreases to 0 within the valid window, no error will occur, and Clock1 is considered to work well within the user-configured tolerance.

22.4.1 DCCOMP counter

Counter0 and Counter1 are configured based on the ratio between Clock0 and Clock1, i.e. F_{CLK0} * Counter1 = F_{CLK1} * Counter0. Valid0 counter is used to provide tolerance and configure based on the error of DCCOMP.

Error sources needing to be considered when configuring counters

Because Clock0 and Clock1 are asynchronous, the start and stop of all counters will not occur synchronously. So when configuring counters, the following error sources must be considered:

(1) The digitalization error is equal to 8 Clock0 cycles

(2) Because Clock0 and Clock1 are asynchronous, DCCOMP error is caused (depending on the frequency of Clock0 and Clock1)

- When F_{CLK1} is unknown, asynchronous error= 2*[max (F_{SYSCLK}/F_{CLK0})]+2, in Clock0 cycles
- When F_{CLK1} is greater than F_{CLK0}, asynchronous error= 2*(F_{SYSCLK}/F_{CLK0})+2, in Clock0 cycles
- When F_{CLK1} is less than F_{CLK0}, asynchronous error= 2*(F_{CLK0}/F_{CLK1})+2*[max(F_{SYSCLK}/F_{CLK0})], in Clock0 cycles

Calculation of total error of DCCOMP

(1) Calculation formula for DCCOMP Error

- DCCOMP error=asynchronous error + digitalization error, in Clock0 cycles

The frequency error under clock measurement is called DCCOMP error. It's caused by DCCOMP, it does not represent the frequency error of clock measurement. When configuring the counter, you need to pay attention to this error and choose the appropriate tolerance, that is, the DCCOMP error determines the measurement window.

(2) Calculation formula for the window

- Window=DCCOMP error/(tolerance*0.01), in Clock0 cycles.

Assuming that the DCCOMP error is 10 and the expected tolerance is ±0.1%, the window is equal to 10,000 Clock0 cycles.

Based on the window calculation formula, it can be seen that the lower the expected tolerance, the larger the counter value, then the larger the measurement window, that is, the counter value when the tolerance is 0.1% is greater than the counter value when the tolerance is 0.2%. Therefore, the measurement window is determined by the tolerance required for the specific application, defined by the Clock0 period.

(3) Calculation formula for allowable frequency error

The measurement clock may have a frequency error range. When it is predicted that this error exists, this error should also be taken into account when configuring the counter. An external crystal oscillator is used as a reference clock when measuring INTOSC1/2's frequency, the allowable frequency tolerance of INTOSC1/2 should be considered in the counter configuration. The formula is as follows:

- Allowable frequency error = (frequency tolerance/100)*window, where the allowable frequency error is in Clock0 cycle and the frequency tolerance is in %.

(4) Calculation formula for total error

- Total error = DCCOMP error + allowable frequency error

Configuration of counter values

- CNTSEED0 = Window - total error
- CNTSEED1 = (F_{CLK1} / F_{CLK0}) * Window
- VALSEED0 = Total Error * 2

The maximum count value of the 20-bit counter Counter1 cannot exceed 1048575. When Counter1 exceeds this value, the target tolerance of DCCOMP error will increase, thereby reducing the measurement window. Therefore, the formula for calculating the minimum tolerance is as follows:

- Tolerance=[(F_{CLK1} / F_{CLK0}) * DCCOMP error * 100]/1048575

22.4.2 One-shot test mode

The DCCOMP module can perform a countdown by enabling the single- shot mode. In this mode, DCCOMP will stop working after Counter0 and Valid0 reach 0

To prevent further counting, DCCOMP will be automatically disabled at the end of the countdown sequence in this mode. This mode is usually used for sampling inspection of signal frequency.

Use case: Verify the frequency of PLLRAWCLK

This case uses XTAL as the reference clock to verify the actual usage of PLL output clock frequency.

Assumption conditions:

- XTAL is 10MHz
- PLL output frequency is 100MHz
- SYSCLK is 100MHz
- The allowable frequency tolerance is 0.1%
- DCCOMP tolerance is 0.1%

The measurement sequence is as follows:

(1) Set the Clock0 source of Counter0 and Valid0 as XTAL, and set the Clock1 source of Counter1 as PLL output clock.

(2) According to the formula in Calculation of DCCOMP Counter, the seed value of the counter is calculated as CNTSEED0 = 29940, CNTSEED1 = 300000, and VALSEED0 = 120

(3) Once DCCOMP is enabled, Counter0/1 will count down from the seed value.

(4) After Counter0 reaches 0, the Valid0 counter will be automatically triggered.

(5) When Valid0 reaches 0 and Counter1 is not 0, ERRFLG will be set and DCCOMP error will be sent to NVIC. At this point, Counter1 stops counting (freezes). Users can enable an interrupt, so that NVIC will generate an interrupt whenever this DCCOMP error occurs.

(6) In order to make the DCCOMP module ready for the next sampling measurement, the application needs to clear the ERRFLG bit and restart this module

When no errors occur at the end of the sequence, SIGMDONEFLG will be set and an interrupt will be generated. The application must clear the SIGMDONEFLG flag bit before restarting DCCOMP.

Error state

When any of the following situations occurs, an error state will be generated:

- Counter 1 counts down to 0 before Counter 0 reaches 0. Indicates that Clock1 is faster than expected, or Clock0 is slower than expected.
- Counter1 has not yet reached 0 after both Counter0 and Valid0 have reached 0. Indicates that Clock1 is slower than expected.

Both errors pause the count. The application can read the value of the counter at this time to help determine the cause of the error.

22.4.3 Interrupt

DCCOMP will generate an interrupt in any of the following events:

(1) When DONEEN=1, DCCOMP completes counting and all counters are within the given window (indicating completion of signal operation), an interrupt will be generated.

(2) When ERREN=1, the counter is not time out within the given window, and a DCCOMP count error occurs (indicating an error event), an interrupt will be generated.

The interrupts caused by completion signal events and error events are subjected to OR operation by phases and marked as DCCOMP interrupts. In order to determine whether the interrupt is caused by a completion signal event or an error event, the interrupt service program of the application needs to check the status flag in the DCCOMP_FLG register.

22.5 Register bank address

Table 61 Register Bank Address

Device register	Register bank	Start address	End address
Dcomp0Regs	DCCOMP_REGS	0x4000_0000	0x4000_03FF

22.6 Register address mapping

Table 62 DCCOMP_REGS address mapping s

Register name	Register description	Offset address	WRPRT
DCCOMP_CTRL	Control register	0x00	-
DCCOMP_REV	Version register	0x08	-
DCCOMP_CNTSEED0	Counter 0 seed value register	0x10	-
DCCOMP_VALSEED0	Valid seed value 0 register	0x18	-
DCCOMP_CNTSEED1	Counter 1 seed value register	0x20	-

Register name	Register description	Offset address	WRPRT
DCCOMP_FLG	Flag register	0x28	-
DCCOMP_CLKSRCCNT0	Counter 0 value register	0x30	-
DCCOMP_CLKSRCVAL0	Valid value 0 register	0x38	-
DCCOMP_CLKSRCCNT1	Counter 1 value register	0x40	-
DCCOMP_CLKSRCSEL1	Clock source 1 register	0x48	-
DCCOMP_CLKSRCSEL0	Clock source 0 register	0x50	-

22.7 Register functional description

22.7.1 Control register (DCCOMP_CTRL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	DCCOMPEN	R/W	DCCOMP Enable 0101: The counter stops Other: Other counters are running	5h
7:4	ERREN	R/W	Error Enable 0101: Disable error signals Other: Enable error signals	5h
11:8	SIGEN	R/W	Single Shot Mode Enable This bit is used to enable or disable repetitive operations of DCCOMP. 1010: Stop counting when both CLKSRCCNT0 and CLKSRCVAL0 reach 0 Others: Reserved	5h
15:12	DONEEN	R/W	DONE Enable This bit is used to enable or disable the completion interrupt signal, but the completion status in the DCCOMP_FLG register is not affected by this bit 0101: Disable completion signals Other: Enable completion signals	5h
31:16	Reserved			0h

22.7.2 Version register (DCCOMP_REV)

Offset address: 0x08

Reset type: SYSRSn

This register is used to specify the version of the module.

Field	Name	R/W	Description	Reset value
5:0	MINOR	R	Minor Revision Number These fields are used to determine minor revision to the module, e.g. enhancing the existing functions.	4h
7:6	Reserved			0h
10:8	MAJOR	R	Major Revision Number These fields are used to determine major revision to the module, e.g. changing or adding new functions.	2h
15:11	Reserved			0h
27:16	Reserved			0h
29:28	Reserved			0h
31:30	Reserved			1h

22.7.3 Counter 0 seed value register (DCCOMP_CNTSEED0)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
19:0	CNTSEED0	R/W	Counter 0 Seed Value This field is used to indicate the additional value of the counter seed of the clock source 0. If the value of this field is set to 0, it will result in undefined operation of DCCOMP.	0h
31:20	Reserved			0h

22.7.4 Valid seed value 0 register (DCCOMP_VALSEED0)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	VALSEED0	R/W	Valid Duration Counter 0 Seed Value This field is used to indicate the additional value of the timeout counter seed of the clock source 0. If the value of this field is set to 0, it will result in undefined operation of DCCOMP. The DCCOMP_CLKSRCVAL0 register defines the time window for counter 1 expiration, and the width of this window should be at least 4 cycles. Programming the values less than 4 into the DCCOMP_VALSEED0 register is disabled.	0h
31:16	Reserved			0h

22.7.5 Counter 1 seed value register (DCCOMP_CNTSEED1)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
19:0	CNTSEED1	R/W	Counter 1 Seed Value This field is used to indicate the additional value of the counter seed of the clock source 0. If the value of this field is set to 0, it will result in undefined operation of DCCOMP.	0h
31:20	Reserved			0h

22.7.6 Flag register (DCCOMP_FLG)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ERRFLG	R/W	Error Flag Clear this bit by writing 1. 0: No error occurs 1: An error occurs	0h
1	SIGMDONEFLG	R/W	Single-Shot Mode Done Flag This bit indicates when the single-shot mode is completed without any errors. Clear this bit by writing 1. 0: Single-shot mode not completed 1: Single-shot mode completed	0h
31:2	Reserved			0h

22.7.7 Counter 0 value register (DCCOMP_CLKSRCCNT0)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
19:0	CLKSRCCNT0	R	Clock Source 0 counter attached Value This field is used to indicate the additional value of the counter of the clock source 0.	0h
31:20	Reserved			0h

22.7.8 Valid value 0 register (DCCOMP_CLKSRCVAL0)

Offset address: 0x38

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CLKSRCVAL0	R	Clock Source 0 valid counter attached Value	0h

Field	Name	R/W	Description	Reset value
			This field is used to indicate the additional value of the valid counter of the clock source 0.	
31:16	Reserved			0h

22.7.9 Counter 1 value register (DCCOMP_CLKSRCCNT1)

Offset address: 0x40

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
19:0	CLKSRCCNT1	R	Clock Source 1 counter attached Value This field is used to indicate the additional value of the counter of the clock source 1.	0h
31:20	Reserved			0h

22.7.10 Clock source 1 register (DCCOMP_CLKSRCSEL1)

Offset address: 0x48

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	CLKSRCSEL1	R/W	Counter 1 Clock Source Select When the CLKSRCWEN field enables the clock source write function of counter 1, specify the clock source of counter 1. 0000: SYSPLLCLKOUT direct output 0001: Reserved 0010: INTOSC1 output clock 0011: INTOSC2 output clock 0100: Reserved 0101: Reserved 0110: System clock 0111: Reserved 1000: Reserved 1001: Reserved 1010: Auxiliary clock input 1011: System clock 1100: Bit clock of SPI and UART modules 1101: ADC conversion clock 1110: WDT frequency division clock 1111: CAN0 bit clock	2h
11:4	Reserved			0h
15:12	CLKSRCWEN	R/W	Counter 1 Clock Source Write Enable	5h

Field	Name	R/W	Description	Reset value
			1010: When a specific key is written, the CLKSRCSEL1 field will be updated to the new selection of clock counter 1 Other: Reserve the previous value, with no impact on new writes to register fields	
31:16	Reserved			0h

22.7.11 Clock source 0 register (DCCOMP_CLKSRCSEL0)

Offset address: 0x50

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	CLKSRCSEL0	R/W	Counter 0 Clock Source Select When the CLKSRCWEN field enables the clock source write function of counter 1, specify the clock source of counter 0. 0000: Crystal oscillator output 0001: INTOSC1 output clock 0010: INTOSC2 output clock 0100: TCK pin input 1000: Auxiliary clock input Others: Reserved	1h
31:4	Reserved			0h

23 General-Purpose Input/Output Pin (GPIO)

23.1 Full Name and Abbreviation Description of Terms

Table 63 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
General-purpose Input/Output	GPIO

23.2 Introduction

In GPIO modules, each pin can be separately configured as digital I/O (i.e. GPIO mode) or connected to peripheral signals. The GPIO port controls the digital and analog I/O multiplexing function of the pins, and through pin sharing, improves the flexibility of pin applications to the full extent. In addition, the input signal can eliminate unnecessary noise.

23.3 Main characteristics

(1) Configurable qualified type for input pins
- Synchronous input qualification
- Asynchronous input qualification
- 3 sampling input qualification
- 6 sampling input qualification

(2) Single-pin setting/clearing/switching

(3) The input path and output path are independent

(4) Support peripheral multiplexing function

(5) Support low-power mode wake-up

23.4 Structure block diagram

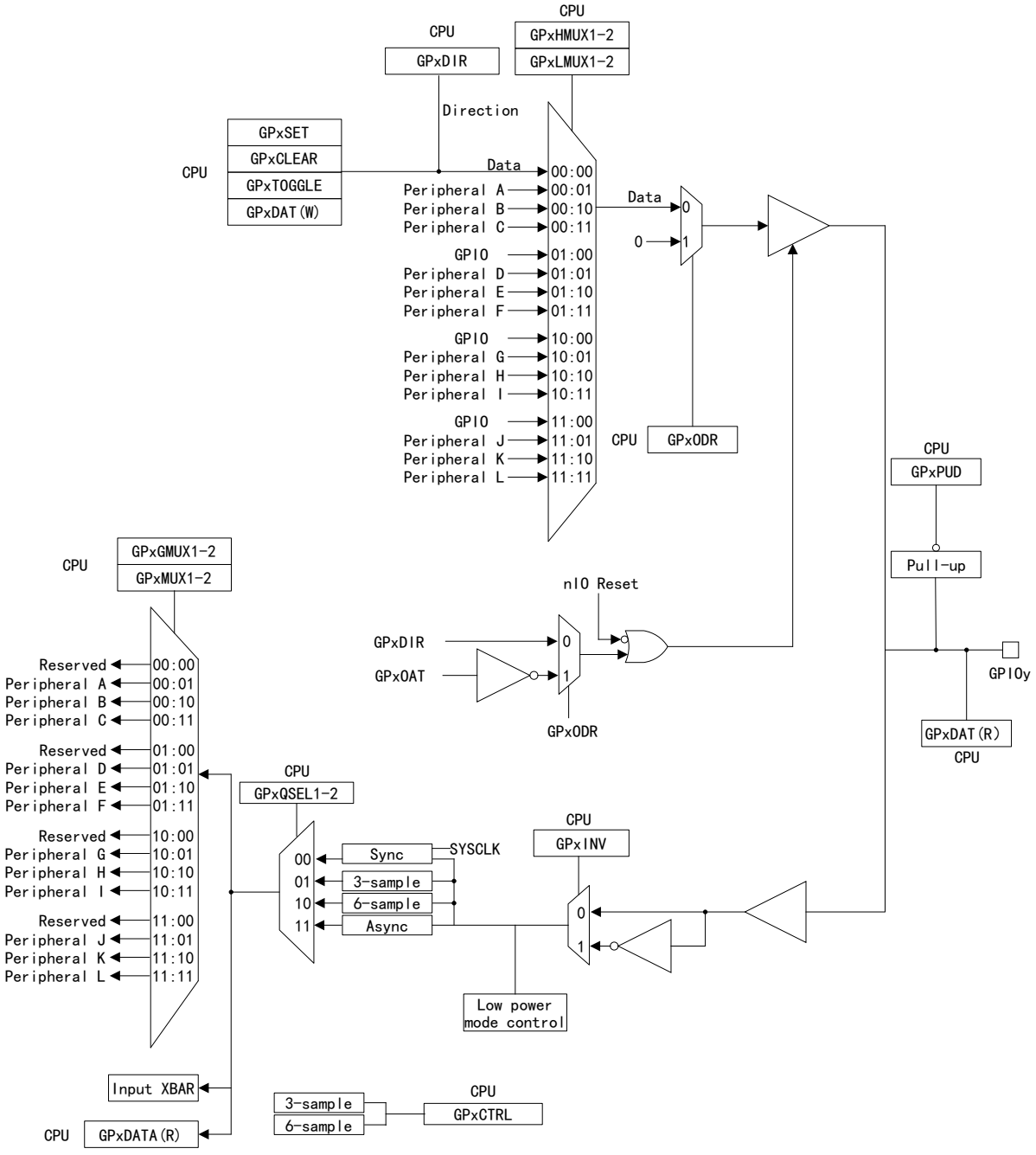
As shown in the structural diagram, GPIO ports have two characteristics:

(1) Perform peripheral multiplexing far away from the pins;

(2) The input path and output path are independent, and only connected at the pins.

Based on the above two characteristics, the process of CPU reading the physical state of pins is always independent of the CPU master controller and peripheral multiplexing. For example, for all hosts and peripherals, all pin options for input qualification and open-drain output are valid.

Figure 20 GPIO Structure Block Diagram



23.5 Functional description

23.5.1 Port introduction

Port composition:

(1) GPIOA: GPIO0-GPIO31

(2) GPIOB: GPIO32-GPIO59

(3) GPIOH: GPIO224-GPIO254

Port function

In the GPIOA and GPIOB ports, CPU or the peripheral can control the output of each pin. The input/output function of the pin can be controlled by the CPU, and a single enabled pin can also be used by peripheral signals through multiplexing function. There are up to 12 peripheral signals, and they are independent of each other.

Of these, GPIO28, GPIO20, GPIO21, GPIO13, GPIO12, GPIO224, GPIO226, GPIO227, GPIO228, GPIO230 and GPIO242 are AGPIO pins, which can be configured via registers as analogue pins, or as digital inputs or outputs.

AIO225, AIO231, AIO232, AIO233, AIO237, AIO238, AIO239, AIO241, AIO244 and AIO245 are AIO pins, which can be configured via registers as analogue pins or as digital input pins.(Analogue pins can be configured for use as digital input pins.)

23.5.2 Pin configuration

23.5.2.1 Configuration overview

The I/O pins are configured through the following steps:

(1) Select pins

According to the target application, select the pins for peripherals and each CPU input/output. Please refer to the Datasheet for details of the pin multiplexing function. All pins are general-purpose I/O by default and are not used as peripheral signals. Configure the selected pin function by writing to the corresponding bits in the GPxMUX1/2 and GPxGMUX1/2 registers.

Note: To avoid glitches during multiplexing, it is required to first set the corresponding bits to 0 in the GPxMUX1/2 register, and then change the value of the corresponding bits in the GPxGMUX1/2 register.

(2) Enable internal pull-up resistor (optional)

By default, the pull-up resistors of all pins are disabled. Write 0 or 1 to the corresponding bits of the pin in the pull-up disable register (GPxPUD) of port x so as to enable or disable the pull-up resistor of the pin. The pull-up resistor can be used to ensure that the state of the pin is known (high or low) when there is no external signal input pin.

(3) Select the input pin qualification

By default, all qualified synchronization and the sampling period is PLLSYSCLK. When GPIO serves as an input pin, the input qualification required for the pin (if there is a qualification type) needs to be configured. The input qualification consists of the

qualification sampling period and the qualification type. The sampling period of the pin is configured in the GPxCTRL register, and the input qualification type of the pin is configured in the GPxQSEL1/2 register. For more description on input qualification, please refer to Input Qualification.

(4) Configure pin direction

All GPIO pins are input pins by default. Specify the input or output direction for each pin configured as GPIO by writing 0 or 1 to the GPxDIR register. The steps to write to the GPxDIR register are as follows:

- Write values to the GPxSET/GPxCLR/GPxDAT registers
- Load the values from the above registers to the output latch (0 by default for all output latches);
- Write the value in the latch to the GPxDIR register;

(5) Select the low-power mode wake-up source

The low-power mode wake-up system can be implemented through GPIO0-59. Write 0 or 1 to the corresponding bit of the GPIOLPMSEL0/1 register in the CPU system register space, and one or more GPIO can be selected for use in wake-up.

(6) Configure external interrupts

Configuring external interrupts contains the following two steps:

- Enable interrupts and configure the polarity of interrupts through the TIM register.
- Select the sources for X-BAR signals 4, 5, 6, 13 and 14 respectively to configure the GPIO pins for XINT1–5 (corresponding to external interrupts EXTI4, EXTI5, EXTI6, EXTI7 and EXTI8).

For more information on the Input X-BAR architecture, please refer to X-BAR.

23.5.2.2 IO configuration after reset

After reset, the default GPIO is the input pin and the internal pull-up resistor is disabled. Therefore, each GPIO pin must be configured as an output mode, with the level driven by another component on the circuit board to the input mode above V_{ih} or below V_{il}, or with one of the three input modes of internal the pull-up resistor enabled, in order to prevent the input pins not driven from floating to the intermediate level and causing waste of penetrating current on the input buffer.

To prevent floating inputs, in the products with a few pins, the pull-up resistor not bound to GPIO is enabled by default. In practical use, do not disable the pull-up resistors of this part in the application code.

23.5.2.3 Input configuration

Input configuration refers to configuration of the input qualification mode of pins in the

input mode.

The input qualification mode of pins can be flexibly configured: configuring the input type of each pin in the register GPxQSEL1/2[GPIOn]. For input pins with different functions, different input types can be selected:

(1) GPIO input pins:
- Only synchronous with SYSCLKOUT ([GPIOn]=00)
- Qualification through sampling window ([GPIOn]=01/10)

(2) Peripheral input pins:
- Only synchronous with SYSCLKOUT ([GPIOn]=00)
- Qualification through sampling window ([GPIOn]=01/10)
- Asynchronous input ([GPIOn]=11)

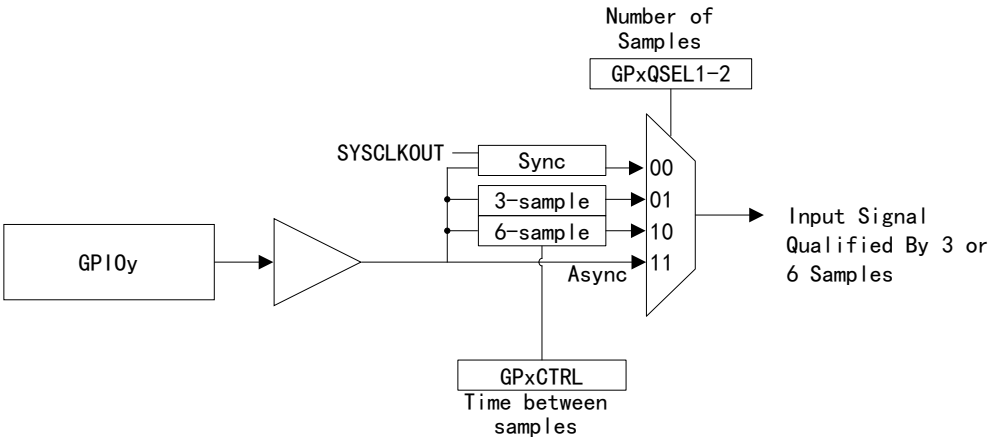
Synchronous input qualification (only synchronous with SYSCLKOUT)

That is, GPxQSEL1/2[GPIOn]=00; this qualification mode by default after all pins are reset. The input signal is only synchronous with the system clock (SYSCLKOUT), and the signals are not further qualified. As the input signal is asynchronous signal, changing the input of the MCU may require a delay time of up to one SYSCLKOUT.

Input qualification using the sampling window

That is, GPxQSEL1/2[GPIOn]=01/10, as shown in the following figure. The signal is first synchronized with the system clock (SYSCLKOUT), and then the input is qualified for a specified number of cycles before the input change is permitted. In this mode, two parameters, i.e. the sampling period (i.e. signal sampling frequency) and the number of samples, need to be configured for qualification.

Figure 21 Input Qualification Using the Sampling Window



As shown in Figure Input Qualification of Sampling Window and Figure Input

Qualification Clock Cycle, excess noise can be eliminated by performing input qualification.

(1) Sampling period (sampling interval)

The input signal is sampled at a fixed interval to ensure that it meets requirements. Users can specify the sampling period relative to the CPU clock (SYSCLKOUT), and determine the duration between sampling points (i.e. sampling frequency).

Configure the sampling period through GPxCTRL[QUALPRDn], as shown in the fields. The sample period is configured in eight input signal groups. For more information on sampling period configuration, please refer to GPxCTRL Register.

The relationship between the configuration of GPxCTRL[QUALPRDn] and the sampling period or sampling frequency is shown in the following table:

Table 64 Relationship between QUALPRDn and Sampling Period or Sampling Frequency

GPxCTRL[QUALPRDn]	Sampling period	Sampling frequency
0	$T_{SYSCLKOUT}$(1)*1	$f_{SYSCLKOUT}$(2)
Not 0	$T_{SYSCLKOUT}$*2*GPxCTRL[QUALPRDn]	$f_{SYSCLKOUT}$*1/(2*GPxCTRL[QUALPRDn])

Note

(1) $T_{SYSCLKOUT}$ is time period of SYSCLKOUT;

(2) $f_{SYSCLKOUT}$ is frequency of SYSCLKOUT.

According to the above relationship, when the frequency of SYSCLKOUT is determined, the minimum and maximum values of the sampling frequency can be calculated, as shown in the following example:

Table 65 Minimum/Maximum Sampling Frequency When $f_{SYSCLKOUT}$=60MHz

GPxCTRL[QUALPRDn]	Calculation formula of sampling frequency	Sampling frequency/period value
00	$f_{SYSCLKOUT}$=60MHz	60MHz/16.67ns
FF	$f_{SYSCLKOUT}$*1/(2*GPxCTRL[QUALPRDn]) =60MHz*1/(2*255)	117.647kHz/8.5μs

(2) Number of samples

According to GPxQSEL1/2[GPIOn] (x=A/B, n represents the pin number) field, the number of samples can be configured as 3 or 6.

If the input signals for three or six consecutive cycles are the same, the changes in the input signals will be transmitted to the MCU; otherwise, the changes will be ignored.

(3) Total sampling window width

The sampling window width refers to the time for sampling the input signal, and it is determined by both the sampling period and the number of samples. In order to enable the input qualifier to detect changes in the input signals, the level of the input signals must remain stable during the sampling window or a longer period of time. The total width of the sampling window is calculated as follows:

Table 66 Calculation of Total Sampling Window Width

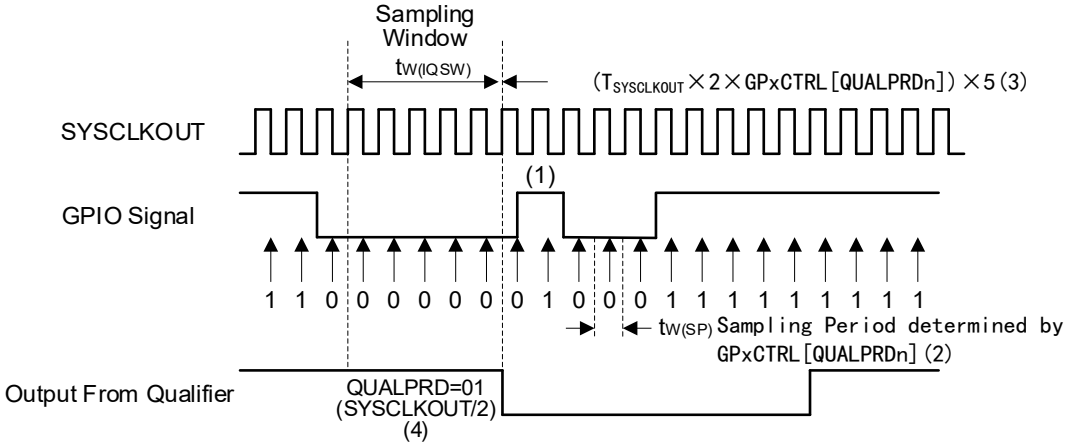
GPxCTRL[QUALPRDn]	Width of 3 sampling windows	Width of 6 sampling windows
0	$T_{SYSCLKOUT}$*2	$T_{SYSCLKOUT}$*5
Not 0	$T_{SYSCLKOUT}$*2*GPxCTRL[QUALPRDn]*2	$T_{SYSCLKOUT}$*2*GPxCTRL[QUALPRDn]*5

As shown in the table above, the number of sampling periods = the number of samples - 1. That is, for 3 sampling windows, the sampling window width is 2 sampling periods (t_w (IQSW)=2*t_w(SP)), while for 6 sampling windows, the sampling window width is 5 sampling periods (t_w(IQSW)=5*t_w(SP)).

Note: There is an asynchronous relationship between external signal changes and SYSCLKOUT and sampling period. Therefore, in order to ensure that external signal changes can be correctly recognized by the logic circuit, the input signal should remain stable for a certain period of time beyond the sampling window width. The time (i.e. extra time) beyond the sampling window can be up to $T_{SYSCLKOUT}$+t_w(SP). For more information on the duration of stable input signals, please refer to Datasheet.

An example of calculation for input qualification time period:

Figure 22 Input Qualification Clock Period



Note:

(1) Since this fault is shorter than the qualification window, the input qualifier will ignore this error. The sampling period is specified by the QUALPRDn field, with a value range of 0x00-0FF. When QUALPRDn=00, the sampling period is 1 $T_{SYSCLKOUT}$; when QUALPRDn≠00, the sampling period is $T_{SYSCLKOUT}$*2n (i.e. the time interval for sampling GPIO pins is $T_{SYSCLKOUT}$*2n).

(2) Choose which pin group consisting of 8 GPIO pins for this qualification period.

(3) Select the number of samples, and either 3 or 6 samples can be selected.

(4) In the example, in order to ensure that the input signal changes can be detected by the qualifier, the input signal should remain stable for more than 10 (i.e. QUALPRDn*5*2) TSYSCLKOUT cycles to ensure smooth detection of 5 sampling periods. Starting from asynchronous driving of external signals, the input pulse can ensure that signal changes can be recognized within up to 13 TSYSCLKOUT periods,

as shown in the above figure:

t_w(SP): Sampling period

t_w(IQSW): Input qualification sampling window

The sampling configuration is as follows:

GPxQSEL1/2[GPIOn]=10, GPxCTRL[QUALPRDn] =01

In this example, the calculation formulas for each parameter are as follows.

- Number of samples: 6,
- Sampling period: $t_w(SP)=T_{SYSCLKOUT}*2*GPxCTRL[QUALPRDn]=T_{SYSCLKOUT}*2$.
- Sampling window width: t_w (IQSW) = t_w (SP)*5= $T_{SYSCLKOUT}*2*GPxCTRL[QUALPRDn]*5= T_{SYSCLKOUT}*2*5$
- Assuming $T_{SYSCLKOUT}$=16.67ns, the parameter values are as follows:
- Sampling period: $t_w(SP)=T_{SYSCLKOUT}*2$=16.67ns*2=33.3ns.
- Sampling window width: $t_w(IQSW)=t_w(SP)*5$=33.3ns*5=166.5ns.

Due to the asynchronous nature of the input signal, the extra time required is:

- $T_{SYSCLKOUT}$+ one additional sampling period=$T_{SYSCLKOUT}+t_w(SP)$=16.67ns+33.3ns=50.0ns

Therefore, the total sampling time is at least:

- $t_w(IQSW)+T_{SYSCLKOUT}+t_w(SP)$=166.5ns+50.0ns=216.5ns

Asynchronous input qualification

GPxQSEL1/2[GPIOn]=11; this mode is used for peripherals that do not need to execute synchronous input and peripherals that perform synchronization themselves. These peripherals include communication ports McBSP, I2C, SPI, and UART. Besides, it is desirable for the function of PWM Trip Zone(TZn) signal to be independent of SYSCLKOUT.

When the pin is configured for GPIO function, this asynchronous mode is invalid. If the pin is configured as GPIO function and configured as asynchronous input qualification, the qualification of this pin is synchronized to SYSCLKOUT by default (for details, see Synchronous Input Qualification (only synchronized with SYSCLKOUT)).

Note: When the peripheral itself performs synchronization, configuring the pin to synchronize input qualification may result in unpredictable outcomes. Therefore, when the peripheral itself performs synchronization, ensure that the GPIO pin is configured as asynchronous input qualification.

23.5.2.4 I/O level

Each I/O port has a corresponding register, and each GPIO pin has a corresponding bit/field. The value of each GPIO pin can be changed by configuring the corresponding field.

GPxDAT port x data registers

The current state of the pin after being qualified is represented by the corresponding bit in this register, regardless of whether the pin is configured as GPIO or peripheral function. For details, please refer to Port x Data Register in Register Functional Description.

GPxSET port x setup registers

The setup register is used to drive the specified GPIO pin to a high level without interfering with other pins. For more information about the setup register, please refer to Port x Setup Register in Register Functional Description.

GPxCLR port x clear registers

The clear register is used to drive the specified GPIO pin to a low level without interfering with other pins. For more information about the clear register, please refer to Port x Clear Register in Register Functional Description.

GPIO_xTOG port x switch register

The switch register is used to drive the specified GPIO pin to an opposite level without interfering with other pins. For more information about the switch register, please refer to Port x Switch Register in Register Functional Description.

23.5.3 Application of peripherals in IO

23.5.3.1 Analog input port that can be reused as a digital input (AIO)

In the GPIOH port, GPIO pins are multiplexed with analog pins, so these pins are also known as AIO. GPIOH can only operate in input mode and does not have digital output function. By default, GPIO has high impedance, and GPIOH (GPIO224-254) is analog pin, and the digital or analog operations of these pins are configured in the GPHAMSEL register.

23.5.3.2 Digital Input and Output on ADC Pins (AGPIO)

Some GPIO pins are multiplexed with analogue pins and possess both digital input and output capabilities. These are also referred to as AGPIO. Unlike AIO, AGPIO has full

input and output capabilities.

By default, AGPIO pins are unconnected and must be configured. The table below shows how to configure AGPIO. To enable analogue functionality, set the AGPIOCTRLx registers in the analogue subsystem. To enable digital functionality, set the GPxAMSEL registers as described in the General-Purpose Input/Output (GPIO) chapter.

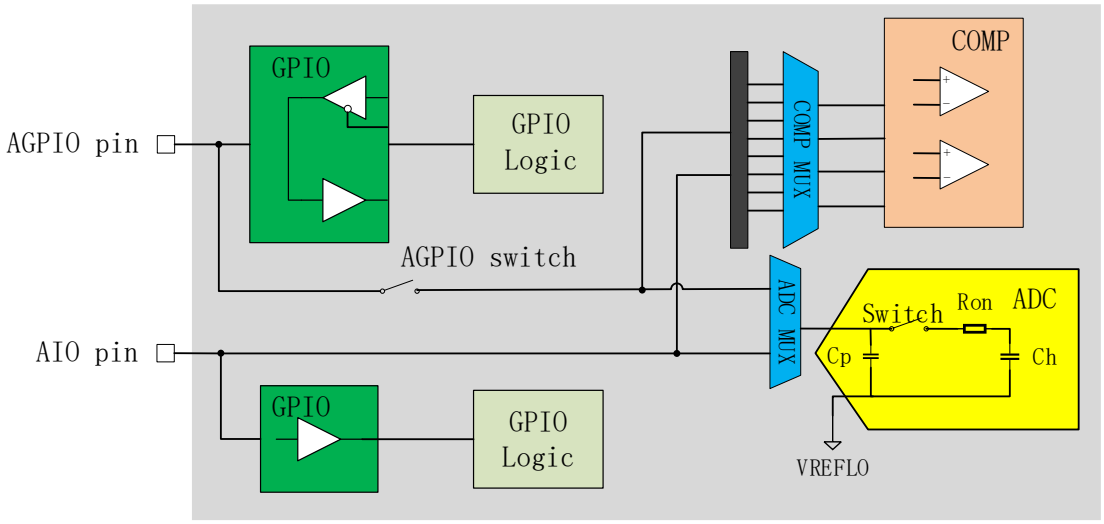
Table67 AGPIO Configuration

AGPIOCTRLx.GPIOy (Default value = 0)	GPxAMSEL.GPIOy (Default = 1)	Pin connected to	
		ADC	GPIOy
0	0	-	Yes
0	1	- (1)	- (1)
1	0	-	Yes
1	1	Yes	-

Note:

(1) By default, no signal is connected to the AGPIO pin. You must select one of the other rows in the table to implement the pin’s functionality.

Figure23 Block diagram of the analogue subsystem with AGPIO



23.5.3.3 SPI non-high-speed mode

SPI supports high-speed mode and can communicate at a rate of up to 40 Mbps. Use special GPIO configuration on single GPIO MUX option of each SPI, so as to reach a high speed as much as possible. These SPI can also use GPIO in non-high speed mode (HSEN=0).

Each pin can multiplex up to 12 kinds of different peripheral functions and general-purpose input/output (GPIO) functions. For more information about multiplexing, please

refer to Datasheet.

23.5.4 Peripheral multiplexing

Pin multiplexing

Configure the multiplexing function of GPIO pins by configuring GPxMUX1/2 and GPxGMUX1/2 registers. Take GPIO6 as an example. According to different configurations of GPAMUX1[GPIO6] and GPAGMUX1[GPIO6], GPIO6 can be set to four different peripheral functions or general-purpose digital I/O.

Table 68 Function Configuration of GPIO6 Pins

GPAGMUX[13:12]	GPAMUX[13:12]	Pin function
00	00	GPIO6
	01	Peripheral 1
	10	Peripheral 2
	11	Peripheral 3
01	00	GPIO6
	01	Peripheral 4
	10	Peripheral 5
	11	-
10	00	GPIO6
	01	-
	10	Peripheral 6
	11	Peripheral 7
11	00	GPIO6
	01	Peripheral 8
	10	Peripheral 9
	11	Peripheral 10

Note:

(1) Different devices adopt different GPIO multiplexing schemes. When a certain peripheral is unavailable on the device, the corresponding multiplexing scheme cannot be used, and the multiplexing scheme is reserved on the device.

(2) When a reserved GPIO multiplexing scheme is selected (i.e. the pin is neither used as a GPIO pin nor configured as a peripheral function), the pin is driven, and its level state is uncertain. In the multiplexing table, "-" and "reserved" are used to represent these unused configurations. Since the

unused configurations are reserved for future expansion, it is advisable to avoid selecting unused configurations.

Peripheral multiplexing configuration

Peripherals can be allocated to different pins by configuring GPxMUX1/2[GPIOy] and GPxGMUX1/2[GPIOy] registers. Take OUTPUTXBAR1 as an example. According to different requirements of the system, GPIO2, GPIO24, GPIO34 or GPIO227 pins can be selected.

Table 69 OUTPUTXBAR1 Multiplexing Scheme

Scheme	GPxGMUX	GPxMUX	Selected GPIO
1	GPAGMUX1[5:4]=01	GPAMUX1[5:4]=01	2
2	GPAGMUX2[17:16]=00	GPAMUX2[17:16]=01	24
3	GPBGMUX1[5:4]=00	GPBMUX1[5:4]=01	34
4	GPBGMUX2[21:20]=01	GPBMUX2[21:20]=01	227

Note: When more than one GPIO pin or no GPIO pin is configured as peripheral input pins, the GPIO pins not configured will be set to the default value of hard wiring.

23.6 GPIO Multiplexing Function Configuration

Table 70 GPIO Multiplexing Function Configuration

0，4，8，12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO0	PWM1_A	CANA_RX	OUTPUTXBAR7	UARTA_RX	I2CA_SDA	SPIA_STE		CANFDA_RX		QEP1_INDEX		PWM3_A	
GPIO1	PWM1_B			UARTA_TX	I2CA_SCL	SPIA_SOMI	QEP1_STROBE	CANFDA_TX				PWM3_B	
GPIO2	PWM2_A			OUTPUTXBAR1	PMBUSA_SDA	SPIA_SIMO	UARTA_TX		I2CB_SDA		CANA_TX	PWM4_A	
GPIO3	PWM2_B	OUTPUTXBAR2		OUTPUTXBAR2	PMBUSA_SCL	SPIA_CLK	UARTA_RX		I2CB_SCL		CANA_RX	PWM4_B	
GPIO4	PWM3_A	I2CA_SCL	CANFDA_TX	OUTPUTXBAR3	CANA_TX						SPIA_SOMI	PWM1_A	
GPIO5	PWM3_B	I2CA_SDA	OUTPUTXBAR3	CANFDA_RX	CANA_RX	SPIA_STE			UARTA_RX			PWM1_B	
GPIO6	PWM4_A	OUTPUTXBAR4	SYNCOUT	QEP1_A								PWM2_A	
GPIO7	PWM4_B	PWM2_A	OUTPUTXBAR5	QEP1_B		SPIA_SIMO			UARTA_TX		CANA_TX	PWM2_B	
GPIO8	PWM5_A		ADCSOCAO	QEP1_STROBE	UARTA_TX	SPIA_SIMO	I2CA_SCL						
GPIO9	PWM5_B	UARTB_TX	OUTPUTXBAR6	QEP1_INDEX	UARTA_RX	SPIA_CLK					I2CB_SCL		
GPIO10	PWM6_A		ADCSOCBO	QEP1_A	UARTB_TX	SPIA_SOMI	I2CA_SDA						

0，4，8，12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO11	PWM6_B	CANA_RX	OUTPUTXBAR7	QEP1_B	UARTB_RX	SPIA_STE				SPIA_SIMO			
GPIO12	PWM7_A		CANFDA_RX	QEP1_STROBE	UARTB_TX	PMBUSA_CTL			SPIA_CLK	CANA_RX		TRACED2	
GPIO13	PWM7_B		CANFDA_TX	QEP1_INDEX	UARTB_RX	PMBUSA_ALERT			SPIA_SOMI	CANA_TX		TRACED3	
GPIO14		UARTB_TX		I2CB_SDA	OUTPUTXBAR3	PMBUSA_SDA				PWM3_A			
GPIO15		UARTB_RX		I2CB_SCL	OUTPUTXBAR4	PMBUSA_SCL				PWM3_B			
GPIO16	SPIA_SIMO		OUTPUTXBAR7	PWM5_A	UARTA_TX		QEP1_STROBE	PMBUSA_SCL	XCLKOUT			TRACED0	
GPIO17	SPIA_SOMI		OUTPUTXBAR8	PWM5_B	UARTA_RX		QEP1_INDEX	PMBUSA_SDA	CANA_TX		PWM6_A	TRGIO	
GPIO18	SPIA_CLK	UARTB_TX	CANA_RX	PWM6_A	I2CA_SCL			PMBUSA_CTL	XCLKOUT				X2
GPIO19	SPIA_STE	UARTB_RX	CANA_TX	PWM6_B	I2CA_SDA			PMBUSA_ALERT					X1
GPIO20	QEP1_A		CANA_TX		SPIA_SIMO		CANFDA_TX		I2CA_SCL			UARTC_TX	
GPIO21	QEP1_B		CANA_RX		SPIA_SOMI		CANFDA_RX		I2CA_SDA			UARTC_RX	
GPIO22	QEP1_STROBE		UARTB_TX				LINA_TX				PWM4_A		
GPIO23	QEP1_INDEX		UARTB_RX				LINA_RX				PWM4_B		

0，4，8，12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO24	OUTPUTXBAR1		SPIA_STE	PWM4_A	SPIA_SIMO			PMBUSA_SCL	UARTA_TX	ERRORSTS			
GPIO25	OUTPUTXBAR2			QEP1_A				PMBUSA_SDA	UARTA_RX				
GPIO26	OUTPUTXBAR3			OUTPUTXBAR3				PMBUSA_CTL	I2CA_SDA				
GPIO27	OUTPUTXBAR4			OUTPUTXBAR4				PMBUSA_ALERT	I2CA_SCL				
GPIO28	UARTA_RX		PWM7_A	OUTPUTXBAR5	QEP1_A			LINA_TX	SPIA_CLK	ERRORSTS	I2CB_SDA	TRACECLK	
GPIO29	UARTA_TX		PWM7_B	OUTPUTXBAR6	QEP1_B			LINA_RX	SPIA_STE	ERRORSTS	I2CB_SCL	TRACED2	
GPIO30	CANA_RX			OUTPUTXBAR7	QEP1_STROBE			CANFDA_RX	PWM1_A			TRACED1	
GPIO31	CANA_TX			OUTPUTXBAR8	QEP1_INDEX			CANFDA_TX	PWM1_B			TRACECLK	
GPIO32	I2CA_SDA	QEP1_INDEX	SPIA_CLK	PWM4_B	LINA_TX			CANA_TX	PMBUSA_SDA	ADCSOCBO		TRGIO	
GPIO33	I2CA_SCL			OUTPUTXBAR4	LINA_RX			CANA_RX		ADCSOCAO		TRACED1	
GPIO34	OUTPUTXBAR1				PMBUSA_SDA						I2CB_SDA	TRACED3	
GPIO35	UARTA_RX	SPIA_SOMI	I2CA_SDA	CANA_RX	PMBUSA_SCL	LINA_RX	QEP1_A	PMBUSA_CTL	PWM5_B			TDI	
GPIO37	OUTPUTXBAR2	SPIA_STE	I2CA_SCL	UARTA_TX	CANA_TX	LINA_TX	QEP1_B	PMBUSA_ALERT	PWM5_A			TDO	

0，4，8，12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO40				PWM2_B	PMBUSA_SDA		UARTB_TX	QEP1_A					
GPIO41	PWM7_A			PWM2_A	PMBUSA_SCL		UARTB_RX	QEP1_B					
GPIO42		LINA_RX	OUTPUTXBAR5	PMBUSA_CTL	I2CA_SDA	UARTC_RX		QEP1_STROBE					
GPIO43			OUTPUTXBAR6	PMBUSA_ALERT	I2CA_SCL	UARTC_TX	PMBUSA_ALERT	QEP1_INDEX					
GPIO44			OUTPUTXBAR7	QEP1_A	PMBUSA_SDA		PMBUSA_CTL						
GPIO45			OUTPUTXBAR8				PMBUSA_ALERT						
GPIO46			LINA_TX	CANFDA_TX			PMBUSA_SDA						
GPIO48	OUTPUTXBAR3		CANA_TX	CANFDA_TX	UARTA_TX		PMBUSA_SDA						
GPIO49	OUTPUTXBAR4		CANA_RX	CANFDA_RX	UARTA_RX		LINA_RX						
GPIO224				OUTPUTXBAR3	SPIA_SIMO		PWM1_A	CANA_TX	QEP1_A		UARTC_TX		
GPIO226			LINA_RX	PWM6_A	SPIA_CLK		PWM1_B		QEP1_STROBE		UARTC_RX		
GPIO227	I2CB_SCL		PWM3_A	OUTPUTXBAR1	PWM2_B								
GPIO228			ADCSOCAO	CANA_TX	SPIA_SOMI		PWM2_B		QEP1_B				

0，4，8，12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO230	I2CB_SDA		PWM3_B	CANA_RX	PWM2_A	I2CA_SDA	PMBUSA_SCL						
GPIO242				OUTPUTXBAR2	SPIA_STE		PWM4_A	CANA_RX	QEP1_INDEX				
AIO225													
AIO231													
AIO232													
AIO233													
AIO237													
AIO238													
AIO239													
AIO241													
AIO244													
AIO245													

23.7 Register bank address

Table 71 GPIO Register Bank Address

Device register	Register bank	Start address	End address
GpioCtrlRegs	GPIO_CTRL_REGS	0x4003 0000	0x4003 07FF
GpioDataRegs	GPIO_DATA_REGS	0x4003 0800	0x4003 08FF
GpioDataReadRegs	GPIO_DATAREAD_REGS	0x4003 0900	0x4003 09FF

23.8 Register address mapping

Table 72 GPIO_CTRL address mapping

Register name	Register description	Offset address	WRPRT
GPACTRL	Port A sampling period register	0x00	√
GPAQSEL1	Port A input type register 1	0x04	√
GPAQSEL2	Port A input type register 2	0x08	√
GPAMUX1	Port A multiplex lower-bit register 1	0x0C	√
GPAMUX2	Port A multiplexing lower-bit register 2	0x10	√
GPADIR	Port A data direction register	0x14	√
GPAPUD	Port A pull-up disable register	0x18	√
GPAINV	Port A input inverting register	0x20	√
GPAODR	Port A open-drain output register	0x24	√
GPAAMSEL	Port A analog mode selection register	0x28	√
GPADSEL1	Port A driver capability selection register 1	0x2C	√
GPADSEL2	Port A driver capability selection register 2	0x30	√
GPAFEN	Port A maximum output frequency enable register	0x34	√
GPAGMUX1	Port A multiplex higher-bit register 1	0x40	√
GPAGMUX2	Port A multiplex higher-bit register 2	0x44	√
GPALOCK	Port A lock register	0x78	√
GPACR	Port A committing lock register	0x7C	√
GPBCTRL	Port B sampling period register	0x80	√
GPBQSEL1	Port B input type register 1	0x84	√
GPBQSEL2	Port B input type register 2	0x88	√
GPBMUX1	Port B multiplex lower-bit register 1	0x8C	√

Register name	Register description	Offset address	WRPRT
GPBMUX2	Port B multiplexing lower-bit register 2	0x90	√
GPBDIR	Port B data direction register	0x94	√
GPBPUD	Port B pull-up disable register	0x98	√
GPBINV	Port B input inverting register	0xA0	√
GPBODR	Port B open-drain output register	0xA4	√
GPBAMSEL	Port B analogue mode selection register	0xA8	√
GPBDSEL1	Port B driver capability selection register 1	0xAC	√
GPBDSEL2	Port B driver capability selection register 2	0xB0	√
GPBFEN	Port B maximum output frequency enable register	0xB4	√
GPBGMUX1	Port B multiplex higher-bit register 1	0xC0	√
GPBGMUX2	Port B multiplex higher-bit register 2	0xC4	√
GPBLOCK	Port B lock register	0xF8	√
GPBCR	Port B committing lock register	0xFC	√
GPHCTRL	Port H sampling period register	0x380	√
GPHQSEL1	Port H input type register 1	0x384	√
GPHQSEL2	Port H input type register 2	0x388	√
GPHMUX1	Port H multiplexed low-order register 1	0x38C	√
GPHMUX2	Port H multiplexed low-order register 2	0x390	√
GPHDIR	Port H data direction register	0x394	√
GPHPUD	Port H pull-up disable register	0x398	√
GPHINV	Port H input inverting register	0x3A0	√
GPHODR	Port H open-drain output register	0x3A4	√
GPHAMSEL	Port H analog mode selection register	0x3A8	√
GPHDSEL1	Port H Drive Capability Selection Register 1	0x3AC	√
GPHDSEL2	Port H Drive Capability Selection Register 2	0x3B0	√
GPHFEN	Port H Maximum Output Frequency Enable Register	0x3B4	√
GPHGMUX1	Port H Multiplexer High-Order Register 1	0x3C0	√
GPHGMUX2	Port H multiplexed high-order register 20x3C4	0x3C4	√
GPHLOCK	Port H lock register	0x3F8	√
GPHCR	Port H committing lock register	0x3FC	√

Table 73 GPIO_DATA address mapping

Register name	Register description	Offset address	WRPRT
GPADAT	Port A data register	0x00	-
GPASET	Port A setting register	0x04	-
GPACLR	Port A clear register	0x08	-
GPATOGGLE	Port A switch register	0x0C	-
GPBDAT	Port B data register	0x10	-
GPBSET	Port B setting register	0x14	-
GPBCLR	Port B clear register	0x18	-
GPBTOGGLE	Port B switch register	0x1C	-
GPHDAT	Port H data register	0x70	-
GPHSET	Port H setting register	0x74	-
GPHCLR	Port H clear register	0x78	-
GPHTOGGLE	Port H switch register	0x7C	-

Table74 GPIO_DATA_READ Address Mapping

Register Name	Register Description	Offset Address	WRPRT
GPADAT_R	GPIO A data read register	0x00	-
GPBDAT_R	GPIO B Data Read Register	0x04	-
GPHDAT_R	GPIO H Data Read Register	0x1C	-

23.9 Register functional description

23.9.1 Port A sampling period register (GPACTRL)

Offset address: 0x00

Reset type: SYSRSn

Qualification sampling period of GPIOA pins (from GPIO0 to GPIO31). For each group consisting of 8 GPIO pins in each field, select the qualification sampling period in the SYSCLK period.

Field	Name	R/W	Description	Reset value
7:0	QUALPRD0	R/W	GPIO0 to GPIO7 sampling cycle Select the sampling period from GPIO0 to GPIO7. 0000 0000: Period=$T_{SYSCLKOUT}$*1 0000 0001: Period=$T_{SYSCLKOUT}$*2 0000 0010: Period=$T_{SYSCLKOUT}$*4 …… 1111 1111: Period=$T_{SYSCLKOUT}$*510	0h

Field	Name	R/W	Description	Reset value
15:8	QUALPRD1	R/W	GPIO8 to GPIO15 sampling cycle Refer to QUALPRD0 for specific description.	0h
23:16	QUALPRD2	R/W	GPIO16 to GPIO23 sampling cycle Refer to QUALPRD0 for specific description.	0h
31:24	QUALPRD3	R/W	GPIO24 to GPIO31 sampling cycle Refer to QUALPRD0 for specific description.	0h

23.9.2 Port A input type register 1 (GPAQSEL1)

Offset address: 0x04

Reset type: SYSRSn

Input qualification type of GPIOA pins (from GPIO0 to GPIO15). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
2y+1:2y	GPIOy	R/W	GPIOy Input type (y=0…15) 00: Synchronous input qualification 01: Input qualification of 3 sampling windows 10: Input qualification of 6 sampling windows 11: Asynchronous input qualification	0h

23.9.3 Port A input type register 2 (GPAQSEL2)

Offset address: 0x08

Reset type: SYSRSn

Input qualification type of GPIOA pins (from GPIO16 to GPIO31). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
1:0	GPIO16	R/W	GPIO16 Input type 00: Synchronous input qualification 01: Input qualification of 3 sampling windows 10: Input qualification of 6 sampling windows 11: Asynchronous input qualification	0h
3:2	GPIO17	R/W	GPIO17 Input type Refer to GPIO16 for specific description.	0h
5:4	GPIO18	R/W	GPIO18 Input type Refer to GPIO16 for specific description.	0h
7:6	GPIO19	R/W	GPIO19 Input type Refer to GPIO16 for specific description.	0h
9:8	GPIO20	R/W	GPIO20 Input type Refer to GPIO16 for specific description.	0h
11:10	GPIO21	R/W	GPIO21 Input type Refer to GPIO16 for specific description.	0h
13:12	GPIO22	R/W	GPIO22 Input type Refer to GPIO16 for specific description.	0h

Field	Name	R/W	Description	Reset value
15:14	GPIO23	R/W	GPIO23 Input type Refer to GPIO16 for specific description.	0h
17:16	GPIO24	R/W	GPIO24 Input type Refer to GPIO16 for specific description.	0h
19:18	GPIO25	R/W	GPIO25 Input type Refer to GPIO16 for specific description.	0h
21:20	GPIO26	R/W	GPIO26 Input type Refer to GPIO16 for specific description.	0h
23:22	GPIO27	R/W	GPIO27 Input type Refer to GPIO16 for specific description.	0h
25:24	GPIO28	R/W	GPIO28 Input type Refer to GPIO16 for specific description.	0h
27:26	GPIO29	R/W	GPIO29 Input type Refer to GPIO16 for specific description.	0h
29:28	GPIO30	R/W	GPIO30 Input type Refer to GPIO16 for specific description.	0h
31:30	GPIO31	R/W	GPIO31 Input type Refer to GPIO16 for specific description.	0h

23.9.4 Port A multiplex lower-bit register 1 (GPAMUX1)

Offset address: 0x0C

Reset value: 0x0000 0000

Reset type: SYSRSn

This register is used to configure the lower bits (from GPIO0 to GPIO15) of the GPIOA pin multiplexing function. When the field value is 0x00, 0x04, 0x08, 0x0C, the corresponding pin is configured as GPIO mode. When the field value is others, the peripheral shall be selected to control the pin. For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of the GPAGMUX1 register, this register must be used to configure the pin as GPIO mode.

Field	Name	R/W	Description	Reset value
2y+1:2y	GPIOy	R/W	GPIOy peripheral mux configuration in the low 2 bits (y=0…15)	0h

23.9.5 Port A multiplex lower-bit register 2 (GPAMUX2)

Offset address: 0x10

Reset type: SYSRSn

This register is used to configure the lower bits (from GPIO16 to GPIO31) of the GPIOA pin multiplexing function. When the field value is 0x00, 0x04, 0x08, 0x0C, the corresponding pin is configured as GPIO mode. When the field value is others, the peripheral shall be selected to control the pin. For the configuration information of each pin, please refer to the Datasheet.

Before changing the corresponding field of the GPAGMUX2 register, this register must be used to configure the pin as GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO16	R/W	GPIO16 peripheral mux configuration in the low 2 bits	0h
3:2	GPIO17	R/W	GPIO17 peripheral mux configuration in the low 2 bits	0h
5:4	GPIO18	R/W	GPIO18 peripheral mux configuration in the low 2 bits	0h
7:6	GPIO19	R/W	GPIO19 peripheral mux configuration in the low 2 bits	0h
9:8	GPIO20	R/W	GPIO20 peripheral mux configuration in the low 2 bits	0h
11:10	GPIO21	R/W	GPIO21 peripheral mux configuration in the low 2 bits	0h
13:12	GPIO22	R/W	GPIO22 peripheral mux configuration in the low 2 bits	0h
15:14	GPIO23	R/W	GPIO23 peripheral mux configuration in the low 2 bits	0h
17:16	GPIO24	R/W	GPIO24 peripheral mux configuration in the low 2 bits	0h
19:18	GPIO25	R/W	GPIO25 peripheral mux configuration in the low 2 bits	0h
21:20	GPIO26	R/W	GPIO26 peripheral mux configuration in the low 2 bits	0h
23:22	GPIO27	R/W	GPIO27 peripheral mux configuration in the low 2 bits	0h
25:24	GPIO28	R/W	GPIO28 peripheral mux configuration in the low 2 bits	0h
27:26	GPIO29	R/W	GPIO29 peripheral mux configuration in the low 2 bits	0h
29:28	GPIO30	R/W	GPIO30 peripheral mux configuration in the low 2 bits	0h
31:30	GPIO31	R/W	GPIO31 peripheral mux configuration in the low 2 bits	0h

23.9.6 Port A data direction register (GPADIR)

Offset address: 0x14

Reset type: SYSRSn

This register configures the data direction of the GPIOA pin. If the pin is not configured as GPIO mode, this register is invalid.

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Data direction (y=0…31) Configure the data direction of pins in GPIO mode. 0: Input 1: Output	0h

23.9.7 Port A pull-up disable register (GPAPUD)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Pull-up disable (y=0…31) Configure the internal pull-up resistor status of the GPIoy pin. 0: Enable internal pull-up resistor 1: Disable internal pull-up resistor	1h

23.9.8 Port A input inverting register (GPAINV)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Input inversion (y=0…31) Whether the input value of the GPIOy pin is configured through an inverter. 0: The input value is not inverted 1: The input value is inverted	0h

23.9.9 Port A open-drain output register (GPAODR)

Offset address: 0x24

Reset type: SYSRSn

Select the output mode of the general-purpose output pins: Push-pull output or open-drain output. In the push-pull output mode, writing 1 to the output data latch will drive the pin to a high level. In the open-drain mode, write 1 to the output data latch to put the output buffer in three states. In either of these two modes, writing 0 to the output data latch will drive the pin to a low level.

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Open-drain output mode (y=0…31) 0: Push-pull output 1: Open-drain output	0h

23.9.10 Port A analog mode selection register (GPAAMSEL)

Offset address: 0x28

Reset type: SYSRSn

Select the analog mode for GPIOA pins. When resetting, the pins are in a special analog mode and their bits in the GPAAMSEL register must be cleared to be reconfigured for GPIO use.

Note: All I/O (including GPIO, AIO and AGPIO) are in analogue mode by default after power-up; they must first be switched to digital mode before the GPIO pull-up function can be used.

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	Analog mode select for GPIOy Select GPIOy as the analog mode or digital mode. 0: Digital mode 1: Analog mode	1h

23.9.11 Port A driver capability selection register 1 (GPADSEL1)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO0	R/W	Driver capability select for GPIO0 When GPAFEN[GPIO0]=1, the maximum output frequency is 25MHz; When GPAFEN[GPIO0]=0, the driver capability is as follows: 00: Maximum output frequency is 20 MHz (with a load of 10 pF), drive capability 2 mA 01: Maximum output frequency is 30 MHz (with a load of 30 pF), drive capability 8 mA 10: Maximum output frequency is 20 MHz (with a load of 20 pF), drive capability is 4 mA 11: Maximum output frequency is 40 MHz (load of 40 pF), drive current is 12 mA Note: The driver capability is not affected by the GPAFEN register.	0h
3:2	GPIO1	R/W	Driver capability select for GPIO1 Refer to GPIO0 for specific description.	0h
5:4	GPIO2	R/W	Driver capability select for GPIO2 Refer to GPIO0 for specific description.	0h
7:6	GPIO3	R/W	Driver capability select for GPIO3 Refer to GPIO0 for specific description.	0h
9:8	GPIO4	R/W	Driver capability select for GPIO4 Refer to GPIO0 for specific description.	0h
11:10	GPIO5	R/W	Driver capability select for GPIO5 Refer to GPIO0 for specific description.	0h
13:12	GPIO6	R/W	Driver capability select for GPIO6 Refer to GPIO0 for specific description.	0h
15:14	GPIO7	R/W	Driver capability select for GPIO7 Refer to GPIO0 for specific description.	0h
17:16	GPIO8	R/W	Driver capability select for GPIO8 Refer to GPIO0 for specific description.	0h
19:18	GPIO9	R/W	Driver capability select for GPIO9 Refer to GPIO0 for specific description.	0h
21:20	GPIO10	R/W	Driver capability select for GPIO10 Refer to GPIO0 for specific description.	0h

Field	Name	R/W	Description	Reset value
23:22	GPIO11	R/W	Driver capability select for GPIO11 Refer to GPIO0 for specific description.	0h
25:24	GPIO12	R/W	Driver capability select for GPIO12 Refer to GPIO0 for specific description.	0h
27:26	GPIO13	R/W	Driver capability select for GPIO13 Refer to GPIO0 for specific description.	0h
29:28	GPIO14	R/W	Driver capability select for GPIO14 Refer to GPIO0 for specific description.	0h
31:30	GPIO15	R/W	Driver capability select for GPIO15 Refer to GPIO0 for specific description.	0h

23.9.12 Port A driver capability selection register 2 (GPADSEL2)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO16	R/W	Driver capability select for GPIO16 When GPAFEN[GPIO16]=1, the maximum output frequency is 25MHz; When GPAFEN[GPIO16]=0, the driver capability is as follows: 00: Maximum output frequency is 20 MHz (with a load of 10 pF), drive capability 2 mA 01: Maximum output frequency is 30 MHz (with a load of 30 pF), drive capability 8 mA 10: Maximum output frequency is 20 MHz (with a load of 20 pF), drive capability is 4 mA 11: Maximum output frequency is 40 MHz (load of 40 pF), drive current is 12 mA Note: The driver capability is not affected by the GPAFEN register.	0h
3:2	GPIO17	R/W	Driver capability select for GPIO17 Refer to GPIO16 for specific description.	0h
5:4	GPIO18	R/W	Driver capability select for GPIO18 Refer to GPIO16 for specific description.	0h
7:6	GPIO19	R/W	Driver capability select for GPIO19 Refer to GPIO16 for specific description.	0h
9:8	GPIO20	R/W	Driver capability select for GPIO20 Refer to GPIO16 for specific description.	0h
11:10	GPIO21	R/W	Driver capability select for GPIO21 Refer to GPIO16 for specific description.	0h
13:12	GPIO22	R/W	Driver capability select for GPIO22 Refer to GPIO16 for specific description.	0h
15:14	GPIO23	R/W	Driver capability select for GPIO23 Refer to GPIO16 for specific description.	0h

Field	Name	R/W	Description	Reset value
17:16	GPIO24	R/W	Driver capability select for GPIO24 Refer to GPIO16 for specific description.	0h
19:18	GPIO25	R/W	Driver capability select for GPIO25 Refer to GPIO16 for specific description.	0h
21:20	GPIO26	R/W	Driver capability select for GPIO26 Refer to GPIO16 for specific description.	0h
23:22	GPIO27	R/W	Driver capability select for GPIO27 Refer to GPIO16 for specific description.	0h
25:24	GPIO28	R/W	Driver capability select for GPIO28 Refer to GPIO16 for specific description.	0h
27:26	GPIO29	R/W	Driver capability select for GPIO29 Refer to GPIO16 for specific description.	0h
29:28	GPIO30	R/W	Driver capability select for GPIO30 Refer to GPIO16 for specific description.	0h
31:30	GPIO31	R/W	Driver capability select for GPIO31 Refer to GPIO16 for specific description.	0h

23.9.13 Port A maximum output frequency enable register (GPAFEN)

Offset address: 0x34

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Maximum Output Frequency Enable(y=0~31) 0: For maximum output frequency of GPIOy, please refer to GPADSEL1/2[GPIOy] 1: The maximum output frequency of GPIOy is 25MHz	0h

23.9.14 Port A multiplex higher-bit register 1 (GPAMUX1)

Offset address: 0x40

Reset type: SYSRSn

This register configures the higher bits of the GPIOA pin multiplexing function (from GPIO0 to GPIO15). For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of this register, the GPAMUX1 register must be used to configure the pin to the GPIO mode.

Field	Name	R/W	Description	Reset value
2y+1:2y	GPIOy	R/W	GPIOy peripheral mux configuration in the High 2 bits (y=0…15)	0h

23.9.15 Port A multiplex higher-bit register 2 (GPAMUX2)

Offset address: 0x44

Reset type: SYSRSn

This register configures the higher bits of the GPIOA pin multiplexing function (from GPIO16 to GPIO31). For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of this register, the GPAMUX2 register must be used to configure the pin to the GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO16	R/W	GPIO16 peripheral mux configuration in the High 2 bits	0h
3:2	GPIO17	R/W	GPIO17 peripheral mux configuration in the High 2 bits	0h
5:4	GPIO18	R/W	GPIO18 peripheral mux configuration in the High 2 bits	0h
7:6	GPIO19	R/W	GPIO19 peripheral mux configuration in the High 2 bits	0h
9:8	GPIO20	R/W	GPIO20 peripheral mux configuration in the High 2 bits	0h
11:10	GPIO21	R/W	GPIO21 peripheral mux configuration in the High 2 bits	0h
13:12	GPIO22	R/W	GPIO22 peripheral mux configuration in the High 2 bits	0h
15:14	GPIO23	R/W	GPIO23 peripheral mux configuration in the High 2 bits	0h
17:16	GPIO24	R/W	GPIO24 peripheral mux configuration in the High 2 bits	0h
19:18	GPIO25	R/W	GPIO25 peripheral mux configuration in the High 2 bits	0h
21:20	GPIO26	R/W	GPIO26 peripheral mux configuration in the High 2 bits	0h
23:22	GPIO27	R/W	GPIO27 peripheral mux configuration in the High 2 bits	0h
25:24	GPIO28	R/W	GPIO28 peripheral mux configuration in the High 2 bits	0h
27:26	GPIO29	R/W	GPIO29 peripheral mux configuration in the High 2 bits	0h
29:28	GPIO30	R/W	GPIO30 peripheral mux configuration in the High 2 bits	0h
31:30	GPIO31	R/W	GPIO31 peripheral mux configuration in the High 2 bits	0h

23.9.16 Port A lock register (GPALOCK)

Offset address: 0x78

Reset type: SYSRSn

Lock the configuration of GPIOA pins (from GPIO0 to GPIO31). This register will write to GPAMUX1/2, GPAGMUX1/2, GPADIR, GPAINV, and GPAODR registers.

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Configuration lock (y=0…31) 0: GPIOy configuration not locked 1: GPIOy configuration locked	0h

23.9.17 Port A committing lock register (GPACR)

Offset address: 0x7C

Reset type: SYSRSn

This bit confirms whether the configuration of the GPIOA pins (from GPIO0 to GPIO31) is locked. Once this bit is set, it can only be cleared through reset. This register will write to the GPALOCK[GPIOy] register.

Field	Name	R/W	Description	Reset value
y	GPIOy	R/WSO	GPIOy lock is committed The bit commits locking of GPALOCK[GPIOy]. 0: The configuration of locking GPSLOCK[GPIOy] is not committed 1: The configuration of locking GPSLOCK[GPIOy] is committed	0h

23.9.18 Port B sampling period register (GPBCTRL)

Offset address: 0x80

Reset type: SYSRSn

Qualification sampling period of GPIOB pins (from GPIO32 to GPIO49). For each group consisting of 8 GPIO pins in each field, select the qualification sampling period in the SYSCLK period, and the number of periods is twice the field value.

Field	Name	R/W	Description	Reset value
7:0	QUALPRD0	R/W	GPIO32 to GPIO39 sampling cycle Select the sampling period from GPIO32 to GPIO39. 0000 0000: Period=$T_{SYSCLKOUT}$*1 0000 0001: Period=$T_{SYSCLKOUT}$*2 0000 0010: Period=$T_{SYSCLKOUT}$*4 …… 1111 1111: Period=$T_{SYSCLKOUT}$*510	0h
15:8	QUALPRD1	R/W	GPIO40 to GPIO47 sampling cycle Refer to QUALPRD0 for specific description.	0h
17:16	QUALPRD2	R/W	GPIO48 to GPIO49 sampling cycle Refer to QUALPRD0 for specific description.	0h
31:18	Reserved			0h

23.9.19 Port B input type register 1 (GPBQSEL1)

Offset address: 0x84

Reset type: SYSRSn

Input qualification type of GPIOB pins (from GPIO32 to GPIO47). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
1:0	GPIO32	R/W	GPIO32 Input type 00: Synchronous input qualification 01: Input qualification of 3 sampling windows 10: Input qualification of 6 sampling windows 11: Asynchronous input qualification	0h

Field	Name	R/W	Description	Reset value
3:2	GPIO33	R/W	GPIO33 Input type Refer to GPIO32 for specific description.	0h
5:4	GPIO34	R/W	GPIO34 Input type Refer to GPIO32 for specific description.	0h
7:6	GPIO35	R/W	GPIO35 Input type Refer to GPIO32 for specific description.	0h
9:8	Reserved			0h
11:10	GPIO37	R/W	GPIO37 Input type Refer to GPIO32 for specific description.	0h
15:12	Reserved			0h
17:16	GPIO40	R/W	GPIO40 Input type Refer to GPIO32 for specific description.	0h
19:18	GPIO41	R/W	GPIO41 Input type Refer to GPIO32 for specific description.	0h
21:20	GPIO42	R/W	GPIO42 Input type Refer to GPIO32 for specific description.	0h
23:22	GPIO43	R/W	GPIO43 Input type Refer to GPIO32 for specific description.	0h
25:24	GPIO44	R/W	GPIO44 Input type Refer to GPIO32 for specific description.	0h
27:26	GPIO45	R/W	GPIO45 Input type Refer to GPIO32 for specific description.	0h
29:28	GPIO46	R/W	GPIO46 Input type Refer to GPIO32 for specific description.	0h
31:30	Reserved			0h

23.9.20 Port B input type register 2 (GPBQSEL2)

Offset address: 0x88

Reset type: SYSRSn

Input qualification type of GPIOB pins (from GPIO48 to GPIO49). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
1:0	GPIO48	R/W	GPIO48 Input type Refer to GPIO32 for specific description.	0h
3:2	GPIO49	R/W	GPIO49 Input type Refer to GPIO32 for specific description.	0h
31:4	Reserved			0h

23.9.21 Port B multiplex lower-bit register 1 (GPBMUX1)

Offset address: 0x8C

Reset type: SYSRSn

This register is used to configure the lower bits of the GPIOB pin multiplexing function. When the field value is 0x00, 0x04, 0x08, 0x0C, the corresponding pin is configured as GPIO mode. When the field value is others, the peripheral shall be selected to control the pin. For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of the GPBGMUX1 register, this register must be used to configure the pin as GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO32	R/W	GPIO32 peripheral mux configuration in the low 2 bits	0h
3:2	GPIO33	R/W	GPIO33 peripheral mux configuration in the low 2 bits	0h
5:4	GPIO34	R/W	GPIO34 peripheral mux configuration in the low 2 bits	0h
7:6	GPIO35	R/W	GPIO35 peripheral mux configuration in the low 2 bits	3h
9:8	Reserved			0h
11:10	GPIO37	R/W	GPIO37 peripheral mux configuration in the low 2 bits	3h
15:12	Reserved			0h
17:16	GPIO40	R/W	GPIO40 peripheral mux configuration in the low 2 bits	0h
19:18	GPIO41	R/W	GPIO41 peripheral mux configuration in the low 2 bits	0h
21:20	GPIO42	R/W	GPIO42 peripheral mux configuration in the low 2 bits	0h
23:22	GPIO43	R/W	GPIO43 peripheral mux configuration in the low 2 bits	0h
25:24	GPIO44	R/W	GPIO44 peripheral mux configuration in the low 2 bits	0h
27:26	GPIO45	R/W	GPIO45 peripheral mux configuration in the low 2 bits	0h
29:28	GPIO46	R/W	GPIO46 peripheral mux configuration in the low 2 bits	0h
31:30	Reserved			0h

23.9.22 Port B multiplex lower-bit register 2 (GPBMUX2)

Offset address: 0x90

Reset type: SYSRSn

This register is used to configure the lower bits (from GPIO48 to GPIO49) of the GPIOB pin multiplexing function. When the field value is 0x00, 0x04, 0x08, 0x0C, the corresponding pin is configured as GPIO mode. When the field value is others, the peripheral shall be selected to control the pin. For the configuration information of each pin, please refer to the Datasheet.

Before changing the corresponding field of the GPBGMUX2 register, this register must be used to configure the pin as GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO48	R/W	GPIO48 peripheral mux configuration in the low 2 bits	0h
3:2	GPIO49	R/W	GPIO49 peripheral mux configuration in the low 2 bits	0h

Field	Name	R/W	Description	Reset value
31:4	Reserved			0h

23.9.23 Port B data direction register (GPBDIR)

Offset address: 0x94

Reset type: SYSRSn

This register configures the data direction of the GPIOB pin. If the pin is not configured as GPIO mode, this register is invalid.

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Data direction Configure the data direction of pins in GPIO mode. 0: Input 1: Output	0h
1	GPIO33	R/W	GPIO33 Data direction Refer to GPIO32 for specific description.	0h
2	GPIO34	R/W	GPIO34 Data direction Refer to GPIO32 for specific description.	0h
3	GPIO35	R/W	GPIO35 Data direction Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/W	GPIO37 Data direction Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/W	GPIO40 Data direction Refer to GPIO32 for specific description.	0h
9	GPIO41	R/W	GPIO41 Data direction Refer to GPIO32 for specific description.	0h
10	GPIO42	R/W	GPIO42 Data direction Refer to GPIO32 for specific description.	0h
11	GPIO43	R/W	GPIO43 Data direction Refer to GPIO32 for specific description.	0h
12	GPIO44	R/W	GPIO44 Data direction Refer to GPIO32 for specific description.	0h
13	GPIO45	R/W	GPIO45 Data direction Refer to GPIO32 for specific description.	0h
14	GPIO46	R/W	GPIO46 Data direction Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/W	GPIO48 Data direction Refer to GPIO32 for specific description.	0h
17	GPIO49	R/W	GPIO49 Data direction Refer to GPIO32 for specific description.	0h

Field	Name	R/W	Description	Reset value
31:18	Reserved			0h

23.9.24 Port B pull-up disable register (GPBPUD)

Offset address: 0x98

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Pull-up disable Configure the internal pull-up resistor status of the GPIO32. 0: Enable internal pull-up resistor 1: Disable internal pull-up resistor	1h
1	GPIO33	R/W	GPIO33 Pull-up disable Refer to GPIO32 for specific description.	1h
2	GPIO34	R/W	GPIO34 Pull-up disable Refer to GPIO32 for specific description.	1h
3	GPIO35	R/W	GPIO35 Pull-up disable Refer to GPIO32 for specific description.	1h
4	Reserved			1h
5	GPIO37	R/W	GPIO37 Pull-up disable Refer to GPIO32 for specific description.	1h
7:6	Reserved			1h
8	GPIO40	R/W	GPIO40 Pull-up disable Refer to GPIO32 for specific description.	1h
9	GPIO41	R/W	GPIO41 Pull-up disable Refer to GPIO32 for specific description.	1h
10	GPIO42	R/W	GPIO42 Pull-up disable Refer to GPIO32 for specific description.	1h
11	GPIO43	R/W	GPIO43 Pull-up disable Refer to GPIO32 for specific description.	1h
12	GPIO44	R/W	GPIO44 Pull-up disable Refer to GPIO32 for specific description.	1h
13	GPIO45	R/W	GPIO45 Pull-up disable Refer to GPIO32 for specific description.	1h
14	GPIO46	R/W	GPIO46 Pull-up disable Refer to GPIO32 for specific description.	1h
15	GPIO47	R/W	GPIO47 Pull-up disable Refer to GPIO32 for specific description.	1h
16	GPIO48	R/W	GPIO48 Pull-up disable Refer to GPIO32 for specific description.	1h
17	GPIO49	R/W	GPIO49 Pull-up disable Refer to GPIO32 for specific description.	1h
31:18	Reserved			1h

23.9.25 Port B input inverting register (GPBINV)

Offset address: 0xA0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INVER32	R/W	GPIO32 Input inversion Whether the input value of the GPIO32 pin is configured through an inverter. 0: The input value is not inverted 1: The input value is inverted	0h
1	INVER33	R/W	GPIO33 Input inversion Refer to INVER32 for specific description.	0h
2	INVER34	R/W	GPIO34 Input inversion Refer to INVER32 for specific description.	0h
3	INVER35	R/W	GPIO35 Input inversion Refer to INVER32 for specific description.	0h
4	Reserved			0h
5	INVER37	R/W	GPIO37 Input inversion Refer to INVER32 for specific description.	0h
7:6	Reserved			0h
8	INVER40	R/W	GPIO40 Input inversion Refer to INVER32 for specific description.	0h
9	INVER41	R/W	GPIO41 Input inversion Refer to INVER32 for specific description.	0h
10	INVER42	R/W	GPIO42 Input inversion Refer to INVER32 for specific description.	0h
11	INVER43	R/W	GPIO43 Input inversion Refer to INVER32 for specific description.	0h
12	INVER44	R/W	GPIO44 Input inversion Refer to INVER32 for specific description.	0h
13	INVER45	R/W	GPIO45 Input inversion Refer to INVER32 for specific description.	0h
14	INVER46	R/W	GPIO46 Input inversion Refer to INVER32 for specific description.	0h
15	Reserved			0h
16	INVER48	R/W	GPIO48 Input inversion Refer to INVER32 for specific description.	0h
17	INVER49	R/W	GPIO49 Input inversion Refer to INVER32 for specific description.	0h
31:18	Reserved			0h

23.9.26 Port B open-drain output register (GPBODR)

Offset address: 0xA4

Reset type: SYSRSn

Select the output mode of the general-purpose output pins: Push-pull output or open-drain output. In the push-pull output mode, writing 1 to the output data latch will drive the pin to a high level. In the open-drain mode, write 1 to the output data latch to put the output buffer in three states. In either of these two modes, writing 0 to the output data latch will drive the pin to a low level.

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Open-drain output mode 0: Push-pull output 1: Open-drain output	0h
1	GPIO33	R/W	GPIO33 Open-drain output mode Refer to GPIO32 for specific description.	0h
2	GPIO34	R/W	GPIO34 Open-drain output mode Refer to GPIO32 for specific description.	0h
3	GPIO35	R/W	GPIO35 Open-drain output mode Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/W	GPIO37 Open-drain output mode Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/W	GPIO40 Open-drain output mode Refer to GPIO32 for specific description.	0h
9	GPIO41	R/W	GPIO41 Open-drain output mode Refer to GPIO32 for specific description.	0h
10	GPIO42	R/W	GPIO42 Open-drain output mode Refer to GPIO32 for specific description.	0h
11	GPIO43	R/W	GPIO43 Open-drain output mode Refer to GPIO32 for specific description.	0h
12	GPIO44	R/W	GPIO44 Open-drain output mode Refer to GPIO32 for specific description.	0h
13	GPIO45	R/W	GPIO45 Open-drain output mode Refer to GPIO32 for specific description.	0h
14	GPIO46	R/W	GPIO46 Open-drain output mode Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/W	GPIO48 Open-drain output mode Refer to GPIO32 for specific description.	0h
17	GPIO49	R/W	GPIO49 Open-drain output mode Refer to GPIO32 for specific description.	0h
31:18	Reserved			0h

23.9.27 Port B analogue mode selection register (GPBAMSEL)

Offset address: 0x0A8

Reset type: SYSRSn

Note: All I/O (including GPIO, AIO and AGPIO) are in analogue mode by default after power-up; you must first switch them to digital mode before you can use the GPIO pull-up function.

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	Analogue mode selection for GPIO32 Select either analogue or digital mode for GPIO32. 0: Digital mode 1: Analogue mode	0h
1	GPIO33	R/W	Analogue mode selection for GPIO33 Refer to GPIO32 for specific description.	0h
2	GPIO34	R/W	Analogue mode selection for GPIO34 Refer to GPIO32 for specific description.	0h
3	GPIO35	R/W	Analogue mode selection for GPIO35 Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/W	Analogue mode selection for GPIO37 Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/W	Analogue mode selection for GPIO40 Refer to GPIO32 for specific description.	0h
9	GPIO41	R/W	Analogue mode selection for GPIO41 Refer to GPIO32 for specific description.	0h
10	GPIO42	R/W	Analogue mode selection for GPIO42 Refer to GPIO32 for specific description.	0h
11	GPIO43	R/W	Analogue mode selection for GPIO43 Refer to GPIO32 for specific description.	0h
12	GPIO44	R/W	Analogue mode selection for GPIO44 Refer to GPIO32 for specific description.	0h
13	GPIO45	R/W	Analogue mode selection for GPIO45 Refer to GPIO32 for specific description.	0h
14	GPIO46	R/W	Analogue mode selection for GPIO46 Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/W	Analogue mode selection for GPIO48 Refer to GPIO32 for specific description.	0h
17	GPIO49	R/W	Analogue mode selection for GPIO49 Refer to GPIO32 for specific description.	0h
31:18	Reserved			0h

23.9.28 Port B driver capability selection register 1 (GPBDSEL1)

Offset address: 0xAC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO32	R/W	Driver capability select for GPIO32 When GPBFEN[GPIO32]=1, the maximum output frequency is 25MHz; When GPBFEN[GPIO32]=0, the driver capability is as follows: 00: The maximum output frequency is 30MHz, and the driver capability is 1mA 01: The maximum output frequency is 50MHz, and the driver capability is 4mA 10: The maximum output frequency is 40MHz, and the driver capability is 2mA 11: The maximum output frequency is 60MHz, and the driver capability is 6mA Note: The driver capability is not affected by the GPBFEN register.	0h
3:2	GPIO33	R/W	Driver capability select for GPIO33 Refer to GPIO32 for specific description.	0h
5:4	GPIO34	R/W	Driver capability select for GPIO34 Refer to GPIO32 for specific description.	0h
7:6	GPIO35	R/W	Driver capability select for GPIO35 Refer to GPIO32 for specific description.	0h
9:8	Reserved			0h
11:10	GPIO37	R/W	Driver capability select for GPIO37 Refer to GPIO32 for specific description.	0h
15:12	Reserved			0h
17:16	GPIO40	R/W	Driver capability select for GPIO40 Refer to GPIO32 for specific description.	0h
19:18	GPIO41	R/W	Driver capability select for GPIO41 Refer to GPIO32 for specific description.	0h
21:20	GPIO42	R/W	Driver capability select for GPIO42 Refer to GPIO32 for specific description.	0h
23:22	GPIO43	R/W	Driver capability select for GPIO43 Refer to GPIO32 for specific description.	0h
25:24	GPIO44	R/W	Driver capability select for GPIO44 Refer to GPIO32 for specific description.	0h
27:26	GPIO45	R/W	Driver capability select for GPIO45 Refer to GPIO32 for specific description.	0h
29:28	GPIO46	R/W	Driver capability select for GPIO46 Refer to GPIO32 for specific description.	0h
31:30	Reserved			0h

23.9.29 Port B driver capability selection register 2 (GPBDSEL2)

Offset address: 0xB0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO48	R/W	Driver capability select for GPIO48 Refer to GPIO32 for specific description.	0h
3:2	GPIO49	R/W	Driver capability select for GPIO49 Refer to GPIO32 for specific description.	0h
31:4	Reserved			0h

23.9.30 Port B maximum output frequency enable register (GPBFEN)

Offset address: 0xB4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Maximum Output Frequency Enable 0: For maximum output frequency of GPIO32, please refer to GPBDSEL1[GPIO32] 1: The maximum output frequency of GPIO32 is 25MHz	0h
1	GPIO33	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
2	GPIO34	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
3	GPIO35	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
9	GPIO41	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
10	GPIO42	R/W	GPIO32 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
11	GPIO43	R/W	GPIO43 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
12	GPIO44	R/W	GPIO44 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h

Field	Name	R/W	Description	Reset value
13	GPIO45	R/W	GPIO45 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
14	GPIO46	R/W	GPIO46 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/W	GPIO48 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
17	GPIO49	R/W	GPIO49 Maximum Output Frequency Enable Refer to GPIO32 for specific description.	0h
31:18	Reserved			0h

23.9.31 Port B multiplex higher-bit register 1 (GPBGMUX1)

Offset address: 0xC0

Reset type: SYSRSn

This register configures the higher bits of the GPIOB pin multiplexing function (from GPIO32 to GPIO47). For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of this register, the GPBMUX1 register must be used to configure the pin to the GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO32	R/W	GPIO32 peripheral mux configuration in the High 2 bits	0h
3:2	GPIO33	R/W	GPIO33 peripheral mux configuration in the High 2 bits	0h
5:4	GPIO34	R/W	GPIO34 peripheral mux configuration in the High 2 bits	0h
7:6	GPIO35	R/W	GPIO35 peripheral mux configuration in the High 2 bits	3h
9:8	Reserved			0h
11:10	GPIO37	R/W	GPIO37 peripheral mux configuration in the High 2 bits	3h
15:12	Reserved			0h
17:16	GPIO40	R/W	GPIO40 peripheral mux configuration in the High 2 bits	0h
19:18	GPIO41	R/W	GPIO41 peripheral mux configuration in the High 2 bits	0h
21:20	GPIO42	R/W	GPIO42 peripheral mux configuration in the High 2 bits	0h
23:22	GPIO43	R/W	GPIO43 peripheral mux configuration in the High 2 bits	0h
25:24	GPIO44	R/W	GPIO44 peripheral mux configuration in the High 2 bits	0h
27:26	GPIO45	R/W	GPIO45 peripheral mux configuration in the High 2 bits	0h
29:28	GPIO46	R/W	GPIO46 peripheral mux configuration in the High 2 bits	0h
31:30	Reserved			0h

23.9.32 Port B multiplex higher-bit register 2 (GPBGMUX2)

Offset address: 0xC4

Reset type: SYSRSn

This register configures the higher bits of the GPIOB pin multiplexing function (from GPIO48 to GPIO49). For the configuration information of each pin, please refer to the Datasheet.

Before changing the field of this register, the GPBMUX2 register must be used to configure the pin to the GPIO mode.

Field	Name	R/W	Description	Reset value
1:0	GPIO48	R/W	GPIO48 peripheral mux configuration in the High 2 bits	0h
3:2	GPIO49	R/W	GPIO49 peripheral mux configuration in the High 2 bits	0h
31:4	Reserved			0h

23.9.33 Port B lock register (GPBLOCK)

Offset address: 0xF8

Reset type: SYSRSn

Lock the configuration of GPIOB pins (from GPIO32 to GPIO49). This register will write to GPBMUX1/2, GPBGMUX1/2, GPBDIR, GPBINV, and GPBODR registers.

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Configuration lock 0: GPIO32 configuration not locked 1: GPIO32 configuration locked	0h
1	GPIO33	R/W	GPIO33 Configuration lock Refer to GPIO32 for specific description.	0h
2	GPIO34	R/W	GPIO34 Configuration lock Refer to GPIO32 for specific description.	0h
3	GPIO35	R/W	GPIO35 Configuration lock Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/W	GPIO37 Configuration lock Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/W	GPIO40 Configuration lock Refer to GPIO32 for specific description.	0h
9	GPIO41	R/W	GPIO41 Configuration lock Refer to GPIO32 for specific description.	0h
10	GPIO42	R/W	GPIO42 Configuration lock Refer to GPIO32 for specific description.	0h
11	GPIO43	R/W	GPIO43 Configuration lock Refer to GPIO32 for specific description.	0h
12	GPIO44	R/W	GPIO44 Configuration lock Refer to GPIO32 for specific description.	0h

Field	Name	R/W	Description	Reset value
13	GPIO45	R/W	GPIO45 Configuration lock Refer to GPIO32 for specific description.	0h
14	GPIO46	R/W	GPIO46 Configuration lock Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/W	GPIO48 Configuration lock Refer to GPIO32 for specific description.	0h
17	GPIO49	R/W	GPIO49 Configuration lock Refer to GPIO32 for specific description.	0h
31:18	Reserved			0h

23.9.34 Port B committing lock register (GPBCR)

Offset address: 0xFC

Reset type: SYSRSn

This bit confirms whether the configuration of the GPIOB pins (from GPIO32 to GPIO49) is locked. Once this bit is set, it can only be cleared through reset. This register will write to the GPBLOCK[GPIOy] register.

Field	Name	R/W	Description	Reset value
0	GPIO32	R/WSO	GPIO32 lock is committed The bit commits locking of GPBLOCK[GPIO32]. 0: The configuration of locking GPBLOCK[GPIO32] is not committed 1: The configuration of locking GPBLOCK[GPIO32] is committed	0h
1	GPIO33	R/WSO	GPIO33 lock is committed Refer to GPIO32 for specific description.	0h
2	GPIO34	R/WSO	GPIO34 lock is committed Refer to GPIO32 for specific description.	0h
3	GPIO35	R/WSO	GPIO35 lock is committed Refer to GPIO32 for specific description.	0h
4	Reserved			0h
5	GPIO37	R/WSO	GPIO37 lock is committed Refer to GPIO32 for specific description.	0h
7:6	Reserved			0h
8	GPIO40	R/WSO	GPIO40 lock is committed Refer to GPIO32 for specific description.	0h
9	GPIO41	R/WSO	GPIO41 lock is committed Refer to GPIO32 for specific description.	0h
10	GPIO42	R/WSO	GPIO42 lock is committed Refer to GPIO32 for specific description.	0h
11	GPIO43	R/WSO	GPIO43 lock is committed Refer to GPIO32 for specific description.	0h

Field	Name	R/W	Description	Reset value
12	GPIO44	R/WSO	GPIO44 lock is committed Refer to GPIO32 for specific description.	0h
13	GPIO45	R/WSO	GPIO45 lock is committed Refer to GPIO32 for specific description.	0h
14	GPIO46	R/WSO	GPIO46 lock is committed Refer to GPIO32 for specific description.	0h
15	Reserved			0h
16	GPIO48	R/WSO	GPIO48 lock is committed Refer to GPIO32 for specific description.	0h
17	GPIO49	R/WSO	GPIO49 lock is committed Refer to GPIO32 for specific description.	0h
31:18	Reserved			0h

23.9.35 Port H sampling period register (GPHCTRL)

Offset address: 0x380

Reset type: SYSRSn

Qualification sampling period of GPIOH pins (from GPIO224 to GPIO245). For each group consisting of 8 GPIO pins in each field, select the qualification sampling period in the SYSCLK period.

Field	Name	R/W	Description	Reset value
7:0	QUALPRD0	R/W	GPIO224 to GPIO231 sampling cycle Select the sampling period from GPIO224 to GPIO231. 0000 0000: Period=$T_{SYSCLKOUT}$*1 0000 0001: Period=$T_{SYSCLKOUT}$*2 0000 0010: Period=$T_{SYSCLKOUT}$*4 …… 1111 1111: Period=$T_{SYSCLKOUT}$*510	0h
15:8	QUALPRD1	R/W	GPIO232 to GPIO239 sampling cycle Refer to QUALPRD0 for specific description.	0h
21:16	QUALPRD2	R/W	GPIO240 to GPIO245sampling cycle Refer to QUALPRD0 for specific description.	0h
31:22	Reserved			0h

23.9.36 Port H input type register 1 (GPHQSEL1)

Offset address: 0x384

Reset type: SYSRSn

Input qualification type of GPIOH pins (from GPIO224 to GPIO239). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
1:0	GPIO224	R/W	GPIO224 Input type 00: Synchronous input qualification	0h

Field	Name	R/W	Description	Reset value
			01: Input qualification of 3 sampling windows 10: Input qualification of 6 sampling windows 11: Asynchronous input qualification	
3:2	GPIO225	R/W	GPIO225 Input type Refer to GPIO224 for specific description.	0h
5:4	GPIO226	R/W	GPIO226 Input type Refer to GPIO224 for specific description.	0h
7:6	GPIO227	R/W	GPIO227 Input type Refer to GPIO224 for specific description.	0h
9:8	GPIO228	R/W	GPIO228 Input type Refer to GPIO224 for specific description.	0h
11:10	Reserved			0h
13:12	GPIO230	R/W	GPIO230 Input type Refer to GPIO224 for specific description.	0h
15:14	GPIO231	R/W	GPIO231 Input type Refer to GPIO224 for specific description.	0h
17:16	GPIO232	R/W	GPIO232 Input type Refer to GPIO224 for specific description.	0h
19:18	GPIO233	R/W	GPIO233 Input type Refer to GPIO224 for specific description.	0h
25:20	Reserved			0h
27:26	GPIO237	R/W	GPIO237 Input type Refer to GPIO224 for specific description.	0h
29:28	GPIO238	R/W	GPIO238 Input type Refer to GPIO224 for specific description.	0h
31:30	GPIO239	R/W	GPIO239 Input type Refer to GPIO224 for specific description.	0h

23.9.37 Port H input type register 2 (GPHQSEL2)

Offset address: 0x388

Reset type: SYSRSn

Input qualification type of GPIOH pins (from GPIO240 to GPIO245). Select the corresponding qualification input type for each pin.

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
3:2	GPIO241	R/W	GPIO241 Input type Refer to GPIO224 for specific description.	0h
5:4	GPIO242	R/W	GPIO242 Input type Refer to GPIO224 for specific description.	0h
7:6	Reserved			0h

Field	Name	R/W	Description	Reset value
9:8	GPIO244	R/W	GPIO244 Input type Refer to GPIO224 for specific description.	0h
11:10	GPIO245	R/W	GPIO245 Input type Refer to GPIO224 for specific description.	0h
31:12	Reserved			0h

23.9.38 Port H multiplexed low-order register 1 (GPHMUX1)

Offset address: 0x38C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO224	R/W	The lower two bits of the GPIO224 peripheral multiplexer configuration	0h
3:2	Reserved			0h
5:4	GPIO226	R/W	GPIO226 peripheral multiplexer configuration in the lower 2 bits	0h
7:6	GPIO227	R/W	GPIO227 peripheral multiplexer configuration in the low 2 bits	0h
9:8	GPIO228	R/W	GPIO228 peripheral multiplexer configuration in the lower 2 bits	0h
11:10	Reserved			0h
13:12	GPIO230	R/W	The lower two bits of the GPIO230 peripheral multiplexer configuration	0h
31:14	Reserved			0h

23.9.39 Port H Multiplexing Low-Order Register 2 (GPHMUX2)

Offset address: 0x390

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	Reserved			0h
5:4	GPIO242	R/W	GPIO242 peripheral multiplexer configuration in the lower 2 bits	0h
31:6	Reserved			0h

23.9.40 Port H Data Direction Register (GPHDIR)

Offset address: 0x394

Reset type: SYSRSn

This register configures the data direction of the GPIOH pin. If the pin is not configured in GPIO mode, this register is invalid.

Field	Name	R/W	Description	Reset Value
0	GPIO224	R/W	GPIO224 Data Direction Configures the data direction of the pin in GPIO mode.	0h

Field	Name	R/W	Description	Reset Value
			0: Input 1: Output	
1	Reserved			0h
2	GPIO226	R/W	GPIO226 Data Direction For a detailed description, refer to GPIO224.	0h
3	GPIO227	R/W	GPIO227 Data Direction For a detailed description, refer to GPIO224.	0h
4	GPIO228	R/W	GPIO228 Data Direction For a detailed description, refer to GPIO224.	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Data Direction For a detailed description, refer to GPIO224.	0h
17:7	Reserved			0h
18	GPIO242	R/W	GPIO242 Data Direction For a detailed description, refer to GPIO224.	0h
31:19	Reserved			0h

23.9.41 Port H pull-up disable register (GPHBPUD)

Offset address: 0x398

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Pull-up disable Configure the internal pull-up resistor status of the GPIO224. 0: Enable internal pull-up resistor 1: Disable internal pull-up resistor	1h
1	GPIO225	R/W	GPIO225 Pull-up disable Refer to GPIO224 for specific description.	1h
2	GPIO226	R/W	GPIO226 Pull-up disable Refer to GPIO224 for specific description.	1h
3	GPIO227	R/W	GPIO227 Pull-up disable Refer to GPIO224 for specific description.	1h
4	GPIO228	R/W	GPIO228 Pull-up disable Refer to GPIO224 for specific description.	1h
5	Reserved			1h
6	GPIO230	R/W	GPIO230 Pull-up disable Refer to GPIO224 for specific description.	1h
7	GPIO231	R/W	GPIO231 Pull-up disable Refer to GPIO224 for specific description.	1h
8	GPIO232	R/W	GPIO232 Pull-up disable Refer to GPIO224 for specific description.	1h
9	GPIO233	R/W	GPIO233 Pull-up disable Refer to GPIO224 for specific description.	1h

Field	Name	R/W	Description	Reset value
10	Reserved			1h
11	Reserved			1h
12	Reserved			1h
13	GPIO237	R/W	GPIO237 Pull-up disable Refer to GPIO224 for specific description.	1h
14	GPIO238	R/W	GPIO238 Pull-up disable Refer to GPIO224 for specific description.	1h
15	GPIO239	R/W	GPIO239 Pull-up disable Refer to GPIO224 for specific description.	1h
16	Reserved			1h
17	GPIO241	R/W	GPIO241 Pull-up disable Refer to GPIO224 for specific description.	1h
18	GPIO242	R/W	GPIO242 Pull-up disable Refer to GPIO224 for specific description.	1h
19	Reserved			1h
20	GPIO244	R/W	GPIO244 Pull-up disable Refer to GPIO224 for specific description.	1h
21	GPIO245	R/W	GPIO245 Pull-up disable Refer to GPIO224 for specific description.	1h
31:22	Reserved			1h

23.9.42 Port H input inverting register (GPHINV)

Offset address: 0x3A0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Input inversion Whether the input value of the GPIO224 pin is configured through an inverter. 0: The input value is not inverted 1: The input value is inverted	0h
1	GPIO225	R/W	GPIO225 Input inversion Refer to GPIO224 for specific description.	0h
2	GPIO226	R/W	GPIO226 Input inversion Refer to GPIO224 for specific description.	0h
3	GPIO227	R/W	GPIO227 Input inversion Refer to GPIO224 for specific description.	0h
4	GPIO228	R/W	GPIO228 Input inversion Refer to GPIO224 for specific description.	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Input inversion Refer to GPIO224 for specific description.	0h

Field	Name	R/W	Description	Reset value
7	GPIO231	R/W	GPIO231 Input inversion Refer to GPIO224 for specific description.	0h
8	GPIO232	R/W	GPIO232 Input inversion Refer to GPIO224 for specific description.	0h
9	GPIO233	R/W	GPIO233 Input inversion Refer to GPIO224 for specific description.	0h
12:10	Reserved			1h
13	GPIO237	R/W	GPIO237 Input inversion Refer to GPIO224 for specific description.	0h
14	GPIO238	R/W	GPIO238 Input inversion Refer to GPIO224 for specific description.	0h
15	GPIO239	R/W	GPIO239 Input inversion Refer to GPIO224 for specific description.	0h
16	Reserved			1h
17	GPIO241	R/W	GPIO241 Input inversion Refer to GPIO224 for specific description.	0h
18	GPIO242	R/W	GPIO242 Input inversion Refer to GPIO224 for specific description.	0h
19	Reserved			1h
20	GPIO244	R/W	GPIO244 Input inversion Refer to GPIO224 for specific description.	0h
21	GPIO245	R/W	GPIO245 Input inversion Refer to GPIO224 for specific description.	0h
31:22	Reserved			0h

23.9.43 Port H open-drain output register (GPHODR)

Offset address: 0x3A4

Reset type: SYSRSn

Select the output mode of the general-purpose output pins: Push-pull output or open-drain output. In the push-pull output mode, writing 1 to the output data latch will drive the pin to a high level. In the open-drain mode, write 1 to the output data latch to put the output buffer in three states. In either of these two modes, writing 0 to the output data latch will drive the pin to a low level.

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Open-drain output mode 0: Push-pull output 1: Open-drain output	0h
1	GPIO225	R/W	GPIO225 Open-drain output mode Refer to GPIO224 for specific description.	0h
2	GPIO226	R/W	GPIO226 Open-drain output mode Refer to GPIO224 for specific description.	0h

Field	Name	R/W	Description	Reset value
3	GPIO227	R/W	GPIO227 Open-drain output mode Refer to GPIO224 for specific description.	0h
4	GPIO228	R/W	GPIO228 Open-drain output mode Refer to GPIO224 for specific description.	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Open-drain output mode Refer to GPIO224 for specific description.	0h
7	GPIO231	R/W	GPIO231 Open-drain output mode Refer to GPIO224 for specific description.	0h
8	GPIO232	R/W	GPIO232 Open-drain output mode Refer to GPIO224 for specific description.	0h
9	GPIO233	R/W	GPIO233 Open-drain output mode Refer to GPIO224 for specific description.	0h
12:10	Reserved			0h
13	GPIO237	R/W	GPIO237 Open-drain output mode Refer to GPIO224 for specific description.	0h
14	GPIO238	R/W	GPIO238 Open-drain output mode Refer to GPIO224 for specific description.	0h
15	GPIO239	R/W	GPIO239 Open-drain output mode Refer to GPIO224 for specific description.	0h
16	Reserved			0h
17	GPIO241	R/W	GPIO241 Open-drain output mode Refer to GPIO224 for specific description.	0h
18	GPIO242	R/W	GPIO242 Open-drain output mode Refer to GPIO224 for specific description.	0h
31:19	Reserved			0h

23.9.44 Port H analog mode selection register (GPHAMSEL)

Offset address: 0x3A8

Reset type: SYSRSn

Selects the output mode for general-purpose output pins: push-pull output or open-drain output. In push-pull output mode, writing a 1 to the output data latch drives the pin high. In open-drain mode, writing a 1 to the output data latch places the output buffer in a tri-state condition. In both output modes, writing a 0 to the output data latch drives the pin low.

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	Analog mode select for GPIO224 Select GPIO224 as the analog mode or digital mode. 0: Digital mode 1: Analog mode	1h

Field	Name	R/W	Description	Reset value
1	GPIO225	R/W	Analog mode select for GPIO225 Refer to GPIO224 for specific description.	1h
2	GPIO226	R/W	Analog mode select for GPIO226 Refer to GPIO224 for specific description.	1h
3	GPIO227	R/W	Analog mode select for GPIO227 Refer to GPIO224 for specific description.	1h
4	GPIO228	R/W	Analog mode select for GPIO228 Refer to GPIO224 for specific description.	1h
5	Reserved			1h
6	GPIO230	R/W	Analog mode select for GPIO230 Refer to GPIO224 for specific description.	1h
7	GPIO231	R/W	Analog mode select for GPIO231 Refer to GPIO224 for specific description.	1h
8	GPIO232	R/W	Analog mode select for GPIO232 Refer to GPIO224 for specific description.	1h
9	GPIO233	R/W	Analog mode select for GPIO233 Refer to GPIO224 for specific description.	1h
10	Reserved			1h
11	Reserved			1h
12	Reserved			1h
13	GPIO237	R/W	Analog mode select for GPIO237 Refer to GPIO224 for specific description.	1h
14	GPIO238	R/W	Analog mode select for GPIO238 Refer to GPIO224 for specific description.	1h
15	GPIO239	R/W	Analog mode select for GPIO239 Refer to GPIO224 for specific description.	1h
16	Reserved			1h
17	GPIO241	R/W	Analog mode select for GPIO241 Refer to GPIO224 for specific description.	1h
18	GPIO242	R/W	Analog mode select for GPIO242 Refer to GPIO224 for specific description.	1h
19	Reserved			1h
20	GPIO244	R/W	Analog mode select for GPIO244 Refer to GPIO224 for specific description.	1h
21	GPIO245	R/W	Analog mode select for GPIO245 Refer to GPIO224 for specific description.	1h
31:22	Reserved			1h

23.9.45 Port H Drive Capability Selection Register 1 (GPHDSEL1)

Offset address: 0x3AC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	GPIO224	R/W	Driver capability selection for GPIO224 When GPBFEN[GPIO224] = 1, the maximum output frequency is 50; When GPBFEN[GPIO224] = 0, the drive capability is as follows: 00: Maximum output frequency is 20 MHz (with a load of 10 pF), drive capability 2 mA 01: Maximum output frequency is 30 MHz (with a load of 30 pF), drive capability 8 mA 10: Maximum output frequency is 20 MHz (load of 20 pF), drive capability 4 mA 11: Maximum output frequency is 40 MHz (load of 40 pF), drive current is 12 mA Note: Drive capability is not affected by the GPBFEN register.	0h
3:2	GPIO225	R/W	Driver capability select for GPIO225 For a detailed description, refer to GPIO224.	0h
5:4	GPIO226	R/W	Driver capability selection for GPIO226 For a detailed description, refer to GPIO224.	0h
7:6	GPIO227	R/W	Driver capability selection for GPIO227 For a detailed description, refer to GPIO224.	0h
9:8	GPIO228	R/W	Driver capability selection for GPIO228 For a detailed description, refer to GPIO224.	0h
11:10	Reserved			0h
13:12	GPIO230	R/W	Driver capability selection for GPIO230 For a detailed description, refer to GPIO224.	0h
15:14	GPIO231	R/W	Driver capability selection for GPIO231 For a detailed description, refer to GPIO224.	0h
17:16	GPIO232	R/W	Driver capability selection for GPIO232 For a detailed description, refer to GPIO224.	0h
19:18	GPIO233	R/W	Driver capability selection for GPIO233 For a detailed description, refer to GPIO224.	0h
25:20	eserved			0h
27:26	GPIO237	R/W	Driver capability selection for GPIO237 For a detailed description, refer to GPIO224.	0h
29:28	GPIO238	R/W	Driver capability selection for GPIO238 For a detailed description, refer to GPIO224.	0h
31:30	GPIO239	R/W	Driver capability selection for GPIO239 For a detailed description, refer to GPIO224.	0h

23.9.46 Port H Driver Capability Select Register 2 (GPHDSEL2)

Offset address: 0x3B0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
3:2	GPIO241	R/W	Driver capability selection for GPIO241 When GPBFEN[GPIO241] = 1, the maximum output frequency is 50; When GPBFEN[GPIO241] = 0, the drive capability is as follows: 00: Maximum output frequency is 20 MHz (with a load of 10 pF), drive capability 2 mA 01: Maximum output frequency is 30 MHz (with a load of 30 pF), drive capability 8 mA 10: Maximum output frequency is 20 MHz (load of 20 pF), drive capability 4 mA 11: Maximum output frequency is 40 MHz (load of 40 pF), drive current is 12 mA Note: Drive capability is not affected by the GPBFEN register.	0h
5:4	GPIO242	R/W	Driver capability selection for GPIO242 For a detailed description, refer to GPIO224.	0h
7:6	Reserved			0h
9:8	GPIO244	R/W	Driver capability selection for GPIO244 For a detailed description, refer to GPIO224.	0h
11:10	GPIO245	R/W	Driver capability selection for GPIO245 For a detailed description, refer to GPIO224.	0h
31:12	Reserved			0h

23.9.47 Port H Maximum Output Frequency Enable Register (GPHFEN)

Offset address: 0x3B4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Maximum Output Frequency Enable 0: For details on the maximum output frequency of GPIO224, please refer to GPHDSEL1[GPIO224] 1: The maximum output frequency of GPIO224 is 50 MHz	0h
1	GPIO225	R/W	GPIO225 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
2	GPIO226	R/W	GPIO226 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
3	GPIO227	R/W	GPIO227 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
4	GPIO228	R/W	GPIO228 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
7	GPIO231	R/W	GPIO231 Maximum Output Frequency Enable	0h

Field	Name	R/W	Description	Reset value
			For a detailed description, refer to GPIO224.	
8	GPIO232	R/W	GPIO232 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
9	GPIO233	R/W	GPIO233 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
12:10	Reserved			0h
13	GPIO237	R/W	GPIO237 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
14	GPIO238	R/W	GPIO238 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
15	GPIO239	R/W	GPIO239 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
16	Reserved			0h
17	GPIO241	R/W	GPIO241 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
18	GPIO242	R/W	GPIO242 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
19	Reserved			0h
20	GPIO244	R/W	GPIO244 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
21	GPIO245	R/W	GPIO245 Maximum Output Frequency Enable For a detailed description, refer to GPIO224.	0h
31:22	Reserved			0h

23.9.48 Port H multiplexed high-order register 1 (GPHGMUX1)

Offset address: 0x3C0

Reset type: SYSRSn

This register configures the high-order portion (GPIO224 to GPIO230) of the GPIOH pin multiplexing function. For configuration details regarding each pin, please refer to the data sheet.

Before changing the bit fields in this register, the pins must be configured to GPIO mode using the GPHMUX1 register.

Field	Name	R/W	Description	Reset value
1:0	GPIO224	R/W	GPIO224 peripheral multiplexer configuration in the high 2 bits	0h
3:2	Reserved			0h
5:4	GPIO226	R/W	GPIO226 peripheral multiplexer configuration in the high 2 bits	0h
7:6	GPIO227	R/W	GPIO227 peripheral multiplexer configuration in the high 2 bits	0h
9:8	GPIO228	R/W	GPIO228 peripheral multiplexer configuration in the high 2 bits	0h

Field	Name	R/W	Description	Reset value
11:10	Reserved			0h
13:12	GPIO230	R/W	GPIO230 peripheral multiplexer configuration in the high 2 bits	0h
31:14	Reserved			0h

23.9.49 Port H Multiplexing High-Order Register 2 (GPHGMUX2)

Offset address: 0x3C4

Reset type: SYSRSn

This register configures the high-order portion (GPIO241 to GPIO245) of the GPIOH pin multiplexing function. For configuration details regarding each pin, please refer to the data sheet.

Before changing the bit fields in this register, the pins must be configured to GPIO mode using the GPHMUX2 register.

Field	Name	R/W	Description	Reset value
3:0	Reserved			0h
5:4	GPIO242	R/W	GPIO242 peripheral multiplexer configuration in the high 2 bits	0h
31:6	Reserved			0h

23.9.50 Port H lock register (GPHLOCK)

Offset address: 0x3F8

Reset type: SYSRSn

Lock the configuration of GPIOH pins (from GPIO224 to GPIO245). This register will be written to the GPHINV register.

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Configuration lock 0: GPIO224 configuration not locked 1: GPIO224 configuration locked	0h
1	GPIO225	R/W	GPIO225 Configuration lock Refer to GPIO224 for specific description.	0h
2	GPIO226	R/W	GPIO226 Configuration lock Refer to GPIO224 for specific description.	0h
3	GPIO227	R/W	GPIO227 Configuration lock Refer to GPIO224 for specific description.	0h
4	GPIO228	R/W	GPIO228 Configuration lock Refer to GPIO224 for specific description.	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Configuration lock Refer to GPIO224 for specific description.	0h
7	GPIO231	R/W	GPIO231 Configuration lock Refer to GPIO224 for specific description.	0h

Field	Name	R/W	Description	Reset value
8	GPIO232	R/W	GPIO232 Configuration lock Refer to GPIO224 for specific description.	0h
9	GPIO233	R/W	GPIO233 Configuration lock Refer to GPIO224 for specific description.	0h
12:10	Reserved			0h
13	GPIO237	R/W	GPIO237 Configuration lock Refer to GPIO224 for specific description.	0h
14	GPIO238	R/W	GPIO238 Configuration lock Refer to GPIO224 for specific description.	0h
15	GPIO239	R/W	GPIO239 Configuration lock Refer to GPIO224 for specific description.	0h
16	Reserved			0h
17	GPIO241	R/W	GPIO241 Configuration lock Refer to GPIO224 for specific description.	0h
18	GPIO242	R/W	GPIO242 Configuration lock Refer to GPIO224 for specific description.	0h
19	Reserved			0h
20	GPIO244	R/W	GPIO244 Configuration lock Refer to GPIO224 for specific description.	0h
21	GPIO245	R/W	GPIO245 Configuration lock Refer to GPIO224 for specific description.	0h
31:22	Reserved			0h

23.9.51 Port H committing lock register (GPHCR)

Offset address: 0x3FC

Reset type: SYSRSn

This bit commits locking for the configuration of the GPIOH pins (from GPIO224 to GPIO245). Once this bit is set, it can only be cleared through reset. This register will write to the GPHLOCK[GPIOy] register.

Field	Name	R/W	Description	Reset value
0	GPIO224	R/WSO	GPIO224 lock is committed The bit commits locking of GPHLOCK[GPIO224]. 0: The configuration of locking GPHLOCK[GPIO224] is not committed 1: The configuration of locking GPHLOCK[GPIO224] is committed	0h
1	GPIO225	R/WSO	GPIO225 lock is committed Refer to GPIO224 for specific description.	0h
2	GPIO226	R/WSO	GPIO226 lock is committed Refer to GPIO224 for specific description.	0h
3	GPIO227	R/WSO	GPIO227 lock is committed Refer to GPIO224 for specific description.	0h

Field	Name	R/W	Description	Reset value
4	GPIO228	R/WSO	GPIO228 lock is committed Refer to GPIO224 for specific description.	0h
5	Reserved			0h
6	GPIO230	R/WSO	GPIO230 lock is committed Refer to GPIO224 for specific description.	0h
7	GPIO231	R/WSO	GPIO231 lock is committed Refer to GPIO224 for specific description.	0h
8	GPIO232	R/WSO	GPIO232 lock is committed Refer to GPIO224 for specific description.	0h
9	GPIO233	R/WSO	GPIO233 lock is committed Refer to GPIO224 for specific description.	0h
12:10	Reserved			0h
13	GPIO237	R/WSO	GPIO237 lock is committed Refer to GPIO224 for specific description.	0h
14	GPIO238	R/WSO	GPIO238 lock is committed Refer to GPIO224 for specific description.	0h
15	GPIO239	R/WSO	GPIO239 lock is committed Refer to GPIO224 for specific description.	0h
16	Reserved			0h
17	GPIO241	R/WSO	GPIO241 lock is committed Refer to GPIO224 for specific description.	0h
18	GPIO242	R/WSO	GPIO242 lock is committed Refer to GPIO224 for specific description.	0h
19	Reserved			0h
20	GPIO244	R/WSO	GPIO244 lock is committed Refer to GPIO224 for specific description.	0h
21	GPIO245	R/WSO	GPIO245 lock is committed Refer to GPIO224 for specific description.	0h
31:22	Reserved			0h

23.9.52 Port A data register (GPADAT)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	GPIOy Read/Write Data Bit (y=0…31) When reading a bit in this register, the corresponding pin will return the input level after qualification and inversion (inversion is optional). The input value is always readable, without considering whether the pin is configured as GPIO output or peripheral signal function. When the system is reset, all output data latches will be cleared to zero.	0h

Field	Name	R/W	Description	Reset value
			Writing to this register will clear or set the corresponding output data latch. The specific operation of the drive pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a high or low level; When the pin is not configured as GPIO output function, the write value will be latched and ignored. The latched value will be driven to the pin only when this pin is subsequently reconfigured as GPIO output function. Note: Since there is a delay between the read-out value and the write value, operation in the sequence of reading – modifying - writing may destroy the state of the output data latch. The GPADAT register indicates the state of the pins rather than the state of the latch, which means that the data register reflects the actual pin value. However, there is a delay in the process from writing the GPADAT to feeding back the updated pin value to the register. When using this data register to change the state of GPIO pins in subsequent program statement, problems may be caused. Therefore, when using the GPADAT registers to change the level of GPIO output pins, check whether the modification of the levels of other pins is interfered with. Solutions: (1) Place some NOP between instructions; (2) Use the GPASET/GPACLR/GPATOGGLE register to replace the GPADAT register. Due to the characteristic of these three types of registers that writing 0 is invalid and reading returns 0, the bits that only need to be changed can be specified without interfering with other bits currently being changed. These three types of registers can be used to safely control the output data latches.	

23.9.53 Port A setup register (GPASET)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	Set the GPIOy output data latch (y=0…31) All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will set the corresponding output data latch to a high level. The specific operation of the driver pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a high level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven.	0h

Field	Name	R/W	Description	Reset value
			The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	

23.9.54 Port A clear register (GPACLR)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	Clear the GPIOy output data latch (y=0…31) All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will clear the corresponding output data latch. The specific operation of the driver pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a low level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h

23.9.55 Port A switch register (GPATOGGLE)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
y	GPIOy	R/W	Toggle the GPIOy output data latch (y=0…31) All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will switch the value of corresponding output data latch, and drive the pin in the opposite direction. The specific operation is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to an opposite level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h

23.9.56 Port B data register (GPBDAT)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	GPIO32 Read/Write Data Bit When reading a bit in this register, the corresponding pin will return the input level after qualification and inversion (inversion is optional). The input value is always readable, without considering whether the pin is configured as GPIO output or peripheral signal function.	0h

Field	Name	R/W	Description	Reset value
			When the system is reset, all output data latches will be cleared to zero. Writing to this register will clear or set the corresponding output data latch. The specific operation of the drive pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a high or low level; When the pin is not configured as GPIO output function, the write value will be latched and ignored. The latched value will be driven to the pin only when this pin is subsequently reconfigured as GPIO output function. Note: Since there is a delay between the read-out value and the write value, operation in the sequence of reading – modifying - writing may destroy the state of the output data latch. The GPBDAT register indicates the state of the pins rather than the state of the latch, which means that the data register reflects the actual pin value. However, there is a delay in the process from writing the GPBDAT to feeding back the updated pin value to the register. When using this data register to change the state of GPIO pins in subsequent program statement, problems may be caused. Therefore, when using the GPBDAT registers to change the level of GPIO output pins, check whether the modification of the levels of other pins is interfered with. Solutions: (1) Place some NOP between instructions; (2) Use the GPBSET/GPBCLR/GPBTOGGLE register to replace the GPBDAT register. Due to the characteristic of these three types of registers that writing 0 is invalid and reading returns 0, the bits that only need to be changed can be specified without interfering with other bits currently being changed. These three types of registers can be used to safely control the output data latches.	
1	GPIO33	R/W	GPIO33 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
2	GPIO34	R/W	GPIO34 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
3	GPIO35	R/W	GPIO35 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
4	Reserved			0h
5	GPIO37	R/W	GPIO37 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
7:6	Reserved			0h
8	GPIO40	R/W	GPIO40 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
9	GPIO41	R/W	GPIO41 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h

Field	Name	R/W	Description	Reset value
10	GPIO42	R/W	GPIO42 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
11	GPIO43	R/W	GPIO43 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
12	GPIO44	R/W	GPIO44 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
13	GPIO45	R/W	GPIO45 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
14	GPIO46	R/W	GPIO46 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
15	Reserved			0h
16	GPIO48	R/W	GPIO48 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
17	GPIO49	R/W	GPIO49 Read/Write Data Bit For a detailed description, refer to GPIO32.	0h
31:18	Reserved			0h

23.9.57 Port B setup register (GPBSET)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	Set the GPIO32 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will set the corresponding output data latch to a high level. The specific operation of the driver pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a high level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h
1	GPIO33	R/W	Set the GPIO33 output data latch For a detailed description, refer to GPIO32.	0h
2	GPIO34	R/W	Set the GPIO34 output data latch For a detailed description, refer to GPIO32.	0h
3	GPIO35	R/W	Set the GPIO35 output data latch For a detailed description, refer to GPIO32.	0h
4	Reserved			0h
5	GPIO37	R/W	Set the GPIO37 output data latch For a detailed description, refer to GPIO32.	0h
7:6	Reserved			0h

Field	Name	R/W	Description	Reset value
8	GPIO40	R/W	Set the GPIO40 output data latch For a detailed description, refer to GPIO32.	0h
9	GPIO41	R/W	Set the GPIO41 output data latch For a detailed description, refer to GPIO32.	0h
10	GPIO42	R/W	Set the GPIO42 output data latch For a detailed description, refer to GPIO32.	0h
11	GPIO43	R/W	Set the GPIO43 output data latch For a detailed description, refer to GPIO32.	0h
12	GPIO44	R/W	Set the GPIO44 output data latch For a detailed description, refer to GPIO32.	0h
13	GPIO45	R/W	Set the GPIO45 output data latch For a detailed description, refer to GPIO32.	0h
14	GPIO46	R/W	Set the GPIO46 output data latch For a detailed description, refer to GPIO32.	0h
15	Reserved			0h
16	GPIO48	R/W	Set the GPIO48 output data latch For a detailed description, refer to GPIO32.	0h
17	GPIO49	R/W	Set the GPIO49 output data latch For a detailed description, refer to GPIO32.	0h
31:18	Reserved			0h

23.9.58 Port B clear register (GPBCLR)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	Clear the GPIO32 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will clear the corresponding output data latch. The specific operation of the driver pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a low level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h
1	GPIO33	R/W	Clear the GPIO33 output data latch For a detailed description, refer to GPIO32.	0h
2	GPIO34	R/W	Clear the GPIO34 output data latch For a detailed description, refer to GPIO32.	0h
3	GPIO35	R/W	Clear the GPIO35 output data latch For a detailed description, refer to GPIO32.	0h
4	Reserved			0h

Field	Name	R/W	Description	Reset value
5	GPIO37	R/W	Clear the GPIO37 output data latch For a detailed description, refer to GPIO32.	0h
7:6	Reserved			0h
8	GPIO40	R/W	Clear the GPIO40 output data latch For a detailed description, refer to GPIO32.	0h
9	GPIO41	R/W	Clear the GPIO41 output data latch For a detailed description, refer to GPIO32.	0h
10	GPIO42	R/W	Clear the GPIO42 output data latch For a detailed description, refer to GPIO32.	0h
11	GPIO43	R/W	Clear the GPIO43 output data latch For a detailed description, refer to GPIO32.	0h
12	GPIO44	R/W	Clear the GPIO44 output data latch For a detailed description, refer to GPIO32.	0h
13	GPIO45	R/W	Clear the GPIO45 output data latch For a detailed description, refer to GPIO32.	0h
14	GPIO46	R/W	Clear the GPIO46 output data latch For a detailed description, refer to GPIO32.	0h
15	Reserved			0h
16	GPIO48	R/W	Clear the GPIO48 output data latch For a detailed description, refer to GPIO32.	0h
17	GPIO49	R/W	Clear the GPIO49 output data latch For a detailed description, refer to GPIO32.	0h
31:18	Reserved			0h

23.9.59 Port B switch register (GPBTOGGLE)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO32	R/W	Toggle the GPIO32 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will switch the value of corresponding output data latch, and drive the pin in the opposite direction. The specific operation is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to an opposite level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h
1	GPIO33	R/W	Toggle the GPIO33 output data latch For a detailed description, refer to GPIO32.	0h

Field	Name	R/W	Description	Reset value
2	GPIO34	R/W	Toggle the GPIO34 output data latch For a detailed description, refer to GPIO32.	0h
3	GPIO35	R/W	Toggle the GPIO35 output data latch For a detailed description, refer to GPIO32.	0h
4	Reserved			0h
5	GPIO37	R/W	Toggle the GPIO37 output data latch For a detailed description, refer to GPIO32.	0h
7:6	Reserved			0h
8	GPIO40	R/W	Toggle the GPIO40 output data latch For a detailed description, refer to GPIO32.	0h
9	GPIO41	R/W	Toggle the GPIO41 output data latch For a detailed description, refer to GPIO32.	0h
10	GPIO42	R/W	Toggle the GPIO42 output data latch For a detailed description, refer to GPIO32.	0h
11	GPIO43	R/W	Toggle the GPIO43 output data latch For a detailed description, refer to GPIO32.	0h
12	GPIO44	R/W	Toggle the GPIO44 output data latch For a detailed description, refer to GPIO32.	0h
13	GPIO45	R/W	Toggle the GPIO45 output data latch For a detailed description, refer to GPIO32.	0h
14	GPIO46	R/W	Toggle the GPIO46 output data latch For a detailed description, refer to GPIO32.	0h
15	Reserved			0h
16	GPIO48	R/W	Toggle the GPIO48 output data latch For a detailed description, refer to GPIO32.	0h
17	GPIO49	R/W	Toggle the GPIO49 output data latch For a detailed description, refer to GPIO32.	0h
31:18	Reserved			0h

23.9.60 Port H data register (GPHDAT)

Offset address: 0x70

Reset type: SYSRSn

Read a field in this register and the input level of the corresponding IO pin will be returned after qualification and (optional) inversion. In digital mode, these pins only support input function.

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	GPIO224 Read/Write Data Bit	0h
1	GPIO225	R/W	GPIO225 Read/Write Data Bit	0h
2	GPIO226	R/W	GPIO226 Read/Write Data Bit	0h
3	GPIO227	R/W	GPIO227 Read/Write Data Bit	0h

Field	Name	R/W	Description	Reset value
4	GPIO228	R/W	GPIO228 Read/Write Data Bit	0h
5	Reserved			0h
6	GPIO230	R/W	GPIO230 Read/Write Data Bit	0h
7	GPIO231	R/W	GPIO231 Read/Write Data Bit	0h
8	GPIO232	R/W	GPIO232 Read/Write Data Bit	0h
9	GPIO233	R/W	GPIO233 Read/Write Data Bit	0h
12:10	Reserved			0h
13	GPIO237	R/W	GPIO237 Read/Write Data Bit	0h
14	GPIO238	R/W	GPIO238 Read/Write Data Bit	0h
15	GPIO239	R/W	GPIO239 Read/Write Data Bit	0h
16	Reserved			0h
17	GPIO241	R/W	GPIO241 Read/Write Data Bit	0h
18	GPIO242	R/W	GPIO242 Read/Write Data Bit	0h
19	Reserved			0h
20	GPIO244	R/W	GPIO244 Read/Write Data Bit	0h
21	GPIO245	R/W	GPIO245 Read/Write Data Bit	0h
31	Reserved			0h

23.9.61 Port H setup register (GPHSET)

Offset address: 0x74

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	Set the GPIO224 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will set the corresponding output data latch to a high level. The specific operation of the driver pin is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to a high level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h
1	GPIO225	R/W	Set the GPIO225 output data latch For a detailed description, refer to GPIO224.	0h
2	GPIO226	R/W	Set the GPIO226 output data latch For a detailed description, refer to GPIO224.	0h

Field	Name	R/W	Description	Reset value
3	GPIO227	R/W	Set the GPIO227 output data latch For a detailed description, refer to GPIO224.	0h
4	GPIO228	R/W	Set the GPIO228 output data latch For a detailed description, refer to GPIO224.	
5	Reserved			0h
6	GPIO230	R/W	Set the GPIO230 output data latch For a detailed description, refer to GPIO224.	0h
7	GPIO231	R/W	Set the GPIO231 output data latch For a detailed description, refer to GPIO224.	
8	GPIO232	R/W	Set the GPIO232 output data latch For a detailed description, refer to GPIO224.	
9	GPIO233	R/W	Set the GPIO233 output data latch For a detailed description, refer to GPIO224.	
12:10	Reserved			0h
13	GPIO237	R/W	Set the GPIO237 output data latch For a detailed description, refer to GPIO224.	0h
14	GPIO238	R/W	Set the GPIO238 output data latch For a detailed description, refer to GPIO224.	0h
15	GPIO239	R/W	Set the GPIO239 output data latch For a detailed description, refer to GPIO224.	0h
16	Reserved			0h
17	GPIO241	R/W	Set the GPIO241 output data latch For a detailed description, refer to GPIO224.	0h
18	GPIO242	R/W	Set the GPIO242 output data latch For a detailed description, refer to GPIO224.	0h
19	Reserved			0h
20	GPIO244	R/W	Set the GPIO244 output data latch For a detailed description, refer to GPIO224.	0h
21	GPIO245	R/W	Set the GPIO245 output data latch For a detailed description, refer to GPIO224.	0h
31:22	Reserved			0h

23.9.62 Port H clear register (GPHCLR)

Offset address: 0x78

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	Clear the GPIO224 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will clear the corresponding output data latch. The specific operation of the driver pin is as follows:	0h

Field	Name	R/W	Description	Reset value
			When the pin is configured as GPIO output function and enabled, this pin will be driven to a low level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	
1	GPIO225	R/W	Clear the GPIO225 output data latch For a detailed description, refer to GPIO224.	0h
2	GPIO226	R/W	Clear the GPIO226 output data latch For a detailed description, refer to GPIO224.	0h
3	GPIO227	R/W	Clear the GPIO227 output data latch For a detailed description, refer to GPIO224.	0h
4	GPIO228	R/W	Clear the GPIO228 output data latch For a detailed description, refer to GPIO224.	
5	Reserved			0h
6	GPIO230	R/W	Clear the GPIO230 output data latch For a detailed description, refer to GPIO224.	0h
7	GPIO231	R/W	Clear the GPIO231 output data latch For a detailed description, refer to GPIO224.	
8	GPIO232	R/W	Clear the GPIO232 output data latch For a detailed description, refer to GPIO224.	
9	GPIO233	R/W	Clear the GPIO233 output data latch For a detailed description, refer to GPIO224.	
12:10	Reserved			0h
13	GPIO237	R/W	Clear the GPIO237 output data latch For a detailed description, refer to GPIO224.	0h
14	GPIO238	R/W	Clear the GPIO238 output data latch For a detailed description, refer to GPIO224.	0h
15	GPIO239	R/W	Clear the GPIO239 output data latch For a detailed description, refer to GPIO224.	0h
16	Reserved			0h
17	GPIO241	R/W	Clear the GPIO241 output data latch For a detailed description, refer to GPIO224.	0h
18	GPIO242	R/W	Clear the GPIO242 output data latch For a detailed description, refer to GPIO224.	0h
19	Reserved			0h
20	GPIO244	R/W	Clear the GPIO244 output data latch For a detailed description, refer to GPIO224.	0h
21	GPIO245	R/W	Clear the GPIO245 output data latch For a detailed description, refer to GPIO224.	0h
31:22	Reserved			0h

23.9.63 Port H switch register (GPHTOGGLE)

Offset address: 0x7C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	Toggle the GPIO32 output data latch All bits of this register are always read as 0, and writing 0 is invalid. Writing 1 to this register will switch the value of corresponding output data latch, and drive the pin in the opposite direction. The specific operation is as follows: When the pin is configured as GPIO output function and enabled, this pin will be driven to an opposite level; When the pin is not configured as GPIO output function, writing 1 will be latched, but this pin will not be driven. The latched value will be driven to the pin only when this pin is subsequently configured as GPIO output function.	0h
1	GPIO225	R/W	Toggle the GPIO225 output data latch For a detailed description, refer to GPIO224.	0h
2	GPIO226	R/W	Toggle the GPIO226 output data latch For a detailed description, refer to GPIO224.	0h
3	GPIO227	R/W	Toggle the GPIO227 output data latch For a detailed description, refer to GPIO224.	0h
4	GPIO228	R/W	Toggle the GPIO228 output data latch For a detailed description, refer to GPIO224.	
5	Reserved			0h
6	GPIO230	R/W	Toggle the GPIO230 output data latch For a detailed description, refer to GPIO224.	0h
7	GPIO231	R/W	Toggle the GPIO231 output data latch For a detailed description, refer to GPIO224.	
8	GPIO232	R/W	Toggle the GPIO232 output data latch For a detailed description, refer to GPIO224.	
9	GPIO233	R/W	Toggle the GPIO233 output data latch For a detailed description, refer to GPIO224.	
12:10	Reserved			0h
13	GPIO237	R/W	Toggle the GPIO237 output data latch For a detailed description, refer to GPIO224.	0h
14	GPIO238	R/W	Toggle the GPIO238 output data latch For a detailed description, refer to GPIO224.	0h
15	GPIO239	R/W	Toggle the GPIO239 output data latch For a detailed description, refer to GPIO224.	0h
16	Reserved			0h
17	GPIO241	R/W	Toggle the GPIO241 output data latch For a detailed description, refer to GPIO224.	0h
18	GPIO242	R/W	Toggle the GPIO242 output data latch For a detailed description, refer to GPIO224.	0h

Field	Name	R/W	Description	Reset value
19	Reserved			0h
20	GPIO244	R/W	Toggle the GPIO244 output data latch For a detailed description, refer to GPIO224.	0h
21	GPIO245	R/W	Toggle the GPIO245 output data latch For a detailed description, refer to GPIO224.	0h
31:22	Reserved			0h

23.9.64 Port A Data Read Register (GPADAT_R)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	DATA	R	On read, returns the contents written to the GPADAT register; write operations to this register are invalid.	0h

23.9.65 Port B Data Read Register (GPBDAT_R)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	DATA	R	On read, returns the contents written to the GPBDAT register; write operations to this register are invalid.	0h

23.9.66 Port H Data Read Register (GPHDAT_R)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	DATA	R	On read, returns the contents written to the GPHDAT register; write operations to this register are invalid.	0h

24 X-BAR

24.1 Introduction

X-BAR provides a flexible cross connection structure, and it can implement fast transmission of data between input, output, and internal resources. According to the position of the X-BAR receiving signals, it is divided into Input X-BAR, Output X-BAR, FLB X BAR, and PWM X-BAR.

24.2 Main characteristics

(1) Flexible cross connection structure

(2) Input X-BAR: Input the external signal into the device through GPIO pins

(3) Output X-BAR: Output the internal signal from the device to GPIO pins

(4) FLB X BAR: Transmit the signal to the FLB module

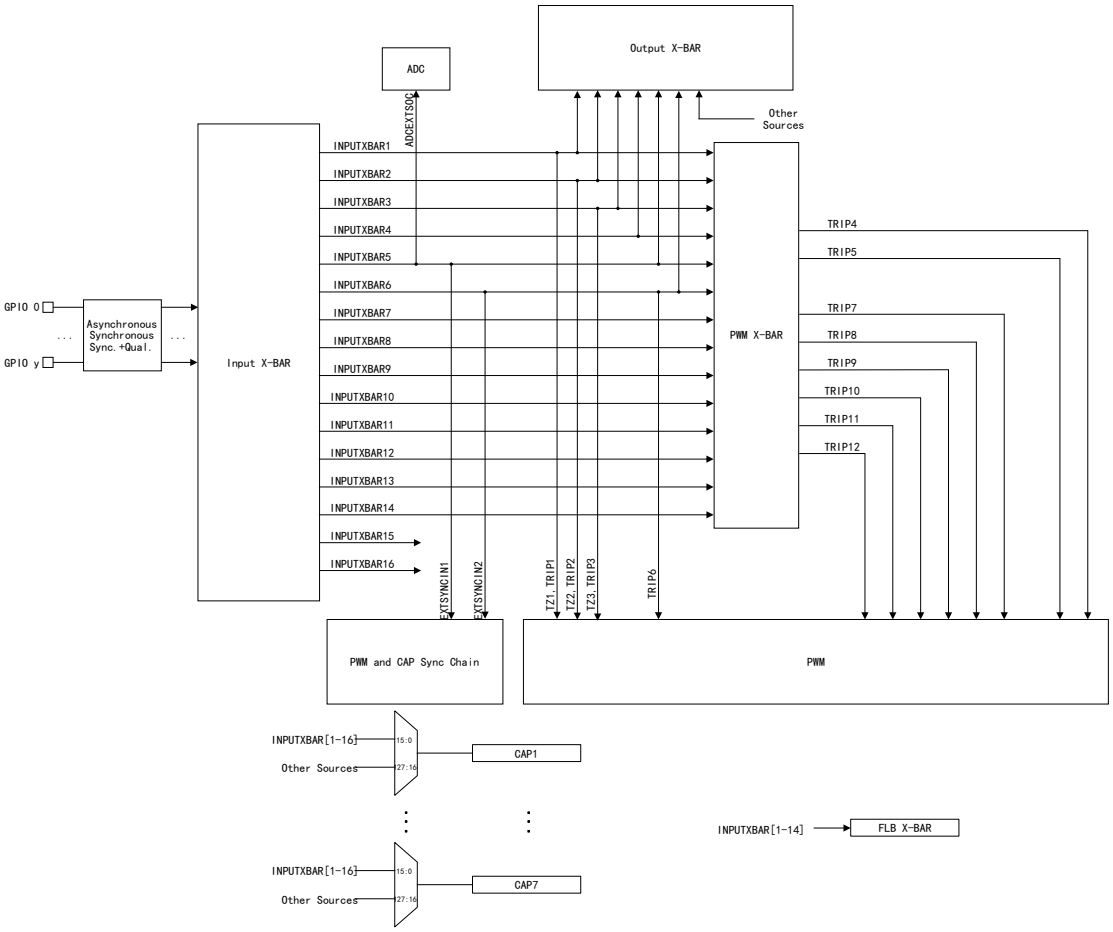
(5) PWM X-BAR: Transmit the signal to the PWM module

24.3 Functional description

24.3.1 Input X-BAR

24.3.1.1 Structure block diagram

Figure 24 Input X-BAR Structure Block Diagram



24.3.1.2 Signal connection

Input X-BAR consists of 16 INPUT signals, and it can access each GPIO pin (including digital input pins of AIO, digital output input pins of AGPIO), and input the external signals to one or more different IP modules through GPIO pins. These IP modules include external interrupts, PWM, ADC, and CAP. As only one available GPIO is needed for flexible signal routing, Input X-BAR can be used to alleviate some limitations of peripheral multiplexing. Input X-BAR only connects the signals in the input buffer to the selected target IP module, and the operation of Input X-BAR is unaffected by GPIO multiplexing function selection. By using the Input X-BAR, the output signal of one peripheral can also be routed to another peripheral, for example, using CAP to measure PWM output during frequency test.

Through the INPUTxSELECT register, GPIO of each INPUT signal can be

configured. For more information on INPUT signal configuration, please refer to XBAR_INPUT Register Bank. The target IP modules that can be selected for each INPUT signal are shown in the following table:

Table 75 Target IP Module of Input X-BAR

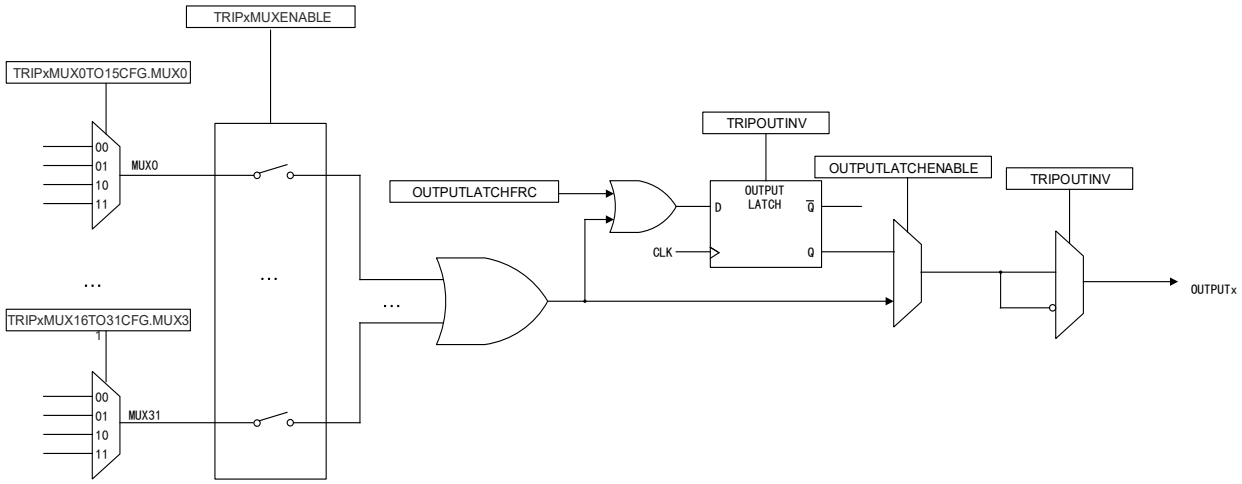
INPUTx	Target IP module
INPUT1	Output X-BAR, PWM X-BAR, CAPx, PWM[TZ1,TRIP1], FLB
INPUT2	Output X-BAR, PWM X-BAR, CAPx, PWM[TZ2,TRIP2], FLB
INPUT3	Output X-BAR, PWM X-BAR, CAPx, PWM[TZ3,TRIP3], FLB
INPUT4	Output X-BAR, PWM X-BAR, CAPx, FLB, CPU XINT1
INPUT5	Output X-BAR, PWM X-BAR, CAPx, ADCEXTSOC, EXTSYNCIN1, FLB, CPU XINT2
INPUT6	Output X-BAR, PWM X-BAR, CAPx, PWM[TRIP2], PWM[TRIP6], EXTSYNCIN2, FLB, CPU XINT3
INPUT7	PWM X-BAR, CAPx, FLB
INPUT8	PWM X-BAR, CAPx, FLB
INPUT9	PWM X-BAR, CAPx, FLB
INPUT10	PWM X-BAR, CAPx, FLB
INPUT11	PWM X-BAR, CAPx, FLB
INPUT12	PWM X-BAR, CAPx, FLB
INPUT13	PWM X-BAR, CAPx, FLB, CPU XINT4
INPUT14	PWM X-BAR, CAPx, FLB, CPU XINT5
INPUT15	CAPx
INPUT16	CAPx

24.3.2 Output X-BAR

24.3.2.1 Structure block diagram

The structure of Output X-BAR is similar to that of PWM X-BAR, except for the output latch.

Figure 25 Output X-BAR Structure Block Diagram



24.3.2.2 Signal transmission

The Output X-BAR consists of 8 OUTPUT, and it can receive signals from the device inside and route them to GPIO pins. Each OUTPUT occupies at least one position on the GPIO multiplexer, represented by OUTPUTXBARx. A single signal can be selected or up to 32 signals of logic or connection can be used.

Signal transmission steps:

(1) Select the signal needing to be transmitted to GPIO: For each OUTPUTXBARx, select a signal from the OUTPUTxMUX0TO15CFG or OUTPUTxMUX16TO31CFG register as the input of each multiplexer (there are a total of 32 multiplexers).

(2) Enable the multiplexer: Enable the multiplexer in the OUTPUTxMUXENABLE register. Before the OUTPUTx signal transmitted to the GPIO module, logic OR operation is performed on all MUX.

(3) (Optional) Signal inversion: Invert the signals in the OUTPUTINV register.

GPIO can recognize signals only when the correct OUTPUTx multiplexing option is configured in the GPxMUX1/2 and GPxGMUX1/2 registers.

Table 76 Output X-BAR Multiplexing Configuration

MUX	0	1	2	3
MUX0	COMP1.CTRIPOUTH	COMP1.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT1	CAP1OUT
MUX1	COMP1.CTRIPOUTL	INPUTXBAR1	FLB1_OUT4	ADCCEVT1
MUX2	COMP2.CTRIPOUTH	COMP2.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT2	CAP2OUT
MUX3	COMP2.CTRIPOUTL	INPUTXBAR2	FLB1_OUT5	ADCCEVT2
MUX4	COMP3.CTRIPOUTH	COMP3.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT3	CAP3OUT

MUX	0	1	2	3
MUX5	COMP3.CTRIPOUTL	INPUTXBAR3	FLB2_OUT4	ADCCEVT3
MUX6	COMP4.CTRIPOUTH	COMP4.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT4	Reserved
MUX7	COMP4.CTRIPOUTL	INPUTXBAR4	FLB2_OUT5	ADCCEVT4
MUX8	Reserved	Reserved	Reserved	Reserved
MUX9	Reserved	INPUTXBAR5	FLB3_OUT4	Reserved
MUX10	Reserved	Reserved	Reserved	Reserved
MUX11	Reserved	INPUTXBAR6	FLB3_OUT5	Reserved
MUX12	Reserved	Reserved	Reserved	Reserved
MUX13	Reserved	ADCSOCAO	FLB4_OUT4	Reserved
MUX14	Reserved	Reserved	Reserved	EXTSYNCOUT
MUX15	Reserved	ADCSOCBO	FLB4_OUT5	Reserved
MUX16	Reserved	Reserved	Reserved	Reserved
MUX17	Reserved	Reserved	Reserved	Reserved
MUX18	Reserved	Reserved	Reserved	Reserved
MUX19	Reserved	Reserved	Reserved	Reserved
MUX20	Reserved	Reserved	Reserved	Reserved
MUX21	Reserved	Reserved	Reserved	Reserved
MUX22	Reserved	Reserved	Reserved	Reserved
MUX23	Reserved	Reserved	Reserved	Reserved

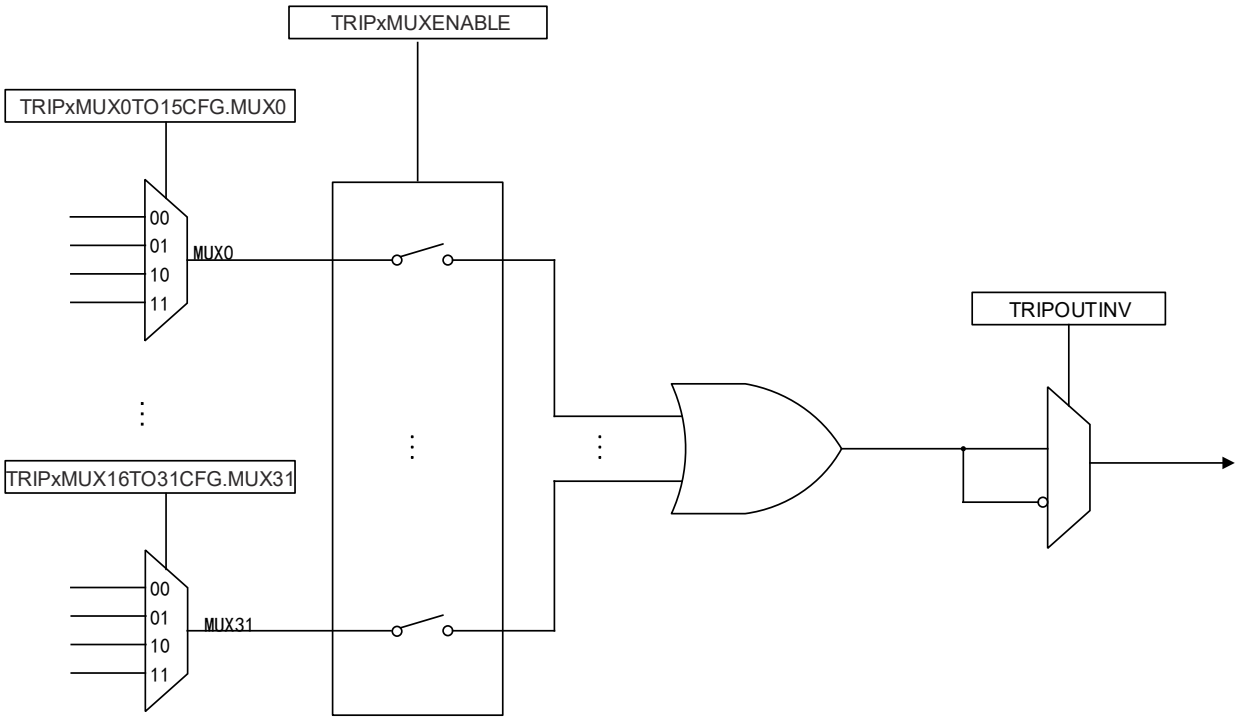
Note: The "Reserved" signals are not usable, and all unused and reserved signals are set to 0.

24.3.3 PWM X-BAR

24.3.3.1 Structure block diagram

The structure of Output X-BAR is similar to that of PWM X-BAR, except for the output latch.

Figure 26 PWM X-BAR Structure Block Diagram



24.3.3.2 Signal transmission

The PWM X-BAR consists of 8 outputs, and it can route signals to the PWM module, connect with the digital compare (DC) submodule of the PWM module, and perform operations such as tripping and synchronization. For more information on DC submodules, please refer to PWM. Each output is represented by TRIPx, and a single signal can be selected on the MUX, or up to 32 signals of logic OR connection can be used.

Signal transmission steps:

(1) Select the signal needing to be transmitted to PWM: For each TRIPx, select a signal from the TRIPxMUX0TO15CFG or TRIPxMUX16TO31CFG register as the input of each MUX (there are a total of 32 MUX).

(2) Enable the multiplexer: Enable the multiplexer in the TRIPxMUXENABLE register. Before the TRIPx signal transmitted to the PWM module, logic OR operation is performed on all multiplexers.

(3) (Optional) Signal inversion: Invert the signals in the TRIPOUTINV register.

Table 77 PWM X-BAR Multiplexing Configuration

MUX	0	1	2	3
MUX0	COMP1.CTRIPH	COMP1.CTRIPH_OR_CTRIPL	ADCAEVT1	CAP1OUT

MUX	0	1	2	3
MUX1	COMP1.CTRIPL	INPUTXBAR1	FLB1_OUT4	ADCCEVT1
MUX2	COMP2.CTRIPH	COMP2.CTRIPH_OR_CTRIPL	ADCAEVT2	CAP2OUT
MUX3	COMP2.CTRIPL	INPUTXBAR2	FLB1_OUT5	ADCCEVT2
MUX4	COMP3.CTRIPH	COMP3.CTRIPH_OR_CTRIPL	ADCAEVT3	CAP3OUT
MUX5	COMP3.CTRIPL	INPUTXBAR3	FLB2_OUT4	ADCCEVT3
MUX6	COMP4.CTRIPH	COMP4.CTRIPH_OR_CTRIPL	ADCAEVT4	Reserved
MUX7	COMP4.CTRIPL	INPUTXBAR4	FLB2_OUT5	ADCCEVT4
MUX8	Reserved	Reserved	Reserved	Reserved
MUX9	Reserved	INPUTXBAR5	FLB3_OUT4	Reserved
MUX10	Reserved	Reserved	Reserved	Reserved
MUX11	Reserved	INPUTXBAR6	FLB3_OUT5	Reserved
MUX12	Reserved	Reserved	Reserved	Reserved
MUX13	Reserved	ADCSOCAO	FLB4_OUT4	Reserved
MUX14	Reserved	Reserved	Reserved	EXTSYNCOUT
MUX15	Reserved	ADCSOCBO	FLB4_OUT5	Reserved
MUX16	Reserved	Reserved	Reserved	Reserved
MUX17	Reserved	INPUTXBAR7	Reserved	Reserved
MUX18	Reserved	Reserved	Reserved	Reserved
MUX19	Reserved	INPUTXBAR8	Reserved	Reserved
MUX20	Reserved	Reserved	Reserved	Reserved
MUX21	Reserved	INPUTXBAR9	Reserved	Reserved
MUX22	Reserved	Reserved	Reserved	Reserved
MUX23	Reserved	INPUTXBAR10	Reserved	Reserved
MUX24	Reserved	Reserved	Reserved	Reserved
MUX25	Reserved	INPUTXBAR11	Reserved	Reserved
MUX26	Reserved	Reserved	Reserved	Reserved
MUX27	Reserved	INPUTXBAR12	Reserved	Reserved
MUX28	Reserved	Reserved	Reserved	Reserved
MUX29	Reserved	INPUTXBAR13	Reserved	Reserved
MUX30	Reserved	Reserved	Reserved	Reserved
MUX31	Reserved	INPUTXBAR14	Reserved	Reserved

Note: The "Reserved" signals are not usable, and all unused and reserved signals are set to 0.

24.3.4 FLB X-BAR

24.3.4.1 Structure block diagram

The structure of Output X-BAR is similar to that of FLB X-BAR, except for the output latch.

Figure 27 FLB X-BAR Structure Block Diagram

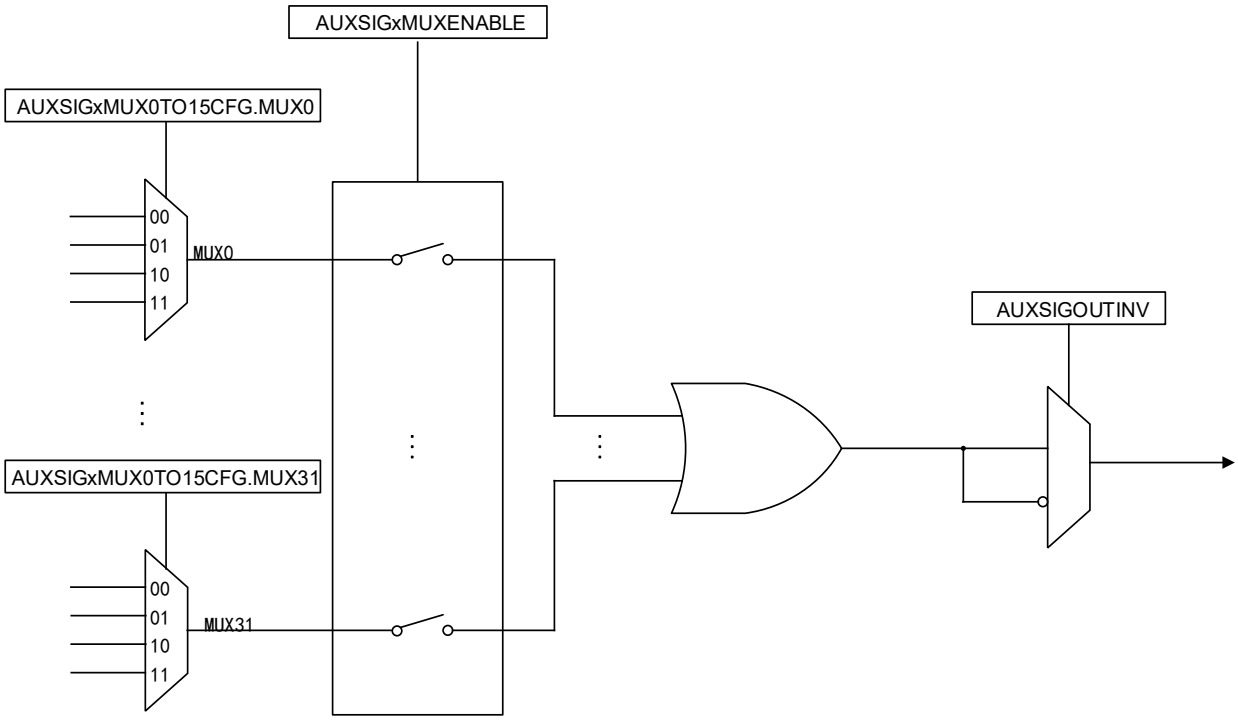
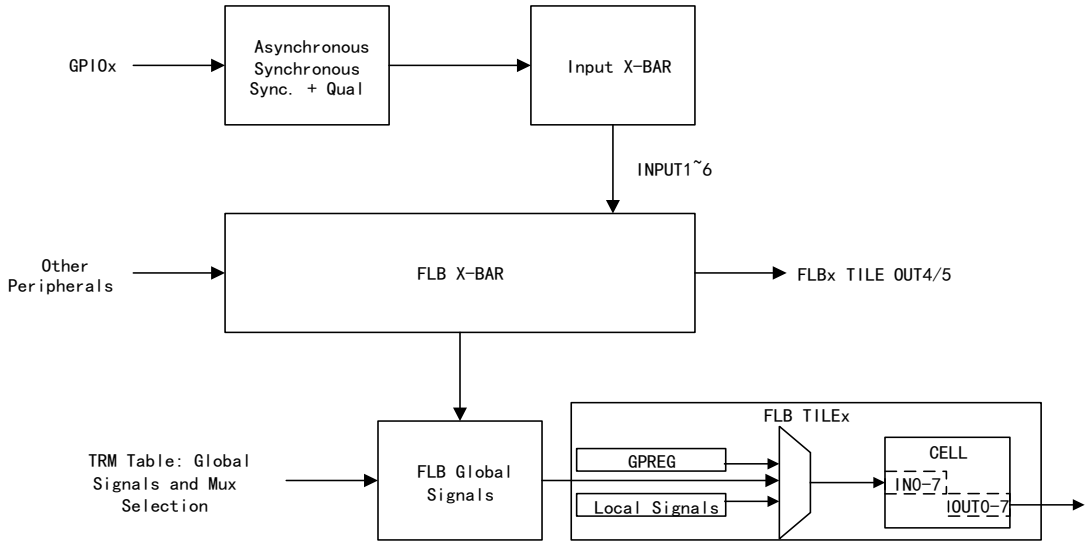


Figure 28 TILE Connection from GPIO to FLB



24.3.4.2 Signal transmission

FLB X-BAR consists of 8 outputs, and it can route signals to the FLB module. Each output is represented by AUXSIGx, and a single signal can be selected on the MUX, or up to 32 signals of logic OR connection can be used.

Signal transmission steps:

(1) Select the signal needing to be transmitted to PWM: For each AUXSIGx, select a signal from the AUXSIGxMUX0TO15CFG or AUXSIGxMUX16TO31CFG register as the input of each MUX (there are a total of 32 MUX).

(2) Enable the multiplexer: Enable the multiplexer in the AUXSIGxMUXENABLE register. Before the AUXSIGx signal transmitted to the PWM module, logic OR operation is performed on all multiplexers.

(3) (Optional) Signal inversion: Invert the signals in the AUXSIGOUTINV register.

Table 78 FLB X-BAR Multiplexing Configuration

MUX	0	1	2	3
MUX0	COMP1.CTRIPH	COMP1.CTRIPH_OR_CTRIPL	ADCAEVT1	CAP1OUT
MUX1	COMP1.CTRIPL	INPUTXBAR1	FLB1_OUT4	ADCCEVT1
MUX2	COMP2.CTRIPH	COMP2.CTRIPH_OR_CTRIPL	ADCAEVT2	CAP2OUT
MUX3	COMP2.CTRIPL	INPUTXBAR2	FLB1_OUT5	ADCCEVT2
MUX4	COMP3.CTRIPH	COMP3.CTRIPH_OR_CTRIPL	ADCAEVT3	CAP3OUT
MUX5	COMP3.CTRIPL	INPUTXBAR3	FLB2_OUT4	ADCCEVT3
MUX6	COMP4.CTRIPH	COMP4.CTRIPH_OR_CTRIPL	ADCAEVT4	Reserved
MUX7	COMP4.CTRIPL	INPUTXBAR4	FLB2_OUT5	ADCCEVT4
MUX8	Reserved	Reserved	Reserved	Reserved
MUX9	Reserved	INPUTXBAR5	FLB3_OUT4	Reserved
MUX10	Reserved	Reserved	Reserved	Reserved
MUX11	Reserved	INPUTXBAR6	FLB3_OUT5	Reserved
MUX12	Reserved	Reserved	Reserved	Reserved
MUX13	Reserved	ADCSOCAO	FLB4_OUT4	Reserved
MUX14	Reserved	Reserved	Reserved	EXTSYNCOUT
MUX15	Reserved	ADCSOCBO	FLB4_OUT5	Reserved
MUX16	Reserved	Reserved	Reserved	Reserved
MUX17	Reserved	INPUTXBAR7	Reserved	Reserved
MUX18	Reserved	Reserved	Reserved	Reserved
MUX19	Reserved	INPUTXBAR8	Reserved	Reserved
MUX20	Reserved	Reserved	Reserved	Reserved
MUX21	Reserved	INPUTXBAR9	Reserved	Reserved
MUX22	Reserved	Reserved	Reserved	Reserved

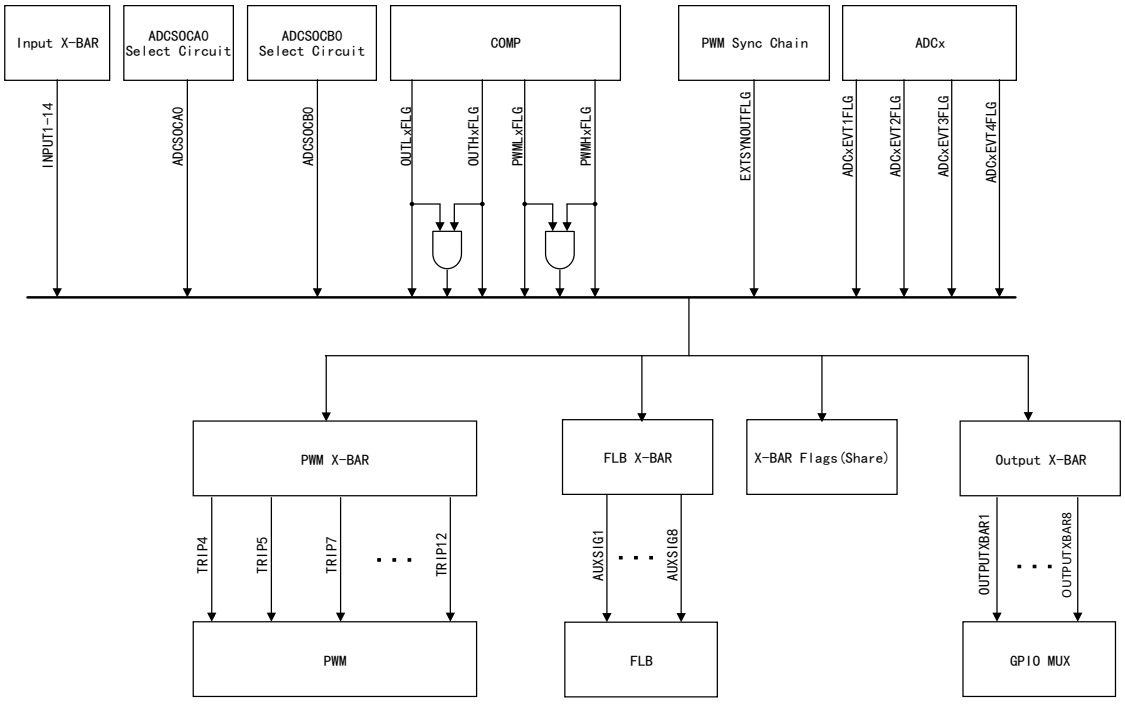
MUX	0	1	2	3
MUX23	Reserved	INPUTXBAR10	Reserved	Reserved
MUX24	Reserved	Reserved	Reserved	Reserved
MUX25	Reserved	INPUTXBAR11	Reserved	Reserved
MUX26	Reserved	Reserved	Reserved	Reserved
MUX27	Reserved	INPUTXBAR12	Reserved	Reserved
MUX28	Reserved	Reserved	Reserved	Reserved
MUX29	Reserved	INPUTXBAR13	Reserved	Reserved
MUX30	Reserved	Reserved	Reserved	Reserved
MUX31	Reserved	INPUTXBAR14	Reserved	Reserved

Note: The "Reserved" signals are not usable, and all unused and reserved signals are set to 0.

24.3.5 X-BAR flag

The input signals of Output X-BAR and PWM X-BAR are the same (except for COMP signal), so Output X-BAR, PWM X-BAR, and FLB X-BAR share a set of input flags to indicate which input signals have been triggered. In the XBAR_FLG register, each input signal has a corresponding flag bit. When the corresponding signal is triggered, the flag bit will remain in the set state until this flag is cleared by the corresponding bit in the XBAR_FLG register. The software can check the corresponding input flag when an event occurs.

Figure 29 X-BAR Input Source



24.4 Register bank address

Table 79 XBAR Register Bank Address

Device register	Register bank	Start address	End address
InputXbarRegs	XBAR_INPUT	0x4003_0C00	0x4003_0C7F
XbarRegs	XBAR_FLG	0x4003_1000	0x4003_103F
PWMXbarRegs	XBAR_PWM	0x4003_11C0	0x4003_123F
FlbXbarRegs	XBAR_FLB	0x4003_1240	0x4003_12BF
OutputXbarRegs	XBAR_OUTPUT	0x4003_12C0	0x4003_133F

24.5 Register address mapping

Table 80 XBAR_INPUT Offset Address

Register name	Register description	Offset address	WRPRT
INPUTxSELECT (n=1…16)	Select register	0x00+(n-1)*0x02	√
INPUTSELECTLOCK	INPUTxSELECT lock register	0x3C	√

Table 81 XBAR_FLG Offset Address

Register name	Register description	Offset address	WRPRT
XBARFLG1	Flag register 1	0x00	-
XBARFLG2	Flag register 2	0x04	-
XBARFLG3	Flag register 3	0x08	-
XBARFLG4	Flag register 4	0x0C	-
XBARCLR1	Clear register 1	0x10	-
XBARCLR2	Clear register 2	0x14	-
XBARCLR3	Clear register 3	0x18	-
XBARCLR4	Clear register 4	0x1C	-

Table 82 XBAR_PWM Offset Address

Register name	Register description	Offset address	WRPRT
TRIP4MUX0TO15CFG	TRIP4 MUX0_15 configuration register	0x00	√
TRIP4MUX16TO31CFG	TRIP4 MUX16_31 configuration register	0x04	√
TRIP5MUX0TO15CFG	TRIP5 MUX0_15 configuration register	0x08	√
TRIP5MUX16TO31CFG	TRIP5 MUX16_31 configuration register	0x0C	√
TRIP7MUX0TO15CFG	TRIP7 MUX0_15 configuration register	0x10	√

Register name	Register description	Offset address	WRPRT
TRIP7MUX16TO31CFG	TRIP7 MUX16_31 configuration register	0x14	√
TRIP8MUX0TO15CFG	TRIP8 MUX0_15 configuration register	0x18	√
TRIP8MUX16TO31CFG	TRIP8 MUX16_31 configuration register	0x1C	√
TRIP9MUX0TO15CFG	TRIP9 MUX0_15 configuration register	0x20	√
TRIP9MUX16TO31CFG	TRIP9 MUX16_31 configuration register	0x24	√
TRIP10MUX0TO15CFG	TRIP10 MUX0_15 configuration register	0x28	√
TRIP10MUX16TO31CFG	TRIP10 MUX16_31 configuration register	0x2C	√
TRIP11MUX0TO15CFG	TRIP11 MUX0_15 configuration register	0x30	√
TRIP11MUX16TO31CFG	TRIP11 MUX16_31 configuration register	0x34	√
TRIP12MUX0TO15CFG	TRIP12 MUX0_15 configuration register	0x38	√
TRIP12MUX16TO31CFG	TRIP12 MUX16_31 configuration register	0x3C	√
TRIP4MUXENABLE	TRIP4 MUX enable register	0x40	√
TRIP5MUXENABLE	TRIP5 MUX enable register	0x44	√
TRIP7MUXENABLE	TRIP7 MUX enable register	0x48	√
TRIP8MUXENABLE	TRIP8 MUX enable register	0x4C	√
TRIP9MUXENABLE	TRIP9 MUX enable register	0x50	√
TRIP10MUXENABLE	TRIP10 MUX enable register	0x54	√
TRIP11MUXENABLE	TRIP11 MUX enable register	0x58	√
TRIP12MUXENABLE	TRIP12 MUX enable register	0x5C	√
TRIPOUTINV	TRIP polarity select register	0x70	√
TRIPLOCK	TRIP lock register	0x7C	√

Table 83 XBAR_FLB Offset Address

Register name	Register description	Offset address	WRPRT
AUXSIG0MUX0TO15CFG	AUXSIG0 MUX0_15 configuration register	0x00	√
AUXSIG0MUX16TO31CFG	AUXSIG0 MUX16_31 configuration register	0x04	√
AUXSIG1MUX0TO15CFG	AUXSIG1 MUX0_15 configuration register	0x08	√
AUXSIG1MUX16TO31CFG	AUXSIG1 MUX16_31 configuration register	0x0C	√
AUXSIG2MUX0TO15CFG	AUXSIG2 MUX0_15 configuration register	0x10	√
AUXSIG2MUX16TO31CFG	AUXSIG2 MUX16_31 configuration register	0x14	√
AUXSIG3MUX0TO15CFG	AUXSIG3 MUX0_15 configuration register	0x18	√
AUXSIG3MUX16TO31CFG	AUXSIG3 MUX16_31 configuration register	0x1C	√
AUXSIG4MUX0TO15CFG	AUXSIG4 MUX0_15 configuration register	0x20	√

Register name	Register description	Offset address	WRPRT
AUXSIG4MUX16TO31CFG	AUXSIG4 MUX16_31 configuration register	0x24	√
AUXSIG5MUX0TO15CFG	AUXSIG5 MUX0_15 configuration register	0x28	√
AUXSIG5MUX16TO31CFG	AUXSIG5 MUX16_31 configuration register	0x2C	√
AUXSIG6MUX0TO15CFG	AUXSIG6 MUX0_15 configuration register	0x30	√
AUXSIG6MUX16TO31CFG	AUXSIG6 MUX16_31 configuration register	0x34	√
AUXSIG7MUX0TO15CFG	AUXSIG7 MUX0_15 configuration register	0x38	√
AUXSIG7MUX16TO31CFG	AUXSIG7 MUX16_31 configuration register	0x3C	√
AUXSIG0MUXENABLE	AUXSIG0 MUX enable register	0x40	√
AUXSIG1MUXENABLE	AUXSIG1 MUX enable register	0x44	√
AUXSIG2MUXENABLE	AUXSIG2 MUX enable register	0x48	√
AUXSIG3MUXENABLE	AUXSIG3 MUX enable register	0x4C	√
AUXSIG4MUXENABLE	AUXSIG4 MUX enable register	0x50	√
AUXSIG5MUXENABLE	AUXSIG5 MUX enable register	0x54	√
AUXSIG6MUXENABLE	AUXSIG6 MUX enable register	0x58	√
AUXSIG7MUXENABLE	AUXSIG7 MUX enable register	0x5C	√
AUXSIGOUTINV	AUXSIG polarity select register	0x70	√
AUXSIGLOCK	AUXSIG lock register	0x7C	√

Table 84 XBAR_OUTPUT Offset Address

Register name	Register description	Offset address	WRPRT
OUTPUTxMUX0TO15CFG (x=1…8)	OUTPUTx MUX0_15 configuration register	0x00+(x-1)*0x08	√
OUTPUTxMUX16TO31CFG (x=1…8)	OUTPUTx MUX16_31 configuration register	0x04+(x-1)*0x08	√
OUTPUTxMUXENABLE (x=1…8)	OUTPUTx MUX enable register	0x40+(x-1)*0x04	√
OUTPUTLATCH	OUTPUT latch register	0x60	-
OUTPUTLATCHCLR	OUTPUT latch clear register	0x64	-
OUTPUTLATCHFRC	OUTPUT latch setup register	0x68	-
OUTPUTLATCHENABLE	OUTPUT latch enable register	0x6C	√
OUTPUTINV	OUTPUT latch inverting register	0x70	√
OUTPUTLOCK	OUTPUT lock register	0x7C	√

24.6 Register functional description

24.6.1 Select register (INPUTxSELECT) (n=1…16)

Offset address: 0x00+(n-1)*0x02

Reset type: CPU1.SYSRSn

Field	Name	R/W	Description	Reset value
15:0	SELECT	R/W	INPUTn signal Select GPIO (n=1…16) When the value of SELECT is greater than the quantity of available GPIO, the target module will be driven to a high level (except for 0xFFFF). 0x0000：GPIO0 0x0001：GPIO1 0x0002：GPIO2 … 0xFFFD: Drive the target module to a high level 0xFFFE: Drive the target module to a high level 0xFFFF: Drive the target module to a low level	FFFEh

24.6.2 INPUTxSELECT lock register (INPUTSELECTLOCK)

Offset address: 0x3C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
n-1	INPUTnSELECT	R/WSO	INPUTxSELECT Register Lock bit (n=1…16) After reset, it can only be cleared through the SYSRSn register. For registers with lock protection, read operation is always allowed. 0: INPUTxSELECT register not locked 1: INPUTxSELECT register locked	0h
31:16	Reserved			

24.6.3 Flag register 1 (XBARFLG1)

Offset address: 0x00

Reset type: SYSRSn

This register indicates the trigger state of the input source.

Field	Name	R/W	Description	Reset value
0	COMP1_CTRIPL	R	COMP1_PWML input flag This bit indicates whether the COMP1_PWML input is triggered. Note: The operation priority of setting this bit is higher than the software clear operation. 0: Not triggered 1: Triggered	0h

Field	Name	R/W	Description	Reset value
1	COMP1_CTRIPH	R	COMP1_PWMH input flag Refer to COMP1_CTRIPL for specific description.	0h
2	COMP2_CTRIPL	R	COMP2_PWML input flag Refer to COMP1_CTRIPL for specific description.	0h
3	COMP2_CTRIPH	R	COMP2_PWMH input flag Refer to COMP1_CTRIPL for specific description.	0h
4	COMP3_CTRIPL	R	COMP3_PWML input flag Refer to COMP1_CTRIPL for specific description.	0h
5	COMP3_CTRIPH	R	COMP3_PWMH input flag Refer to COMP1_CTRIPL for specific description.	0h
6	COMP4_CTRIPL	R	COMP4_PWML input flag Refer to COMP1_CTRIPL for specific description.	0h
7	COMP4_CTRIPH	R	COMP4_PWMH input flag Refer to COMP1_CTRIPL for specific description.	0h
15:8	Reserved			0h
16	COMP1_CTRIPOUTL	R	COMP1_OUTL input flag Refer to COMP1_CTRIPL for specific description.	0h
17	COMP1_CTRIPOUTH	R	COMP1_OUTH input flag Refer to COMP1_CTRIPL for specific description.	0h
18	COMP2_CTRIPOUYL	R	COMP2_OUTL input flag Refer to COMP1_CTRIPL for specific description.	0h
19	COMP2_CTRIPOUTH	R	COMP2_OUTH input flag Refer to COMP1_CTRIPL for specific description.	0h
20	COMP3_CTRIPOUTL	R	COMP3_OUTL input flag Refer to COMP1_CTRIPL for specific description.	0h
21	COMP3_CTRIPOUTH	R	COMP3_OUTH input flag Refer to COMP1_CTRIPL for specific description.	0h
22	COMP4_CTRIPOUTL	R	COMP4_OUTL input flag Refer to COMP1_CTRIPL for specific description.	0h
23	COMP4_CTRIPOUTH	R	COMP4_OUTH input flag Refer to COMP1_CTRIPL for specific description.	0h

Field	Name	R/W	Description	Reset value
31:24	Reserved			0h

24.6.4 Flag register 2 (XBARFLG2)

Offset address: 0x04

Reset type: SYSRSn

This register indicates the trigger state of the input source.

Field	Name	R/W	Description	Reset value
0	INPUT1	R	INPUT1 input flag This bit indicates whether the INPUT1 input is triggered. Note: The operation priority of setting this bit is higher than the software clear operation. 0: Not triggered 1: Triggered	0h
1	INPUT2	R	INPUT2 input flag Refer to INPUT1 for specific description.	0h
2	INPUT3	R	INPUT3 input flag Refer to INPUT1 for specific description.	0h
3	INPUT4	R	INPUT4 input flag Refer to INPUT1 for specific description.	0h
4	INPUT5	R	INPUT5 input flag Refer to INPUT1 for specific description.	0h
5	INPUT6	R	INPUT6 input flag Refer to INPUT1 for specific description.	0h
6	ADCSOCA	R	ADCSOCA input flag Refer to INPUT1 for specific description.	0h
7	ADCSOCB	R	ADCSOCB input flag Refer to INPUT1 for specific description.	0h
8	INPUT7	R	INPUT7 input flag Refer to INPUT1 for specific description.	0h
9	INPUT8	R	INPUT8 input flag Refer to INPUT1 for specific description.	0h
10	INPUT9	R	INPUT9 input flag Refer to INPUT1 for specific description.	0h
11	INPUT10	R	INPUT10 input flag Refer to INPUT1 for specific description.	0h
12	INPUT11	R	INPUT11 input flag Refer to INPUT1 for specific description.	0h
13	INPUT12	R	INPUT12 input flag Refer to INPUT1 for specific description.	0h
14	INPUT13	R	INPUT13 input flag Refer to INPUT1 for specific description.	0h

Field	Name	R/W	Description	Reset value
15	INPUT14	R	INPUT14 input flag Refer to INPUT1 for specific description.	0h
16	CAP1_OUT	R	CAP1_OUT input flag Refer to INPUT1 for specific description.	0h
17	CAP2_OUT	R	CAP2_OUT input flag Refer to INPUT1 for specific description.	0h
18	CAP3_OUT	R	CAP3_OUT input flag Refer to INPUT1 for specific description.	0h
21:19	Reserved			0h
22	EXTSYNCOUT	R	EXTSYNOUT input flag Refer to INPUT1 for specific description.	0h
23	ADCAEVT1	R	ADCAEVT1 input flag Refer to INPUT1 for specific description.	0h
24	ADCAEVT2	R	ADCAEVT2 input flag Refer to INPUT1 for specific description.	0h
25	ADCAEVT3	R	ADCAEVT3 input flag Refer to INPUT1 for specific description.	0h
26	ADCAEVT4	R	ADCAEVT4 input flag Refer to INPUT1 for specific description.	0h
30:27	Reserved			0h
31	ADCCEVT1	R	ADCCEVT1 input flag Refer to INPUT1 for specific description.	0h

24.6.5 Flag register 3 (XBARFLG3)

Offset address: 0x08

Reset type: SYSRSn

This register indicates the trigger state of the input source.

Field	Name	R/W	Description	Reset value
0	ADCCEVT2	R	ADCCEVT2 input flag This bit indicates whether the ADCCEVT2 input is triggered. Note: The operation priority of setting this bit is higher than the software clear operation. 0: Not triggered 1: Triggered	0h
1	ADCCEVT3	R	ADCCEVT3 input flag Refer to ADCCEVT2 for specific description.	0h
2	ADCCEVT4	R	ADCCEVT4 input flag Refer to ADCCEVT2 for specific description.	0h
31:3	Reserved			0h

24.6.6 Flag register 4 (XBARFLG4)

Offset address: 0x0C

Reset type: SYSRSn

This register indicates the trigger state of the input source.

Field	Name	R/W	Description	Reset value
15:0	Reserved			0h
16	FLB1_OUT4	R	FLB1_OUT4 input flag This bit indicates whether the FLB1_OUT4 input is triggered. Note: The operation priority of setting this bit is higher than the software clear operation. 0: Not triggered 1: Triggered	0h
17	FLB1_OUT5	R	FLB1_OUT5 input flag Refer to FLB1_OUT4 for specific description.	0h
18	FLB2_OUT4	R	FLB2_OUT4 input flag Refer to FLB1_OUT4 for specific description.	0h
19	FLB2_OUT5	R	FLB2_OUT5 input flag Refer to FLB1_OUT4 for specific description.	0h
20	FLB3_OUT4	R	FLB3_OUT4 input flag Refer to FLB1_OUT4 for specific description.	0h
21	FLB3_OUT5	R	FLB3_OUT5 input flag Refer to FLB1_OUT4 for specific description.	0h
22	FLB4_OUT4	R	FLB4_OUT4 input flag Refer to FLB1_OUT4 for specific description.	0h
23	FLB4_OUT5	R	FLB4_OUT5 input flag Refer to FLB1_OUT4 for specific description.	0h
31:24	Reserved			0h

24.6.7 Clear register 1 (XBARCLR1)

Offset address: 0x10

Reset type: SYSRSn

This register is used to clear the corresponding flag bit in the XBARFLG1 register.

Field	Name	R/W	Description	Reset value
0	COMP1_CTRIPL	R-0/W1S	COMP1_PWML input flag clear Clear the COMP1_PWML input flag bit COMP1_CTRIPL. 0: No effect 1: Clear the COMP1_CTRIPL flag bit	0h
1	COMP1_CTRIPH	R-0/W1S	COMP1_PWMH input flag clear Refer to COMP1_CTRIPL for specific description.	0h

Field	Name	R/W	Description	Reset value
2	COMP2_CTRIPL	R-0/W1S	COMP2_PWML input flag clear Refer to COMP1_CTRIPL for specific description.	0h
3	COMP2_CTRIPH	R-0/W1S	COMP2_PWMH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
4	COMP3_CTRIPL	R-0/W1S	COMP3_PWML input flag clear Refer to COMP1_CTRIPL for specific description.	0h
5	COMP3_CTRIPH	R-0/W1S	COMP3_PWMH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
6	COMP4_CTRIPL	R-0/W1S	COMP4_PWML input flag clear Refer to COMP1_CTRIPL for specific description.	0h
7	COMP4_CTRIPH	R-0/W1S	COMP4_PWMH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
8	COMP5_CTRIPL	R-0/W1S	COMP5_PWML input flag clear Refer to COMP1_CTRIPL for specific description.	0h
15:8	Reserved			0h
16	COMP1_CTRIPOUTL	R-0/W1S	COMP1_OUTL input flag clear Refer to COMP1_CTRIPL for specific description.	0h
17	COMP1_CTRIPOUTH	R-0/W1S	COMP1_OUTH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
18	COMP2_CTRIPOUTL	R-0/W1S	COMP2_OUTL input flag clear Refer to COMP1_CTRIPL for specific description.	0h
19	COMP2_CTRIPOUTH	R-0/W1S	COMP2_OUTH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
20	COMP3_CTRIPOUTL	R-0/W1S	COMP3_OUTL input flag clear Refer to COMP1_CTRIPL for specific description.	0h
21	COMP3_CTRIPOUTH	R-0/W1S	COMP3_OUTH input flag clear Refer to COMP1_CTRIPL for specific description.	0h
22	COMP4_CTRIPOUTL	R-0/W1S	COMP4_OUTL input flag clear Refer to COMP1_CTRIPL for specific description.	0h
23	COMP4_CTRIPOUTH	R-0/W1S	COMP4_OUTH input flag clear Refer to COMP1_CTRIPL for specific description.	0h

Field	Name	R/W	Description	Reset value
31:24	Reserved			0h

24.6.8 Clear register 2 (XBARCLR2)

Offset address: 0x14

Reset type: SYSRSn

This register is used to clear the corresponding flag bit in the XBARFLG2 register.

Field	Name	R/W	Description	Reset value
0	INPUT1	R-0/W1S	INPUT1 input flag clear Clear the INPUT1 input flag bit INPUT1. 0: No effect 1: Clear the INPUT1 flag bit	0h
1	INPUT2	R-0/W1S	INPUT2 input flag clear Refer to INPUT1 for specific description.	0h
2	INPUT3	R-0/W1S	INPUT3 input flag clear Refer to INPUT1 for specific description.	0h
3	INPUT4	R-0/W1S	INPUT4 input flag clear Refer to INPUT1 for specific description.	0h
4	INPUT5	R-0/W1S	INPUT5 input flag clear Refer to INPUT1 for specific description.	0h
5	INPUT6	R-0/W1S	INPUT6 input flag clear Refer to INPUT1 for specific description.	0h
6	ADCSOCA	R-0/W1S	ADCSOCA input flag clear Refer to INPUT1 for specific description.	0h
7	ADCSOCB	R-0/W1S	ADCSOCB input flag clear Refer to INPUT1 for specific description.	0h
8	INPUT7	R-0/W1S	INPUT7 input flag clear Refer to INPUT1 for specific description.	0h
9	INPUT8	R-0/W1S	INPUT8 input flag clear Refer to INPUT1 for specific description.	0h
10	INPUT9	R-0/W1S	INPUT9 input flag clear Refer to INPUT1 for specific description.	0h
11	INPUT10	R-0/W1S	INPUT10 input flag clear Refer to INPUT1 for specific description.	0h
12	INPUT11	R-0/W1S	INPUT11 input flag clear Refer to INPUT1 for specific description.	0h
13	INPUT12	R-0/W1S	INPUT12 input flag clear Refer to INPUT1 for specific description.	0h
14	INPUT13	R-0/W1S	INPUT13 input flag clear Refer to INPUT1 for specific description.	0h
15	INPUT14	R-0/W1S	INPUT14 input flag clear Refer to INPUT1 for specific description.	0h

Field	Name	R/W	Description	Reset value
16	CAP1_OUT	R-0/W1S	CAP1_OUT input flag clear Refer to INPUT1 for specific description.	0h
17	CAP2_OUT	R-0/W1S	CAP2_OUT input flag clear Refer to INPUT1 for specific description.	0h
18	CAP3_OUT	R-0/W1S	CAP3_OUT input flag clear Refer to INPUT1 for specific description.	0h
21:19	Reserved			0h
22	EXTSYNCOUT	R-0/W1S	EXTSYNOUT input flag clear Refer to INPUT1 for specific description.	0h
23	ADCAEVT1	R-0/W1S	ADCAEVT1 input flag clear Refer to INPUT1 for specific description.	0h
24	ADCAEVT2	R-0/W1S	ADCAEVT2 input flag clear Refer to INPUT1 for specific description.	0h
25	ADCAEVT3	R-0/W1S	ADCAEVT3 input flag clear Refer to INPUT1 for specific description.	0h
26	ADCAEVT4	R-0/W1S	ADCAEVT4 input flag clear Refer to INPUT1 for specific description.	0h
30:27	Reserved			0h
31	ADCCEVT1	R-0/W1S	ADCCEVT1 input flag clear Refer to INPUT1 for specific description.	0h

24.6.9 Clear register 3 (XBARCLR3)

Offset address: 0x18

Reset type: SYSRSn

This register is used to clear the corresponding flag bit in the XBARFLG3 register.

Field	Name	R/W	Description	Reset value
0	ADCCEVT2	R-0/W1S	ADCCEVT2 input flag clear Clear the ADCCEVT2 input flag bit ADCCEVT2. 0: No effect 1: Clear the ADCCEVT2 flag bit	0h
1	ADCCEVT3	R-0/W1S	ADCCEVT3 input flag clear Refer to ADCCEVT2 for specific description.	0h
2	ADCCEVT4	R-0/W1S	ADCCEVT4 input flag clear Refer to ADCCEVT2 for specific description.	0h
31:3	Reserved			0h

24.6.10 Clear register 4 (XBARCLR4)

Offset address: 0x1C

Reset type: SYSRSn

This register is used to clear the corresponding flag bit in the XBARFLG4 register.

Field	Name	R/W	Description	Reset value
15:0	Reserved			0h
16	FLB1_OUT4	R-0/W1S	FLB1_OUT4 flag clear Clear the FLB1_OUT4 input flag bit FLB1_OUT4. 0: No effect 1: Clear the FLB1_OUT4 flag bit	0h
17	FLB1_OUT5	R-0/W1S	FLB1_OUT5 flag clear Refer to FLB1_OUT4 for specific description.	0h
18	FLB2_OUT4	R-0/W1S	FLB2_OUT4 flag clear Refer to FLB1_OUT4 for specific description.	0h
19	FLB2_OUT5	R-0/W1S	FLB2_OUT5 flag clear Refer to FLB1_OUT4 for specific description.	0h
20	FLB3_OUT4	R-0/W1S	FLB3_OUT4 flag clear Refer to FLB1_OUT4 for specific description.	0h
21	FLB3_OUT5	R-0/W1S	FLB3_OUT5 flag clear Refer to FLB1_OUT4 for specific description.	0h
22	FLB4_OUT4	R-0/W1S	FLB4_OUT4 flag clear Refer to FLB1_OUT4 for specific description.	0h
23	FLB4_OUT5	R-0/W1S	FLB4_OUT5 flag clear Refer to FLB1_OUT4 for specific description.	0h
31:24	Reserved			0h

24.6.11 TRIPx MUX0_15configuration register (TRIPxMUX0TO15CFG) (x=4...5, 7...12)

Offset address: 0x00 (TRIP4MUX0TO15CFG)
0x08(TRIP5MUX0TO15CFG)
0x10(TRIP7MUX0TO15CFG)
0x18(TRIP8MUX0TO15CFG)
0x20(TRIP9MUX0TO15CFG)
0x28(TRIP10MUX0TO15CFG)
0x30(TRIP11MUX0TO15CFG)
0x38(TRIP12MUX0TO15CFG)
Reset type: SYSRSn
Configure the input signal of XBAR_PWM TRIPx Muxy (y=0…15).

Field	Name	R/W	Description	Reset value
2y+1:2y	MUXy	R/W	XBAR_PWM TRIPx Muxy Select input (y=0…15) Select the input signal of XBAR_PWM TRIPx Muxy 00: Select No. 1 input signal of Muxy 01: Select No. 2 input signal of Muxy 10: Select No. 3 input signal of Muxy 11: Select No. 4 input signal of Muxy	0h

24.6.12 TRIPx MUX16_31 MUX16_31 configuration register (TRIPxMUX16TO31CFG (x=4...5, 7...12)

Offset address: 0x04 (TRIP4MUX16TO31CFG)
0x0C(TRIP5MUX16TO31CFG)
0x14(TRIP7MUX16TO31CFG)
0x1C(TRIP8MUX16TO31CFG)
0x24(TRIP9MUX16TO31CFG)
0x2C(TRIP10MUX16TO31CFG)
0x34(TRIP11MUX16TO31CFG)
0x3C(TRIP12MUX16TO31CFG)
Reset type: SYSRSn
Configure the input signal of XBAR_PWM TRIPx Muxy (y=16…31).

Field	Name	R/W	Description	Reset value
1:0	MUX16	R/W	XBAR_PWM TRIPx Mux16 Select input Select the input signal of XBAR_PWM TRIPx Mux16 00: Select No. 1 input signal of Mux16 01: Select No. 2 input signal of Mux16 10: Select No. 3 input signal of Mux16 11: Select No. 4 input signal of Mux16	0h
3:2	MUX17	R/W	XBAR_PWM TRIPx Mux17 Select input Refer to MUX16 for specific description.	0h
5:4	MUX18	R/W	XBAR_PWM TRIPx Mux18 Select input Refer to MUX16 for specific description.	0h
7:6	MUX19	R/W	XBAR_PWM TRIPx Mux19 Select input Refer to MUX16 for specific description.	0h
9:8	MUX20	R/W	XBAR_PWM TRIPx Mux20 Select input Refer to MUX16 for specific description.	0h
11:10	MUX21	R/W	XBAR_PWM TRIPx Mux21 Select input Refer to MUX16 for specific description.	0h
13:12	MUX22	R/W	XBAR_PWM TRIPx Mux22 Select input Refer to MUX16 for specific description.	0h
15:14	MUX23	R/W	XBAR_PWM TRIPx Mux23 Select input Refer to MUX16 for specific description.	0h
17:16	MUX24	R/W	XBAR_PWM TRIPx Mux24 Select input Refer to MUX16 for specific description.	0h

Field	Name	R/W	Description	Reset value
19:18	MUX25	R/W	XBAR_PWM TRIPx Mux25 Select input Refer to MUX16 for specific description.	0h
21:20	MUX26	R/W	XBAR_PWM TRIPx Mux26 Select input Refer to MUX16 for specific description.	0h
23:22	MUX27	R/W	XBAR_PWM TRIPx Mux27 Select input Refer to MUX16 for specific description.	0h
25:24	MUX28	R/W	XBAR_PWM TRIPx Mux28 Select input Refer to MUX16 for specific description.	0h
27:26	MUX29	R/W	XBAR_PWM TRIPx Mux29 Select input Refer to MUX16 for specific description.	0h
29:28	MUX30	R/W	XBAR_PWM TRIPx Mux30 Select input Refer to MUX16 for specific description.	0h
31:30	MUX31	R/W	XBAR_PWM TRIPx Mux31 Select input Refer to MUX16 for specific description.	0h

24.6.13 TRIPx MUX enable register (TRIPxMUXENABLE) (x=4...5, 7...12)

Offset address: 0x40 (TRIP4MUXENABLE)
0x44(TRIP5MUXENABLE)
0x48(TRIP7MUXENABLE)
0x4C(TRIP8MUXENABLE)
0x50(TRIP9MUXENABLE)
0x54(TRIP10MUXENABLE)
0x58(TRIP11MUXENABLE)
0x5C(TRIP12MUXENABLE)
Reset type: SYSRSn
Enable the Muxy (y=0…31) output, and the Muxy output can be used for driving PWM-XBAR TRIPx.

Field	Name	R/W	Description	Reset value
y	MUXy	R/W	Muxy output Enable (y=0…31) Enable the input signal of XBAR_PWM TRIPx Muxy 0: Disable; PWM-XBAR TRIPx cannot be driven 1: Enable; PWM-XBAR TRIPx can be driven	0h

24.6.14 TRIP polarity select register (TRIPOUTINV)

Offset address: 0x70
Reset type: SYSRSn
Enable the Muxy (y=0…31) output, and the Muxy output can be used for driving PWM-XBAR TRIPx.

Field	Name	R/W	Description	Reset value
0	TRIP4	R/W	XBAR_PWM TRIP4 polarity Select 0: Drive TRIP4 output to active high 1: Drive TRIP4 output to active low	0h
1	TRIP5	R/W	XBAR_PWM TRIP5 polarity Select Refer to TRIP4 for specific description.	0h
2	TRIP7	R/W	XBAR_PWM TRIP7 polarity Select Refer to TRIP4 for specific description.	0h
3	TRIP8	R/W	XBAR_PWM TRIP8 polarity Select Refer to TRIP4 for specific description.	0h
4	TRIP9	R/W	XBAR_PWM TRIP9 polarity Select Refer to TRIP4 for specific description.	0h
5	TRIP10	R/W	XBAR_PWM TRIP10 polarity Select Refer to TRIP4 for specific description.	0h
6	TRIP11	R/W	XBAR_PWM TRIP11 polarity Select Refer to TRIP4 for specific description.	0h
7	TRIP12	R/W	XBAR_PWM TRIP12 polarity Select Refer to TRIP4 for specific description.	0h
31:8	Reserved			0h

24.6.15 TRIP lock register (TRIPLOCK)

Offset address: 0x7C

Reset type: SYSRSn

This register is used to prevent write operation to the XBAR_PWM register bank, and read operation is unaffected.

The locked registers include:

- TRIPxMUX0TO15CFG (x=4...5, 7...12)
- TRIPxMUX16TO31CFG (x=4...5, 7...12)
- TRIPxMUXENABLE (x=4...5, 7...12)
- TRIPOUTINV

Field	Name	R/W	Description	Reset value
0	LOCK	R/WSO	XBAR_PWM configuration Lock When this bit is set to 1, locking the configuration of the XBAR_PWM register bank will disable write operation to these registers. 0: Write operation to the XBAR_PWM registers is enabled 1: Write operation to the XBAR_PWM registers is disabled	0h
15:1	Reserved			0h

Field	Name	R/W	Description	Reset value
31:16	KEY	R-0/W	Key This field determines whether the LOCK bit is activated. The LOCK bit can only be set when KEY=0x5A5A.	0h

24.6.16 AUXSIGx MUX0_15 configuration register (AUXSIGxMUX0TO15CFG) (x=4...5, 7...12)

Offset address: 0x00 (AUXSIG0MUX0TO15CFG)
0x08(AUXSIG1MUX0TO15CFG)
0x10(AUXSIG2MUX0TO15CFG)
0x18(AUXSIG3MUX0TO15CFG)
0x20(AUXSIG4MUX0TO15CFG)
0x28(AUXSIG5MUX0TO15CFG)
0x30(AUXSIG6MUX0TO15CFG)
0x38(AUXSIG7MUX0TO15CFG)
Reset type: SYSRSn
Configure the input signal of XBAR_FLB AUXSIGx Muxy (y=0…15).

Field	Name	R/W	Description	Reset value
2y+1:2y	MUXy	R/W	XBAR_FLB AUXSIGx Muxy Select input (y=0…15) Select the input signal of XBAR_FLB AUXSIGx Muxy 00: Select No. 1 input signal of Muxy 01: Select No. 2 input signal of Muxy 10: Select No. 3 input signal of Muxy 11: Select No. 4 input signal of Muxy	0h

24.6.17 AUXSIGx MUX16_31 configuration register (AUXSIGxMUX16TO31CFG) (x=4...5, 7...12)

Offset address: 0x04 (AUXSIG0MUX16TO31CFG)
0x0C(AUXSIG1MUX16TO31CFG)
0x14(AUXSIG2MUX16TO31CFG)
0x1C(AUXSIG3MUX16TO31CFG)
0x24(AUXSIG4MUX16TO31CFG)
0x2C(AUXSIG5MUX16TO31CFG)
0x34(AUXSIG6MUX16TO31CFG)
0x3C(AUXSIG7MUX16TO31CFG)
Reset type: SYSRSn
Configure the input signal of XBAR_FLB AUXSIGx Muxy (y=16…31).

Field	Name	R/W	Description	Reset value
1:0	MUX16	R/W	XBAR_FLB AUXSIGx Mux16 Select input Select the input signal of XBAR_FLB AUXSIGx Mux16 00: Select No. 1 input signal of Mux16 01: Select No. 2 input signal of Mux16 10: Select No. 3 input signal of Mux16 11: Select No. 4 input signal of Mux16	0h
3:2	MUX17	R/W	XBAR_FLB AUXSIGx Mux17 Select input Refer to MUX16 for specific description.	0h
5:4	MUX18	R/W	XBAR_FLB AUXSIGx Mux18 Select input Refer to MUX16 for specific description.	0h
7:6	MUX19	R/W	XBAR_FLB AUXSIGx Mux19 Select input Refer to MUX16 for specific description.	0h
9:8	MUX20	R/W	XBAR_FLB AUXSIGx Mux20 Select input Refer to MUX16 for specific description.	0h
11:10	MUX21	R/W	XBAR_FLB AUXSIGx Mux21 Select input Refer to MUX16 for specific description.	0h
13:12	MUX22	R/W	XBAR_FLB AUXSIGx Mux22 Select input Refer to MUX16 for specific description.	0h
15:14	MUX23	R/W	XBAR_FLB AUXSIGx Mux23 Select input Refer to MUX16 for specific description.	0h
17:16	MUX24	R/W	XBAR_FLB AUXSIGx Mux24 Select input Refer to MUX16 for specific description.	0h
19:18	MUX25	R/W	XBAR_FLB AUXSIGx Mux25 Select input Refer to MUX16 for specific description.	0h
21:20	MUX26	R/W	XBAR_FLB AUXSIGx Mux26 Select input Refer to MUX16 for specific description.	0h
23:22	MUX27	R/W	XBAR_FLB AUXSIGx Mux27 Select input Refer to MUX16 for specific description.	0h
25:24	MUX28	R/W	XBAR_FLB AUXSIGx Mux28 Select input Refer to MUX16 for specific description.	0h
27:26	MUX29	R/W	XBAR_FLB AUXSIGx Mux29 Select input Refer to MUX16 for specific description.	0h
29:28	MUX30	R/W	XBAR_FLB AUXSIGx Mux30 Select input Refer to MUX16 for specific description.	0h
31:30	MUX31	R/W	XBAR_FLB AUXSIGx Mux31 Select input Refer to MUX16 for specific description.	0h

24.6.18 AUXSIGx MUX enable register (AUXSIGxMUXENABLE) (x=4...5, 7...12)

Offset address: 0x40 (AUXSIG0MUXENABLE)
0x44(AUXSIG1MUXENABLE)
0x48(AUXSIG2MUXENABLE)
0x4C(AUXSIG3MUXENABLE)

0x50(AUXSIG4MUXENABLE)
0x54(AUXSIG5MUXENABLE)
0x58(AUXSIG6MUXENABLE)
0x5C(AUXSIG7MUXENABLE)
Reset type: SYSRSn
Enable the Muxy (y=0…31) output, and the Muxy output can be used for driving FLB-XBAR AUXSIGx.

Field	Name	R/W	Description	Reset value
y	MUXy	R/W	Muxy output Enable (y=0…31) Enable the input signal of XBAR_FLB AUXSIGx Muxy 0: Disable; FLB-XBAR AUXSIGx cannot be driven 1: Enable; FLB-XBAR AUXSIGx can be driven	0h

24.6.19 AUXSIG polarity select register (AUXSIGOUTINV)

Offset address: 0x70
Reset type: SYSRSn
Enable the Muxy (y=0…31) output, and the Muxy output can be used for driving FLB-XBAR AUXSIGx.

Field	Name	R/W	Description	Reset value
0	AUXSIG0	R/W	XBAR_FLB AUXSIG0 polarity Select 0: Drive AUXSIG0 output to active high 1: Drive AUXSIG0 output to active low	0h
1	AUXSIG1	R/W	XBAR_FLB AUXSIG1 polarity Select Refer to AUXSIG0 for specific description.	0h
2	AUXSIG2	R/W	XBAR_FLB AUXSIG2 polarity Select Refer to AUXSIG0 for specific description.	0h
3	AUXSIG3	R/W	XBAR_FLB AUXSIG3 polarity Select Refer to AUXSIG0 for specific description.	0h
4	AUXSIG4	R/W	XBAR_FLB AUXSIG4 polarity Select Refer to AUXSIG0 for specific description.	0h
5	AUXSIG5	R/W	XBAR_FLB AUXSIG5 polarity Select Refer to AUXSIG0 for specific description.	0h
6	AUXSIG6	R/W	XBAR_FLB AUXSIG6 polarity Select Refer to AUXSIG0 for specific description.	0h
7	AUXSIG7	R/W	XBAR_FLB AUXSIG7 polarity Select Refer to AUXSIG0 for specific description.	0h
31:8	Reserved			0h

24.6.20 AUXSIG lock register (AUXSIGLOCK)

Offset address: 0x7C
Reset type: SYSRSn
This register is used to prevent write operation to the XBAR_FLB register bank, and read operation is unaffected.

The locked registers include:

- AUXSIGxMUX0TO15CFG (x=0…7)
- AUXSIGxMUX16TO31CFG (x=0…7)
- AUXSIGxMUXENABLE (x=0…7)
- AUXSIGOUTINV

Field	Name	R/W	Description	Reset value
0	LOCK	R/WSO	XBAR_FLB configuration Lock When this bit is set to 1, locking the configuration of the XBAR_FLB register bank will disable write operation to these registers. 0: Write operation to the XBAR_FLB registers is enabled 1: Write operation to the XBAR_FLB registers is disabled	0h
15:1	Reserved			0h
31:16	KEY	R-0/W	Key This field determines whether the LOCK bit is activated. The LOCK bit can only be set when KEY=0x5A5A.	0h

24.6.21 OUTPUTx MUX0_15 configuration register (OUTPUTxMUX0TO15CFG) (x=1...8)

Offset address: 0x00+(x-1)*0x08

Reset type: SYSRSn

Configure the input signal of XBAR_OUTPUTx Muxy (y=0…15).

Field	Name	R/W	Description	Reset value
2y+1:2y	MUXy	R/W	OUTPUTx Muxy Select input (y=0…15) Select the input signal of OUTPUTx Muxy 00: Select No. 1 input signal of Muxy 01: Select No. 2 input signal of Muxy 10: Select No. 3 input signal of Muxy 11: Select No. 4 input signal of Muxy	0h

24.6.22 OUTPUTx MUX16_31 configuration register (OUTPUTxMUX16TO31CFG) (x=1...8)

Offset address: 0x04+(x-1)*0x08

Reset type: SYSRSn

Configure the input signal of XBAR_OUTPUTx Muxy (y=16…31).

Field	Name	R/W	Description	Reset value
1:0	MUX16	R/W	OUTPUTx Mux16 Select input Select the input signal of OUTPUTx Mux16 00: Select No. 1 input signal of Mux16 01: Select No. 2 input signal of Mux16 10: Select No. 3 input signal of Mux16 11: Select No. 4 input signal of Mux16	0h
3:2	MUX17	R/W	OUTPUTx Mux17 Select input Refer to MUX16 for specific description.	0h
5:4	MUX18	R/W	OUTPUTx Mux18 Select input Refer to MUX16 for specific description.	0h
7:6	MUX19	R/W	OUTPUTx Mux19 Select input Refer to MUX16 for specific description.	0h
9:8	MUX20	R/W	OUTPUTx Mux20 Select input Refer to MUX16 for specific description.	0h
11:10	MUX21	R/W	OUTPUTx Mux21 Select input Refer to MUX16 for specific description.	0h
13:12	MUX22	R/W	OUTPUTx Mux22 Select input Refer to MUX16 for specific description.	0h
15:14	MUX23	R/W	OUTPUTx Mux23 Select input Refer to MUX16 for specific description.	0h
17:16	MUX24	R/W	OUTPUTx Mux24 Select input Refer to MUX16 for specific description.	0h
19:18	MUX25	R/W	OUTPUTx Mux25 Select input Refer to MUX16 for specific description.	0h
21:20	MUX26	R/W	OUTPUTx Mux26 Select input Refer to MUX16 for specific description.	0h
23:22	MUX27	R/W	OUTPUTx Mux27 Select input Refer to MUX16 for specific description.	0h
25:24	MUX28	R/W	OUTPUTx Mux28 Select input Refer to MUX16 for specific description.	0h
27:26	MUX29	R/W	OUTPUTx Mux29 Select input Refer to MUX16 for specific description.	0h
29:28	MUX30	R/W	OUTPUTx Mux30 Select input Refer to MUX16 for specific description.	0h
31:30	MUX31	R/W	OUTPUTx Mux31 Select input Refer to MUX16 for specific description.	0h

24.6.23 OUTPUTx MUX enable register (OUTPUTxMUXENABLE) (x=1... 8)

Offset address: 0x40+(x-1)*0x08

Reset type: SYSRSn

Enable the Muxy (y=0…31) output, and the Muxy output can be used for driving PWM-XBAR OUTPUTx.

Field	Name	R/W	Description	Reset value
y	MUXy	R/W	Muxy output Enable (y=0…31) Enable the input signal of OUTPUTx Muxy 0: Disable; OUTPUTx of output XBAR cannot be driven 1: Enable; OUTPUTx of output XBAR can be driven	0h

24.6.24 OUTPUT latch register (OUTPUTLATCH)

Offset address: 0x60

Reset type: SYSRSn

This register represents the trigger status of OUTPUTx.

Field	Name	R/W	Description	Reset value
0	OUTPUT1	R	OUTPUT1 Latch This bit indicates whether to trigger the OUTPUT1 latch of Output X-BAR. Note: The operation priority of setting this bit is higher than the software clear operation. 0: Not triggered 1: Triggered	0h
1	OUTPUT2	R	OUTPUT2 Latch Refer to OUTPUT1 for specific description.	0h
2	OUTPUT3	R	OUTPUT3 Latch Refer to OUTPUT1 for specific description.	0h
3	OUTPUT4	R	OUTPUT4 Latch Refer to OUTPUT1 for specific description.	0h
4	OUTPUT5	R	OUTPUT5 Latch Refer to OUTPUT1 for specific description.	0h
5	OUTPUT6	R	OUTPUT6 Latch Refer to OUTPUT1 for specific description.	0h
6	OUTPUT7	R	OUTPUT7 Latch Refer to OUTPUT1 for specific description.	0h
7	OUTPUT8	R	OUTPUT8 Latch Refer to OUTPUT1 for specific description.	0h
31:8	Reserved			0h

24.6.25 OUTPUT latch clear register (OUTPUTLATCHCLR)

Offset address: 0x60

Reset type: SYSRSn

This register is used to clear the corresponding bit in the OUTPUT latch register (OUTPUTLATCH).

Field	Name	R/W	Description	Reset value
0	OUTPUT1	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT1	0h

Field	Name	R/W	Description	Reset value
			Clear the OUTPUT1 bit. 0: No effect 1: Clear the OUTPUT1 bit	
1	OUTPUT2	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT2 Refer to OUTPUT1 for specific description.	0h
2	OUTPUT3	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT3 Refer to OUTPUT1 for specific description.	0h
3	OUTPUT4	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT4 Refer to OUTPUT1 for specific description.	0h
4	OUTPUT5	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT5 Refer to OUTPUT1 for specific description.	0h
5	OUTPUT6	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT6 Refer to OUTPUT1 for specific description.	0h
6	OUTPUT7	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT7 Refer to OUTPUT1 for specific description.	0h
7	OUTPUT8	W	Clear the OUTPUTLATCH for XBAR_OUTPUT OUTPUT8 Refer to OUTPUT1 for specific description.	0h
31:8	Reserved			0h

24.6.26 OUTPUT latch setup register (OUTPUTLATCHFRC)

Offset address: 0x60

Reset type: SYSRSn

This register is used to set the corresponding bit in the OUTPUT latch register (OUTPUTLATCH).

Field	Name	R/W	Description	Reset value
0	OUTPUT1	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT1 Set the OUPUT1 bit. 0: No effect 1: Set the OUPUT1 bit	0h
1	OUTPUT2	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT2 Refer to OUTPUT1 for specific description.	0h
2	OUTPUT3	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT3 Refer to OUTPUT1 for specific description.	0h

Field	Name	R/W	Description	Reset value
3	OUTPUT4	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT4 Refer to OUTPUT1 for specific description.	0h
4	OUTPUT5	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT5 Refer to OUTPUT1 for specific description.	0h
5	OUTPUT6	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT6 Refer to OUTPUT1 for specific description.	0h
6	OUTPUT7	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT7 Refer to OUTPUT1 for specific description.	0h
7	OUTPUT8	W	Set the OUTPUTLATCH for XBAR_OUTPUT OUTPUT8 Refer to OUTPUT1 for specific description.	0h
31:8	Reserved			0h

24.6.27 OUTPUT latch enable register (OUTPUTLATCHENABLE)

Offset address: 0x6C

Reset type: SYSRSn

This register is used to enable the corresponding bit in the OUTPUT latch register (OUTPUTLATCH).

Field	Name	R/W	Description	Reset value
0	OUTPUT1	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT1 Enable the OUPUT1 bit so as to drive OUTPUT1. 0: Disable; OUTPUT1 cannot be driven 1: Enable; OUTPUT1 can be driven	0h
1	OUTPUT2	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT2 Refer to OUTPUT1 for specific description.	0h
2	OUTPUT3	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT3 Refer to OUTPUT1 for specific description.	0h
3	OUTPUT4	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT4 Refer to OUTPUT1 for specific description.	0h
4	OUTPUT5	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT5 Refer to OUTPUT1 for specific description.	0h
5	OUTPUT6	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT6 Refer to OUTPUT1 for specific description.	0h

Field	Name	R/W	Description	Reset value
6	OUTPUT7	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT7 Refer to OUTPUT1 for specific description.	0h
7	OUTPUT8	R/W	Enable the OUTPUTLATCH for XBAR_OUTPUT OUTPUT8 Refer to OUTPUT1 for specific description.	0h
31:8	Reserved			0h

24.6.28 OUTPUT latch inverting register (OUTPUTINV)

Offset address: 0x70

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	OUTPUT1	R/W	XBAR_OUTPUT OUTPUT1 polarity Select 0: Drive OUTPUT1 to active high 1: Drive OUTPUT1 to active low	0h
1	OUTPUT2	R/W	XBAR_OUTPUT OUTPUT2 polarity Select Refer to OUTPUT1 for specific description.	0h
2	OUTPUT3	R/W	XBAR_OUTPUT OUTPUT3 polarity Select Refer to OUTPUT1 for specific description.	0h
3	OUTPUT4	R/W	XBAR_OUTPUT OUTPUT4 polarity Select Refer to OUTPUT1 for specific description.	0h
4	OUTPUT5	R/W	XBAR_OUTPUT OUTPUT5 polarity Select Refer to OUTPUT1 for specific description.	0h
5	OUTPUT6	R/W	XBAR_OUTPUT OUTPUT6 polarity Select Refer to OUTPUT1 for specific description.	0h
6	OUTPUT7	R/W	XBAR_OUTPUT OUTPUT7 polarity Select Refer to OUTPUT1 for specific description.	0h
7	OUTPUT8	R/W	XBAR_OUTPUT OUTPUT8 polarity Select Refer to OUTPUT1 for specific description.	0h
31:8	Reserved			0h

24.6.29 OUTPUT lock register (OUTPUTLOCK)

Offset address: 0x7C

Reset type: SYSRSn

This register is used to prevent write operation to the following registers in the XBAR_OUTPUT register bank, and read operation is unaffected.

The locked registers include:

- OUTPUTxMUX0TO15CFG (x=1...8)
- OUTPUTxMUX16TO31CFG (x=1…8)
- OUTPUTxMUXENABLE (x=1…8)
- OUTPUTLATCHENABLE
- OUTPUTINV

Field	Name	R/W	Description	Reset value
0	LOCK	R/WSO	XBAR_PWM configuration Lock When this bit is set to 1, locking the configuration of the XBAR_PWM register bank will disable write operation to these registers. 0: Write operation to registers is enabled 1: Write operation to registers is disabled	0h
15:1	Reserved			0h
31:16	KEY	W	Key This field determines whether the LOCK bit is activated. The LOCK bit can only be set when KEY=0x5A5A.	0h

25 Direct memory access (DMA)

25.1 Full Name and Abbreviation Description of Terms

Table 85 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
pointer	PTR
ping-pong	-

25.2 Introduction

The DMA module provides a hardware method for transmitting data between peripherals and memory without CPU intervention, to free bandwidth for other system functions. DMA also has the ability to orthogonally reconstruct data during transmission and "ping-pong" data between buffers. This is very useful for structuring data into blocks for CPU processing.

DMA channel 1 can be configured to have a higher priority than other channels, and all channels except channel 1 are completely the same. DMA is an event-based machine, so a peripheral or software trigger is needed to initiate DMA transmission. The core of DMA is a state machine and the tightly coupled address control logic. It allows data blocks to be reordered during transfer and data to and from buffers. DMA itself cannot start memory transfers periodically, but it can be done by configuring timers as trigger sources.

The strength of the controller needs to be measured by the capability of the entire system. If the CPU bandwidth decreases, the system capability will become stronger. In general, the format of data is not conducive to optimal CPU processing, but DMA can free CPU bandwidth and rearrange data into a more streamlined processing mode.

25.3 Main characteristics

(1) 6 DMA channels
- The channels with independent NVIC interrupts can be configured separately
- They can be independently triggered from multiple peripheral trigger sources

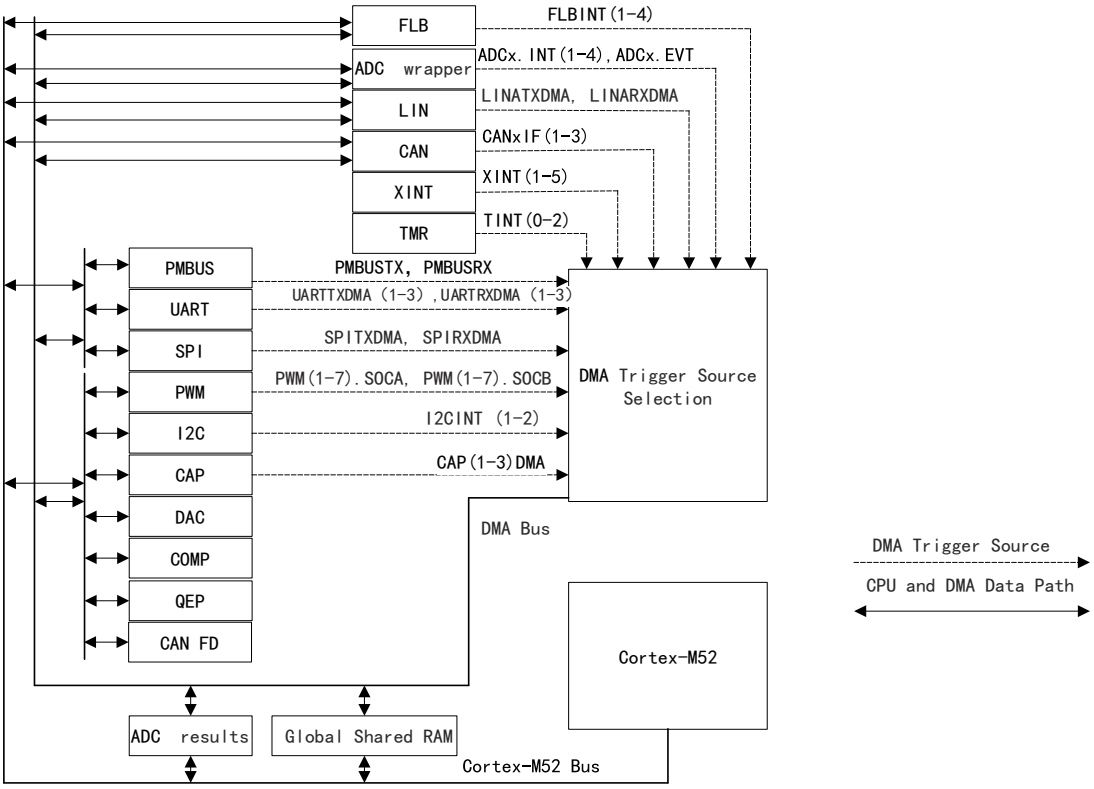
(2) There are three data transmission modes: from off-chip memory to on-chip memory, from peripheral to RAM, and from RAM to peripheral

(3) Word length: 16 bits or 32 bits (SPI is limited to 16 bits)

(4) Throughput: 3 cycles/words without arbitration

25.4 Structure block diagram

Figure 30 Structure Block Diagram



25.5 Functional description

25.5.1 DMA bus

The 32-bit address bus, 32-bit data write bus, and 32-bit data read bus are connected to the memory and register positions of the DMA bus through interfaces, sometimes sharing resources with CPU memory or peripheral buses.

25.5.2 Peripheral interrupt event trigger source

All DMA channels can be triggered by software and other peripheral trigger events. The CHx register and PERINTSEL field can configure DMA trigger sources and select peripheral event trigger sources for each channel. The software can always force trigger using the PERINTFRC bit or clear the DMA trigger using the PERINTCLR bit. The DMA trigger sources include five external interrupt signals, and they can be connected to most GPIO pins in order to enhance the flexibility of event trigger function. When receiving a peripheral interrupt event signal, DMA will automatically transmit a clear signal to the interrupt source so that interrupt events will occur subsequently.

If a specific peripheral trigger event sets the PERINTFLG bit, this bit will remain set until the priority logic of the state machine initiates the burst transmission of

this channel. If a new peripheral trigger event is generated during a burst transmission, the burst will be completed before responding to the new peripheral trigger event. If a new peripheral trigger event occurs before the bit is cleared, an overflow will occur to set the OVRFLG bit. If a peripheral trigger event occurs while clearing the latch flag, the priority of the trigger event is higher, and the PERINTFLG will remain set.

Table 86 DMA Trigger Sources

Selection value (8 bit)	DMA trigger sources
0	None
1	ADCA.1
2	ADCA.2
3	ADCA.3
4	ADCA.4
5	ADCAEVT
6-10	None
11	ADCC.1
12	ADCC.2
13	ADCC.3
14	ADCC.4
15	ADCCEVT
16-28	None
29	XINT1
30	XINT2
31	XINT3
32	XINT4
33	XINT5
34-35	None
36	PWM1.SOCA
37	PWM1.SOCB
38	PWM2.SOCA
39	PWM2.SOCB
40	PWM3.SOCA
41	PWM3.SOCB
42	PWM4.SOCA
43	PWM4.SOCB

Selection value (8 bit)	DMA trigger sources
44	PWM5.SOCA
45	PWM5.SOCB
46	PWM6.SOCA
47	PWM6.SOCB
48	PWM7.SOCA
49	PWM7.SOCB
50-67	None
68	TINT0
69	TINT1
70	TINT2
71-74	None
75	CAP1DMA
76	CAP2DMA
77	CAP3DMA
78	PMBUSTX
79	PMBUSRX
80-95	None
96	UARTATX
97	UARTARX
98	UARTBTX
99	UARTBRX
100-108	None
109	SPIATX
110	SPIARX
111	UARTCTX
112	UARTCRX
113-116	None
117	LINATX
118	LINARX
119-126	None
123	I2CATX
124	None
125	I2CARX

Selection value (8 bit)	DMA trigger sources
126	None
127	FLB1INT
128	FLB2INT
129	FLB3INT
130	FLB4INT
131-166	None
167	CANAIF1
168	CANAIF2
169	CANAIF3
170	I2CBRX
171	I2CBTX
172-255	None

Figure 31 DMA Trigger Structure

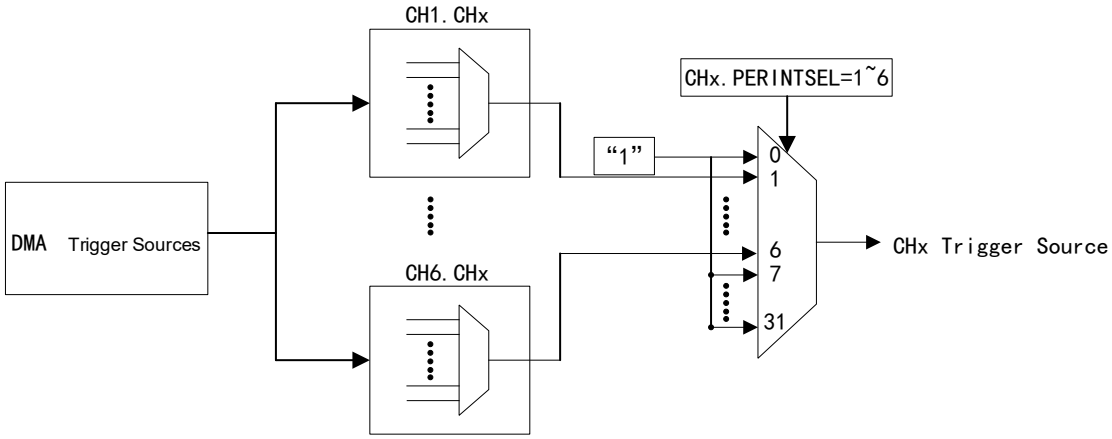
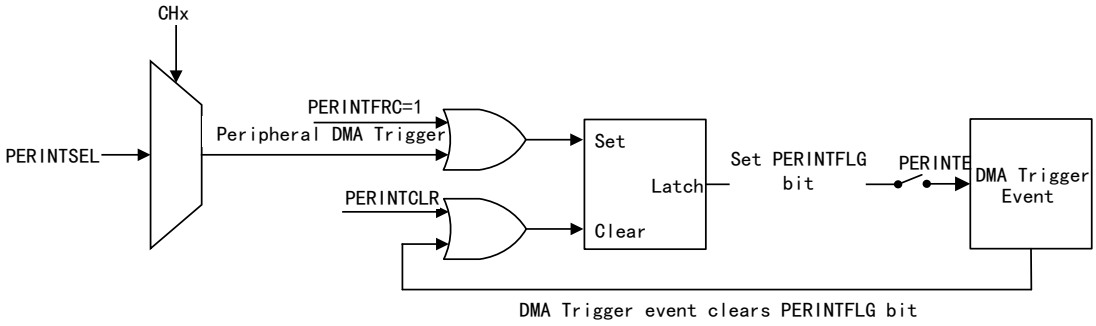


Figure 32 DMA Peripheral Interrupt Triggers Input



25.5.3 Channel priority

Channel priority can be determined according to the following two priority schemes:

- Loop polling mode

- Channel 1 high-priority mode

Loop polling mode

All channels have the same priority, and each enabled channel serves in a loop polling manner. After one channel completes transmission of a burst data, the next channel starts to work. The burst size of transmission for each channel can be configured. If there are no other channels to be processed after the transmission of the last enabled channel has been completed, the loop polling state machine enters an idle state.

The PRIORITYRESET bit can reset the loop polling state machine to an idle state. Starting from the idle state, priority is given to the response to channel 1. However, if DMA is processing another channel x, it needs to process all other pending channels between x and the end of polling before processing the enabled channel 1. All channels have the same priority.

Channel 1 high-priority mode

Channel 1 has a higher priority than all other channels. The priority of other channels is the same, and the enabled channels serve in a loop polling manner. If there are no other channels to be processed after the transmission of the last enabled channel has been completed, the loop polling state machine enters an idle state.

In this mode, channel 1 is generally used for ADC with a high data rate, but it can also be used with other peripherals together. On channel 1, the high-priority mode and single-shot mode cannot be used simultaneously. Other channels can use the single-shot mode when channel 1 is in high-priority mode.

25.5.4 Address pointer and transmission control

Burst (internal) loop

When the burst loop receives a DMA channel trigger, it will transmit the word configured in the BURSTSIZE field, and the number of the words is BURSTSIZE+1. BURSTSIZE allows for transmission of up to 32 16-bit words in one burst. All channels can support burst of 16-bit or 32-bit words through selection of the DATASIZE bit.

All channels contain the shadow address pointers of source address and destination address. At the beginning of each transmission, the shadow version of each pointer will be copied to the corresponding active register. In the burst loop, each transmitted word will add the signed values in SRCBURSTSTEP or DSTBURSTSTEP field to the corresponding active register.

Transmission (external) loop

Each channel's transmission loop transmits a burst of the TRANSFERSIZE field configuration. As the TRANSFERSIZE bit is 16 bits, the allowed transmission quantity exceeds all actual requirements. In the transmission loop, after transmission of each burst is completed, the active address pointer can be modified by disabling the address wrapping or when the number of bursts is equal to WRAPSIZE+1 of the SRC_WRAP_SIZE register or WRAPSIZE+1 of the DST_WRAP_SIZE register. After DMA transmission is completed, DMA can transmit BURSTSIZE+1 x TRANSFERSIZE+ 1 words.

Single-shot mode

The single-shot mode is disabled by default. When single-shot mode is disabled, a burst will be transmitted every time a DMA channel trigger is received. After the burst transmission is completed, the state machine will continue to process the next pending channel in the priority scheme whether there is another trigger waiting in the just completed channel or not, so that it can prevent a single channel from monopolizing the DMA bus. When the single-shot mode is enabled, DMA will transmit all bursts when a single DMA channel is triggered. This mode may cause a trigger to use most of the DMA bandwidth.

Continuous mode

The continuous mode is disabled by default. When the continuous mode is disabled, the state machine will disable the channel after the loop transmission of all bursts is completed. Setting the RUN bit can activate the channel and start another transmission. When the continuous mode is enabled, the state machine will keep the channel active after the loop transmission of all bursts is completed. Each DMA channel can use the CHINTMODE bit to trigger its own NVIC interrupt at the beginning or end of DMA transmission.

(1) Pointer of source/destination address

The start address of the first position where data is written or read will be written into the shadow register. At the beginning of the transmission, the active register loads the data of the shadow register. The active register serves as a pointer of the current address.

(2) Pointer of source/destination start address

Packaging pointer. The value written to the shadow start register will be loaded into the active start register at the beginning of the transmission. Under the wrapping conditions, the active start register will be loaded into the active register after adding the signed value in the corresponding wrapping step register.

(3) Size of start/target burst

Specify the number of words to be transmitted in a burst, and load them into the

BURST_COUNT register at the beginning of each burst transmission. Every time a word is transmitted, the BURST_COUNT register will decrease. When the register is zero, the burst transmission is completed, and it can provide service for next channel. The ONESHOT bit and peripherals determine the usage mode of channels and the maximum burst size.

Table 87 Relationship between BURSTSIZE and DATASIZE

BURST_SIZE	DATASIZE=0	DATASIZE=1
0	1	2
1	2	2
2	3	4
3	4	4
4	5	6
5	6	6
…	…	…
30	31	32
31	32	32

(4) Size of start/target burst

Specify the number of bursts to be transmitted when CPU interrupt is enabled. The CONTINUOUS bit determines the transmission mode, and the CHINTMODE bit determines whether an interrupt is generated at the beginning or end of the transmission. The TRANSFERSIZE bit is loaded into the TRANSFERCOUNT bit at the beginning of each transmission. When the number of transmitted bursts in the channel is zero, DMA transmission is completed.

(5) Size of start/target wrapping

The function of circular addressing type can be implemented. Specify the number of bursts to be transmitted before the current address pointer returns to its initial position. They will be loaded into the appropriate wrapping counter register at the beginning of each transmission. When the number of transmitted bursts in the channel is zero, the wrapping process will be executed on the appropriate start/destination address pointer. The start and destination pointers both have separate size and count registers. The wrapping function can be disabled only when these registers are greater than the TRANSFERSIZE bit.

(6) Start/target burst step

Specify the step of start and destination addresses in burst transmission. This is a signed binary complement number, and it can increment or decrement the address pointer. When the pointer does not need to increment, the values of

these registers can be set to zero.

(7) Start/target transmission step

Specify the address offset for starting the next burst transmission after the current burst transmission is completed. This is a signed binary complement number, and it can increment or decrement the address pointer. Used when the register or data storage locations are arranged at constant intervals.

(8) Start/target wrapping step

This is a signed binary complement number, and it can increment or decrement the address pointer. When the wrapping counter is zero, it represents the number of words to be increased or decreased from the pointer of the start address register, in order to set a new start address. This implements a circular addressing mode.

Note: The step registers are only applicable to 16-bit addresses. The value of 2 can be written to these registers to increment a 32-bit address.

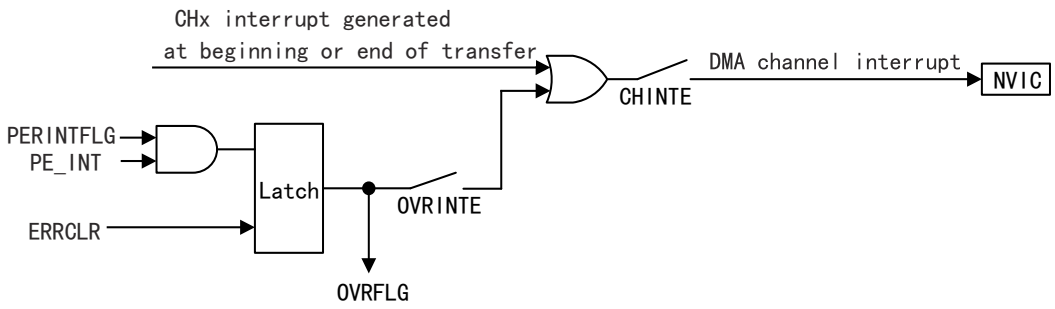
(9) Channel interrupt mode

This mode bit selects whether the DMA interrupt for the corresponding channel occurs at the start or end of a new transmission. If continuous mode is not used, an interrupt is generally generated when the transfer is complete. If DMA uses a continuous mode ping-pong buffer scheme, interrupts can be generated after an active register is copied to a shadow register group.

25.5.5 Overflow detection

When DMA receives a peripheral event trigger, the PERINTFLG bit will be set to suspend the channel to the DMA state machine, and it will be cleared to zero when the burst transmission of this channel begins. If an additional event trigger occurs during the time between setting and clearing of the PERINTFLG bit, the second trigger will be lost and the OVRFLG bit will be set. If overflow interrupt is enabled, the channel interrupt will be generated to the NVIC module.

Figure 33 DMA Overflow Detection Logic



25.5.6 Assembly line and throughput

The DMA module consists of a three-stage assembly line.

Figure 34 Three-stage Assembly Line DMA Transmission

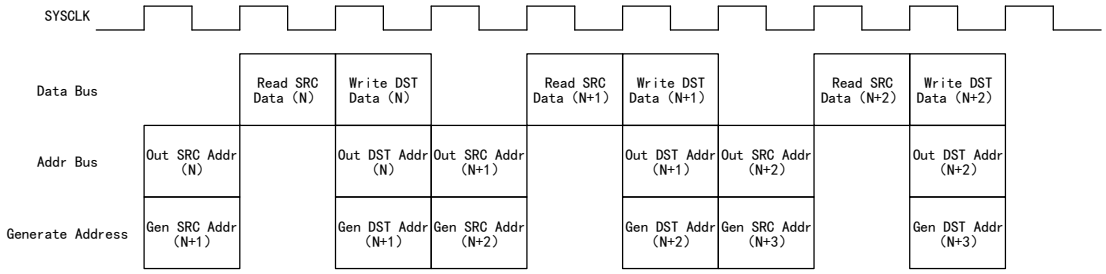
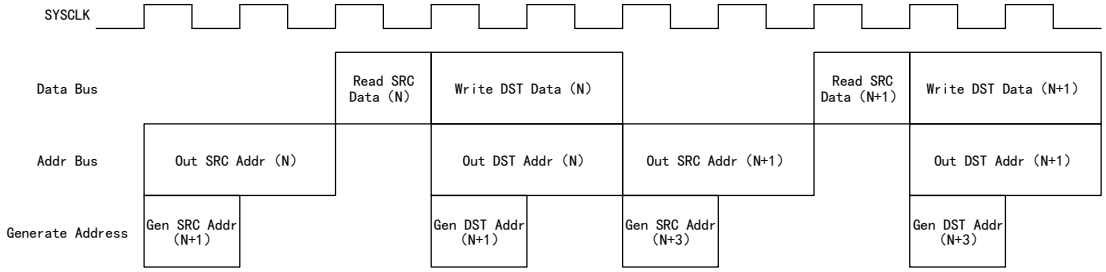


Figure 35 Three-stage Assembly Line Transmission with a Read Gear



The total throughput of DMA is affected by the assembly line and the following conditions:

- Conflicts with CPU may increase the latency slots
- At the beginning of each burst transmission, a delay of one cycle will be added
- When returning from CH1 high-priority interrupt, a delay of one cycle will be added
- The speed of 32-bit transmission is twice that of 16-bit transmission

To transmit 128 16-bit words from SRAM1 to SRAM3, a channel can be configured to transmit 8 bursts with 16 words per burst. The transmission requires the following time:

8 bursts * [(3 cycles/word * 16 words/burst) + 1] = 392 cycles

If the channel is configured to transmit the same amount of 32-bit data at a time, the transmission requires the following time:

8 bursts * [(3 cycles/word * 8 words/burst) + 1] = 200 cycles

25.5.7 CPU arbitration

DMA data transfer operates independently of CPU. When DMA and CPU attempt to simultaneously access an address, the arbitration logic will solve this conflict.

The priority order for DMA, CPU0 to access any slave is DMA>CPU.

Instances of the same peripheral type conflict with each other, and because the instances of different peripheral types can share a bus interface, more access conflicts will result. The peripherals of these shared buses include:

- Peripheral frame 1: COMP, DAC, QEP, CAP, PWM
- Peripheral frame 2: PMBUS, SPI

But the ADC result register is an exception, and it is duplicated for each bus host. CPU and DMA can read these result registers simultaneously without pause or arbitration of any host. DMA transmission includes four stages: transmitting source address, reading source data, transmitting destination address, and writing destination data. When the CPU accesses peripherals/memory during DMA transfer resulting in a conflict, the CPU will remain stagnant until the current DMA access is completed, rather than until the entire DMA transmission is completed.

The priority order for different interfaces on the device is as follows:

(1) Arbitration adopts a fixed highest-priority arbitration scheme:DMA write> DMA read> CPU write > CPU read

(2) The priority scheme for SRAM access is polling.

(3) The ADC result is CPU and DMA duplication, so arbitration is not needed when reading the result register. This allows all hosts to access the ADC result register simultaneously without delay.

Note:

(1) If CPU is performing the read-write-read operation while DMA is executing a write operation, the data written by DMA will be lost. Therefore, it is necessary to avoid executing CPU write operation and DMA write operation in the same beat (exceptions: DMA and CPU zero-wait access to ADC result registers can be executed in the same beat).

(2) DMA cannot be configured when transmitting data.

25.6 Register bank address

Table 88 Register Bank Address

Device register	Register bank	Start address	End address
DMAregs	DMA_REGS	0x4001_8400	0x4001_843F
DMACH1regs	DMA_CH	0x4001_8440	0x4001_847F
DMACH2regs	DMA_CH	0x4001_8480	0x4001_84BF
DMACH3regs	DMA_CH	0x4001_84C0	0x4001_84FF
DMACH4regs	DMA_CH	0x4001_8500	0x4001_853F

Device register	Register bank	Start address	End address
DMACH5regs	DMA_CH	0x4001_8540	0x4001_857F
DMACH6regs	DMA_CH	0x4001_8580	0x4001_85BF
DMASRCSELregs	DMA_SRC_SEL	0x4001_302C	0x4001_3030

25.7 Register address mapping

Table 89 DMA_REGS Register Address Mapping

Register name	Register description	Offset address	WRPRT
DMACTRL	Control register	0x00	√
DEBUGCTRL	Debug control register	0x02	√
PRIORITYCTRL1	Priority control 1 register	0x08	√
PRIORITYSTAT	Priority status register	0x0C	√

Table 90 DMA_CH Register Address Mapping

Register name	Register description	Offset address	WRPRT
MODE	Mode register	0x00	√
CONTROL	Control register	0x02	√
BURST_SIZE	Burst size register	0x04	√
BURST_COUNT	Burst counter register	0x06	√
SRC_BURST_STEP	Source burst step register	0x08	√
DST_BURST_STEP	Destination burst step register	0x0A	√
TRANSFER_SIZE	Transmitter size register	0x0C	√
TRANSFER_COUNT	Transmitter counter register	0x0E	√
SRC_TRANSFER_STEP	Source transmitter step register	0x10	√
DST_TRANSFER_STEP	Destination transmitter step register	0x12	√
SRC_WRAP_SIZE	Source wrapping size register	0x14	√
SRC_WRAP_COUNT	Source wrapping counter register	0x16	√
SRC_WRAP_STEP	Source wrapping step register	0x18	√
DST_WRAP_SIZE	Destination wrapping size register	0x1A	√
DST_WRAP_COUNT	Destination wrapping counter register	0x1C	√
DST_WRAP_STEP	Destination wrapping step register	0x1E	√
SRC_BEG_ADDR_SHADOW	Shadow source start address register	0x20	√
SRC_ADDR_SHADOW	Shadow source address register	0x24	√

Register name	Register description	Offset address	WRPRT
SRC_BEG_ADDR_ACTIVE	Active source start address register	0x28	√
SRC_ADDR_ACTIVE	Active source address register	0x2C	√
DST_BEG_ADDR_SHADOW	Shadow destination start address register	0x30	√
DST_ADDR_SHADOW	Shadow destination address register	0x34	√
DST_BEG_ADDR_ACTIVE	Active destination start address register	0x38	√
DST_ADDR_ACTIVE	Active destination address register	0x3C	√

Table 91 DMA_SRC_SEL Register Address Mapping

Register name	Register description	Offset address	WRPRT
DMACHSRCSELLOCK	Channel trigger source selection lock register	0x08	√
DMACHSRCSEL1	Channel trigger source selection register 1	0x2C	√
DMACHSRCSEL2	Channel trigger source selection register 2	0x30	√

25.8 Register functional description

25.8.1 Control register (DMACTRL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	HARDRESET	R-0/W1S	Hard Reset When writing 1, the entire DMA will be reset and the current access will be terminated. Software reset provides one bit for each channel to perform a mild reset. If it is necessary to access other DMA registers, delay of one cycle is required in the software. Writing 0 will be ignored, and 0 will be read back.	0h
1	PRIORITYRESET	R-0/W1S	Priority reset When writing 1, this bit will reset the polling state machine. The service starts from the first enabled channel. The suspended burst transmission needs to be completed in order to reset the channel priority machine. If CH1 has a high priority, the state machine will be restarted from the next highest enabled channel. If this bit is written simultaneously during the CH1 service burst, both the CH1 burst and the next waiting low-priority burst will be completed before the state machine is	0h

Field	Name	R/W	Description	Reset value
			reset. Writing 0 will be ignored, and 0 will be read back.	
15:2	Reserved			0h

25.8.2 Debug control register (DEBUGCTRL)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
14:0	Reserved			0h
15	FREE	R/W	Configure DMA Work Status When Core is in Halted 0: Stop after the current read/write operation is completed 1: Continue to run during simulation pause period	0h

25.8.3 Priority control 1 register (PRIORITYCTRL1)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
15:1	CH1PRIORITY	R/W	DMA Channel 1 Priority configuration To change the priority, all channels need to be disabled. After the priority is modified, reset the priority and then restart the channel 0: CH1 has the same priority as other channels 1: The priority of CH1 is higher than that of other channels	0h

25.8.4 Priority status register (PRIORITYSTAT)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	ACTIVESTS	R	Active Channel Status It indicates the channel that is currently active or executing transmission. 0: No active channel 1: CH 1 2: CH 2 … 6: CH 6 7: Reserved	0h
3	Reserved			0h

Field	Name	R/W	Description	Reset value
6:4	ACTIVESTS_SHADOW	R	Active Channel Status Shadow Applied in channel 1 high-priority mode. When CH1 is serviced, the ACTIVESTS bit will be copied to this bit and indicates the channel interrupted by CH1. After the CH1 service is completed, this bit will be copied back to the ACTIVESTS bit. If this bit is zero or the same as the ACTIVESTS bit, no channel will be suspended. 0: No active channel 1: CH 1 2: CH 2 … 6: CH 6 7: Reserved	0h
15:7	Reserved			0h

25.8.5 Mode register (MODE)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	PERINTSEL	R/W	Peripheral Event Trigger Source Select It is a remaining bit, which should be set as the channel number. The actual source selection is completed through the DMACHSRCSELn register, and it is part of the DMA_SRC_SEL bank.	0h
6:5	Reserved			0h
7	OVRINTE	R/W	Overflow Interrupt Enable 0: Disable 1: Enable	0h
8	PERINTE	R/W	Peripheral Event Trigger Enable 0: Disable; the selected peripherals and software cannot initiate DMA burst 1: Enable	0h
9	CHINTMODE	R/W	Generation Channel Interrupt Mode configuration 0: An interrupt is generated at the beginning of new transmission 1: An interrupt is generated at the end of transmission	0h
10	ONESHOT	R/W	Single Shot Mode disable 0: Enable; each trigger in the channel only executes a burst once 1: Disable; each peripheral event trigger causes the channel to execute the entire transmission	0h

Field	Name	R/W	Description	Reset value
11	CONTINUOUS	R/W	Continuous Mode Enable Whether to enable continuous mode when TRANSFERCOUNT is zero. 0: Disable; DMA stops and the RUNSTS bit is cleared 1: Enable; reinitialize the channel and wait for the next event trigger	0h
13:12	Reserved			0h
14	DATASIZE	R/W	Data Size configuration This bit determines the size of the transmitted data in each read/write operation of the DMA channel. All data lengths and offsets in other DMA registers point to 16-bit words. The pointer step increment is configured to accommodate 32-bit words. 0: 16 bits 1: 32 bits	0h
15	CHINTE	R/W	Channel Interrupt Enable 0: Disable 1: Enable	0h

25.8.6 Control register (CONTROL)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	RUN	R-0/W1S	Channel Enable This bit is used for recovery after channel interrupt. It is usually used to enable DMA channels after configuration. Then wait for the first peripheral event trigger to enable the burst event. 0: Disable 1: Enable, and make CHENFLG=1	0h
1	HALT	R-0/W1S	Halt Channel After the current read/write access is complete, setting this bit will stop the DMA channel in the current state.	0h
2	SOFTRESET	R-0/W1S	Channel Soft Reset Before accessing other DMA registers, delay of one cycle is required in the software. Setting this bit will cause the channel to enter the default state after the current read-write access is completed: TRANSFERSTS= 0 BURSTSTS= 0 RUNSTS= 0 BURSTCOUNT = 0 TRANSFERCOUNT = 0 WRAPSIZE of SRC_WRAP_COUNT = 0	0h

Field	Name	R/W	Description	Reset value
			WRAPSIZE of DST_WRAP_COUNT = 0	
3	PERINTFRC	R-0/W1S	Force Peripheral Event Trigger This bit can be used to enable DMA transmission in the software. If the PERINTE bit is set, setting it will set the PERINTFLG bit, thus triggering a DMA burst.	0h
4	PERINTCLR	R-0/W1S	Peripheral Event Trigger Clear Setting this bit during DMA module initialization can clear PERINTFLG, and cancel the suspended event triggers. If an event trigger arrives when setting this bit, the trigger has priority and PERINTFLG will be set.	0h
6:5	Reserved			0h
7	ERRCLR	R-0/W1S	Error Clear Setting this bit when the DMA module is initialized or an overflow condition is detected can clear the OVRFLG bit. If an overflow event occurs when setting this bit, set the overflow priority and OVRFLG bit.	0h
8	PERINTFLG	R	Peripheral Event Trigger Flag When set, this bit indicates that the peripheral event trigger has been suspended. This bit is automatically cleared at the start of the first burst transmission. 0: Wait for the event trigger 1: The event trigger is suspended	0h
10:9	Reserved			0h
11	TRANSFERSTS	R	Transferring Flag This bit is set at the start of the DMA transfer. Copy the contents of the address register to the shadow set and set TRANSFERCOUNT to be equal to TRANSFERSIZE. When TRANSFERCOUNT reaches zero or the HARDRESET bit or SOFTRESET bit is set, clear the bit. 0: Not transmit 1: Currently in the process of transmission, no matter whether DMA is transmitting burst data	0h
12	BURSTSTS	R	Burst Flag This bit is set at the start of the DMA burst. Set BURSTCOUNT to equal BURSTSIZE. When BURSTCOUNT reaches zero or the HARDRESET bit or SOFTRESET bit is set, clear the bit. 0: No burst 1: DMA is currently serving or pausing the burst transmission from this channel	0h
13	RUNSTS	R	Channel Enable Flag	0h

Field	Name	R/W	Description	Reset value
			It indicates that the DMA channel has been ready to respond to peripheral event triggers. Set this bit when setting the RUN bit. Clear this bit when the following situations occur: when the transmission is completed and the continuous mode is disabled, the HARDRESET bit or SOFTRESET or HALT bit is set. 0: Disable 1: Enable	
14	OVRFLG	R	Overflow Flag Indicates that the peripheral event trigger has been received repeatedly. Setting the ERRCLR bit can clear this bit. 0: No overflow 1: Overflow	0h
15	Reserved			0h

25.8.7 Burst size register (BURST_SIZE)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	BURSTSIZE	R/W	Burst Size Represent the burst size with 16-bit words. The actual size is equal to BURSIZE +1.	0h
15:5	Reserved			0h

25.8.8 Burst counter register (BURST_COUNT)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	BURSTCNT	R	Burst Count It indicates the remaining word count in the current burst. 00000: 0 00001: 1 00010: 2 … 11111: 31	0h
15:5	Reserved			0h

25.8.9 Source burst step register (SRC_BURST_STEP)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	SRCBURSTSTEP	R/W	Source Burst Step It indicates changes in the source address after a burst is completed, a 16-bit binary complement with the size between -4096 and 4095. Add this value to the source address after each read/write operation. 0000000000000000: No address change 0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094 0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2 1111111111111111: Address-1	0h

25.8.10 Destination burst step register (DST_BURST_STEP)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DESBURSTSTEP	R/W	Destination Burst Step Indicates the changes in destination address after a burst is completed. A 16-bit binary complement with the size between -4096 and 4095. Add this value to the destination address after each read/write operation. 0000000000000000: No address change 0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094 0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2 1111111111111111: Address-1	0h

25.8.11 Transmitter size register (TRANSFER_SIZE)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	TRANSFERSIZE	R/W	Transfer Size This field specifies the transmission size in a burst form. The actual size is equal to TXSIZE+1.	0h

25.8.12 Transmitter counter register (TRANSFER_COUNT)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	TRANSFERCNT	R	Transfer Count It indicates the remaining burst count in the current transmission.	0h

25.8.13 Source transmitter step register (SRC_TRANSFER_STEP)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	SRCTRANSFERSTEP	R/W	Source Transfer Step It indicates changes in the source address after a burst is completed, a 16-bit binary complement with the size between -4096 and 4095. Add this value to the source address after each read/write operation. 0000000000000000: No address change 0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094 0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2 1111111111111111: Address-1	0h

25.8.14 Destination transmitter step register (DST_TRANSFER_STEP)

Offset address: 0x12

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DSTTRANSFERSTEP	R/W	Destination Burst Step Indicates the changes in destination address after a burst is completed. A 16-bit binary complement with the size between -4096 and 4095. Add this value to the destination address after each read/write operation. 0000000000000000: No address change	0h

Field	Name	R/W	Description	Reset value
			0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094 0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2 1111111111111111: Address-1	

25.8.15 Source wrapping size register (SRC_WRAP_SIZE)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPSIZE	R/W	Source Wrap Size It indicates the number of bursts to be transmitted before the source address goes back to the start address. The actual quantity is equal to SRCWSIZE+1. If this bit is greater than the value of the TXSIZE bit, the line feed function can be turned off.	FFFFh

25.8.16 Source wrapping counter register (SRC_WRAP_COUNT)

Offset address: 0x16

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPCNT	R	Source Wrap Count It indicates the number of remaining bursts before the wrapping source address.	0h

25.8.17 Source wrapping step register (SRC_WRAP_STEP)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPSTEP	R/W	Source Transfer Step It indicates changes in the source start address when the wrapping counter reaches zero. A 16-bit binary complement with the size between -4096 and 4095. Add this value to the source address when wrapping occurs. 0000000000000000: No address change 0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094	0h

Field	Name	R/W	Description	Reset value
			0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2 1111111111111111: Address-1	

25.8.18 Destination wrapping size register (DST_WRAP_SIZE)

Offset address: 0x1A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPSIZE	R/W	Destination Wrap Size It indicates the number of bursts to be transmitted before the destination address goes back to the start address. The actual quantity is equal to DESWSIZE+1. If this bit is greater than the value of the TXSIZE bit, the line feed function can be turned off.	FFFFh

25.8.19 Destination wrapping counter register (DST_WRAP_COUNT)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPCNT	R	Destination Wrap Count It indicates the number of remaining bursts before the wrapping destination address.	0h

25.8.20 Destination wrapping step register (DST_WRAP_STEP)

Offset address: 0x1E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	WRAPSTEP	R/W	Source Transfer Step It indicates changes in the destination start address when the wrapping counter reaches zero. A 16-bit binary complement with the size between -4096 and 4095. Add this value to the destination address when wrapping occurs. 0000000000000000: No address change 0000000000000001: Address+1 0000000000000010: Address+2 … 0000111111111110: Address+4094 0000111111111111: Address+4095 0001000000000000: Address-4096 0001000000000001: Address-4095 … 1111111111111110: Address-2	0h

Field	Name	R/W	Description	Reset value
			1111111111111111: Address-1	

25.8.21 Shadow source start address register (SRC_BEG_ADDR_SHADOW)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	BEGADDR	R/W	Shadow Source Begin Address The value in this bit is loaded into the BEGADDR bit field of the SRC_BEG_ADDR_ACTIVE register at the start of transmitting and is used as the starting value of the source address. This register can be safely updated in the transmission process.	0h

25.8.22 Shadow source address register (SRC_ADDR_SHADOW)

Offset address: 0x24

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ADDR	R/W	Shadow Source Address The value in this bit is loaded into the ADDR bit field of the SRC_ ADDR_ACTIVE register at the start of transmitting and is used as the starting value of the source address.This register can be safely updated in the transmission process.	0h

25.8.23 Active source start address register (SRC_BEG_ADDR_ACTIVE)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	BEGADDR	R	Active Source Begin Address This field has two functions,During transmission, stores the current starting value of the source address, which may be updated after wrapping. When transmission begins, will load the shadow address from the BEGADDR field of the SRC_BEG_ADDR_SHADOW register.	0h

25.8.24 Active source address register (SRC_ADDR_ACTIVE)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ADDR	R	Active Source Address During transmission, this field stores the current starting value of the source address, which may be updated after wrapping, write or burst. When transmission begins, this	0h

Field	Name	R/W	Description	Reset value
			field will load the shadow address from the ADDR field of the SRC_ADDR_SHADOW register	

25.8.25 Shadow destination start address register (DST_BEG_ADDR_SHADOW)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	BEGADDR	R/W	Shadow Destination Begin Address The value of this bit field is loaded into the BEGADDR bit field of the DST_BEG_ADDR_ACTIVE register at the beginning of the transmission and is used as the starting value of the destination address. This register can be safely updated in the transmission process.	0h

25.8.26 Shadow destination address register (DST_ADDR_SHADOW)

Offset address: 0x34

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ADDR	R/W	Shadow Destination Address The value of this bit field is loaded into the ADDR field of the DST_ADDR_ACTIVE register as the value of the destination address when the transmission begins.This register can be safely updated in the transmission process.	0h

25.8.27 Active destination start address register (DST_BEG_ADDR_ACTIVE)

Offset address: 0x38

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	BEGADDR	R	Active Destination Begin Address This field has two functions,During transmission, stores the current destination value of the source address. This address may be updated after wrapping. When transmission begins, will load the shadow address from the BEGADDR field of the DST_BEG_ADDR_SHADOW register.	0h

25.8.28 Active destination address register (DST_ADDR_ACTIVE)

Offset address: 0x3C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	ADDR	R	Active Destination Address The current value of the destination address is saved during transmission and may be updated after wrapping, write or burst.	0h

25.8.29 Channel trigger source selection lock register (DMACHSRCSELLOCK)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DMACHSRCSEL1	R/WSonce	DMACHSRCSEL1 Register Lock The locking mechanism only applies to write. Read operation can be performed on the registers with LOCK protection 1: Unlocked 1: Locked	0h
1	DMACHSRCSEL1	R/WSonce	DMACHSRCSEL2 Register Lock The locking mechanism only applies to write. Read operation can be performed on the registers with LOCK protection 1: Unlocked 1: Locked	0h
31:2	Reserved			0h

25.8.30 Channel trigger source selection register 1 (DMACHSRCSEL1)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	CH1	R/W	DMA CH1 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 1	0h
15:8	CH2	R/W	DMA CH2 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 2	0h
23:16	CH3	R/W	DMA CH3 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 3	0h
31:24	CH4	R/W	DMA CH4 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 4	0h

25.8.31 Channel trigger source selection register 2 (DMACHSRCSEL2)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	CH5	R/W	DMA CH5 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 5	0h
15:8	CH6	R/W	DMA CH6 Trigger and Sync Source Selects Select the trigger and synchronization source for channel 6	0h
31:16	Reserved			0h

26 Analog Subsystem (AS)

26.1 Introduction

The analog module of this device contains an analog-to-digital converter (ADC), a temperature sensor, a buffer digital-to-analog converter (DAC), and a comparator (COMP).

26.2 Main characteristics

Flexible voltage reference

(1) ADC takes VREFHIx and VREFLOx pins as reference
- The internal voltage reference range is 0V~3.3V or 0V~2.5V
- The voltage of VREFHIx pin is driven externally or generated by internal reference voltage

(2) Pin reference for DAC
- The buffer DAC takes VREFHIx and VSSA as reference or takes VDAC pin and VSSA as reference
- The comparator DAC takes VDDA and VSSA as reference

Flexible use of pins

(1) Internally connected to VREFLO on all ADC, it can be used for bias self-calibration

(2) Buffer DAC output, comparator input and digital input are multiplexed with ADC input

26.3 Structure block diagram

The block diagram shows the connections between the different integrated analog modules and the input/output and reference pins of the analog modules.

The analog pins form several analog groups with COMP together. The block diagram shows the pins connected to each analog group, not show the connections from the pins to the ADC, DAC, and COMP modules.

The VDAC reference pin used to set other ranges for DAC A, DAC B. If VDAC pin is used to provide a reference voltage, this channel cannot be used as the ADC input. The reference selection for buffer DAC can be configured through the configuration register of the module.

Precautions:

(1) The voltage range of VREFHI and VREFLO needs to be determined.

(2) The VREFHI pin needs to connect an external capacitor.

(3) Not all analog pins are applicable to all devices.

(4) Whether the high reference voltage of the buffer DAC is VREFHIx or VDAC, the VSSA is a low reference voltage.

(5) The high reference voltage of COMP is VDDA, the VSSA is a low reference voltage.

Figure 36 Analog Subsystem Block Diagram

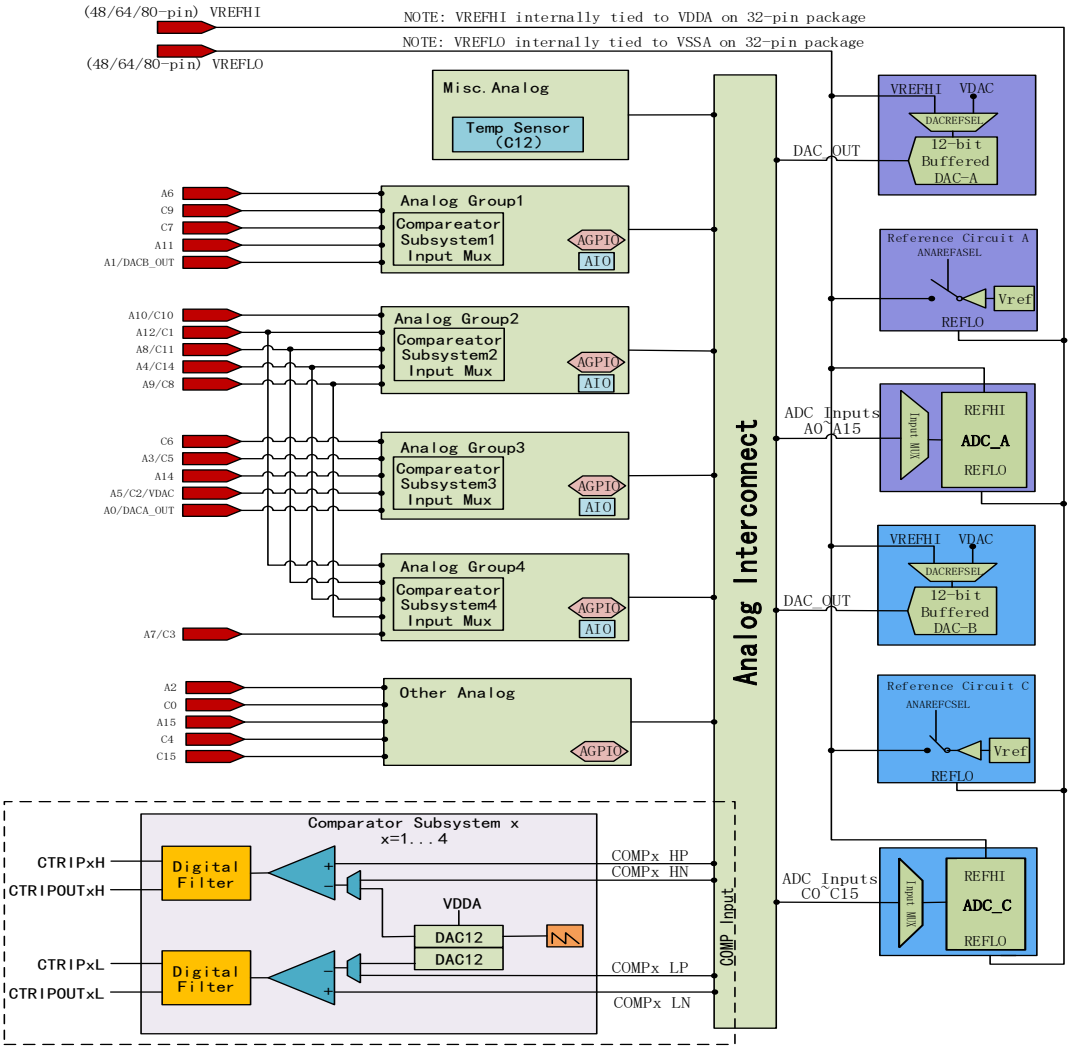
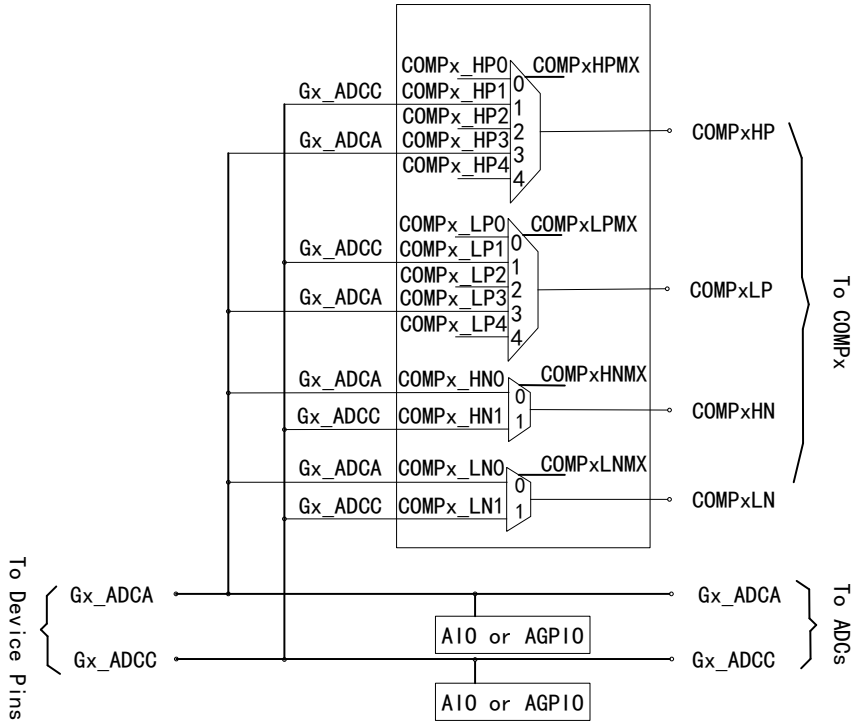


Figure 37 Analog Group Connection



26.4 Functional description

26.4.1 Analog signal

Table 92 Analog Signals

Signal name	Description
AIOx	Analog input port that can be reused as a digital input
Ax	ADCA Input
Cx	ADCC Input
VDAC	Optional external reference voltage for on-chip DAC. There is a 100pF VSSA capacitor on this pin, and neither the ADC input nor the DAC reference can be disabled. If it is used as the reference for on-chip DAC, place at least a 1μF capacitor on this pin.
DACx_OUT	Buffer DAC output
COMPx_DACH	High-level DAC Output of Comparator Subsystem
COMPx_DACL	Low-level DAC Output of Comparator Subsystem
COMPx_HPy	Positive Input of High-side comparator of Comparator Subsystem
COMPx_HNy	Negative Input of High-side comparator of Comparator Subsystem
COMPx_LPy	Positive Input of Low-side comparator of Comparator Subsystem
COMPx_LNy	Negative Input of Low-side comparator of Comparator Subsystem
TempSensor	Internal temperature sensor

26.4.2 Analog pins and internal connection

Table 93 Analog Pins and Internal Connection

Pin Name	Pin/Package				ADC		DAC	Comparator Subsystem (Multiplexer)				AIO 输入/ GPIO
	LQFP80	LQFP64	HTQFP48	VQFN32	A	C		High positive （HPMXSEL）	High Negative (HNMXSEL)	Low positive (LPMXSEL)	Low Negative (LNMXSEL)	
VREFHI	20	16	12	-(4)								
VREFLO	21	17	13	-(4)	A13	C13						
Analogue Group 1								COMP1				
A6	10	6	4(1)	2(1)	A6	-		2		2		GPIO228(3)
C9	13	9	6	4	-	C9		0		0		GPIO224(3)
C7	14	10	7(1)	5(1)	-	C7		3	0	3	0	AIO233
A11	16	12	8	6(1)	A11	-		1	1	1	1	AIO237
A1/DACB_OUT	18	14	10	7(1)	A1	-	DACB_OUT	4		4		AIO232
Analogue Group 2								COMP2				
A10/C10	29	25	21	13(1)	A10	C10		3	0	3	0	GPIO230(3)
Analogue Group 3								COMP3				
C6	11	7	4(1)	2(1)	-	C6		0		0		GPIO226(3)
A3/C5	12	8	5	3	A3	C5		3	0	3	0	GPIO242(3)
A14	15	11	7(1)	5(1)	A14	-		4		4		AIO239
A5/C2/VDAC	17	13	9	6(1)	A5	C2	VDAC	1	1	1	1	AIO244
A0/DACA_OUT	19	15	11	7(1)	A0	-	DACA_OUT	2		2		AIO231
Analogue Group 4								COMP4				

Pin Name	Pin/Package				ADC		DAC	Comparator Subsystem (Multiplexer)				AIO 输入/ GPIO
	LQFP80	LQFP64	HTQFP48	VQFN32	A	C		High positive （HPMXSEL）	High Negative (HNMXSEL)	Low positive (LPMXSEL)	Low Negative (LNMXSEL)	
A7/C3	23	19	15	8(1)	A7	C3		1	1	1	1	AIO245
Analogue Group 2/4								COMP2/4				
A12/C1	22	18	14	8(1)	A12	C1		COMP2：1 COMP4：2	1	COMP2：1 COMP4 ：2	COMP2：1	AIO238
A8/C11	24	20	16	9	A8	C11		COMP2：4 COMP4 ：4		COMP2：4 COMP4：4		AIO241
A4/C14	27	23	19	12	A4	C14		COMP2：0 COMP4：3	COMP4：0	COMP2：0 COMP4：3	COMP4：0	AIO225
A9/C8	28	24	20	13(1)	A9	C8		COMP2：2 COMP4：0		COMP2：2 COMP4：0		GPIO227(3)
Other analogue												
Temperature sensors(2)	-	-	-	-	-	C12		COMP2：5				
A2	4	2	2	32	-	-						GPIO28(3)(5)
C0	33	27	22	-	-	-						GPIO20(3)
A15	34	28	-	-	-	-						GPIO21(3)
C4	35	29	23	-	-	-						GPIO13(3)
C15	36	30	24	-	-	-						GPIO12(3)

Note:

(1) The signal is bonded to another signal to form a single pin on this package.

(2) This is an internal connection only and is not brought out to the device pins.

(3) These analogue pins support full digital input and output functionality, known as AGPIO. By default, AGPIO is not enabled; that is, both analogue and digital functions are disabled. For configuration details, please refer to the section ‘Digital Input and Output on ADC Pins (AGPIO)’.

(4) On the 32-pin RHB package, VREFHI is internally connected to VDDA, and VREFLO is internally connected to VSSA.

(5) The functions of GPIO28 and Comparator 4 (COMP4) cannot be used simultaneously.

26.4.3 Optimize power-on time

The ADC and DAC share a reference circuit. When using an application with one or more modules, a shared reference circuit can optimize the power-on time. If one module of the shared reference circuit has been initialized in the internal reference mode, the subsequent modules can optimize their entire power-on time by subtracting the reference power-on time from the required minimum power-on time.

When switching between 0V~2.5 V and 0V~3.3 V, there is also a waiting time for powering on the internal reference mode. Please refer to the *Datasheet* for relevant waiting time values.

26.4.4 Analog input port that can be reused as a digital input (AIOs)

The GPIO on AIOs port H, with only digital input function and only in input mode, multiplexed with analog pins. These pins are generally used as analog pins, and GPIO is in a high-impedance state. The GPHAMSEL register can be used to configure the digital or analog operations of these pins.

If the adjacent channels are used for analog function, the slew rate of signals connected to AIOs should be limited. Otherwise, connecting digital signals with sharp edges to AIOs may result in crosstalk with the adjacent analog signals.

26.4.5 Analog input/output that can be reused as digital input (AGPIOs)

Agpios are GPIOs that are multiplexed with analog pins and have full digital input and output capabilities.

AGPIOs is not connected by default and must be configured. The simulation function can be enabled by setting the GPXAMSEL register of the simulation subsystem. Setting the Universal Input/Output (GPIO) GPXAMSEL register enables the digital function.

Table 94 AGPIOs Configuration

AGPIOCTRLx.GPIOy (Default = 0)	**GPxAMSEL.GPIOy (Default = 1)**	**Pin connection**	
		GPIOy	ADC
0	0	√	-
0	1	There is generally no signal connected to the AGPIO pin. Another row in the table must be selected for the pin functions	
1	0	√	-
1	1	-	√

If the adjacent channels are used for analog function, the edge rate of signals connected to AGPIOs should be limited. Otherwise, connecting digital signals with sharp edges to AGPIOs may result in crosstalk with the adjacent analog signals.

26.5 Register bank address

Table 95 Register Bank Address

Device register	Register bank	Start address	End address
ASRegs	AS_REGS	0x5002_8000	0x5002_83FF

26.6 Register address mapping

Table 96 Register Address Mapping

Register name	Register description	Offset address	WRPRT
CONFIGLOCK	Control Register Lock Register	0xBC	√
TSNSCTL	Temperature sensor control register	0xC0	√
ANAREFCTL	Analog reference control register	0xD0	√
VMONCTL	Voltage monitor control register	0xE0	√
CMPHPMXSEL	COMP_HP select source register	0x104	√
CMPLPMXSEL	COMP_LP select source register	0x108	√
CMPHNMXSEL	COMP_HN select source register	0x10C	√
CMPLNMXSEL	COMP_LN select source register	0x10E	√
ADCDACLOOPBACK	Loopback register	0x110	-
LOCK	Lock register	0x11C	√
AGPIOCTRLA	AGPIO Control Register	0x214	√
AGPIOCTRLH	AGPIO Control Register	0x230	√

26.7 Register functional description

26.7.1 Control Register Lock Register (CONFIGLOCK)

Offset address: 0xBC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	Reserved			0h
3	AGPIOCTRL	R/Wsonce	Lock all AGPIOCTRL registers Setting this bit will prevent any subsequent writes to this register. This bit can only be cleared by a reset.	0h
15:4	Reserved			0h

26.7.2 Temperature sensor control register (TSNSCTL)

Offset address: 0xC0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ENABLE	R/W	Temperature Sensor Enable 0: Disable 1: Enable	0h
15:1	Reserved			0h

26.7.3 Analog reference control register (ANAREFCTL)

Offset address: 0xD0

Reset type: XRSn

Field	Name	R/W	Description	Reset value
1:0	ANAREFASEL	R/W	Analog reference A mode select If multiple VREFHI pins are connected together, the reference voltage of the connecting pins will always be the same. 00: Internal reference mode 01: Reserved 10: Reserved 11: External reference mode	3h
2	ANAREFCSEL	R/W	Analog reference C mode select If multiple VREFHI pins are connected together, the reference voltage of the connecting pins will always be the same. 0: Internal reference mode 1: External reference mode	1h
3	Reserved			1h
7:4	Reserved			0h
9:8	ANAREFA2P5SEL	R/W	Analog reference A 2.5V source select In the internal reference mode, this bit selects the voltage driven from the internal reference buffer to the VREFHI pin. 00: Internal 1.65V reference mode 01: Reserved 10: Reserved 11: Internal 2.5V reference mode Can select a reference range. The reference range ranges from 0 to 2.5V when the 2.5V reference mode is configured, and from 0 to 3.3V when the 1.65V reference mode is configured. When switching modes, sufficient time must be allowed for the external capacitor to charge to the new voltage before the ADC or buffer DAC can be used. If multiple VREFHI pins are connected together, the reference voltage of the connecting pins will always remains as the same settings.	0h

Field	Name	R/W	Description	Reset value
10	ANAREFC2P5SEL	R/W	Analog reference C 2.5V source select In the internal reference mode, this bit selects the voltage driven from the internal reference buffer to the VREFHI pin. 0: Internal 1.65V reference mode 1: Internal 2.5V reference mode Can select a reference range. The reference range ranges from 0 to 2.5V when the 2.5V reference mode is configured, and from 0 to 3.3V when the 1.65V reference mode is configured. When switching modes, sufficient time must be allowed for the external capacitor to charge to the new voltage before the ADC or buffer DAC can be used. If multiple VREFHI pins are connected together, the reference voltage of the connecting pins will always remains as the same settings.	0h
15:11	Reserved			0h

26.7.4 Voltage monitor control register (VMONCTL)

Offset address: 0xE0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	Reserved			0h
8	BORLVMONDIS	R/W	BORL disable on VDDIO Whether the BOR circuit will be triggered when VDDIO is below the lower BOR threshold of VDDIO. 0: Enable, it will be triggered 1: Disable, it will not be triggered	0h
15:9	Reserved			0h

26.7.5 COMP_HP select source register (CMPHPMXSEL)

Offset address: 0x104

Reset type: XRSn

Field	Name	R/W	Description	Reset value
2:0	CMP1HPMXSEL	R/W	Select source on the COMP1_HP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
5:3	CMP2HPMXSEL	R/W	Select source on the COMP2_HP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group	0h

Field	Name	R/W	Description	Reset value
			Connection Diagram. The range of values is 0~4, and others are reserved.	
8:6	CMP3HPMXSEL	R/W	Select source on the COMP3_HP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
11:9	CMP4HPMXSEL	R/W	Select source on the COMP4_HP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
31:12	Reserved			0h

26.7.6 COMP_LP select source register (CMPLPMXSEL)

Offset address: 0x108

Reset type: XRSn

Field	Name	R/W	Description	Reset value
2:0	CMP1LPMXSEL	R/W	Select source on the COMP1_LP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
5:3	CMP2LPMXSEL	R/W	Select source on the COMP2_LP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
8:6	CMP3LPMXSEL	R/W	Select source on the COMP3_LP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h
11:9	CMP4LPMXSEL	R/W	Select source on the COMP4_LP input For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram. The range of values is 0~4, and others are reserved.	0h

Field	Name	R/W	Description	Reset value
31:12	Reserved			0h

26.7.7 COMP_HN select source register (CMPHNMXSEL)

Offset address: 0x10C

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	CMPxHNMXSEL	R/W	Select source on the COMPx_HN input (x=1-4) For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram.	0h
15:4	Reserved			0h

26.7.8 COMP_LN select source register (CMPLNMXSEL)

Offset address: 0x10E

Reset type: XRSn

Field	Name	R/W	Description	Reset value
x-1	CMPxLNMXSEL	R/W	Select source on the COMPx_LN input (x=1-4) For the specific connection of the COMP input multiplexer, please refer to Analog Pin and Internal Connection Table. For the general structure, please refer to Analog Group Connection Diagram.	0h
15:4	Reserved			0h

26.7.9 Loopback register (ADCDACLOOPBACK)

Offset address: 0x110

Reset type: The KEY bit is SYSRSn, and others are XRSn

Field	Name	R/W	Description	Reset value
0	ENLB2ADCA	R/W	Loopback COMPDACA output to ADCA When this bit is set, the CHANSEL of ADC will be overwritten. 0: The loop is disconnected 1: Loopback COMPDACA is output to ADCA.	0h
1	Reserved			0h
2	ENLB2ADCC	R/W	Loopback COMPDACA output to ADCC When this bit is set, the CHANSEL of ADC will be overwritten. 0: The loop is disconnected 1: Loopback COMPDACA is output to ADCC.	0h
15:3	Reserved			0h
31:16	KEY	R-0/W	Key	0h

Field	Name	R/W	Description	Reset value
			The value 0xA5A5 must be included in this bit so that it can be written to this register. Otherwise, the register will remain in the state before the attempt to write. The read operation returns 0.	

26.7.10 Lock register (LOCK)

Offset address: 0x11C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TSNSCTL	R/WSonce	TSNSCTL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
1	ANAREFCTL	R/WSonce	ANAREFCTL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
2	VMONCTL	R/WSonce	VMONCTL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
4:3	Reserved			0h
5	CMPHPMXSEL	R/WSonce	CMPHPMXSEL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
6	CMPLPMXSEL	R/WSonce	CMPLPMXSEL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
7	CMPHNMXSEL	R/WSonce	CMPHNMXSEL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
8	CMPLNMXSEL	R/WSonce	CMPLNMXSEL Register Lock This bit controls writing to the corresponding register, and setting 1 disables it. This bit can be cleared by a reset.	0h
31:9	Reserved			0h

26.7.11 AGPIO Control Register (AGPIOCTRLA)

Offset address: 0x214

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	Reserved			0h
12	GPIO12	R/W	One-time configuration of GPIO12 Used to determine whether the AGPIO function is enabled: 0: Disable AGPIO 1: Enable AGPIO	0h
13	GPIO13	R/W	One-time configuration of GPIO13 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
19:14	Reserved			0h
20	GPIO20	R/W	One-time configuration of GPIO20 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
21	GPIO21	R/W	One-time configuration of GPIO21 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
27:22	Reserved			0h
28	GPIO28	R/W	One-time configuration of GPIO28 Used to determine whether the AGPIO function is enabled: 0: Disable AGPIO 1: Enable AGPIO	0h
31:29	Reserved			0h

26.7.12 AGPIO Control Register (AGPIOCTRLH)

Offset address: 0x230

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GPIO224	R/W	One-time configuration of GPIO224 Used to determine whether the AGPIO function is enabled: 0: Disable AGPIO 1: Enable AGPIO	0h
1	Reserved			0h

Field	Name	R/W	Description	Reset value
2	GPIO226	R/W	One-time configuration of GPIO226 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
3	GPIO227	R/W	One-time configuration of GPIO227 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
4	GPIO228	R/W	One-time configuration of GPIO228 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
5	Reserved			0h
6	GPIO230	R/W	One-time configuration of GPIO230 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
17:7	Reserved			0h
18	GPIO242	R/W	One-time configuration of GPIO242 Used to determine whether to enable the AGPIO function: 0: Disable AGPIO 1: Enable AGPIO	0h
31:19	Reserved			0h

27 Analog-to-digital converter (ADC)

27.1 Full Name and Abbreviation Description of Terms

Table 97 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Multiplexer	MUX
Sample and hold	S/H
ADC Start of Conversion	ADCSOC

27.2 Introduction

Two 12-bit, 3.72 MSPS modules, each with a maximum of 16 channels, are connected to the AHB bus.

The 12-bit precision ADC includes a wrapper and a core. The wrapper is based on the start of conversion (SOC) and consists of digital circuits that configure and control the ADC. These circuits include result registers, programmable conversion logic, analog circuit interfaces, peripheral bus interfaces, post-processing circuits, and other on-chip module interfaces. The core is made up of analog circuits, which include sampling/holding (S/H) circuits, successive approximation circuits, channel selection MUX, reference voltage circuits, and other analog support circuits. Each ADC module consists of a sampling/holding (s/h) circuit. Multiple ADC are allowed to sample simultaneously or operate independently.

27.3 Main characteristics

(1) Internal reference voltage options of ADC: 2.5V or 3.3V

(2) Single-ended conversion mode

(3) 12-bit precision resolution

(4) External reference is set by VREFHI and VREFLO pins

(5) Burst mode

(6) 16 result registers, which can be addressed separately

(7) 16-channel input multiplexer

(8) 16 configurable SOC

(9) 4 flexible PIE interrupts

(10) Trigger source

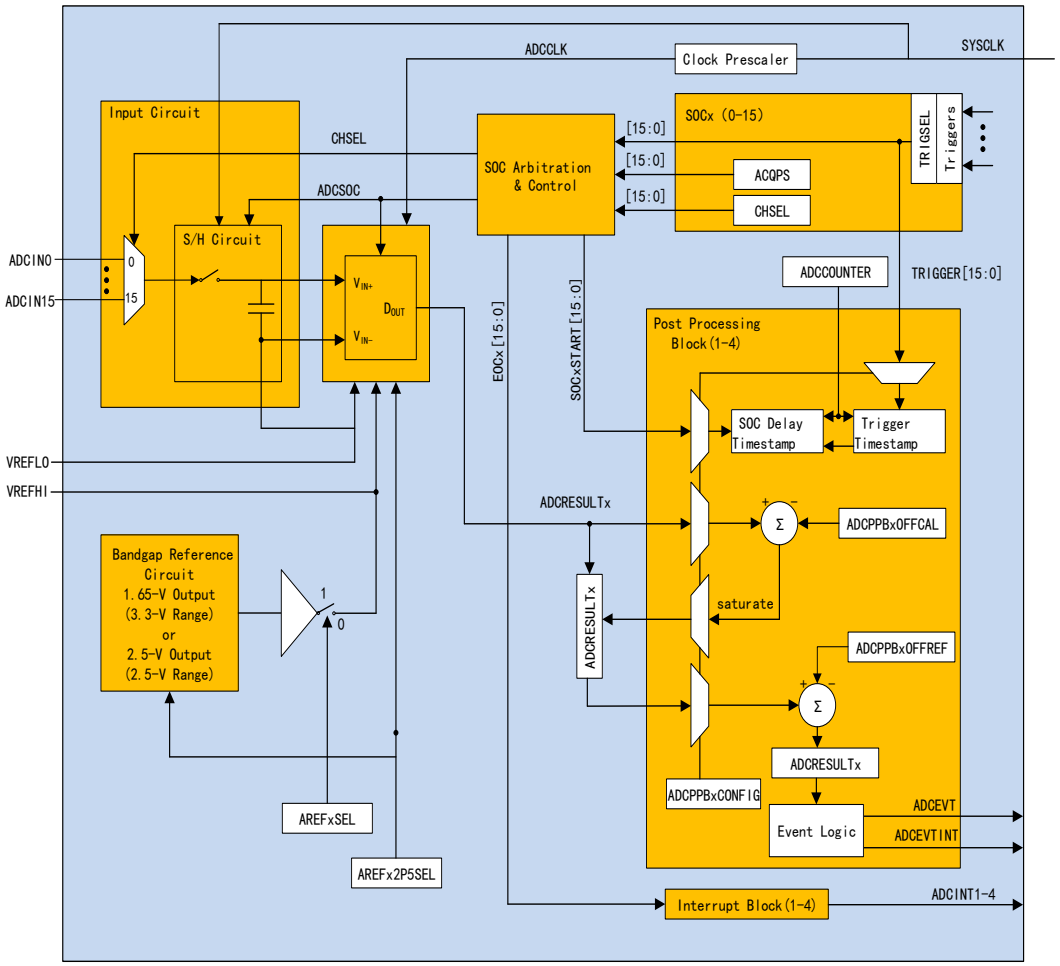
- Software starts immediately
- TMR0/1/2
- ADCINT1/2
- All PWM - ADCSOC A or B
- Input X-Bar INPUT5

(11) All the four post-processing blocks have:

- Saturation offset calibration
- Delay capture from trigger to sampling
- Set value calculation error
- High, low, and zero-crossing comparisons can trigger an interrupt or PWM

27.4 Structure block diagram

Figure 38 Structure Block Diagram



27.4.1 ADC Connections and Configuration with External Devices

For connections between the ADC channels and external I/O, refer to the multiplexing table in the Analogue Subsystem chapter.

27.5 Functional description

27.5.1 Configuration

ADC configuration is divided into independent control by SOC and global control by each ADC module, as follows:

Table 98 ADC Options and Configuration

Option	Configuration
Resolution	Not configurable (only 12 bits)
Signal mode	Not configurable (single-ended only)
Conversion channel	SOC level
Trigger source	SOC level (1)
Sampling window duration	
EOC position	Module level
Burst mode	Module level (1)
Clock	
Reference voltage source	Module level (external or internal) (2)

Note:

(1) In these configurations, writing different values to different ADC modules may result in asynchronous operation of the ADC.

(2) Multiple ADCs share a single VREFHI pin. At this time, the ADC that shares the reference pin must be configured as the same reference mode.

27.5.2 Clock configuration

The APBCLK clock provides a basic ADC clock, and it can be used to generate ADC sampling windows. ADCCLK is used for clock processing of the converter, and is determined by ADCCTL2 [PRESCALE]. The core requires approximately 10.5 ADCCLK cycles to process the voltage into conversion results. Users need to determine the duration of the required sampling window.

27.5.3 Signal mode

ADC supports single-ended signals. In single-ended mode, the input voltage of the converter is sampled through a pin ADCINx. For details, refer to VREFLO.

27.5.4 Reference voltage

External reference mode

Each ADC has one VREFHI input and one VREFLO input. In the external reference mode, in order to determine the input range of ADC conversion, these

two pins need to be used as ratio references. The external reference mode needs to connect to an external capacitor on the VREFHI pin. If there is no external VREFLO signal, VREFLO is internally connected to the analog ground VSSA of the device.

Internal reference mode

In the internal reference mode, the device drives the voltage to the VREFHI pin. The VREFHI and VREFLO pins are used to set the ADC conversion range, and the internal reference voltage can be configured as 2.5V or 1.65V. When configured as 1.65V, the valid ADC conversion input range is from VREFLO to 3.3V. The internal reference mode also needs an external capacitor on the VREFHI pin.

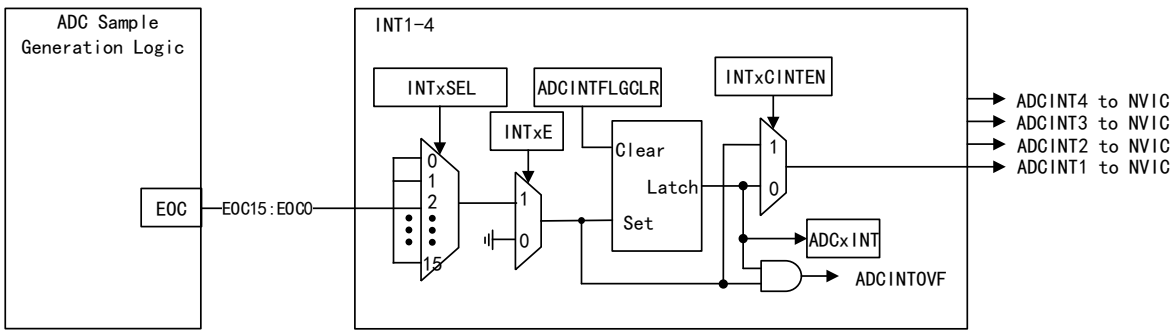
Select the reference mode

The reference voltage mode needs to be configured using SetVREF or ADC_setVREF function. Any of these functions can be loaded into the ADCOFFTRIM register. They cannot be directly written to the ANAREFCTL register for configuration. This function must be called at least once after the device is reset.

27.5.5 EOC and interrupt

Each SOC has a corresponding end of conversion (EOC) signal. The EOC signal can be used to trigger ADC interrupts. ADCCTL1 [INTPULSEPOS] determines the generation of EOC pulses at the end of the sampling window or at the end of voltage conversion. Each ADC module has 4 configurable ADC interrupts. These interrupts can be triggered by 16 EOC signals at will. The flag bit of ADCINTx can be directly read so as to determine whether the relevant SOC is completed or whether the interrupt can be passed to NVIC. ADCCTL1 [ADCBSY] cleared to zero indicates that the ADC has been ready to process the next conversion. To determine if the SOC sequence is complete, the ADCINTx flag needs to be linked to the last SOC in the sequence and be monitored.

Figure 39 ADC Interrupt Structure Block Diagram



Interrupt overflow

Interrupt overflow occurs when the EOC signal sets a flag that has already been set in the ADCINTFLG register. In general, overflow interrupt will not be passed to the NVIC module. When the flag in the ADCINTFLG register overflows, the corresponding flag in the ADCINTOVF register will be set, and this flag is only used to detect whether an overflow has occurred. When ADC interrupt overflow might occur, the application needs to check the appropriate overflow flag in the ISR or background loop, and take corresponding actions when an overflow is detected.

Continuous interrupt mode

By default, this mode is disabled and no additional overlapping interrupts will be sent to NVIC. The INTxCONT bit determines how to handle interrupts when ADCINTFLG is not cleared from previous interrupts. By activating this mode, the ADC interrupts will always reach NVIC. If an interrupt occurs when ADCINTFLG is set, regardless of the configuration, ADCINTOVF is still set.

Early interrupt configuration mode

DELAY determines the time for ADC interrupts to enter the early interrupt mode. The early interrupt mode allows applications to enter the ADC interrupt service program before the ADC result is ready. This allows the application to carry out preliminary work, and when the ADC result is available, it can be operated immediately. But if the timing of early interrupt is too early, the application needs to wait until the updated ADC results are available. The time at which the ADC interrupt is entered in early interrupt mode can be configured via the DELAY field in the ADCINTCYCLE register.

- INTPULSEPOS can be cleared to put the ADC in an early interrupt mode so that the configurable interrupt times can be used
- When INTPULSEPOS is set to 1, writing DELAY will not affect the interrupts
- The DELAY value determines the additional number of APBCLK cycles after the falling edge of SOC pulse and before the ADCINTx flag is set
- If the DELAY value exceeds EOC, both ADC interrupt and EOC will be generated simultaneously

27.5.6 Resolution

12-bit resolution is supported. The resolution determines the precision at which the analog range is quantified into digital values.

27.5.7 Expected conversion results

Based on the given analog input voltage, the analog-to-digital conversion formula is as follows:

- When ADCINx≤VREFLO, ADC_RULTx = 0

- When VREFLO＜ADCINx＜VREFHI, $\mathrm{ADC_RULTx} = \frac{4096(\mathrm{ADCINx}-\mathrm{VREFLO})}{\mathrm{VREFHI}-\mathrm{VREFLO}}$
- When ADCINx≥VREFHI, ADC_RULTx =4095

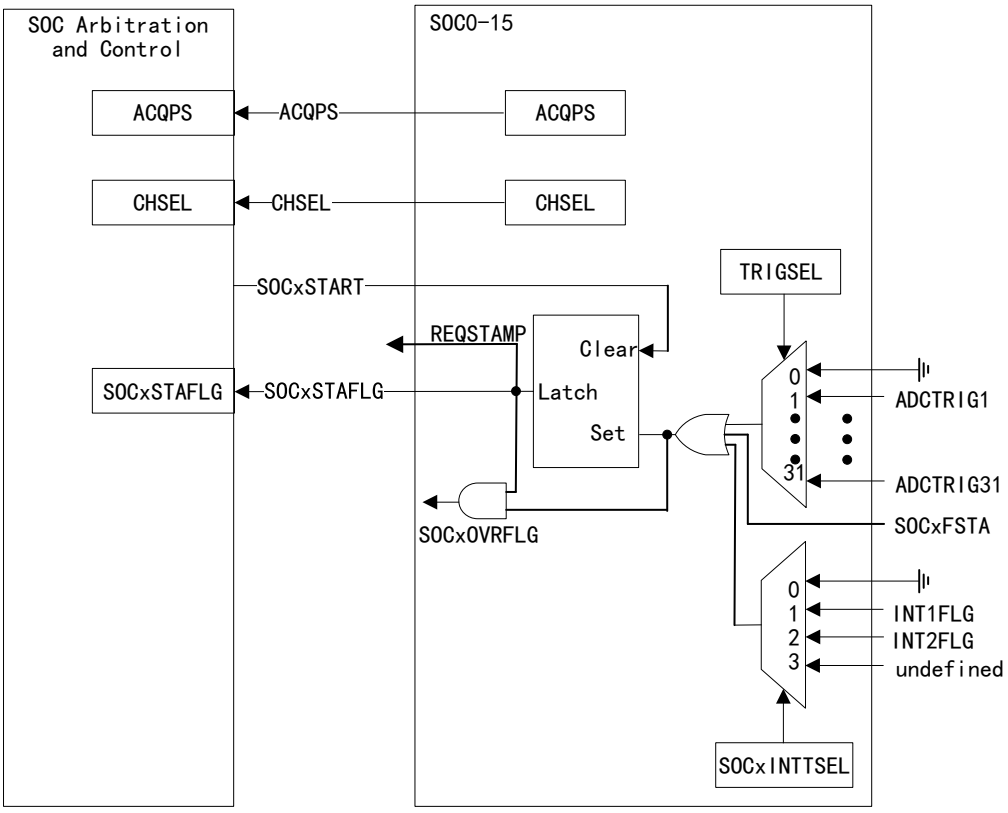
27.5.8 Explanation to the conversion results

According to the given ADC conversion result, the corresponding analog input is as follows. This corresponds to the center of the analog voltage range that may produce this conversion result.

- When ADCRESULTx=0, ADCINx≤VREFLO
- When 0＜ADCRESULTx＜4095,$\mathrm{ADCINx} = (\mathrm{VREFHI} - \mathrm{VREFLO})(\frac{ADC_RULTx}{4096}) + \mathrm{VREFLO}$
- When ADCRESULTx=4095, ADCINx≥VREFHI

27.5.9 SOC

Figure 40 SOC Structure Block Diagram



Working principle

ADC completes triggering and conversion sequence through configurable SOC. Each SOC is a configuration set, and it determines the single conversion of the single channel. There are three configurations in this set: the duration of the sampling window, the channel to be converted, and the trigger source for

initiating the conversion.

After receiving the trigger configured for SOC, the wrapper will ensure that the specified sampling window duration is used to capture the specified channel. Multiple SOC can be configured for the same sampling window, channel, and/or trigger as needed:

- Configuring multiple SOC to use the same channel and trigger will enable oversampling
- Configuring multiple SOC to use the same trigger will enable the trigger to generate a series of conversions

SOC Configuration

Each SOC has a separate control register, which that can configure the trigger source, channels to be converted, and sampling window duration for SOCx.

Trigger

Each SOC can be configured to activate on any input trigger. Select the main trigger by ADCSOCxCTL [TRIGSEL]. The specific selection is as follows:

- Disable in software
- TMR0/1/2
- ADCINT1/2
- GPIO: Input X-Bar INPUT5
- All PWM - ADCSOCA or ADCSOCB

ADC sampling window

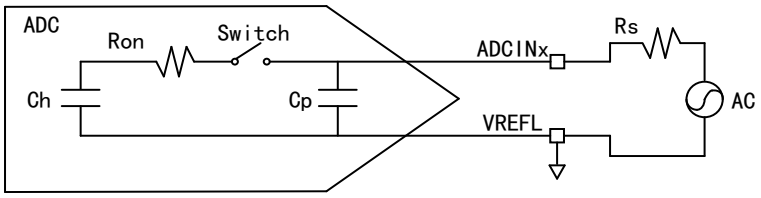
The ability of each external signal source to drive analog signals varies. The signal source needs to charge the sampling capacitor in the ADC core to be within 0.5 LSBs of the signal voltage so as to reach the rated resolution. The sampling window is configured by ADCSOCxCTL [ACQPS]. The sampling window is the amount of time the sampling capacitor is allowed to charge, which can be set to a value between 0 and 511. Therefore, the duration of the sampling window is: sampling window=(ACQPS+1) x (APB clock cycle time).

It can be seen that the duration of the sampling window is based on APBCLK. The selected sampling window duration must be at least as long as one ADCCLK cycle.

ADC input model

In single-ended operation, the input model needs to be used together with the actual signal source impedance so as to determine the sampling window duration.

Figure 41 Single-ended Input Model



Channel selection

Each SOC can be configured to convert any ADC channel. It is selected by ADCSOCxCTL [CHSEL].

Table 99 Selection Input Pin Channels

Input mode	CHSEL	Input
Single-ended type	0	ADCIN0
	1	ADCIN1
	2	ADCIN2
	3	ADCIN3
	4	ADCIN4
	5	ADCIN5
	6	ADCIN6
	7	ADCIN7
	8	ADCIN8
	9	ADCIN9
	10	ADCIN10
	11	ADCIN11
	12	ADCIN12
	13	ADCIN13
	14	ADCIN14
	15	ADCIN15

27.5.10 SOC Configuration

Single conversion of PWM trigger

To configure ADCA to perform a single conversion on channel ADCIN1 when the PWM timer reaches its cycle matching, PWM3 needs to be configured first to generate SOCA or SOCB signals. Any one of the 16 SOC can be used. If the frequency of APBCLK is 100MHz and a sampling window of 100 ns is required, the duration of the sampling window should be 100 ns/10 ns=10 cycles. So ACQPS should be set to 10-1=9.

When PWM3 matches its cycle and generates a SOCB signal, if ADC is idle, it will start to sample the channel ADCINA1 immediately. If ADC is busy, ADCINA1 will only start sampling when channel gets the priority. The steps are as follows:

(1) The ADC control logic will sample ADCINA1 with a specified 100ns sampling window width.

(2) After the sampling is completed, the ADC will immediately start converting the sampling voltage into a digital value

(3) After the conversion is completed, the result will be presented in the ADCRESULT5 register

Oversampling conversion of PWM trigger

When PWM3 matches its cycle and generates a SOCB signal, if ADC is idle, it will start to sample the channel ADCINA1 immediately. If ADC is busy, ADCINA1 will only start sampling when SOC5 gets the priority. Once the conversion of SOC5 is completed, SOC6 will start converting ADCINA1. Generally, the conversion will be made in sequence, and SOC5 converts first, depending on the position of the polling pointer when the PWM trigger is received. The correspondence between the conversion results and the placed registers is as follows:

Table 100 Correspondence between Conversion Results and Registers

Conversion result	Register
SOC5	ADCRESULT5
SOC6	ADCRESULT6
SOC7	ADCRESULT7
SOC8	ADCRESULT8

Multiple conversions of timer triggers

TMR2 is used to generate triggers. To design a sampling scheme that includes multiple signals, the following steps need to be followed:

(1) List the signals and required sampling windows

(2) Calculate the number of APBCLK cycles required for each signal

(3) Set ACQPS

(4) Confirm that the ADC pins are connected to each signal

(5) Confirm the CHSEL value

Table 101 Multi-signal Sampling

Signal name	Sampling window requirement (ns)	APBCLK cycle of sampling window =Sampling window requirement/10ns (ns)	ACQPS value=Sampling window APBCLK cycle-1	ADC pins	CHSEL value
Signal 1	Min=240	24	23	ADCINA5	5
Signal 2	Min=883.333	89	88	ADCINA0	0
Signal 3	Min=220	22	21	ADCINA3	3
Signal 4	Min=585	59	58	ADCINA2	2

When TMR2 generates an event, SOC0, SOC1, SOC2, and SOC3 will be sampled and converted in sequence. Generally, the conversion will be made in sequence, and SOC0 converts first, depending on the position of the polling pointer when the timer trigger is received. The correspondence between the conversion results and the placed registers is as follows:

Table 102 Correspondence between Conversion Results and Registers

Conversion result	Register
ACINA5	ADCRESULT0
ACINA0	ADCRESULT1
ACINA3	ADCRESULT2
ACINA2	ADCRESULT3

Software trigger of SOC

No matter whether the SOC is configured to accept a specific trigger, as long as a bit is written into the ADCSOCFRC1 register, the software trigger can set the SOC to be converted.

27.5.11 ADC conversion priority

When multiple SOC flags are set simultaneously, their conversion sequence needs to be determined. The default priority method is pooling, determined by the polling pointer. ADCSOCPRICTL [RRPOINTER] points to the last converted SOC. The highest-priority SOC is assigned to the next value larger than the RRPOINTER value, and after SOC15, it loops back to SOC0. When RRPOINTER is equal to 16, the highest-priority SOC is SOC0. When reset, this value is 16, because 0 indicates that a conversion has occurred. When the ADC is reset or the reset value is written to the ADCSOCPRICTL register, RRPOINTER will be reset. Reset the ADC by writing and clearing the SOFTPRES register corresponding to the ADC instance.

ADCSOCPRICTL [SOCPRIORITY] can assign the high priority from a single SOC to all SOC. The assigned SOC will insert itself into the next conversion after the current conversion is completed. After the conversion is completed, the polling will continue to run where it was interrupted. If two high-priority SOC are triggered simultaneously, the lower-priority SOC will have the priority. The high-

priority mode is first assigned to SOC0 and then increases in numerical order. The value written to the SOCPRIORITY bit defines the first SOC with a low priority. If SOCPRIORITY is configured to 3, SOC0–SOC2 are assigned high priority, whilst SOC3–SOC15 operate in polling mode.

27.5.12 Burst mode

Setting the ADCBURSTCTL [BURSTEN] bit can configure the ADC wrapper as the burst mode. In this mode, the TRIGSEL field is ignored (only for SOCs configured for polled operation; does not apply to high-priority SOCs).The burst mode allows a single trigger to pass through one or more cycles of SOC at a time. It is only applicable to polling SOC. ADCBURSTCTL [BURSTTRIGSEL] controls the triggering of all polling SOC. After receiving the burst trigger, the ADC wrapper sets (BURST_SIZE+ 1) SOC. Set the first SOC based on the highest-priority SOC of the polling pointer, and then set other SOC until BURST_SIZE SOC is set. When the ADC is configured to the burst mode, the next burst trigger can be received only after each burst conversion is completed.

Example

All signals are converted using a sampling window with a width of 20 APBCLK cycles, and each SOC can be configured with different durations. The burst mode can sample a set of different signals on other triggers. For example, ADCIN7 and ADCIN5 are converted on the first trigger of TMR2 and each subsequent trigger.

If the ADC is idle when the first TMR2 trigger is received, SOC12 and SOC13 will be immediately converted. If the ADC is busy, it will be converted only when the SOC of SOC12 and SOC13 gets the priority. After SOC13 is completed, the polling pointer will give SOC14 the highest priority. So when the next TMR2 trigger is received, SOC14 and SOC15 will be suspended and converted. The subsequent triggers will continue to switch between converting SOC12 and SOC13, and converting SOC14 and SOC15. The correspondence between the conversion results and the placed registers is as follows:

Table 103 Correspondence between Conversion Results and Registers

Conversion result	Register
SOC12	ADCRESULT12
SOC13	ADCRESULT13
SOC14	ADCRESULT14
SOC15	ADCRESULT15

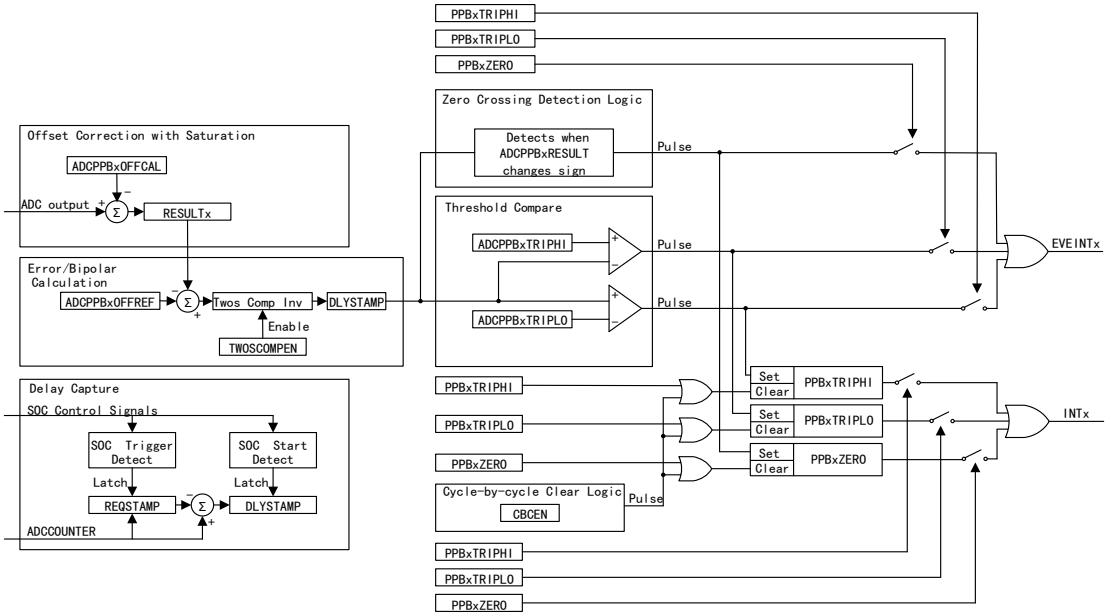
Note: Although this example only switches between two groups of conversion, similar methods can be used to implement three or more different conversions.

27.5.13 Post-processing module

Each ADC contains four post-processing modules. These blocks can be

associated using the ADCPPBxCONFIG and ADCRESULTx registers. Each PPB can simultaneously remove the offsets related to ADCIN channels, mark the zero-crossing points, subtract the reference value, and mark the upper or lower limits of comparison. Zero-crossing and comparison flags can also trip PWM and/or generate interrupts. PPB can also record the delay time between triggering SOC related to PPB and the actual start of sampling.

Figure 42 PPB Structure Block Diagram



PPB offset correction

The external sensors and signal sources may generate some offsets, which vary depending on channels, and the global trimming of ADC offset is insufficient to compensate for it. In order to save cycles in the strict control loop, the post-processing module can eliminate these offsets with zero overhead. To correct the offsets, first it is required to point ADCPPBxCONFIG to the desired SOC, and then write the offset correction value into the ADCPPBxOFFCAL register. Writing 0 to OFFCAL can disable the offset correction function and pass the original result to the ADCRESULT register. The post-processing block will automatically add or subtract the value of OFFCAL from the original conversion result and store it in the ADCRESULT register. This addition/subtraction will have a saturation of 4095 at the high end and 0 at the low end.

Note: Multiple PPB can be pointed to the same SOC, and then the actual ADCPPBxOFFCAL value is the value of the PPB with the highest value. Because all PPB points to SOC0 by default, you should be cautious when using PPB on SOC0. Otherwise, the offset correction of the lower-number PPB may be unintentionally overwritten by the higher-number PPB.

PPB error calculation

In general, bipolar signals can be used to control the calculation of the errors in the set or expected value. In special cases, the error must be calculated from the digital output of the ADC conversion. PPB can automatically perform these functions, reducing the delay from sampling to output and minimizing software overhead. To calculate the error, first it is required to point ADCPPBxCONFIG to the desired SOC, and then write a value into the ADCPPBxOFFCAL register. Writing 0 to OFFREF can disable the error calculation function and pass ADCRESULT to the ADCPPBxRESULT register. The post-processing module will automatically subtract the value in OFFREF from the ADCRESULT value and store it in the ADCPPBxRESULT register. Then a 32-bit result of sign extension will be generated. Setting the ADCPPBxCONFIG [TWOSCOMPEN] can selectively invert the calculated values before the calculation value is stored in the ADCPPBxRESULT register.

Note:

(1) The value written to OFFREF must be less than or equal to 12 bits.

(2) Since the ADCPPBxRESULT register is unique for each PPB, multiple PPB can point to the same SOC and different results can be obtained for each PPB.

PPB limit detection and zero-crossing detection

In special circumstances, limit detection will be performed on the ADC conversion results. PPB can automatically perform detection based on the upper or lower limit when ADCPPBxRESULT changes signs. The zero-crossing detection circuit considers the result of zero as positive. Based on these comparisons, trips and/or interrupts to PWM can be automatically generated, reducing the delay from sampling to PWM and minimizing the software overhead. To perform the detection, it is required to first point ADCPPBxCONFIG to the desired SOC, and then write the value to ADCPPBxTRIPHI [LIMITHI] and ADCPPBxTRIPLO [LIMITLO]. Once the limit is exceeded, LIMITHI or LIMITLO will be set in the ADCEVTSTAT register.

This function also has a security conscious application, and it can trigger PWM based on the ADC conversion out of range without any CPU intervention. The ADCEVTSTAT [PPBxZERO] bit is gated by EOC, and there are corresponding bits in the ADCEVTCLR register to clear these flags. The ADCEVTSEL register has corresponding bits that allow it to propagate through PWM. The ADCEVTSEL register has corresponding bits that allow it to propagate through NVIC. An NVIC is shared among all PPB of the given ADC module. Pointing multiple PPB to the same SOC can generate different PWM trips for high/low comparison and/or zero crossing.

PPB sampling delay capture

When multiple control loops are running asynchronously on the same ADC, the

ADC requests from two or more loops may conflict, causing delay to one sampling, and this is called measurement error. By understanding the occurrence time of delay and the amount of delay that has already occurred, extrapolation techniques are used for this software to reduce errors. Therefore, each PPB has a DLYSTAMP field that contains the number of APBCLK cycles between the triggering of associated SOC and the start of the conversion. This is achieved through a 12-bit counter located in the FREECOUNT field, which is based on APBCLK. When very slow conversion is used, the following situations can easily occur: if there are more than 4,096 APBCLK cycles between the SOC trigger and the actual start of SOC acquisition, FREECOUNT may overflow multiple times, resulting in incorrect DLYSTAMP values.

When the trigger of the relevant SOC arrives, the value of this counter will be loaded into REQSTAMP. When the actual example window of SOC starts, the value in REQSTAMP will be subtracted from the current DLYSTAMP value and stored in DLYSTAMP.

If the relevant SOC is triggered by software, sampling delay capture will not work. However, if software trigger of different SOC causes SOC delay related to PPB, it will correctly record the delay.

27.5.14 Open/short circuit detection circuit

It can be used to detect pin faults in the system. After the circuit is connected to the ADC input channel selection multiplexer, before the S+H circuit, performing write operation to the DETECTCFG field can control this circuit, and the circuit provides voltage to the input end during the S+H phase period of any conversion.

Figure43 Open-Circuit/Short-Circuit Detection Circuit

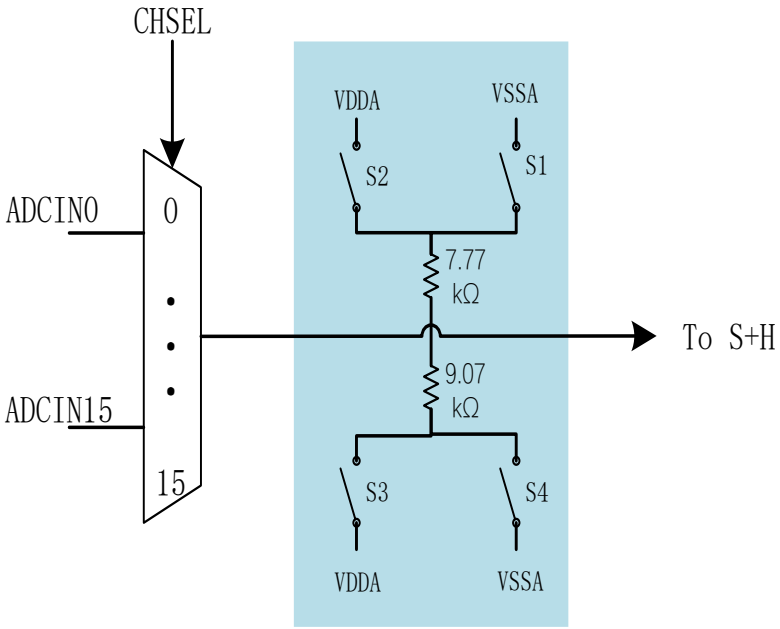


Table 104 DETECTCFG Configuration

DETECTCFG	Voltage source	S1	S2	S3	S4	Driver impedance
000	Disable	On	On	On	On	On
001	Zero scale	OFF	On	On	OFF	7.77K||9.07K
010	Full scale	On	OFF	OFF	On	7.77K||9.07K
011	9.07/16.84 VDDA	On	OFF	On	OFF	7.77K||9.07K
100	7.77/16.84 VDDA	OFF	On	OFF	On	7.77K||9.07K
101	Zero scale	On	On	On	OFF	7.77K
110	Full scale	On	On	OFF	On	7.77K
111	Zero scale	OFF	On	On	On	9.07K

Implementation

The signal source of shunt capacitor C_P and series resistor R_S, equivalent DETECTCFG resistor $R_{DETECTCFG}$, and voltage $V_{DETECTCFG}$ constitute the representative circuit implemented by DETECTCFG, and this circuit can serve as the basis for calculating the signal level entering the sampling capacitor. The DETECTCFG circuit provides equivalent input resistance $R_{DETECTCFG}$ and voltage source $V_{DETECTCFG}$. Please refer to the above table for specific configuration. When the DETECTCFG characteristic is enabled, if the signal source V_S is in the driving state, the input signal will be exported as S/H.

The input impedance R_S and C_P can be a part of the signal source, and they can pre-process the signal in the design or control the signal settling time to meet S/H requirements. Because the input path may affect the conversion result, it must be considered when using the DETECTCFG characteristic. The C_P values greater than several hundred pF require higher ACQPS to ensure that the signals at the input end is stable before conversion.

The configuration steps for the enable circuit are as follows:

(1) Configure ADC used for conversion.

(2) Configure the ADCOSDETECT register for the voltage divider connection.

(3) Start the conversion and check the conversion results.

The results need to be explained based on the factors driven by the input end and the values of R_S and C_P. If the V_S signal can be disconnected from the input pin, the circuit can be used to detect the input pins of disconnection and short circuit.

Detect open circuit input pin

Circulate various DETECTCFG configurations, and the input signal will be pulled towards the source voltage. When the pin is not turned on, the inputs with good driving strength will be minimally affected. If the pin is turned on, the sampling voltage will approach the voltage source shown in the above table.

Detect short circuit input pin

Circulate various DETECTCFG configurations, and the input signal will be pulled towards the source voltage. When the pins are not shorted, the inputs with limited driving strength will be pulled towards each source voltage. If the pin is shorted, the signal will remain at the same voltage.

27.5.15 Ascending order

When the device is powered on or the system is reset, the ADC will be powered off and disabled. The steps to power on the ADC are as follows:

(1) Set the ADC clock in the PCLKCR13 register.

(2) Set the ADC clock divider in ADCCTL2 [PRESCALE].

(3) Set the ADCCTL1 [ADCPWDNZ] bit to power on ADC.

(4) Allow delay before sampling.

If multiple ADC are powered on simultaneously, steps 1 and 3 can be completed separately for all ADC in one write instruction. As long as it occurs after all ADC start power-on, only one delay is needed.

27.5.16 Calibration

This function can calibrate the gain, offset, and linearity of ADC and buffer the offset of DAC in manufacturing and testing processes. These trimming settings are embedded into the reserved OTP memory. Appropriate factory configuration must be loaded to ensure that the ADC runs properly, and the trimming values also need to be factory settings. The boot ROM will call the calibration function, so the trimming value should be initially filled without intervention. If the trimming is cleared due to module reset or modification for other reasons, the calibration function can be called.

(1) The Device_cal() function will copy the trimming value of ADC and DAC offset from OTP memory to their respective registers.

(2) The trim function in Device_cal() can call ADC_setOFFSETTRIM(), ADC_setINLTRIM(), and DAC_setDACTRIM(). These functions obtain the trimming values from the destination of the analog module register and the OTP memory source location.

(3) Each combination of resolution and signal mode requires different trimming of offsets. The GetAdcOffsetTrimOTP (Uint16) function accepts input values corresponding to ADC, resolution, and signal mode, and

returns the corresponding trimming values of offset from OTP memory, and then moves it to ADCOFFTRIM.

ADC zero-offset calibration

The difference from 0 that occurs during VREFLO voltage conversion is called zero offset error, which includes positive and negative. To correct this error, equal amplitude and opposite polarity trimming needs to be written into the ADCOFFTRIM register in the ADC core. These values will be applied before ADCRESULTx is available. The timing of the results will not be affected, and the full dynamic range of the ADC will be maintained for all trimming values. The ADCOFFTRIM register can load the factory-calibrated offset amount for error correction using the GetAdcOffsetTrimOTP(Uint16) function. If necessary, the ADCOFFTRIM register can be modified to compensate for the additional offset errors caused by the application environment.

The steps to recalibrate the ADC offset in 12-bit single-ended mode are as follows:

(1) Set ADCOFFTRIM to +112 steps (0x70), and add an artificial offset to explain the negative offset that may reside in the ADC core.

(2) Perform the multiple of 16 conversions on VREFLO, and obtain the cumulative result.

(3) Divide the cumulative result by a multiple of 16.

(4) Set ADCOFFTRIM to 112- the result of step 3.

Note: Regardless of the converter resolution, the size of each ADCOFFTRIM is (VREFHI-VREFLO)/ 65536.

27.5.17 Timing

The process of converting the analog voltage into the digital value is divided into S+H phase and conversion phase. ADCCTL2 [PRESCALE] can generate ADCCLK for APBCLK frequency division. The APBCLK clock controls the sampling and holding circuit (S+H) of ADC, while the ADCCLK clock controls the ADC conversion process. The conversion time is an integer of the APBCLK cycle, approximately 10.5 ADCCLK cycles. Ensure that this duration exceeds the minimum S+H duration and 1 ADCCLK cycle.

Timing parameters

(1) t_{INT}: The time from the end of the S+H window to the setting of the ADCINT flag (if configured).

- If ADCCTL1 [$_{INTPULSEPOS}$] is set, t_{INT} is consistent with the conversion result latched into ADCRESULTx.

- If the INTPULSEPOS bit is 0, and ADCINTCYCLE [DELAY] is not 0, there will be a delay of DELAY APBCLK cycles before the ADCINTx flag is set. This delay can be used to enter ISR or trigger DMA when sampling is ready.
- If the INTPULSEPOS bit is 0, and ADCINTCYCLE [DELAY] is 0, t_{INT} will coincide with the end of the S+H window.
- If t_{INT} triggers a read of ADCRESULTx, ensure that the read occurs after the result is latched; otherwise, the previous result will be read.

(2) t_{LAT}: The duration from the end of the S+H window to the time of latching the ADC result to the ADCRESULTx register. If the ADCRESULTx register is read before this time, the previous conversion result will be returned.

(3) t_{EOC}: The duration from the end of the S+H window to the time when the S+H window of the next ADC conversion can start. Subsequent sampling can begin before the conversion result is latched.

(4) t_{SH}: Duration of S+H window. At the end of this window, the value on the S+H capacitor becomes the voltage to be converted into a digital value. The duration is determined by (ACQPS+1) APBCLK cycles. ACQPS can be configured separately for each SOC, so t_{SH} may be different for different SOC.

Note: No matter how the device clock is set, the value on the S+H capacitor will be captured approximately 5ns before the end of the S+H window.

Timing Diagram

The ADC conversion timing of two SOC in the following situations:

- SOC0 and SOC1 use the same trigger
- The polling pointer points to SOC0 and it converts first
- When a trigger occurs, no other SOC is converting or hanging
- ADCINTSEL is configured to set the ADCINTx flag at the end of SOC0 conversion

Figure 44 ADC Timing of 12-bit Mode in Early Interrupt Mode

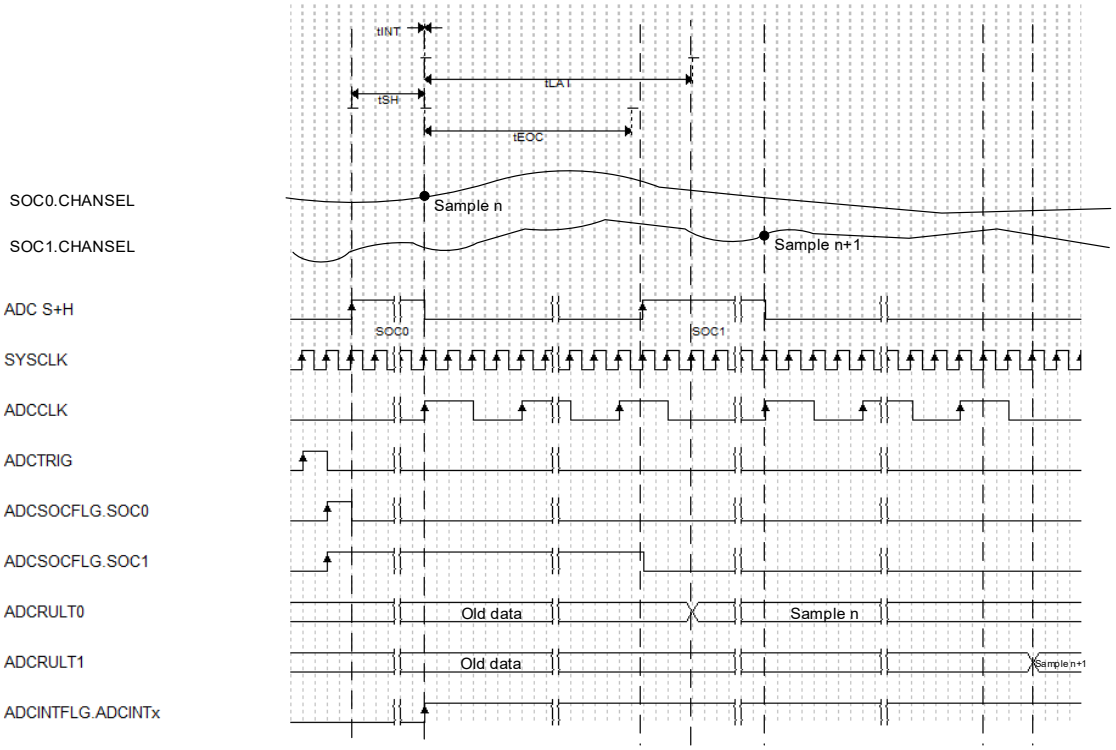


Figure 45 ADC Timing of 12-bit Mode in Delayed Interrupt Mode

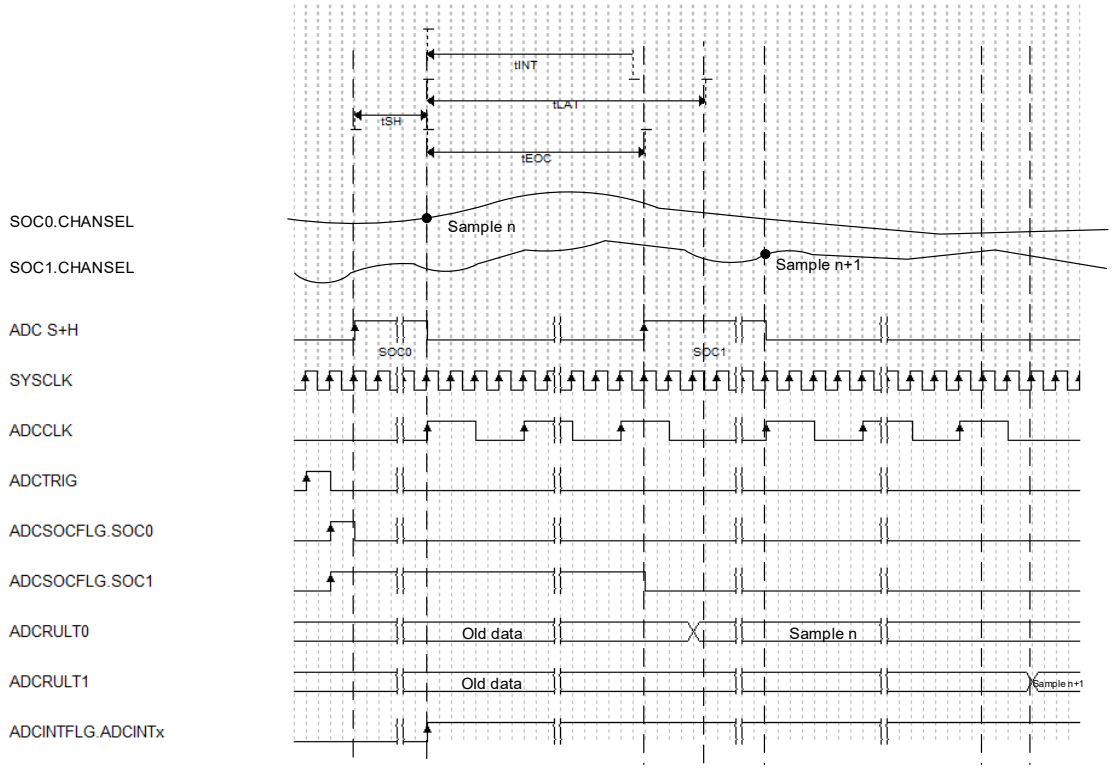


Table 105 ADC Timing of 12-bit Mode

ADCCLK prescale		APBCLK cycle			
Prescale ratio	PRESCALE	tLAT	tEOC	t_{INT} (early stage)	t_{INT} (later stage)
1	0	13	11	1	11
2	2	23	21	1	21
3	4	34	31	1	31
4	6	44	41	1	41
5	8	55	51	1	51
6	10	65	61	1	61
7	12	76	71	1	71
8	14	86	81	1	81

27.5.18 Synchronous operation

To produce the best effect, all ADC on the devices need to operate synchronously. Writing configurations to all ADC can be synchronized, and these configurations precisely align the sampling and conversion stages of all ADC. Writing the same values for the trigger selection of each ADC and the SOC configuration of ACQPS (S+H duration) can be achieved.

Basic synchronization

Configuring two SOC on ADCA and ADCC, each SOC with the same trigger selection and ACQPS value will result in synchronization between ADCA and ADCC. The principle of configuring two or more ADC is the same. The ACQPS values of SOC with the same number must be the same; otherwise, they can be different. Synchronous operation does not require a single global S+H time, and only the channels sampled simultaneously require the same S+H duration. Any channel selection value can be used for SOC. If a high-priority SOC is used, the priority of all ADC must be set to be consistent.

Synchronization with multiple trigger sources

To use multiple trigger sources and synchronization, each group of SOC needs to have the same trigger selection and ACQPS settings. When three SOC and two trigger sources are used, ADCA and ADCC can also be synchronized.

Any trigger source that TRIGSEL can choose can be used, except for software trigger. Failure to simultaneously issue software triggers for all ADC may result in asynchronous operation. If the ADCINTSOCSEL1 and ADCINTSOCSEL2 registers have the same configuration for all ADC and do not use software triggers to start the conversion chain, ADCINT1 and ADCINT2 can also be used as triggers.

Uneven synchronization of SOC number

If only one trigger source is used, one ADC can use more SOC than other ADC, but can still perform synchronous operation. If ADCA and ADCC are triggered by the same trigger source, with ADCA using 3 SOCs and ADCC using 2 SOCs, and if the trigger occurs again before all SOC completes the conversion, ADCC will immediately start the conversion on SOC0, and ADCA will not start the conversion on SOC0 again until SOC2 completes conversion. This will result in asynchronous operation, so it is important to ensure that the triggers do not overflow.

Non-overlapping conversion

If the conversion time does not overlap, it is unnecessary to achieve synchronous operation. For example, if two ADC triggers in the system come from two PWM sources that are always out of phase for 180 degrees, SOC0 can be used for ADCA and ADCC with different trigger sources and different ACQPS values.

27.5.19 Synchronous sampling

Although ADC does not have dual S+H circuits, synchronous sampling can be achieved by setting SOC triggers on two or more ADC in order to use the same trigger source. Based on PWM3 event x3 simultaneous sampling, the ADCINA3, ADCINB5 and ADCINC5 can be sampled. A sampling window of 20 APBCLK cycles is used, but different durations can be used. Upon receiving the PWM3 trigger, the three ADC will immediately start parallel conversion. The result will be stored in the ADCRESULT0 register of each ADC.

Note: At this point, all ADC needs to be in an idle state when receiving the trigger; otherwise, sampling will not occur at exactly the same time.

27.5.20 Internal temperature sensor

The internal temperature sensor can measure the junction temperature of the device, and can be turned on by setting the ENABLE bit of the temperature sensor. The output of the sensor can be sampled together with the ADC through internal connection. To convert the temperature sensor reading to temperature, the temperature sensor reading is passed to the ADC_getTemperatureC () function.

For specific pins of TempSensor, please refer to Analog Pins and Internal Connection of AS Module.

The temperature sensor voltage can be connected to the ADCCIN12 (C12) internal channel; this voltage value can be sampled by configuring the ADCC.

27.5.21 Choose to get the window duration

The input signal of ADC must have sufficient time to charge the sample and

maintain the capacitance. In general, the S+H duration can charge the sampling capacitor to the $\frac{1}{2}$LSB or $\frac{1}{4}$LSB range of the final value, depending on the tolerable settling error. The approximate value of the required settling time can be determined using the RC settlement model.

The S+H time is as follows:

$$t = \tau * k$$

The time constant of model is as follows:

$$\tau = (Rs + Ron) * Ch + Rs * (Cs + Cp)$$

The number of time constants is as follows:

$$k = \frac{2^n}{\text{settling error}} - ln\frac{Cs + Cp}{Ch}$$

Table 106 Parameter Description

Parameter	Description
Rs	Source impedance of ADC drive circuit
Ron	Resistance of ADC sampling switch
Ch	ADC sampling capacitor
Cs	Capacitance of ADC input pin
Cp	Parasitic input capacitance of ADC channel
settling error	Tolerable settling error
n	ADC resolution

27.5.22 Result register mapping

The ADC results and ADC PPB results are repeated for each memory bus controller in the system. The bus controller can directly make read access to the result register without conflicts when multiple bus controllers read ADC result simultaneously.

27.5.23 Design of external reference circuit

Sharing a reference voltage source among all ADC modules can minimize the mismatch of reference voltages between ADC modules. The capacitors between the high/low reference pins should be placed as close to the pins as possible on the PCB, which can help absorb the high-frequency current. A precision operational amplifier with good bandwidth and low output impedance needs to buffer the reference voltage before driving it to the reference pin. The stability of the operational amplifier should be ensured in order to connect the series resistor and capacitor in series. Two reference pins can be shared between an operational amplifier driver. But its performance is slightly lower than that when

each reference pin has a dedicated operational amplifier buffer, and it can still achieve all ADC specifications.

27.6 Register bank address

Table 107 Register Bank Address

Device register	Register bank	Start address	End address
ADCARegs	ADC_REGS	0x5000_1C00	0x5000_1CFF
ADCCRegs	ADC_REGS	0x5000_1D00	0x5000_1FFF
ADCARESULTRegs	ADC_RESULT	0x5000_2100	0x5000_23FF
ADCCRESULTRegs	ADC_RESULT	0x5000_2000	0x5000_20FF

27.7 Register address mapping

Table 108 ADC_REGS Register Address Mapping

Register name	Register description	Offset address	WRPRT
ADCCTL1	Control register 1	0x00	√
ADCCTL2	Control register 2	0x02	√
ADCBURSTCTL	Burst control register	0x04	√
ADCINTFLG	Interrupt flag register	0x06	-
ADCINTFLGCLR	Clear interrupt flag register	0x08	-
ADCINTOVF	Interrupt overflow register	0x0A	-
ADCINTOVFCLR	Clear interrupt overflow register	0x0C	-
ADCINTSEL1N2	Select interrupt 1/2 register	0x0E	√
ADCINTSEL3N4	Select interrupt 3/4 register	0x10	√
ADCSOCPRICTL	SOC priority control register	0x12	√
ADCINTSOCSEL1	Select SOC interrupt trigger register 1	0x14	√
ADCINTSOCSEL2	Select SOC interrupt trigger register 2	0x16	√
ADCSOCFLG1	SOC wait flag register 1	0x18	-
ADCSOCFRC1	SOC force start register 1	0x1A	-
ADCSOCOVF1	SOC overflow flag register 1	0x1C	-
ADCSOCOVFCLR1	Clear SOC overflow register 1	0x1E	-
ADCSOC0CTL	SOC0 control register	0x20	√
ADCSOC1CTL	SOC1 control register	0x24	√

Register name	Register description	Offset address	WRPRT
ADCSOC2CTL	SOC2 control register	0x28	√
ADCSOC3CTL	SOC3 control register	0x2C	√
ADCSOC4CTL	SOC4 control register	0x30	√
ADCSOC5CTL	SOC5 control register	0x34	√
ADCSOC6CTL	SOC6 control register	0x38	√
ADCSOC7CTL	SOC7 control register	0x3C	√
ADCSOC8CTL	SOC8 control register	0x40	√
ADCSOC9CTL	SOC9 control register	0x44	√
ADCSOC10CTL	SOC10 control register	0x48	√
ADCSOC11CTL	SOC11 control register	0x4C	√
ADCSOC12CTL	SOC12 control register	0x50	√
ADCSOC13CTL	SOC13 control register	0x54	√
ADCSOC14CTL	SOC14 control register	0x58	√
ADCSOC15CTL	SOC15 control register	0x5C	√
ADCEVTSTAT	Event status register	0x60	-
ADCEVTCLR	Clear event register	0x64	-
ADCEVTSEL	Enable event register	0x68	√
ADCEVTINTSEL	Enable event interrupt register	0x6C	√
ADCOSDETECT	Configure ADC open circuit and short circuit detection register	0x70	√
ADCCOUNTER	Free running counter register	0x72	-
ADCREV	Version register	0x74	-
ADCOFFTRIM	Offset trim register	0x76	√
ADCPPB1CONFIG	Configuration PPB1 register	0x80	√
ADCPPB1STAMP	PPB1 sampling delay timestamp register	0x82	-
ADCPPB1OFFCAL	PPB1 offset calibration register	0x84	√
ADCPPB1OFFREF	PPB1 offset reference register	0x86	-
ADCPPB1TRIPHI	PPB1 upper limit register	0x88	√
ADCPPB1TRIPLO	PPB1 lower limit register	0x8C	√
ADCPPB2CONFIG	Configuration PPB2 register	0x90	√
ADCPPB2STAMP	PPB2 sampling delay timestamp register	0x92	-
ADCPPB2OFFCAL	PPB2 offset calibration register	0x94	√
ADCPPB2OFFREF	PPB2 offset reference register	0x96	-

Register name	Register description	Offset address	WRPRT
ADCPPB2TRIPHI	PPB2 upper limit register	0x98	√
ADCPPB2TRIPLO	PPB2 lower limit register	0x9C	√
ADCPPB3CONFIG	Configuration PPB3 register	0xA0	√
ADCPPB3STAMP	PPB3 sampling delay timestamp register	0xA2	-
ADCPPB3OFFCAL	PPB3 offset calibration register	0xA4	√
ADCPPB3OFFREF	PPB3 offset reference register	0xA6	-
ADCPPB3TRIPHI	PPB3 upper limit register	0xA8	√
ADCPPB3TRIPLO	PPB3 lower limit register	0xAC	√
ADCPPB4CONFIG	Configuration PPB4 register	0xB0	√
ADCPPB4STAMP	PPB4 sampling delay timestamp register	0xB2	-
ADCPPB4OFFCAL	PPB4 offset calibration register	0xB4	√
ADCPPB4OFFREF	PPB4 offset reference register	0xB6	-
ADCPPB4TRIPHI	PPB4 upper limit register	0xB8	√
ADCPPB4TRIPLO	PPB4 lower limit register	0xBC	√
ADCINTCYCLE	Early interrupt cycle delay register	0xDE	√
ADCINLTRIM2	Linearity trim register 2	0xE4	√
ADCINLTRIM3	Linearity trim register 3	0xE8	√

Table 109 ADC_RESULT Register Address Mapping

Register name	Register description	Offset address	WRPRT
ADCRESULT0	Result register 0	0x00	-
ADCRESULT1	Result register 1	0x02	-
ADCRESULT2	Result register 2	0x04	-
ADCRESULT3	Result register 3	0x06	-
ADCRESULT4	Result register 4	0x08	-
ADCRESULT5	Result register 5	0x0A	-
ADCRESULT6	Result register 6	0x0C	-
ADCRESULT7	Result register 7	0x0E	-
ADCRESULT8	Result register 8	0x10	-
ADCRESULT9	Result register 9	0x12	-
ADCRESULT10	Result register 10	0x14	-
ADCRESULT11	Result register 11	0x16	-
ADCRESULT12	Result register 12	0x18	-

Register name	Register description	Offset address	WRPRT
ADCRESULT13	Result register 13	0x1A	-
ADCRESULT14	Result register 14	0x1C	-
ADCRESULT15	Result register 15	0x1E	-
ADCPPB1RESULT	Post-processing block 1 result register	0x20	-
ADCPPB2RESULT	Post-processing block 2 result register	0x24	-
ADCPPB3RESULT	Post-processing block 3 result register	0x28	-
ADCPPB4RESULT	Post-processing block 4 result register	0x2C	-

27.8 Register functional description

27.8.1 Control register 1 (ADCCTL1)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	Reserved			0h
2	INTPULSEPOS	R/W	ADC Interrupt Pulse Position Configure 0: The interrupt pulse is generated when the ADC starts conversion, and the APBCLK cycle specified by ADCINTCYCLE is added. 1: The interrupt pulse occurs at the end of the conversion, one cycle before the ADC result is stored in its result register.	0h
6:3	Reserved			0h
7	ADCPWDNZ	R/W	ADC Power Down Configure Control the power-on and power-down of all analog circuits within the analog core. 0: Power down 1: Power on	0h
11:8	ADCBSYCHN	R	ADC Busy Channel Flag. Set when the SOC of the ADC is generated. When ADCBSY =0: Save the SOC value of the last conversion. When ADCBSY =1: SOC being processed. 0000: SOC0 being processed or last SOC conversion 0001: SOC1 being processed or last SOC conversion … 1111: SOC15 being processed or last SOC conversion	0h
12	Reserved			0h

Field	Name	R/W	Description	Reset value
13	ADCBSY	R	ADC Busy Flag Determine whether ADC can be used for sampling. Set when the SOC of the ADC is generated, cleared by four ADC clock hardware after the negative edge of the S/H pulse. 0: ADC can be used for the next channel sampling. 1: ADC is busy and unable to sample another channel.	0h
15:14	Reserved			0h

27.8.2 Control register 2 (ADCCTL2)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	PRESCALE	R/W	ADC Clock Prescaler 0000: ADCCLK=Input clock/1.0 0001: Reserved 0010: ADCCLK=Input clock/2.0 0011: Reserved … 1110: ADCCLK=Input clock/8.0 1111: Reserved	0h
15:4	Reserved			0h

27.8.3 Burst control register (ADCBURSTCTL)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
5:0	BURSTTRIGSEL	R/W	SOC Burst Trigger Source Select Configure the trigger that will activate the burst conversion sequence. 000000: BURSTTRIG0 - software only 000001: BURSTTRIG1 - CPU1 timer 0, TINT0 000010: BURSTTRIG2 - CPU1 timer 1, TINT1 000011: BURSTTRIG3 - CPU1 timer 2, TINT2 000100: BURSTTRIG4 - GPIO, Input X-Bar INPUT5 000101: BURSTTRIG5 - ePWM1, ADCSOCA 000110: BURSTTRIG6 - ePWM1, ADCSOCB 000111: BURSTTRIG7 - ePWM2, ADCSOCA 001000: BURSTTRIG8 - ePWM2, ADCSOCB … 010011: BURSTTRIG19 - ePWM8, ADCSOCA 010100: BURSTTRIG20 - ePWM8, ADCSOCB 010101-111111: Reserved	0h
7:6	Reserved			0h

Field	Name	R/W	Description	Reset value
11:8	BURSTSIZE	R/W	SOC Burst Size Select Select how many SOC will be converted when a burst conversion sequence begins. The first SOC to be converted is determined by the polling pointer, and the pointer advances with the conversion of each SOC. 0000: 1 SOC conversion 0001: 2 SOC conversions … 1111: 16 SOC conversions	0h
14:12	Reserved			0h
15	BURSTEN	R/W	SOC Burst Mode Enable 0: Disable 1: Enable	0h

27.8.4 Interrupt flag register (ADCINTFLG)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x-1	ADCINTx	R	ADC Interrupt x Flag (x=1-4) It indicates whether relevant ADCINT pulses are generated since the last clearing. If the ADC interrupt is placed in continuous interrupt mode, further interrupt pulses will be generated when a selected EOC event occurs, whether the flag bit is set or not. If the continuous mode is not enabled, no further interrupt pulses will be generated until this bit is cleared.	0h
15:4	Reserved			0h

27.8.5 Clear interrupt flag register (ADCINTFLGCLR)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	ADCINTx	R-0/W1C	ADC Interrupt x Flag Clear (x=1-4) 0: Not processed 1: Clear the corresponding interrupt flag in the ADCINTFLG register. If the software sets the clear bit and the flag bit that the hardware attempts to set on the same cycle, the hardware will have priority and the overflow bit will not be set. Read returns 0.	0h
15:4	Reserved			0h

27.8.6 Interrupt overflow register (ADCINTOVF)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	ADCINTx	R	ADC Interrupt x Overflow Flag (x=1-4) It indicates whether overflow occurs when the ADCINT pulse is generated. If the corresponding flag bit is set and a selected additional EOC trigger is generated, overflow will occur. 0: ADC interrupt overflow not detected 1: ADC interrupt overflow detected	0h
15:4	Reserved			0h

27.8.7 Clear interrupt overflow register (ADCINTOVFCLR)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	ADCINTx	R-0/W1C	ADC Interrupt x Overflow Flag Clear (x=1-4) 0: Not processed 1: Clear the corresponding overflow flag in the ADCINTOVF register. If the software sets the clear bit and the overflow bit that the hardware attempts to set on the same cycle, the hardware will have priority and the overflow bit will not be set.	0h
15:4	Reserved			0h

27.8.8 Select interrupt 1/2 register (ADCINTSEL1N2)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	INT1SEL	R/W	ADC INT1 EOC Source Select 0000: EOC0 triggers ADCINT1 0001: EOC1 triggers ADCINT1 … 1111: EOC15 triggers ADCINT1	0h
4	Reserved			0h
5	INT1E	R/W	ADC INT1 Enable 0: Disable 1: Interrupt	0h
6	INT1CONT	R/W	ADC INT1 Continue to Interrupt Mode Enable 0: No more ADCINT1 pulse will be generated until the ADCINT1 flag is cleared. 1: Regardless of whether the flag bit is cleared, as long as the EOC pulse is generated, the ADCINT1 pulse will be generated.	0h
7	Reserved			0h

Field	Name	R/W	Description	Reset value
11:8	INT2SEL	R/W	ADC INT2 EOC Source Select 0000: EOC0 triggers ADCINT2 0001: EOC1 triggers ADCINT2 … 1111: EOC15 triggers ADCINT2	0h
12	Reserved			0h
13	INT2E	R/W	ADC INT2 Enable 0: Disable 1: Interrupt	0h
14	INT2CONT	R/W	ADC INT2 Continue to Interrupt Mode Enable 0: No more ADCINT2 pulse will be generated until the ADCINT2 flag is cleared. 1: Regardless of whether the flag bit is cleared, as long as the EOC pulse is generated, the ADCINT2 pulse will be generated.	0h
15	Reserved			0h

27.8.9 Select interrupt 3/4 register (ADCINTSEL3N4)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	INT3SEL	R/W	ADC INT3 EOC Source Select 0000: EOC0 triggers ADCINT3 0001: EOC1 triggers ADCINT3 … 1111: EOC15 triggers ADCINT3	0h
4	Reserved			0h
5	INT3E	R/W	ADC INT3 Enable 0: Disable 1: Interrupt	0h
6	INT3CONT	R/W	ADC INT3 Continue to Interrupt Mode Enable 0: No more ADCINT3 pulse will be generated until the ADCINT3 flag is cleared. 1: Regardless of whether the flag bit is cleared, as long as the EOC pulse is generated, the ADCINT3 pulse will be generated.	0h
7	Reserved			0h
11:8	INT4SEL	R/W	ADC INT4 EOC Source Select 0000: EOC0 triggers ADCINT4 0001: EOC1 triggers ADCINT4 … 1111: EOC15 triggers ADCINT4	0h
12	Reserved			0h

Field	Name	R/W	Description	Reset value
13	INT4E	R/W	ADC INT4 Enable 0: Disable 1: Interrupt	0h
14	INT4CONT	R/W	ADC INT4 Continue to Interrupt Mode Enable 0: No more ADCINT4 pulse will be generated until the ADCINT4 flag is cleared. 1: Regardless of whether the flag bit is cleared, as long as the EOC pulse is generated, the ADCINT4 pulse will be generated.	0h
15	Reserved			0h

27.8.10 SOC priority control register (ADCSOCPRICTL)

Offset address: 0x12

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	SOCPRIORITY	R/W	SOC Priority Configure Determine the priority mode of SOCx and the cutoff point of polling arbitration. 00000: SOC priority is processed in polling mode of all channels. 00001: SOC0 has high priority, and other channels are in polling mode. 00010: SOC0-SOC1 have high priority, and SOC2-SOC15 are in polling mode. 00011: SOC0-SOC2 have high priority, and SOC3-SOC15 are in polling mode. … 01111: SOC0-SOC14 have high priority, and SOC15 are in polling mode. 10000: All SOC are in high priority mode and arbitrate based on the SOC number. 10001-11111: Reserved	0h
9:5	RRPOINTER	R	Round Robin Pointer Save the value of the polling SOCx of the last conversion, and this value will be used by the polling mode to determine the order of conversion. 00000: SOC0 is the last polling SOC conversion, and SOC1 has the highest polling priority. 00001: SOC1 is the last polling SOC conversion, and SOC2 has the highest polling priority. … 01111: SOC15 is the last polling SOC conversion, and SOC0 has the highest polling priority. 10000: Reset value, indicating that SOC is not converted. SOC0 has the highest polling priority. When the device is reset, this value will be set. If the conversion is ongoing when writing to the ADCSOCPRICTL register, the conversion is completed and the new priority takes effect.	10h

Field	Name	R/W	Description	Reset value
			10001-11111: Reserved	
15:10	Reserved			0h

27.8.11 Select SOC interrupt trigger register 1 (ADCINTSOCSEL1)

This register selects the ADCINT to trigger SOCx, which, together with ADCSOCxCTL [TRIGSEL], determines the trigger for SOC.

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2x+1:2x	SOCx	R/W	SOCx ADC Interrupt Trigger Select (x=0-7) 00: No ADCINT triggers SOCx 01: ADCINT1 triggers SOCx 10: ADCINT2 triggers SOCx 11: Reserved	0h

27.8.12 Select SOC interrupt trigger register 2 (ADCINTSOCSEL2)

This register selects the ADCINT to trigger SOCx, which, together with ADCSOCxCTL [TRIGSEL], determines the trigger for SOC.

Offset address: 0x16

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	SOC8	R/W	SOC8 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC8 01: ADCINT1 triggers SOC8 10: ADCINT2 triggers SOC8 11: Reserved	0h
3:2	SOC9	R/W	SOC9 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC9 01: ADCINT1 triggers SOC9 10: ADCINT2 triggers SOC9 11: Reserved	0h
5:4	SOC10	R/W	SOC10 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC10 01: ADCINT1 triggers SOC10 10: ADCINT2 triggers SOC10 11: Reserved	0h
7:6	SOC11	R/W	SOC11 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC11 01: ADCINT1 triggers SOC11 10: ADCINT2 triggers SOC11 11: Reserved	0h
9:8	SOC12	R/W	SOC12 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC12 01: ADCINT1 triggers SOC12	0h

Field	Name	R/W	Description	Reset value
			10: ADCINT2 triggers SOC12 11: Reserved	
11:10	SOC13	R/W	SOC13 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC13 01: ADCINT1 triggers SOC13 10: ADCINT2 triggers SOC13 11: Reserved	0h
13:12	SOC14	R/W	SOC14 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC14 01: ADCINT1 triggers SOC14 10: ADCINT2 triggers SOC14 11: Reserved	0h
15:14	SOC15	R/W	SOC15 ADC Interrupt Trigger Select 00: No ADCINT triggers SOC15 01: ADCINT1 triggers SOC15 10: ADCINT2 triggers SOC15 11: Reserved	0h

27.8.13 SOC wait flag register 1 (ADCSOCFLG1)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x	SOCx	R	OCx Wait Flag (x=0-15) 0: No sampling to be processed 1: Sampling is waiting When SOCx conversion starts, this bit will be automatically cleared. If there is contention for this bit and both a set request and a clear request are received within the same cycle, the set request will have a higher priority and the clear request will be ignored, whether the bit of the ADCSOCOVF1 register has been set or not.	0h

27.8.14 SOC force start register 1 (ADCSOCFRC1)

Offset address: 0x1A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x	SOCx	R-0/W1S	SOCx Force Start Conversion（x=0-15） 0: No action. 1: Force the SOCx bit of ADCSOCFLG1 to be 1. Cause the conversion to start after priority is given to SOCx. If the software sets this bit and the SOCx bit of ADCSOCFLG1 that the hardware attempts to clear on the same cycle, the software will have the priority and the SOCx bit of ADCSOCFLG1 will be set. At this point, whether the SOCx bit of ADCSOCFLG1 was set or not before, the corresponding overflow bit will not	0h

Field	Name	R/W	Description	Reset value
			be affected. It can be used to start software start conversion. This bit always reads as 0.	

27.8.15 SOC overflow flag register 1 (ADCSOCOVF1)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x	SOCx	R	SOCx Event Overflow Flag It indicates that a SOCx event has been generated in the hardware when a SOCx event has already been suspended. The overflow conditions will not prevent the processing of SOCx events. Writing to ADCSOCFRC1 does not affect this bit. 0: No overflow 1: Overflow	0h

27.8.16 Clear SOC overflow register 1 (ADCSOCOVFCLR1)

Offset address: 0x1E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
x	SOCx	R-0/W1S	SOCx Overflow Clear 0: No action 1: Clear SOCx overflow If the software sets this bit and the SOCx bit of ADCSOCOVF1 that the hardware attempts to set on the same cycle, the hardware will have the priority and the SOCx bit of ADCSOCOVF1 will be set. This bit always reads as 0.	0h

27.8.17 SOCx control register (ADCSOCxCTL) (x=0-15)

Offset address: 0x20+a*0x04 (a=0-15)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
8:0	ACQPS	R/W	SOCx Sample Window Duration Configure The configured sampling time must be at least one ADCCLK cycle so that the ADC can function properly. 000000000: The sampling window is 1 APBCLK cycle wide 000000001: The sampling window is 2 APBCLK cycles wide 000000010: The sampling window is 3 APBCLK cycles wide … 111111111: The sampling window is 512 APBCLK cycles wide	0h
14:9	Reserved			0h

Field	Name	R/W	Description	Reset value
18:15	CHSEL	R/W	SOCx Convert Channel Select 0000: ADCIN0 0001：ADCIN1 … 1111: ADCIN15	0h
19	Reserved			0h
24:20	TRIGSEL	R/W	SOCx Trigger Source Select This field and ADCINTSOCSELx jointly determine the trigger of SOC. 00000: ADCTRIG0 - software only 00001: ADCTRIG1 -TMR0，TINT0 00010: ADCTRIG2 - TMR1, TINT1 00011: ADCTRIG3 - TMR2, TINT2 00100: ADCTRIG4 - GPIO Input X-Bar INPUT5 00101: ADCTRIG5 - PWM1 ADCSOCA 00110: ADCTRIG6 - PWM1 ADCSOCB 00111: ADCTRIG7 - PWM2 ADCSOCA 01000: ADCTRIG8 - PWM2 ADCSOCB … 10001: ADCTRIG17 – PWM7, ADCSOCA 10010: ADCTRIG18 – PWM7, ADCSOCB 10011-11111: Reserved	0h
31:25	Reserved			0h

27.8.18 Event status register (ADCEVTSTAT)

Offset address: 0x60

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PPB1TRIPHI	R/W	Post Processing Block 1 Trip High Flag 0: Not occur 1: A digital compare trip high event occurred	0h
1	PPB1TRIPLO	R/W	Post Processing Block 1 Trip Low Flag 0: Not occur 1: A digital compare trip low event occurred	0h
2	PPB1ZERO	R/W	Post Processing Block 1 Zero Crossing Flag This bit is gated by the EOC signal. 0: No change 1: The ADCPPB1RESULT register has changed the sign	0h
3	Reserved			0h
4	PPB2TRIPHI	R/W	Post Processing Block 2 Trip High Flag 0: Not occur 1: A digital compare trip high event occurred	0h

Field	Name	R/W	Description	Reset value
5	PPB2TRIPLO	R/W	Post Processing Block 2 Trip Low Flag 0: Not occur 1: A digital compare trip low event occurred	0h
6	PPB2ZERO	R/W	Post Processing Block 1 Zero Crossing Flag This bit is gated by the EOC signal. 0: No change 1: The ADCPPB2RESULT register has changed the sign	0h
7	Reserved			0h
8	PPB3TRIPHI	R/W	Post Processing Block 3 Trip High Flag 0: Not occur 1: A digital compare trip high event occurred	0h
9	PPB3TRIPLO	R/W	Post Processing Block 3 Trip Low Flag 0: Not occur 1: A digital compare trip low event occurred	0h
10	PPB3ZERO	R/W	Post Processing Block 3 Zero Crossing Flag This bit is gated by the EOC signal. 0: No change 1: The ADCPPB3RESULT register has changed the sign	0h
11	Reserved			0h
12	PPB4TRIPHI	R	Post Processing Block 4 Trip High Flag 0: Not occur 1: A digital compare trip high event occurred	0h
13	PPB4TRIPLO	R	Post Processing Block 4 Trip Low Flag 0: Not occur 1: A digital compare trip low event occurred	0h
14	PPB4ZERO	R	Post Processing Block 4 Zero Crossing Flag This bit is gated by the EOC signal. 0: No change 1: The ADCPPB4RESULT register has changed the sign	0h
15	Reserved			0h

27.8.19 Clear event register (ADCEVTCLR)

Offset address: 0x64

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PPB1TRIPHI	R-0/W1S	Post Processing Block 1 Trip High Clear 0: Not clear 1: Clear	0h
1	PPB1TRIPLO	R-0/W1S	Post Processing Block 1 Trip Low Clear 0: Not clear 1: Clear	0h

Field	Name	R/W	Description	Reset value
2	PPB1ZERO	R-0/W1S	Post Processing Block 1 Zero Crossing Clear 0: Not clear 1: Clear	0h
3	Reserved			0h
4	PPB2TRIPHI	R-0/W1S	Post Processing Block 2 Trip High Clear 0: Not clear 1: Clear	0h
5	PPB2TRIPLO	R-0/W1S	Post Processing Block 2 Trip Low Clear 0: Not clear 1: Clear	0h
6	PPB2ZERO	R-0/W1S	Post Processing Block 2 Zero Crossing Clear 0: Not clear 1: Clear	0h
7	Reserved			0h
8	PPB3TRIPHI	R-0/W1S	Post Processing Block 3 Trip High Clear 0: Not clear 1: Clear	0h
9	PPB3TRIPLO	R-0/W1S	Post Processing Block 3 Trip Low Clear 0: Not clear 1: Clear	0h
10	PPB3ZERO	R-0/W1S	Post Processing Block 3 Zero Crossing Clear 0: Not clear 1: Clear	0h
11	Reserved			0h
12	PPB4TRIPHI	R-0/W1S	Post Processing Block 4 Trip High Clear 0: Not clear 1: Clear	0h
13	PPB4TRIPLO	R-0/W1S	Post Processing Block 4 Trip Low Clear 0: Not clear 1: Clear	0h
14	PPB4ZERO	R-0/W1S	Post Processing Block 4 Zero Crossing Clear 0: Not clear 1: Clear	0h
15	Reserved			0h

27.8.20 Enable event register (ADCEVTSEL)

Offset address: 0x68

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PPB1TRIPHI	R/W	Post Processing Block 1 Trip High Event Enable Allow the corresponding rising trip high flag to activate the event signal to the PWM block It must be cleared before an additional event is generated to the PWM block.	0h

Field	Name	R/W	Description	Reset value
			0: Disable 1: Enable	
1	PPB1TRIPLO	R/W	Post Processing Block 1 Trip Low Event Enable Allow the corresponding rising trip low flag to activate the event signal to the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
2	PPB1ZERO	R/W	Post Processing Block 1 Zero Crossing Event Enable Allow the corresponding rising zero crossing flag to activate the event signal of the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
3	Reserved			0h
4	PPB2TRIPHI	R/W	Post Processing Block 2 Trip High Event Enable Allow the corresponding rising trip high flag to activate the event signal to the PWM block It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
5	PPB2TRIPLO	R/W	Post Processing Block 2 Trip Low Event Enable Allow the corresponding rising trip low flag to activate the event signal to the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
6	PPB2ZERO	R/W	Post Processing Block 2 Zero Crossing Event Enable Allow the corresponding rising zero crossing flag to activate the event signal of the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
7	Reserved			0h
8	PPB3TRIPHI	R/W	Post Processing Block 3 Trip High Event Enable Allow the corresponding rising trip high flag to activate the event signal to the PWM block It	0h

Field	Name	R/W	Description	Reset value
			must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	
9	PPB3TRIPLO	R/W	Post Processing Block 3 Trip Low Event Enable Allow the corresponding rising trip low flag to activate the event signal to the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
10	PPB3ZERO	R/W	Post Processing Block 3 Zero Crossing Event Enable Allow the corresponding rising zero crossing flag to activate the event signal of the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
11	Reserved			0h
12	PPB4TRIPHI	R/W	Post Processing Block 4 Trip High Event Enable Allow the corresponding rising trip high flag to activate the event signal to the PWM block It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
13	PPB4TRIPLO	R/W	Post Processing Block 4 Trip Low Event Enable Allow the corresponding rising trip low flag to activate the event signal to the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
14	PPB4ZERO	R/W	Post Processing Block 4 Zero Crossing Event Enable Allow the corresponding rising zero crossing flag to activate the event signal of the PWM block. It must be cleared before an additional event is generated to the PWM block. 0: Disable 1: Enable	0h
15	Reserved			0h

27.8.21 Enable event interrupt register (ADCEVTINTSEL)

Offset address: 0x6C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PPB1TRIPHI	R/W	Post Processing Block 1 Trip High Interrupt Enable Allow the corresponding rising trip high flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
1	PPB1TRIPLO	R/W	Post Processing Block 1 Trip Low Interrupt Enable Allow the corresponding rising trip low flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
2	PPB1ZERO	R/W	Post Processing Block 1 Zero Crossing Interrupt Enable Allow the corresponding rising zero crossing flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
3	Reserved			0h
4	PPB2TRIPHI	R/W	Post Processing Block 2 Trip High Interrupt Enable Allow the corresponding rising trip high flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
5	PPB2TRIPLO	R/W	Post Processing Block 2 Trip Low Interrupt Enable Allow the corresponding rising trip low flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
6	PPB2ZERO	R/W	Post Processing Block 2 Zero Crossing Interrupt Enable Allow the corresponding rising zero crossing flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
7	Reserved			0h
8	PPB3TRIPHI	R/W	Post Processing Block 3 Trip High Interrupt Enable Allow the corresponding rising trip high flag to activate the event interrupt signal to the NVIC. It must	0h

Field	Name	R/W	Description	Reset value
			be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	
9	PPB3TRIPLO	R/W	Post Processing Block 3 Trip Low Interrupt Enable Allow the corresponding rising trip low flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
10	PPB3ZERO	R/W	Post Processing Block 3 Zero Crossing Interrupt Enable Allow the corresponding rising zero crossing flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
11	Reserved			0h
12	PPB4TRIPHI	R/W	Post Processing Block 4 Trip High Interrupt Enable Allow the corresponding rising trip high flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
13	PPB4TRIPLO	R/W	Post Processing Block 4 Trip Low Interrupt Enable Allow the corresponding rising trip low flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
14	PPB4ZERO	R/W	Post Processing Block 4 Zero Crossing Interrupt Enable Allow the corresponding rising zero crossing flag to activate the event interrupt signal to the NVIC. It must be cleared before an additional interrupt is generated to NVIC. 0: Disable 1: Enable	0h
15	Reserved			0h

27.8.22 Configure ADC open and short circuit detection register (ADCOSDETECT)

Offset address: 0x70

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	DETECTCFG	R/W	ADC Opens and Shorts Detect Configure 000: Turn off the open/short circuit detection circuit 001: Enable open/short circuit detection circuit at zero range 010: Enable open/short circuit detection circuit at full range 0101: Enable open/short circuit detection circuit at (nominal) 9.07/16.84 scale 100: Enable open/short circuit detection circuit at (nominal) 7.77/16.84 scale 101: Enable open/short circuit detection circuit, with (nominal) 7.77K pulled down to VSSA 110: Enable open/short circuit detection circuit, with (nominal) 7.77K pulled up to VDDA 111: Enable open/short circuit detection circuit, with (nominal) 9.07K pulled down to VSSA	0h
15:3	Reserved			0h

27.8.23 Free running counter register (ADCCOUNTER)

Offset address: 0x72

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	FREECOUNT	R	ADC Free Running Counter Value It indicates the counter value with the ADC running freely.	0h
15:12	Reserved			0h

27.8.24 Version register (ADCREV)

Offset address: 0x74

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TYPE	R	ADC Type The clock is set to 5.	5h
15:8	REV	R	ADC Revision It records the differences between different versions, with the first version marked as 00.	0h

27.8.25 Offset trim register (ADCOFFTRIM)

Offset address: 0x76

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	OFFTRIM	R/W	ADC Offset Trim Adjust the conversion result of the converter upward or downward to consider the offset error in the ADC. Load the factory configuration during device startup period. The offset can be corrected within the range of +7 to -8 LSB;	0h

Field	Name	R/W	Description	Reset value
			for the offset with a value of 16*8-bit complement, the details are as follows: 7 LSB (16* 7) = 112 6 LSB (16* 6) = 96 5 LSB (16* 5) = 80 4 LSB (16* 4) = 64 3 LSB (16* 3) = 48 2 LSB (16* 2) = 32 1 LSB (16* 1) = 16 0 LSB (16* 0) = 0 -1 LSB (16*(-1)) = 240 … -7LSB(16*(-7)) = 144	
15:8	Reserved			0h

27.8.26 Configure PPBx register (ADCPPBxCONFIG) (x=1-4)

Offset address: 0x80+(a-1)*0x10 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	CONFIG	R/W	ADC Post Processing Block x Configure Configure the SOC/EOC/RESULT associated with this post-processing block. 0000: SOC0 / EOC0 / RESULT0 0001: SOC1 / EOC1 / RESULT1 … 1111: SOC15/EOC15/RESULT15	0h
4	TWOSCOMPEN	R/W	ADC Post Processing Block x Two's Complement Enable Before the processing block stores the results in the ADCPPBxRESULT register, perform a complement of 2 on the output of the offset/reference subtraction unit. 0: ADCPPBxRESULT = ADCRESULTx – ADC_OFFREF 1: ADCPPBxRESULT = ADC_OFFREF – ADCRESULTx	0h
5	CBCEN	R/W	ADC Post Processing Block Cycle By Cycle Enable If the event condition does not exist, the hardware processing circuit after conversion can automatically clear the ADCEVTSTAT at the time of conversion.	0h
15:6	Reserved			0h

27.8.27 PPBx sampling delay timestamp register (ADCPPBxSTAMP) (x=1-4)

Offset address: 0x82+(a-1)*0x10 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DLYSTAMP	R	ADC Post Processing Block x Sample Delay Timestamp When SOC starts sampling, the values contained in REQSTAMP are subtracted from the value of FREECOUNT and loaded into this field, so as to provide the number of clock cycle delays between SOC trigger and actual start of sampling.	0h
15:12	Reserved			0h

27.8.28 PPBx offset calibration register (ADCPPBxOFFCAL) (x=1-4)

Offset address: 0x84+(a-1)*0x10(a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	OFFCAL	R/W	ADC Post Processing Block x Offset Calibrate It can be used to digitally eliminate the inherent system level offset in ADCIN circuit. It needs to be subtracted from the ADC output before this value is stored in the ADCRESULT register. 0000000000: No change. The ADC output is directly stored in ADCRESULT. 0000000001: ADC output -1 is stored in ADCRESULT. 0000000010: ADC output -2 is stored in ADCRESULT. ... 1000000000: ADC output +512 is stored in ADCRESULT. ... 1111111111: ADC output+ 1 is stored in ADCRESULT. In 12-bit mode, subtraction will be saturated at 000000000000 and 111111111111 and stored in ADCRESULT register. In 16-bit mode, subtraction will be saturated at 0000000000000000 and 1111111111111111 and stored in ADCRESULT register.	0h
15:10	Reserved			0h

27.8.29 PPBx offset reference register (ADCPPBxOFFREF) (x=1-4)

Offset address: 0x86+(a-1)*0x10 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	OFFREF	R/W	ADC Post Processing Block x Offset Reference It can be used to calculate feedback error or convert unipolar signals into bipolar signals by subtracting the reference value. This value is subtracted from the ADCRESULT register, then passed through an optional binary complement function and stored in the ADCPPBxRESULT register. This subtraction is unsaturated. In 12-bit mode, the size of this register will not change from 16 bits. It is necessary to ensure that only 12-bit values are written into the register in 12-bit mode.	0h

Field	Name	R/W	Description	Reset value
			0000000000000000: No change. Pass the ADCRESULT value 0000000000000001: Pass ADCRESULT - 1 0000000000000010: Pass ADCRESULT – 2 ... 1111111111111111: Pass ADCRESULT - 65,535	

27.8.30 PPBx upper limit register (ADCPPBxTRIPHI) (x=1-4)

Offset address: 0x88+(a-1)*0x10 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	LIMITHI	R/W	ADC Post Processing Block x Trip upper Threshold Upper limit for the digital comparator trip. In 16-bit mode, all 17 bits will be compared with the 17 bits of the PPBRESULTx field. In 12-bit mode, the [12:0] of this bit will be compared with PPBRESULTx[12:0].	0h
16	HSIGN	R/W	Upper Threshold Sign The 17th bit (sign bit) of LIMITHI in 16-bit ADC mode.	0h
31:17	Reserved			0h

27.8.31 PPBx lower limit register (ADCPPBxTRIPLO) (x=1-4)

Offset address: 0x8C+(a-1)*0x10 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	LIMITLO	R/W	ADC Post Processing Block x Trip Lower Threshold Lower limit for the digital comparator trip. In 16-bit mode, all 17 bits will be compared with the 17 bits of the PPBRESULTx field. In 12-bit mode, the [12:0] of this bit will be compared with PBRESULTx [12:0].	0h
16	LSIGN	R/W	Lower Threshold Sign The 17th bit (sign bit) of LIMITLO in 16-bit ADC mode.	0h
19:17	Reserved			0h
31:20	REQSTAMP	R/W	ADC Post Processing Block x Request Timestamp When the trigger sets the relevant SOC flag in ADCSOCFLGx, FREECOUNT is loaded into this field.	0h

27.8.32 Early interrupt cycle delay register (ADCINTCYCLE)

Offset address: 0xDE

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DELAY	R/W	ADC Early Interrupt Cycle Delay Define the delay from the falling edge of ADCSOC based on the clock cycle in order to generate interrupts.	0h

27.8.33 Linearity trim register 1 (ADCINLTRIM1)

Offset address: 0xE0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	INLTRIM31TO0	R/W	ADC Linearity Trim 31-0 Bits Modifying the contents of this register may cause the ADC to run outside of specifications, so this register should not be modified except under special circumstances.	0h

27.8.34 Linearity trim register 2 (ADCINLTRIM2)

Offset address: 0xE4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	INLTRIM63TO32	R/W	ADC Linearity Trim 63-32 Bits Modifying the contents of this register may cause the ADC to run outside of specifications, so this register should not be modified except under special circumstances.	0h

27.8.35 Linearity trim register 3 (ADCINLTRIM3)

Offset address: 0xE8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	INLTRIM95TO64	R/W	ADC Linearity Trim 95_64 Bits Modifying the contents of this register may cause the ADC to run outside of specifications, so this register should not be modified except under special circumstances.	0h

27.8.36 Result register x (ADCRESULTx) (x=0-15)

Offset address: 0x00 + a*0x02 (a=0-15)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RESULT	R	ADC Conversion Result x (x=0-15) 16-bit ADC result. After the ADC completes the SOCx conversion, the digital result is stored in this field.	0h

27.8.37 Post processing block x result register (ADCPPBxRESULT) (x=1-4)

Offset address: 0x1C+a*0x04 (a=1-4)

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	PPBRESULT	R	ADC Post Processing Block Conversion Result x (x=1-4) The conversion result after the offset/reference subtraction is stored in this field. If ADCINTFLG is polled when reading	0h

Field	Name	R/W	Description	Reset value
			this field, a NOP instruction needs to be added to ensure that the conversion post-processing is filled in this register. Note: If the conversion related to this post-processing block is a 12-bit conversion, this field is [12:0].	
31:16	SIGN	R	Sign Extended These bits reflect the same values as the 16 bits. Note: If the conversion related to this post-processing block is a 12-bit conversion, this field extends down to 12 bits and reflects the same value as the 12 bits.	0h

28 Buffer digital-to-analog converter (DAC)

28.1 Introduction

The buffer DAC is a universal DAC, and it consists of an internal 12-bit DAC and an analog output buffer that can drive external loads. It can generate AC waveforms such as sine waves, square waves, and triangular waves, as well as DC voltages. To drive a load higher than the typical one, the load size can be balanced against the output voltage swing. The software's writing to DACVALS can take effect immediately or be synchronized with the PWMSYNCPER event. The reference voltage can be selected as either VDAC or VREFHI, with the VREFHI input supporting both external and internal input modes.

28.2 Main characteristics

(1) 12-bit programmable internal DAC

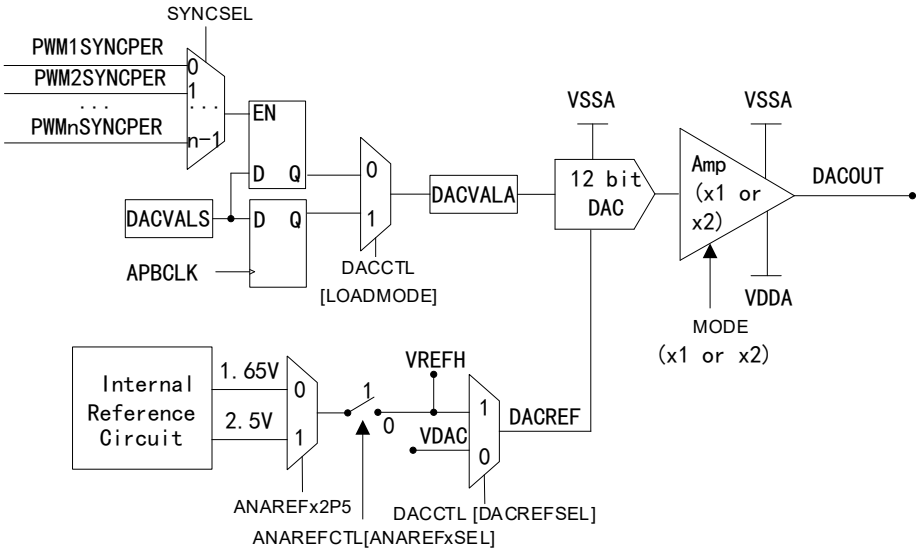
(2) Optional reference voltage source

(3) Can synchronize with PWMSYNCPER

(4) In x1 and x2 mode, use the internal VREFHI

28.3 Structure block diagram

Figure 46 Structure Block Diagram



28.4 Functional description

he DAC’s reference voltage can be set to either VDAC or VREFHI, where VDAC is an external reference voltage and VREFHI can be either an internal or

external reference voltage. When using an external reference, only the X1 function is available.When using an internal reference, the X2 function is available when the reference voltage is 1.65 V; the X1 function is available when the reference voltage is 2.5 V. Even when the buffered DAC is in X2 gain mode, the maximum output voltage is less than VDDA.

To ensure the chip operates correctly, the VDAC pin must be kept below VDDA + 0.3 V. If the VDAC pin exceeds this level, the blocking circuit may be activated and the internal value of VDAC may float to 0 V, resulting in an incorrect DAC output.The VREFHI pin must be kept below VDDA + 0.3V to ensure correct operation. If the VREFHI pin exceeds this level, the blocking circuit may be activated and the internal value of VREFHI may float to 0V, resulting in incorrect ADC conversions or DAC outputs.

The buffered DAC supports the following combinations of gain modes:

Table 110 Gain Mode Combinations Supported by DAC

DACREFSEL	ANAREFxSEL (A/B)	ANAREFx2P5 (A/B)	Reference source	Voltage (V)	Mode	Maximum DAC output (V)	Support status
0	0/1	0/1	External	VDAC	0	2.5	Supported
0	0/1	0/1	External	VDAC	1	2.5	Not supported
1	1	0/1	External	VREFHI	0	2.5	Supported
1	1	0/1	External	VREFHI	1	2.5	Not supported
1	0	0	Internal	1.65	0	1.65	Not supported
1	0	0	Internal	1.65	1	3.3	Supported
1	0	1	Internal	2.5	0	2.5	Supported
1	0	1	Internal	2.5	1	2.5	Not supported

Note: VDAC/VREFHI = 2.5V, VDDA = 3.3V, VAL = 4095.

The buffer DAC contains DACVALA and DACVALA registers. DACVALS is a writable shadow register, which can be immediately loaded into DACVALA or be synchronized with the next PWMSYNCPER event. DACVALA is a read-only register, which actively controls the DAC value of buffer. If the clock of the buffer DAC is disabled when its voltage is outputted, the output voltage is not affected, but DACVALA and DACVALS are no longer updated through the register writing. Re-enabling the clock of buffer DAC can restore it to its state before the clock is disabled. The output buffer of the buffer DAC may exhibit nonlinear behavior near VDDA/VSSA.

The ideal output of the internal DAC is as follows:

$$\text{DACOUT} = \frac{DACVALA \times DACREF}{4096}$$

28.4.1 Initialization

The initialization steps are as follows:

(1) Enable the DAC buffer clock

(2) Select the DACREF

(3) Enable the buffer DAC

(4) The voltage is outputted after the power-on time, and the power-on time is confirmed according to the DAC Electrical Characteristics Table

(5) Continuous writing to DACVALS based on the stable time interval of the buffer DAC can support the predictable behavior of the buffer DAC. Please refer to DAC Electrical Characteristics Table for specific parameters.

28.4.2 DAC offset adjustment

The voltage difference between the middle code (2048) and 1.25V (2.5V reference voltage) is called zero offset error. The DAC offset error is calibrated at 2.5V reference voltage, included in the Device_cal() function, and loaded into DACTRIM. If the DAC is not used at 2.5V reference voltage, the offset must be adjusted to ensure that the offset error performance remains within the limit range. DACTRIM [OFFSET_TRIM] is a signed offset adjustment. The DAC_tuneOffsetTrim () function can be used to adjust the offset.

28.4.3 PWMSYNCPER signal

PWMSYNCPER comes from the time base submodule of PWM. When setting the LOADMODE bit, the PWMSYNCPER signal of DACVALA is triggered to be loaded by level. If both TBCNT and TBPRD of PWM are 0, PWMSYNCPER will remain at a high level and be immediately loaded from DACVALS to DACVALA. So you need to configure the PWM first and then set the LOADMODE.

Note: The name of synchronization signal received by GPDAC from PWM has been updated from PWMSYNC to PWMSYNCPER to avoid confusion with other PWM synchronization signals.

28.5 Lock register

A DACLOCK register is provided to prevent the false write operations from modifying the DACCTL, DACVALS and DACOUTEN registers. If the registers are protected by DACLOCK, the write access will be locked until the device is reset.

28.6 Register bank address

Table 111 Register Bank Address

Device register	Register bank	Start address	End address
DACARegs	DAC_REGS	0x4002_4000	0x4002_43FF
DACBRegs	DAC_REGS	0x4002_4400	0x4002_47FF

28.7 Register address mapping

Table 112 Register Address Mapping

Register name	Register description	Offset address	WRPRT
DACREV	Revision register	0x00	-
DACCTL	Control register	0x02	√
DACVALA	DAC value active register	0x04	-
DACVALS	DAC value shadow register	0x06	-
DACOUTEN	Enable output register	0x08	√
DACLOCK	Lock register	0x0A	√
DACTRIM	Offset trim register	0x0C	√

28.8 Register functional description

28.8.1 Revision register (DACREV)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	REV	R/W	DAC Revision	0h
15:8	Reserved			0h

28.8.2 Control register (DACCTL)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DACREFSEL	R/W	DAC reference voltage select 0: VDAC/VSSA is reference voltage 1: ADC VREFHI/VSSA is reference voltage	0h

Field	Name	R/W	Description	Reset value
1	MODE	R/W	DAC gain mode select Select gain mode for buffer output Only used when setting the DACREFSEL bit and selecting the ADC internal reference mode. 0: The gain is 1 1: The gain is 2	0h
2	LOADMODE	R/W	DACVALA load mode select Select the time to update the DACVALA register with the value in DACVALS. 0: Load the next APBCLK 1: Load the next PWMSYNCPER specified by SYNCSEL	0h
3	Reserved			0h
7:4	SYNCSEL	R/W	Update DACVALA signal Select n is the maximum number of PWMSYNCPER signals that the device can obtain: 0: PWM1SYNCPER 1: PWM2SYNCPER 2: PWM3SYNCPER … n-1: PWMnSYNCPER	0h
15:8	Reserved			0h

28.8.3 DAC value active register (DACVALA)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVALA	R	Active output code currently driven by the DAC	0h
15:12	Reserved			0h

28.8.4 DAC value shadow register (DACVALS)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVALS	R/W	Shadow output code to be loaded into DACVALA	0h
15:12	Reserved			0h

28.8.5 Enable output register (DACOUTEN)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DACOUTEN	R/W	DAC output enable 0: Disable 1: Enable	0h

Field	Name	R/W	Description	Reset value
15:1	Reserved			0h

28.8.6 Lock register (DACLOCK)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DACCTL	R/WSonce	DACCTL register Lock write-access 1: Unlocked. Write "0" and this bit will be invalid. 1: Locked. Only system reset can clear this bit	0h
1	DACVAL	R/WSonce	DACVALS register Lock write-access 1: Unlocked. Write "0" and this bit will be invalid. 1: Locked. Only system reset can clear this bit	0h
2	DACOUTEN	R/WSonce	DACOUTEN register Lock write-access 1: Unlocked. Write "0" and this bit will be invalid. 1: Locked. Only system reset can clear this bit	0h
11:3	Reserved			0h
15:12	KEY	R-0/W	Key This register can only be written when this field is 0xA. Only 16-bit write will succeed after KEY matching. read-modify-writes to a single bit in this register are ignored.	0h

28.8.7 Offset trim register (DACTRIM)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	OFFSET_TRIM	R/W	DAC Offset Trim This register is not modified except under special circumstances. Otherwise, the module may run outside the specifications.	0h
15:8	Reserved			0h

29 Comparator (COMP)

29.1 Full Name and Abbreviation Description of Terms

Table 113 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Comparator DAC	Compdac

29.2 Introduction

The comparator (COMP) module integrates the analog comparator and related circuits, and is suitable for such applications as voltage jump monitoring, switch mode power supply, peak current mode control, and power factor correction. Each subsystem of the COMP module consists of 2 analog comparators, 1 decreasing ramp generator, 2 digital filters, and 2 12-bit DAC.

Note: In this chapter, COMP represents the comparator module, while comp represents the analog comparator in the comparator module subsystem.

29.3 Main characteristics

(1) Each subsystem of COMP consists of:
- 2 analog comparators
- 1 decreasing ramp generator
- 2 digital filters
- 2 12-bit DAC

(2) DAC reference voltage is programmable, and can be configured as V_{DDA} or V_{DAC}

(3) The maximum clock prescale value of the digital filter is 2^{10}

(4) Synchronous output with SYSCLK signal

(5) Input/output function
- (Optional) Hysteresis input
- Inverted output
- Latched output

(6) Positive/negative input of comparator
- Positive input end: Driven by external signals
- Negative input end: Driven by external signals or reference DAC

(7) PWMBLANK signal, used to extend the clear signal

(8) PWMSYNC signal, used for synchronizing submodules

29.4 Structure block diagram

Figure 47 COMP Structure Block Diagram

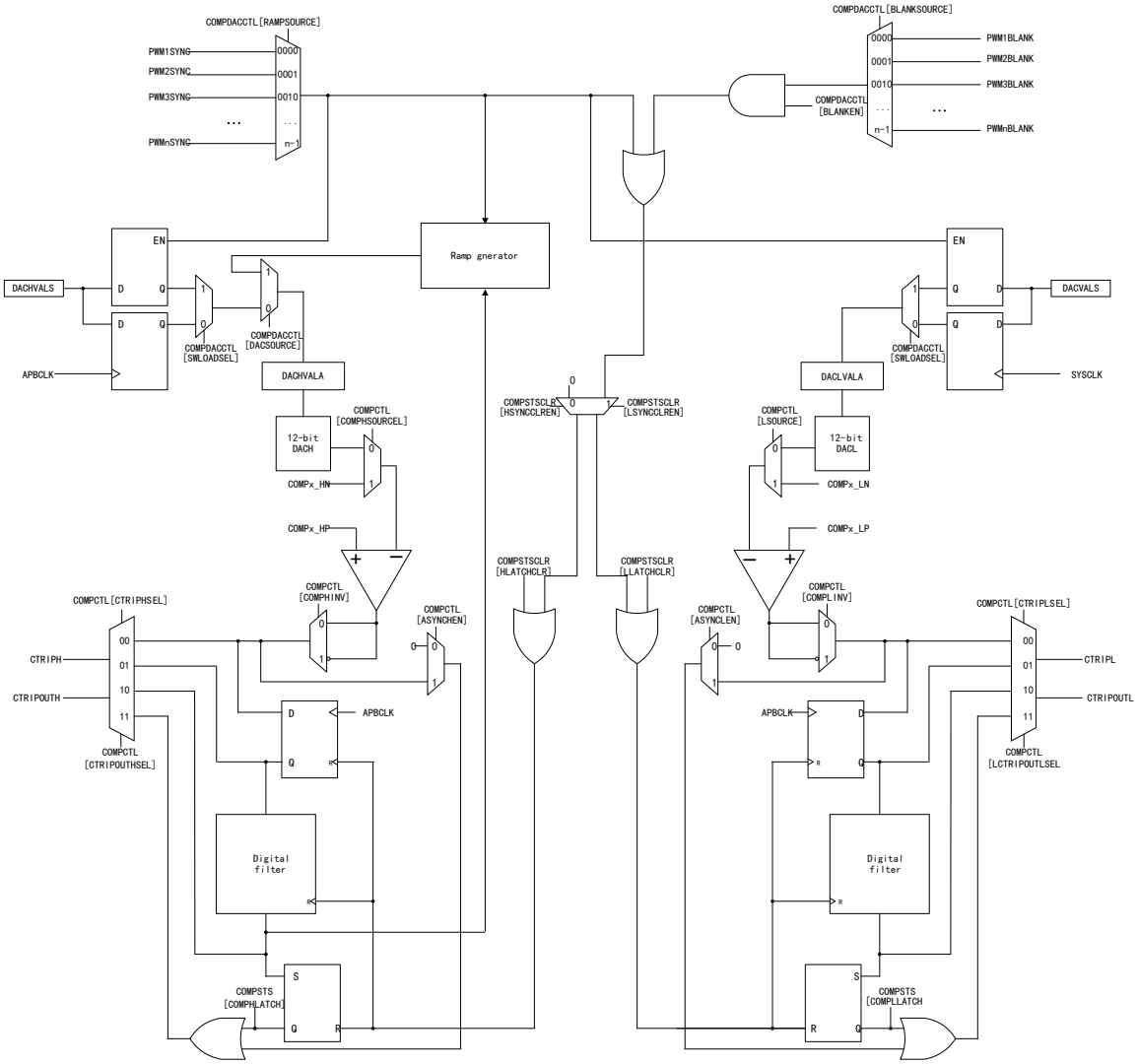
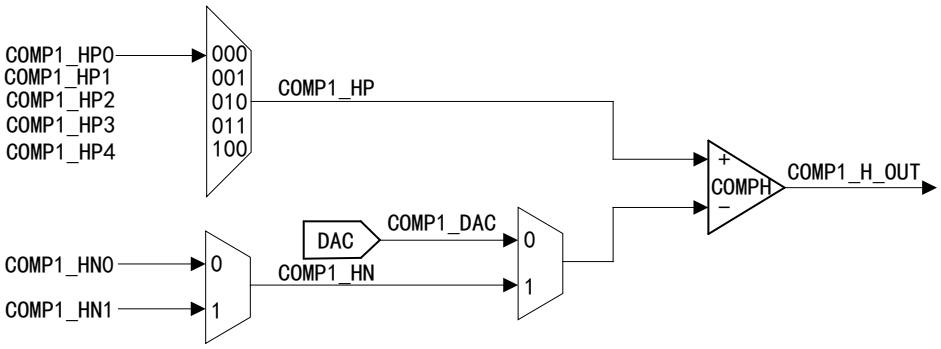


Figure 48 External input pin selection (taking COMP1 as an example)



As shown in the diagram, each COMP module consists of multiple submodules:

(1) High-bit comparator and low-bit comparator

- “H” and “L” represent the high-bit comparator and the low-bit comparator respectively;
- The positive input end of the comparator is driven by external pins, while the negative input end is driven by external pins or 12-bit programmable reference DAC.
- Each comparator can generate a digital output, indicating whether the voltage on the positive input end is greater than the voltage on the negative input end. Each comparator output will pass through a programmable digital filter, which can remove the accidentally triggered signals. If filtering is not needed, unfiltered output can be used.

(2) 2 12-bit reference DAC

(3) 2 digital filters

(4) 1 decreasing ramp generator

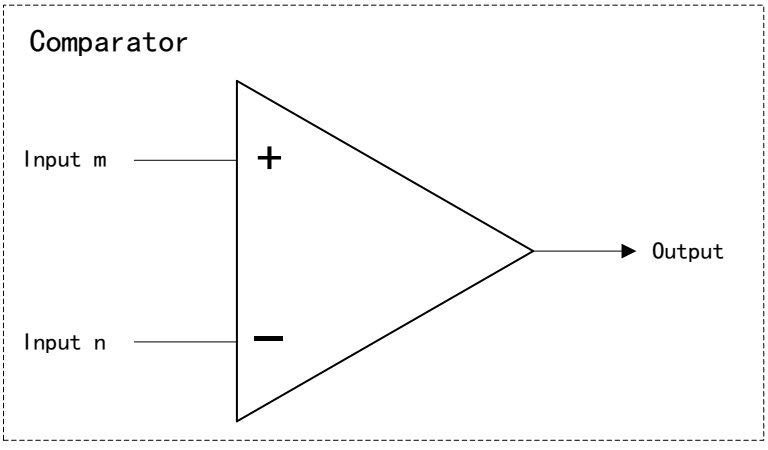
- (Optional) 12-bit reference DAC value used to control the high-bit comparator

As shown in the above figure, the CTRIPH/L signal is connected to the PWM X-BAR for PWM trip response, and the CTRIPOUTH/ signal is connected to the Output X-BAR for external signal transmission. For information about the multiplexing configuration of PWM X-BAR and Output X-BAR, please refer to PWM and GPIO.

29.5 Functional description

29.5.1 Basic structure of the comparator

Figure 49 Basic Structure of the Comparator



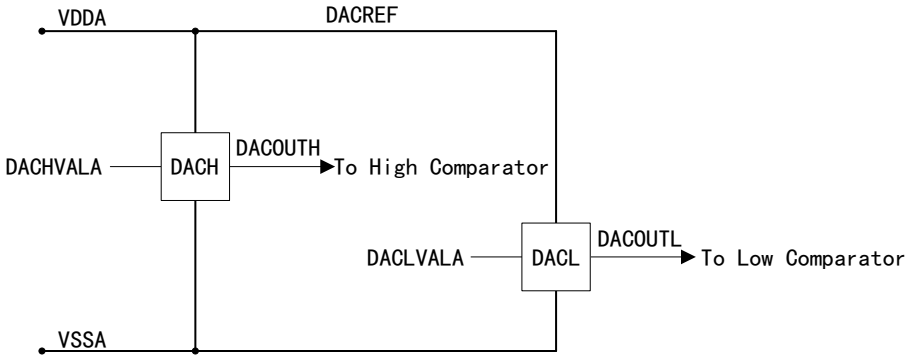
The basic structure of the comparator is shown in the above figure:

(1) When the voltage at the positive input end of the comparator is greater than the voltage at the negative input end, a high level will be outputted, that is, when m is greater than n, Output=1;

(2) When the voltage at the positive input end of the comparator is less than the voltage at the negative input end, a low level will be outputted, that is, when m is less than n, Output=0.

29.5.2 Reference DAC

Figure 50 Internal DAC Block Diagram



(1) Internal reference voltage output:

- Each 12-bit reference DAC is configured to generate a reference voltage, which is used as the negative input end of the corresponding comparator;
- The output of the 12-bit reference DAC is internal, so the internal reference voltage output cannot be directly observed from the outside.

(2) Double register bank, DACxVALA register bank, and DACxVALS register bank (x represents H/L):

- DACxVALS: The DAC shadow register is a writable register, and the register value can be immediately loaded into the corresponding DACxVALA register, or it can be synchronously loaded into the DACxVALA register with the next PWMSYNC event

(3) DACx input source: The value of DACx can be obtained from either the DACxVALA register or the ramp generator

(4) 12-bit reference DAC operating range: V_{DACREF}~ V_{SSA}

(5) The reference voltage source cannot be configured and is fixed at V_{DDA}.

(6) Under ideal conditions, the formula for calculating the output of the 12-bit reference DAC is as follows:

$$Out_{DAC} = \frac{(1 + DACxVALA) * DACREF}{4096}$$

Note: The COMP module instance can be enabled before the benchmark DAC value is configured.

29.5.3 Operation of ramp generator

By setting COMPDACCTL[DACSOURCE]=1, select the ramp generator as the

source of the high 12-bit reference DAC. At this point, the ramp generator generates a downward ramp input to supply the high 12-bit reference DAC.

When COMPDACCTL[DACSOURCE]=1, the value of the ramp value register (RAMPSTS) is loaded from the ramp maximum reference shadow register (RAMPMAXREFS) and remains static. Once the selected PWMSYNC signal is received, the value of RAMPMAXREFS will be subtracted from RAMPSTS in each subsequent APBCLK cycle.

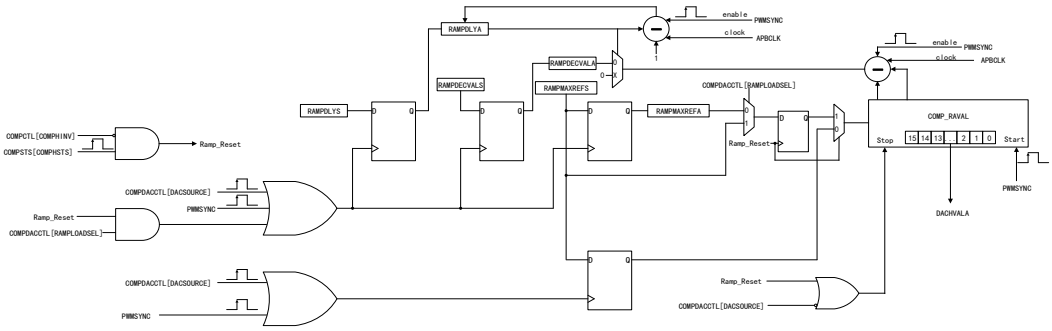
When selecting the ramp generator, the [15:4] bit in the ramp value register (RAMPSTS) is used as the input of the high 12-bit reference DAC, and the [3:0] bit is used as the prescale value of the decreasing ramp rate, and is configured through the RAMPDECVALA register.

Delay counter

The ramp delay value active register (RAMPDLYA) serves as the delay counter, and is used to control the time of starting subtracting the value of RAMPMAXREFS. After the selected PWMSYNC signal is received, RAMPDLYA will decrease by 1 in each APBCLK cycle, and when the value of RAMPDLYA reaches 0, the operation of subtracting the value of RAMPMAXREFS from RAMPSTS begins.

29.5.3.1 Operation under normal circumstances

Figure 51 Ramp Generator



As shown in the above figure, on the rising edge of each COMPDACCTL[DACSOURCE], PWMSYNC and COMPSTS[COMPHSTS], the status of the ramp generator will change:

Table 114 Operation of Ramp Generator

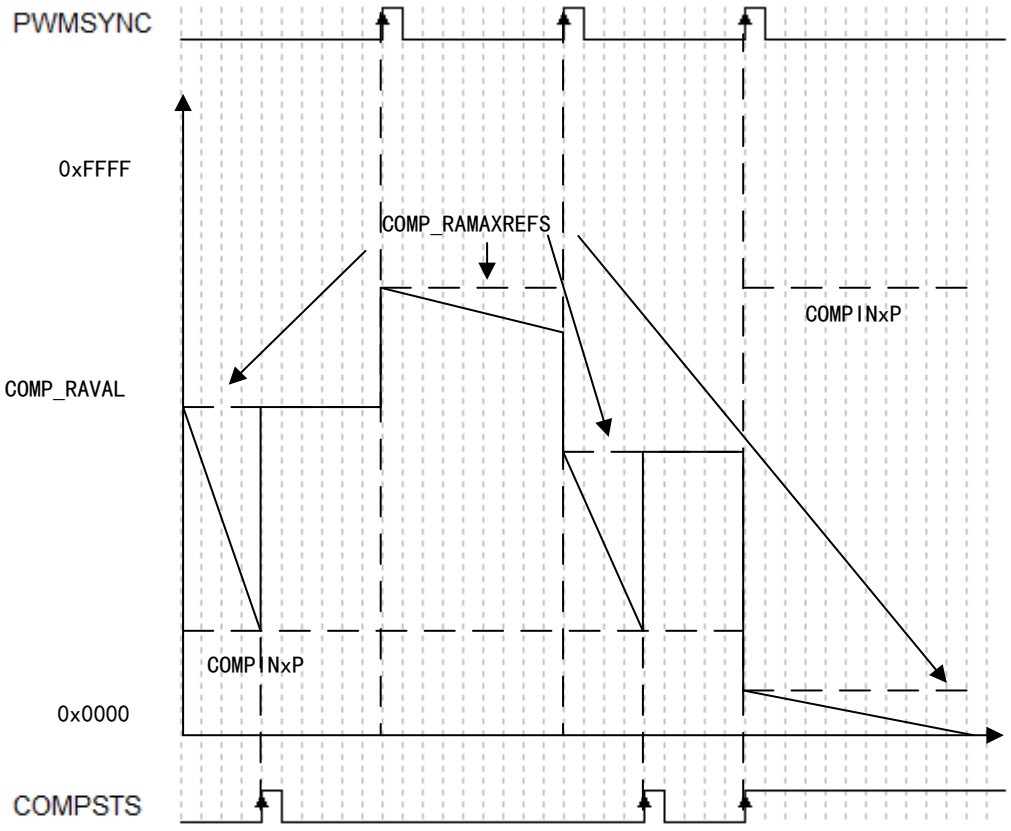
Condition	RAMPMAXREFA, RAMPDECVALA and RAMPDLYA registers	RAMPSTS register
Rising edge of COMPDACCTL[DACSOURCE]	Load values from the corresponding shadow registers	Load values from the RAMPMAXREFS register

Condition	RAMPMAXREFA, RAMPDECVALA and RAMPDLYA registers	RAMPSTS register
Rising edge of COMPSTS[COMPHSTS], and COMPDACCTL[RAMPLOADSEL]=0	-	Load values from the RAMPMAXREFA register, and stop decreasing
Rising edge of COMPSTS[COMPHSTS], and COMPDACCTL[RAMPLOADSEL]=1	Load values from the corresponding shadow registers	Load values from the RAMPMAXREFS register, and stop decreasing.
PWMSYNC	Load values from the corresponding shadow registers	Load values from the RAMPMAXREFS register, and start decreasing when the RAMPDLYA counter is zero

Besides, when the value of RAMPSTS decreases to 0, it will remain static until the next selected PWMSYNC signal is received.

29.5.3.2 Operation under extreme circumstances

Figure 52 Operation of Ramp Generator Under Extreme Circumstances



As shown in the above figure, since the state of the ramp generator changes at the rising edge of each PWMSYNC and COMPSTS[COMPHSTS], when these

two events are simultaneously on the rising edge or the rising edge time is very close, the following operation will occur:

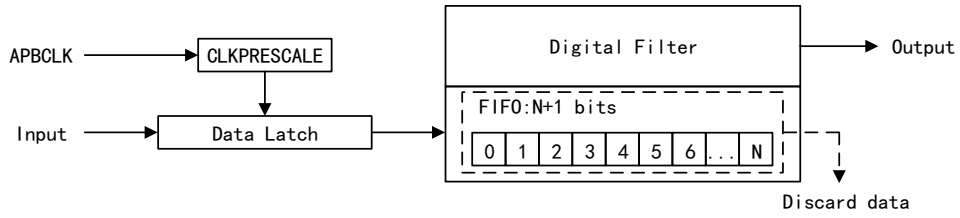
Table 115 Operation under Extreme Circumstances

Condition	Operation
COMPSTS[COMPHSTS] rising edge occurs within one or more cycles before the rising edge of PWMSYNC	When COMPSTS[COMPHSTS] rising edge occurs, RAMPSTS stops decreasing; when the RAMPDLYA counter decreases to 0, the RAMPSTS register begins to decrease after the rising edge of PWMSYNC is received.
Occur simultaneously	The rising edge event of PWMSYNC takes priority, the rising edge event of COMPSTS[COMPHSTS] is ignored, and the RAMPSTS register will not stop decreasing. When the RAMPDLYA counter decreases to 0, the RAMPSTS register begins to decrease.
One or more cycles after the rising edge of PWMSYNC, before the RAMPDLYA counter decreases to 0, the COMPSTS[COMPHSTS] rising edge occurs	When the RAMPDLYA counter decreases to 0, the RAMPSTS register will not decrease.
After the rising edge of PWMSYNC, when the RAMPDLYA counter decreases to 0, the COMPSTS[COMPHSTS] rising edge occurs	The RAMPSTS register will not decrease.

29.5.4 Digital filter

29.5.4.1 Conceptual model

Figure 53 Conceptual Model of Digital Filter



The conceptual model of digital filter is shown in the above figure.

Sampling window

The operation of the digital filter is based on a FIFO sampling window (SAMPWIN). This sampling window samples from the filter input. The actual size of the internal sampling window is SAMPWIN+1.

Filter output

The filter output is the majority threshold (i.e. the value with the most

occurrences) in the sampling window, and the majority threshold is defined in THRESH. The output state of the filter will change only when there are at least THRESH samples that are opposite to the current output state in the sampling window; otherwise, the output of the filter remains unchanged. The actual value of the internal majority thresholds is THRESH+1.

Conditions of majority threshold

In order to ensure normal operation of the filter, the majority thresholds must meet the following conditions:

$$\frac{\text{SAMPWIN}}{2} < \text{THRESH} \leq \text{SAMPWIN}$$

Sampling ratio of the filter

The sampling ratio of the filter is defined by CLKPRESCALE, and the filter FIFO performs a sampling operation every prescaler clock. The actual value of the prescaler is CLKPRESCALE+1. The filter will discard the old data in the FIFO in order to store new data.

Example:

As shown in the conceptual model, when SAMPWIN is set to 8, the value of the internal sampling window is 9, and when the majority threshold is set to 4, the internal threshold is 5.

(1) When the number of "1" inputted in the sampling window is 7, assuming it is 110111110, the output state of the filter is changed and outputs "1".

(2) When the number of "1" inputted in the sampling window is 5, assuming it is 110101100, the output state of the filter is changed and outputs "1".

(3) When the number of "1" inputted in the sampling window is 3, assuming it is 011000010, the output state of the filter remains unchanged and outputs "0".

29.5.4.2 Filter initialization sequence

(1) Configure and enable the comparator

(2) Configure filter parameters: Configure the sampling window (SAMPWIN), majority threshold (THRESH), and clock prescale value (CLKPRESCALE)

(3) Initialize the sample value in the digital FIFO window: Set FILINIT to initialize all sample values as the filter input value.

(4) Clear COMPSTS latch: When a latch path is required, use the COMPSTSCLR register to clear the COMPSTS latch.

(5) Configure CTRIP and CTRIPOUT signal paths

(6) Configure target modules (if needed): Configure the target modules such as GPIO module as required so as to receive filtered signals.

29.5.5 Delay

The delay added to this module is as follows:

(1) Synchronization block: Add a delay of 1~2 clocks;

(2) Digital filter: When the digital filter is in a bypassed state (i.e. all digital filters are set to 0), a delay of 2 clocks will be increased;

(3) Latch: Add a delay of 1 clock.

29.5.6 LATCLR, PWMSYNC and PWMBLANK signals

LATCLR signal

After the required delay, the LATCHCLR signal is used to keep the outputs of the digital filter, synchronization block, and latch in a reset state.

The LATCHCLR signal can be activated through software or through the PWMSYNC signal.

- Software activation: Activate by setting xLATCLR
- PWMSYNC signal activation: When xSYNCCLR is set, the LATCLR signal can be activated through the PWMSYNC signal.

Extend LATCLR signal: When a longer-time LATCLR signal is required, it can be extended by setting the BLANKEN and using the PWMBLANK signal.

PWMSYNC and PWMBLANK signals

The PWMSYNC signal is generated by the time base submodule of the PWM module, and the PWMBLANK signal is generated by the digital compare submodule of the PWM module. For details, please refer to Functional Characteristics of Time Base Submodule and Digital Compare Submodule of PWM module.

When COMPDACCTL [SWLOADSEL] is set to 1, the PWMSYNC signal for loading DACxVALA is a level-triggered loading. If both the time base counter (TBCNT) and time base period (TBPRD) of PWM are 0, PWMSYNC will remain in a high-level state, and DACxVALA will immediately be loaded from DACxVALS, regardless of the value of COMPDACCTL [SWLOADSEL].

It is recommended to configure PWM first, and then set COMPDACCTL [SWLOADSEL] to 1, which is because: when COMPDACCTL [SWLOADSEL] is set to 1, the PWMSYNC signal will load DACLVALA through level-triggered loading. If both TBCNT and TBPRD of PWM are 0, PWMSYNC will remain in a high-level state, and regardless of the value of COMPDACCTL [SWLOADSEL],

DACLVALA will immediately be loaded from DACLVALS.

It is recommended to configure the PWM first, then set COMPDACCTL [SWLOADSEL] to 1, for the following reasons: When COMPDACCTL [SWLOADSEL] is set to 1, the PWMSYNC signal will load DACLVALA via level-triggered loading.If both TBCNT and TBPRD of the PWM are 0, PWMSYNC will remain high, and DACLVALA will be loaded immediately from DACLVALS regardless of the value of COMPDACCTL [SWLOADSEL].

Note: For information about the description of the PWMSNC signal, please refer to PWM.

29.5.7 COMP calibration

29.5.7.1 Offset error source

In the COMP module, there are two types of offset error sources. For actual values of the errors, please refer to the Datasheet.

(1) Comparator offset error: It is referred to as "input offset error" in the Datasheet. When both inputs of the comparator are driven by one pin, only the comparator offset error is considered;

(2) Comparator DAC offset error (hereinafter referred to as compdac offset error): It is referred to as "static offset error" in the Datasheet. When the inverted input of the comparator is driven by compdac, only the compdac offset error is considered because it includes the comparator offset error.

29.5.7.2 Calibration

Due to the existence of the above two types of errors, the COMP module needs to be calibrated to ensure correct tripping at the expected level.

When the inverted input of the comparator is driven by compdac, the module can be calibrated by scanning down the compdac, and the calibration process is as follows:

(1) Set the initial value of compdac: Set the value of compdac to the maximum value, i.e. 0xFF;

- (Optional) If the approximate DC voltage (Vtarget) at the non-inverted input is known, the initial value of compdac can be set to Vtarget + static offset error +Margin for quick calibration. Margin represents a certain degree of protection margin. When Vtarget is unknown, it can be converted by ADC.

(2) compdac value decrease: compdac value decreases by 1;

(3) Wait for compdac to be stable: After the compdac value decreases, wait for its output to be stable;

(4) Clear latch

(5) Wait to see whether the latch is set

(6) Check the latch status: When the latch is set, it indicates that a trip code has been found and the process ends.

- (Optional) Verify the trip code through the following steps: increase the compdac value by 1, clear latch, wait to see if the latch is set, and then the latch should be in an unset state, to verify that the trip code is correct.

(7) When the latch is not set, return to step (2) until the correct trip code is found.

Note:

(1) Before calibration, ensure that there is a static DC signal at the non-inverted input end of the comparator.

(2) During calibration, the hysteresis function of the comparator should be disabled, and be enabled again after calibration is completed.

(3) Input noise processing: If the signal at the non-inverted input end has noise, a latch with a non-zero filter can be used and appropriate filtering parameters can be set according to the noise level to compare the noise interference in the calibration process.

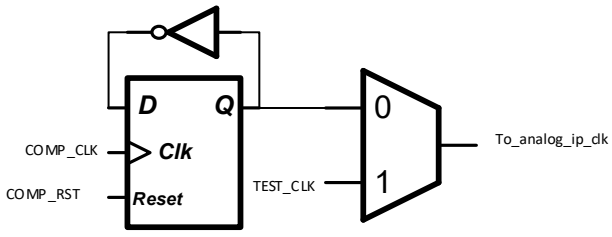
When both inputs of the comparator are driven by one pin, the COMP module can also be calibrated. In such case, the process remains unchanged, but an external scanning voltage needs to be applied to the inverted input pin of the comparator.

29.5.8 COMP clock

29.5.8.1 Analogue clock

The clock frequency of the analogue IP is set to half the controller clock frequency, as shown in the figure below.

Figure54 Clock scheme for the analogue comparator



29.5.8.2 Digital Clock

When the comparator is activated, if the COMP module clock is disabled, each structure in the COMP module will perform the following operations:

(1) Comparator: Even if the COMP module clock is disabled, the comparator will still trigger based on changes in input voltage, unaffected by the disabled clock.

(2) Ramp generator, synchronization block, and digital filter: Frozen in current state

(3) 12-bit reference DAC: When the 12-bit reference DAC drives the negative input end of the comparator, the voltage on the negative input end remains static and is unaffected by the disabled clock, but DACLVALA cannot update data from the ramp generator or DACLVALS.

29.6 Register bank address

Table 116 COMP Register Bank Address

Device register	Register bank	Start address	End address
Comp1Regs	COMP_REGS	0x4002_1000	0x4002_13FF
Comp2Regs	COMP_REGS	0x4002_1400	0x4002_17FF
Comp3Regs	COMP_REGS	0x4002_1800	0x4002_1BFF
Comp4Regs	COMP_REGS	0x4002_1C00	0x4002_1FFF

29.7 Register address mapping

Table 117 COMP_REGS Offset Address

Register name	Register description	Offset address	WRPRT
COMPCTL	Control register	0x00	√
COMPHYSCTL	Hysteresis control register	0x02	√
COMPSTS	Status register	0x04	-
COMPSTSCLR	Status clear register	0x06	√
COMPDACCTL	DAC control register	0x08	√
DACHVALS	High-bit DAC shadow register	0x0C	-
DACHVALA	High-bit DAC active register	0x0E	-
RAMPMAXREFA	Ramp maximum reference value active register	0x10	-
RAMPMAXREFS	Ramp maximum reference value shadow register	0x14	-
RAMPDECVALA	Ramp decrement value active register	0x18	-
RAMPDECVALS	Ramp decrement value shadow register	0x1C	-
RAMPSTS	Ramp value register	0x20	-
DACLVALS	Low-bit DAC shadow register	0x24	-
DACLVALA	Low-bit DAC active register	0x26	-

Register name	Register description	Offset address	WRPRT
RAMPDLYA	Ramp delay value active register	0x28	-
RAMPDLYS	Ramp delay value shadow register	0x2A	-
CTRIPLFILCTL	Low-bit filter control register	0x2C	√
CTRIPLFILCLKCTL	Low-bit filter clock control register	0x2E	√
CTRIPHFILCTL	High-bit filter control register	0x30	√
CTRIPHFILCLKCTL	High-bit filter clock control register	0x32	√
COMPLOCK	Lock register	0x34	√

29.8 Register functional description

29.8.1 Control register (COMPCTL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	COMPHSOURCE	R/W	High comparator input source select 0: Select the comparator inverting phase input driven by internal DAC 1: Select the comparator inverting phase input driven by external pins	0h
1	COMPHINV	R/W	High comparator invert output enable 0: Disable (phase not inverted) 1: Enable (phase inverted)	0h
3:2	CTRIPHSEL	R/W	High comparator PWMH source select Select the output of driving PWMH 00: Asynchronous comparator 01: Synchronous comparator 10: Digital filter 11: Latched digital filter	0h
5:4	CTRIPOUTHSEL	R/W	High comparator OUTH source select Select the output of driving OUTH 00: Asynchronous comparator 01: Synchronous comparator 10: Digital filter 11: Latched digital filter	0h
6	ASYNCHEN	R/W	High comparator asynchronous output transmit enable The output of the asynchronous comparator and the output of a latched digital filter signal are connected to an OR logic gate. This field is used to control whether the output result of the asynchronous comparator is allowed to be fed back to the OR logic gate when CTRIPHSEL =11 or CTRIPOUTHSEL =11.	0h

Field	Name	R/W	Description	Reset value
			0: The output of the asynchronous comparator will not be fed into the OR logic gate 1: The output of the asynchronous comparator will be fed back to the OR logic gate	
7	Reserved			0h
8	COMPLSOURCE	R/W	Low comparator input source 0: Select the comparator inverting phase input driven by internal DAC 1: Select the comparator inverting phase input driven by external pins	0h
9	COMPLINV	R/W	Low comparator invert output enable 0: Disable (phase not inverted) 1: Enable (phase inverted)	0h
11:10	CTRIPLSEL	R/W	Low comparator PWML source select 00: Asynchronous comparator 01: Synchronous comparator 10: Digital filter 11: Latched digital filter	0h
13:12	CTRIPOUTLSEL	R/W	Low comparator OUTL source select 00: Asynchronous comparator 01: Synchronous comparator 10: Digital filter 11: Latched digital filter	0h
14	ASYNCLEN	R/W	Low comparator asynchronous output transmit enable The output of the asynchronous comparator and the output of a latched digital filter signal are connected to an OR logic gate. This field is used to control whether the output result of the asynchronous comparator is allowed to be fed back to the OR logic gate when CTRIPLSEL =11 or CTRIPOUTLSEL =11. 0: The output of the asynchronous comparator will not be fed into the OR logic gate 1: The output of the asynchronous comparator will be fed back to the OR logic gate	0h
15	COMPDACE	R/W	Comparator/DAC enable 0: Disable 1: Enable	0h

29.8.2 Hysteresis control register (COMPHYSCTL)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	COMPHYS	R/W	Comparator hysteresis select Select the hysteresis value of the comparator. 0000: No hysteresis 0001: The hysteresis value is the typical value 0010: The hysteresis value is the typical value*2 0011: The hysteresis value is the typical value*3 0100: The hysteresis value is the typical value*4 Others: Undefined	0h
15:4	Reserved			0h

29.8.3 State register (COMPSTS)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	COMPHSTS	R	High comparator digital filter output Flag	0h
1	COMPHLATCH	R	High comparator digital filter output Latched value	0h
7:2	Reserved			0h
8	COMPLSTS	R	Low comparator digital filter output Flag	0h
9	COMPLLATCH	R	Low comparator digital filter output Latched value	0h
15:10	Reserved			0h

29.8.4 State clear register (COMPSTSCLR)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	HLATCHCLR	R-0/W1S	High comparator latch clear This bit is used for software to clear COMPSTS[COMPHLATCH]. 0: No effect 1: Generate a pulse signal so as to clear COMPSTS[COMPHLATCH]	0h
2	HSYNCCLREN	R/W	High comparator latch PWMSYNC clear This bit is used to control whether to enable PWMSYNC to clear COMPSTS[COMPHLATCH] 0: PWMSYNC will not clear COMPSTS[COMPHLATCH] 1: PWMSYNC is enabled to clear COMPSTS[COMPHLATCH]	0h
8:3	Reserved			0h

Field	Name	R/W	Description	Reset value
9	LLATCHCLR	R-0/W1S	Low comparator latch clear This bit is used for software to clear COMPSTS[COMPLLATCH]. 0: No effect 1: Generate a pulse signal so as to clear COMPSTS[COMPLLATCH]	0h
10	LSYNCCLREN	R/W	Low comparator latch PWMSYNC clear This bit is used to control whether to enable PWMSYNC to clear COMPSTS[COMPLLATCH] 0: PWMSYNC will not clear COMPSTS[COMPLLATCH] 1: PWMSYNC is enabled to clear COMPSTS[COMPLLATCH]	0h
15:11	Reserved			0h

29.8.5 DAC control register (COMPDACCTL)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DACSOURCE	R/W	DAC source select Select the source to update to DACHVALA. 0: DACHVALA is updated by DACHVALS 1: DACHVALA is updated by the ramp generator	0h
4:1	RAMPSOURCE	R/W	PWMSYNC source select Select the PWMSYNC signal for the COMP module 0000: PWM1SYNC 0001: PWM2SYNC 0010: PWM3SYNC …… n: PWM(n-1)SYNC n (decimal) represents the maximum value of the available PWMSYNC signal	0h
5	SELREF	R/W	DAC reference voltage Select Select the reference voltage source for the internal comparator DAC. 0: VDDA 1：VDAC	0h
6	RAMPLOADSEL	R/W	Ramp load select After COMPSTS[COMPHSTS] is triggered, select the source to update to RAMPSTS. 0: RAMPMAXREFA 1: RAMPMAXREFS	0h
7	SWLOADSEL	R/W	Software load select Select the source to update to DACxVALA(x=H/L). 0: DACxVALS is updated by SYSCLK 1: DACxVALS is updated by PWMSYNC signal	0h

Field	Name	R/W	Description	Reset value
11:8	BLANKSOURCE	R/W	PWMBLANK source select Select PWMnBLANK as the PWMBLANK signal 0000: PWM1BLANK 0001: PWM2BLANK 0010: PWM3BLANK …… n: PWM (n-1) BLANK n (decimal) represents the maximum value of the available PWMSYNC signal	0h
12	BLANKEN	R/W	PWMBLANK enable Enable the PWMBLANK signal. 0: Disable 1: Enable	0h
13	Reserved			0h
15:14	FREESOFT	R/W	ramp generator run mode select When the simulation pauses, select the running mode of the ramp generator. 00: The ramp generator stops running immediately 01: The ramp generator completes the current ramp and stops at the next PWMSYNC Other: The ramp generator runs freely and is not affected by the pause of simulation	0h

29.8.6 High-bit DAC shadow register (DACHVALS)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVAL	R/W	High DAC shadow value When COMPDACCTL[DACSOURCE]=0, the value of DACHVALS register is loaded into the DACHVALA register on the trigger signal selected by COMPDACCTL[SWLOADSEL].	0h
15:12	Reserved			0h

29.8.7 High-bit DAC active register (DACHVALA)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVAL	R	High DAC Active value The value actively driven by the high-bit DAC.	0h
15:12	Reserved			0h

29.8.8 Ramp maximum reference value active register (RAMPMAXREFA)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RAMPMAXREF	R	Ramp maximum reference active value Latched value to be loaded into the ramp generator (i.e. RAMPSTS register).	0h

29.8.9 Ramp maximum reference value shadow register (RAMPMAXREFS)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RAMPMAXREF	R/W	Ramp maximum reference shadow value Unlatched value to be loaded into the ramp generator (i.e. RAMPSTS register).	0h

29.8.10 Ramp decrement value active register (RAMPDECVALA)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RAMPDECVAL	R	Ramp decrement value active Latched value to be subtracted from RAMPSTS.	0h

29.8.11 Ramp decrement value shadow register (RAMPDECVALS)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RAMPDECVAL	R/W	Ramp decrement value shadow Unlatched value to be loaded into RAMPDECVALA.	0h

29.8.12 Ramp value register (RAMPSTS)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	RAMPVALUE	R	Ramp value The current value of the ramp generator.	0h

29.8.13 Low-bit DAC shadow register (DACLVALS)

Offset address: 0x24

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVAL	R/W	Low DAC shadow value The value of the DACVAL register is loaded into the DACLVALA register on the trigger signal selected by COMPDACCTL[SWLOADSEL].	0h

Field	Name	R/W	Description	Reset value
15:12	Reserved			0h

29.8.14 Low-bit DAC active register (DACLVALA)

Offset address: 0x26

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	DACVAL	R	Low DAC Active value The value actively driven by the low-bit DAC.	0h
15:12	Reserved			0h

29.8.15 Ramp delay value active register (RAMPDLYA)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
12:0	DELAY	R	Ramp delay active value Latched value of the number of clock cycles that the ramp generator must wait for before starting decreasing operation after receiving the PWMSYNC signal.	0h
15:13	Reserved			0h

29.8.16 Ramp delay value shadow register (RAMPDLYS)

Offset address: 0x2A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
12:0	DELAY	R/W	Ramp delay shadow value Unlatched value to be loaded into RAMPDLYA register.	0h
15:13	Reserved			0h

29.8.17 Low-bit filter control register (CTRIPLFILCTL)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	Reserved			0h
8:4	SAMPWIN	R/W	Low filter sample window size The actual value of the number of samples to be monitored is SAMPWIN+1.	0h
13:9	THRESH	R/W	Low filter majority threshold In order to change the output state, there must be at least THRESH samples that are opposite to the current output state within the sampling window. The actual value of the threshold quantity is THRESH+1.	0h

Field	Name	R/W	Description	Reset value
14	Reserved			0h
15	FILINIT	R-0/W1S	Low filter initialization 0: No effect 1: Initialize all sample values as the filter input value	0h

29.8.18 Low-bit filter clock control register (CTRIPLFILCLKCTL)

Offset address: 0x2E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	CLKPRESCALE	R/W	Low filter sample clock prescale The actual value of the clock count for the sampling interval is LFLTCLKPSC+1.	0h
15:10	Reserved			0h

29.8.19 High-bit filter control register (CTRIPHFILCTL)

Offset address: 0x30

Reset type: SYSRSn

Field	**Name**	**R/W**	**Description**	**Reset value**
3:0	Reserved			0h
8:4	SAMPWIN	R/W	High filter sample window size The actual value of the number of samples to be monitored is SAMPWIN+1.	0h
13:9	THRESH	R/W	High filter majority threshold In order to change the output state, there must be at least THRESH samples that are opposite to the current output state within the sampling window. The actual value of the threshold quantity is THRESH+1.	0h
14	Reserved			0h
15	FILINIT	R-0/W1S	High filter initialization 0: No effect 1: Initialize all sample values as the filter input value	0h

29.8.20 High-bit filter clock control register (CTRIPHFILCLKCTL)

Offset address: 0x32

Reset type: SYSRSn

Field	**Name**	**R/W**	**Description**	**Reset value**
9:0	CLKPRESCALE	R/W	High filter sample clock prescale The actual value of the clock count for the sampling interval is CLKPRESCALE+1.	0h
15:10	Reserved			0h

29.8.21 Lock register (DACLOCK)

Offset address: 0x34

Reset type: SYSRSn

Writing 0 to the [3:0] bit of this register has no effect.

Field	Name	R/W	Description	Reset value
0	COMPCTL	R/WSO	write-access to the COMPCTL register Lock 0: The configuration of COMPCTL is not locked 1: The configuration of COMPCTL is locked This bit can only be cleared through system reset.	0h
1	COMPHYSCTL	R/WSO	write-access to the COMPHYSCTL register Lock 0: The configuration of COMPHYSCTL is not locked 1: The configuration of COMPHYSCTL is locked This bit can only be cleared through system reset.	0h
2	DACCTL	R/WSO	write-access to the COMPDACCTL register Lock 0: The configuration of COMPDACCTL is not locked 1: The configuration of COMPDACCTL is locked This bit can only be cleared through system reset.	0h
3	CTRIP	R/WSO	write-access to the COMPCTL register Lock 0: The configuration of CTRIPxFILTCTL and CTRIPxFILCLKCTL is not locked 1: The configuration of CTRIPxFILTCTL and CTRIPxFILCLKCTL is locked This bit can only be cleared through system reset.	0h
15:4	Reserved			0h

30 Pulse width modulator (PWM)

30.1 Full Name and Abbreviation Description of Terms

Table 118 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Rising Edge Delay	RED
Falling Edge Delay	FED
Time-Base	TB
Time Base Counter	TBCTR
Time-Base Clock	TBCLK
Counter-Compare	CMP
Action-Qualifier	AQ
Dead-Band Generator	DB
PWM Chopper	PC
Trip-Zone	TZ
One-Shot	OST
Cycle-by-Cycle	CBC
Event-Trigger	ETRG
Digital Compare	DC
High Resolution Positioner	HRP

30.2 Introduction

PWM is a key element used to control many power electronic systems, which is often applied in consumer and industrial equipment. These power electronic systems include UPS, switching power supply control, digital motor control, and other forms of energy conversion. At the same time, PWM also has the function of DAC, and the duty cycle is equivalent to the analog value of DAC.

Each PWM module consists of eight submodules: time base submodule, counter compare submodule, action qualifier submodule, dead-band generator submodule, PWM chopper submodule, trip zone submodule, event trigger submodule, and digital compare submodule. Each submodule performs specific functions and can be configured by software.

30.3 Main characteristics

(1) It has two PWM outputs, namely PWMxA and PWMxB, which can be configured as unidirectional mode, bilateral symmetric mode, and bilateral asymmetric mode

(2) Asynchronous overwriting control of PWM signals through software

(3) A dedicated 16-bit time base counter can control the period and frequency of PWM

(4) Through programming, adjust the phase of the PWM signal to produce a lag or lead effect with the signals from other PWM modules

(5) Able to independently control the dead band generation of rising edge and falling edge delay

(6) PWM chopping through high-frequency carrier signal, suitable for pulse transformer grid drive

(7) Hardware locked phase relationship can be implemented cycle by cycle

(8) In the event of a fault, the trip zone can be assigned to either cycle-by-cycle trip or one-shot trip

(9) The trip state can force PWM to output high or low logic level or high impedance state

(10) Frequency division can be conducted through programming event to reduce the overhead of CPU response to interrupts

(11) All events can trigger CPU interrupts and ADCSOC

(12) Under all temperature and voltage conditions, the delay error of PWM channels A and B is less than 10 ns, with a fluctuation in delay error between channels A and B of less than 2 ns

30.3.1 Functional characteristics of time base submodule

(1) Time base counter mode:
- Count-up mode
- Count-down mode
- Center-aligned count mode

(2) Time base clock: Obtained through PWMCLK prescale

(3) By configuring the TBCTR frequency or period, the frequency of event occurrence can be controlled

(4) The TBCTR=0 or TBCTR=TBPRD event is generated

(5) The time base phase relative to other PWM modules can be configured

(6) Implement TBCTR synchronization between PWM modules through software or hardware

(7) After synchronization event, configure the TBCTR direction to be up or down

(8) The same value can be written into multiple TBPRD registers of PWM module simultaneously

(9) When the simulator pauses the device, the behavior of TBCTR can be configured

(10) Synchronous output signal source of the PWM module:
- Synchronous input signal
- TBCTR=0
- TBCTR=CMPB
- TBCTR=CMPC
- TBCTR=CMPD
- No synchronous output signal is generated

(11) Configure single and global load of registers in PWM module

30.3.2 Functional characteristics of counter compare submodule

(1) The moment for switching the output levels of PWMxA and PWMxB can be specified

(2) The duty cycle of the output of PWMxA and PWMxB can be specified, and the duty cycle can be controlled by configuring the CMPA and CMPB registers

(3) The same value can be written into multiple CMPx registers of PWM module simultaneously

(4) Configure single and global load of registers in PWM module

(5) The programmable delay interrupt and SOC generation can be specified through the additional comparator (CMPC/D)

(6) Shadow buffering of new comparison values can avoid burrs or waveform distortion during the effective period of PWM

(7) Use the CMPx register to generate events based on the programmable timestamp:
- TBCTR=CMPA
- TBCTR=CMPB
- TBCTR=CMPC
- TBCTR=CMPD

30.3.3 Functional characteristics of action qualifier submodule

(1) The PWM output status can be controlled by force through software

(2) The PWM dead band can be configured and controlled through software

(3) When a time base counter comparison, the trip zone submodule or comparator event occurs, the following actions can be performed:
- Switching of PWMxA and PWMxB to a high level
- Switching of PWMxA and PWMxB to a low level
- Flipping of PWMxA and PWMxB levels
- Not any action

(4) Perform qualitative identification and generate actions (setting, clearing, and flipping) in the following events:
- TBCTR=0
- TBCTR=TBPRD
- TBCTR=CMPA
- TBCTR=CMPB

(5) T1 and T2 events: Trigger events based on comparator, trip, or synchronization events; manage the priority when the above events occur simultaneously

(6) Provide the ability to independently control events when TBCTR is increasing or decreasing

(7) Configure single and global load of registers in PWM module

30.3.4 Functional characteristics of dead-band generator submodule

(1) Generate corresponding signal pairs with dead band relationship, namely PWMxA and PWMxB, from a single PWMxA input

(2) The half-cycle clock mode can be enabled to double the resolution

(3) Allow PWMxB output for phase shift relative to PWMxA

(4) Program the signal pair as:
- Active high (AH)
- Active low (AL)
- Active high complement (AHC)
- Active low complement (ALC)

(5) Insert programmable RED and FED values

(6) Able to control the traditional complementary dead band relationship between the up and down switches

(7) Able to fully bypass the dead-band generator submodule
- The PWM waveform can pass through the dead band generator submodule without any modification

(8) Configure single and global load of registers in PWM module

30.3.5 Functional characteristics of PWM chopper submodule

(1) Programmable chopping frequency

(2) The first pulse has a programmable pulse width, and the second and subsequent pulses have a programmable duty cycle

(3) Able to completely bypass the PWM chopper submodule:
- The PWM waveform can pass through the PWM chopper submodule without any modification

30.3.6 Functional characteristics of trip zone submodule

(1) The trip inputs TZ1n to TZ6n can be flexibly mapped to any PWM module

(2) Support software-forced tripping

(3) Configure the PWM module to respond to the trip signals or digital compare events:
- A trip signal or digital compare event
- All trip signals or digital compare events
- Make no response

(4) In a fault state, the tripping action can be configured as:
- PWMxA and/or PWMxB forcing the output to a high-impedance state
- PWMxA and/or PWMxB forcing the output to a high level
- PWMxA and/or PWMxB forcing the output to a low level
- Ignore any tripping action

(5) The DCAEVT1/2 or DCBEVT1/2 forced events and each trip zone input of the DC submodule can be assigned to single or cycle-by-cycle operation

(6) One-shot trip (OST Trip): Support short circuit and overcurrent protection.

(7) Cycle-by-cycle trip (CBC Trip): Support current limiting operation.

(8) Enable the trip zone to initiate an interrupt, and any input of the trip zone can generate an interrupt

(9) Support the digital compare trip (DC) based on the on-chip analog comparator and/or TZ1n to TZ3n

(10) It can completely bypass the trip zone submodule

30.3.7 Functional characteristics of event trigger submodule

(1) Enable PWM events that trigger interrupts or ADCSOC

(2) Allow software to forcibly trigger interrupts and ADCSOC

(3) Use the prescaler circuit to specify the frequency of event trigger (Mmaximum of 15 times)

(4) The event counters and flag bit provide complete visibility of event generation

(5) Receive event inputs from TB, CC, and DC submodules.

(6) Use the time base direction information to distinguish (up/down) events

(7) Poll, set, or clear the event flag bits

30.3.8 Functional characteristics of digital compare submodule

(1) Use COMP module output and trip zone signals to generate events, and these signals are transmitted through input X-BAR

(2) Generate DCxH and DCxL signals through the following signals:
- Internal NVIC
- TZ1n, TZ2n, and TZ3n inputs
- Output using the Input X-BAR, PWM X-BAR, or COMP module which is inputted through external GPIO peripherals

(3) The event filtering options can be specified to capture TBCTR and generate a blank window, or insert a delay in PWM output or TBCTR based on the capture value

(4) The event filtering (blank window logic circuit) can be used to filter input signals to remove noise

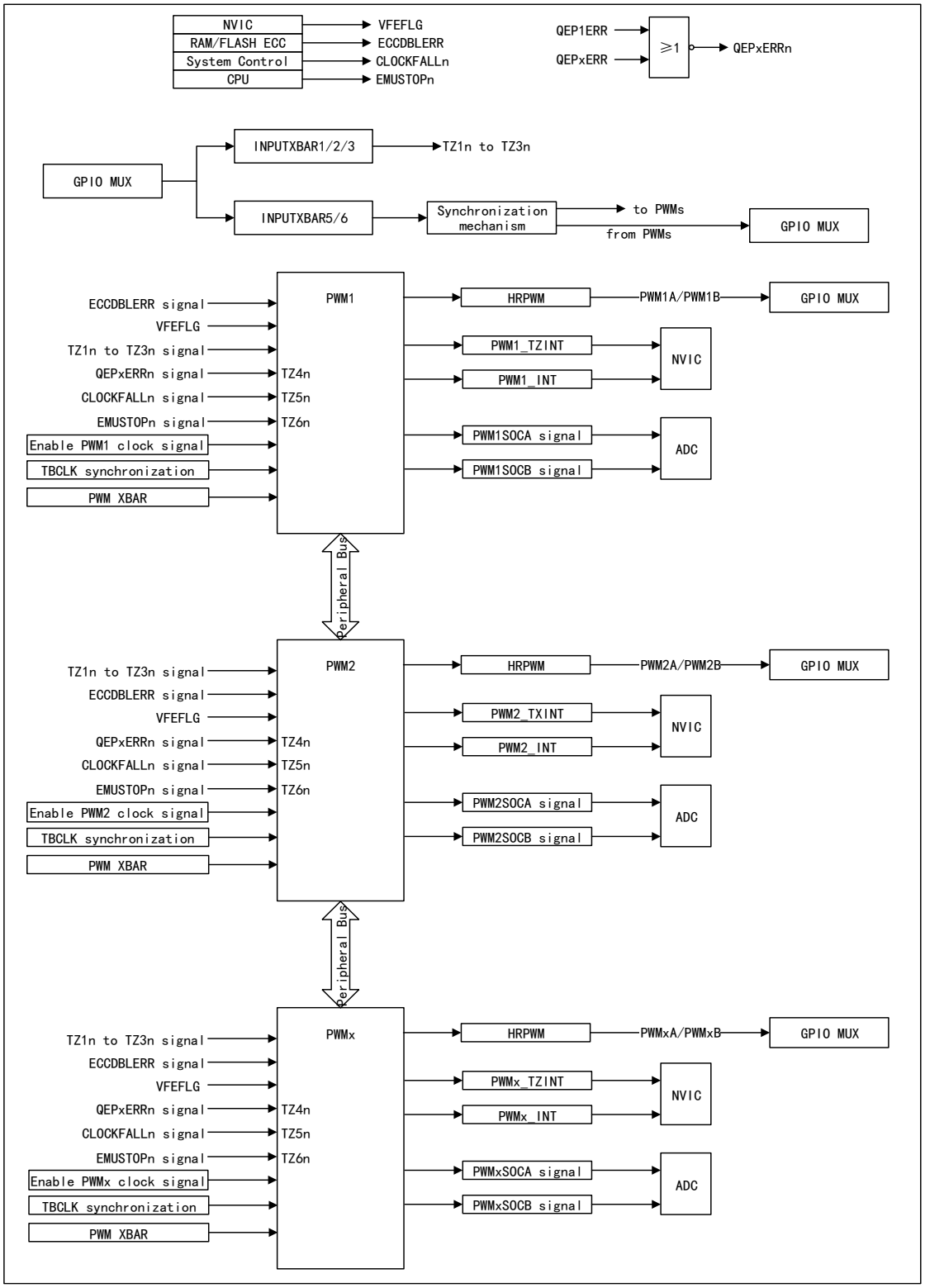
(5) The DCxH and DCxL signals can trigger events, and these events can be filtered or directly passed to the TZ, ETRG, and TB submodules, in order to:
- Generate synchronization events for synchronizing the PWM module TBCTR
- Forcefully generate an event
- Generate ADCSOC
- Generate trip zone interrupt

30.4 Structure block diagram

30.4.1 Structure Block Diagram of Multiple PWM Modules

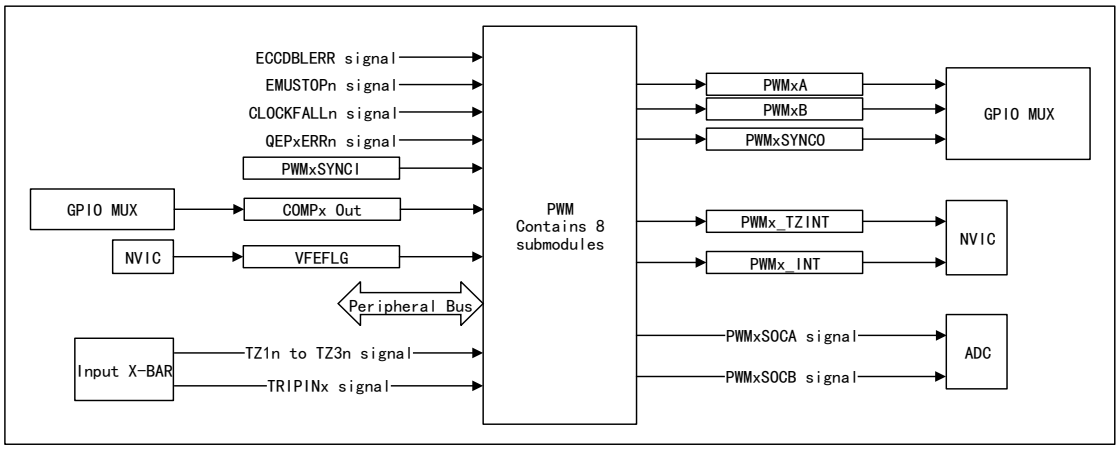
As shown in the figure below, the connection sequence of PWM modules is different. For details, please refer to Synchronization of Time Base Counter. For signals of connection to each PWM module, please refer to Main Signals of PWM.

Figure 55 Structure Block Diagram of Multiple PWM Modules



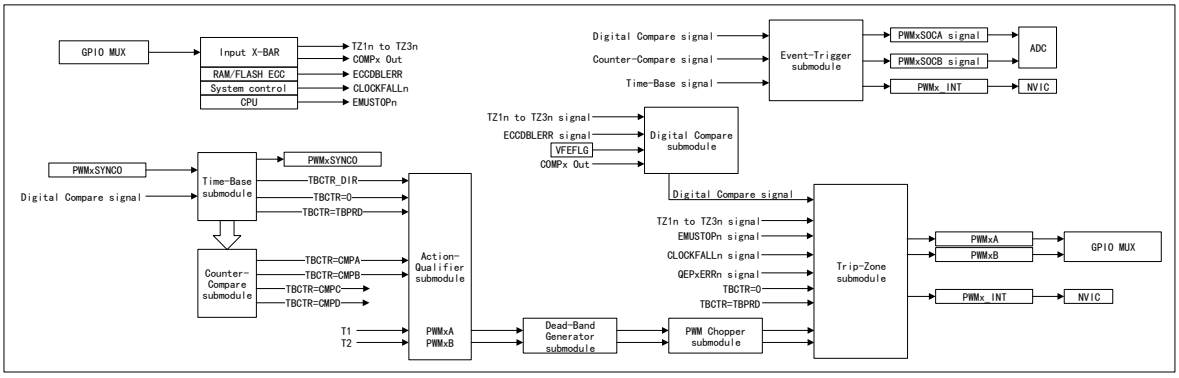
30.4.2 Structure Block Diagram of Single PWM Module

Figure 56 Structure Block Diagram of Single PWM Module



30.4.3 Structure Block Diagram of PWM Submodules

Figure 57 Structure Block Diagram of PWM Submodules



30.5 Functional description

There are multiple instances of PWM modules in the PWM device, with one exception, and each PWM instance is the same. The PWM modules are connected through clock synchronization, and its submodules can operate independently.

30.5.1 PWM enhancement function

(1) Digital compare submodule: This module provides filtering, shielding, and improved trip functions, enhancing the functions of ETRG and TZ submodules.

(2) Dead band resolution: Enhance the dead band clock function, which can increase the resolution of one and a half cycle clocks.

(3) High-resolution cycle: Enable the high-resolution cycle function.

(4) Interrupt and ADCSOC generation: Interrupts and ADCSOC can be generated on TBCTR=0 or TBCTR=TBPRD events. Double-edge PWM

control is implemented and ADCSOC can be generated through the events defined by the digital compare submodule.

(5) PWMxB output: High-resolution extension can be used on this output to provide the function of enabling the high-resolution cycle and duty cycle control.

(6) High-resolution dead band capability:The rising edge delay and falling edge delay of the dead band have high-resolution capability in half-cycle clock mode.

(7) Counter compare submodule: Generate interrupts and SOC events for CMPC and CMPD are compared by other counters.

(8) Dead-band generator submodule: Possess 14-bit dead-band counter, dead-band register and dead-band high-resolution register, and have the function of shadow register. RED and FED functions can be enabled on PWM output.

(9) Event trigger submodule: The prescaler logic can be extended to issue interrupt requests and ADCSOC every 15 events, and software initialization can be conducted for the event counter on the synchronization events.

(10) Digital compare submodule: The digital comparison trip selection logic [DCTRIPSEL] has up to 12 external trip sources selected by the input X-BAR logic, and can perform OR operations on all of these sources (up to 14 [external and internal sources]) to create corresponding DCxEVT.

(11) Simultaneously write to CMPx and TBPRD registers:

- Writing of CMPx, CMPxHR, and TBPRD in any PWM module can be bound to any other PWM module, and all PWM modules can be bound to specific PWM module.
- When synchronizing events, CMPx and TBPRD active registers are loaded into shadow registers

(12) Register address mapping: Additional registers are required. For better alignment and ease of use, the PWM register address space is remapped.

(13) Delay trip function: Add the changes to implement dead band insertion functions so as to support this function, e.g. the delay trip function required for peak current mode control type application scenarios. This is achieved by allowing the comparator events to enter the Action Qualifier (events T1 and T2) as trigger events. If the comparator T1/T2 events are used to change PWM, the change in PWM waveform will not occur immediately. On the contrary, the waveform is synchronized with the next TBCLK.

(14) Enhancement of the dead-band generator submodule: Add the DBCTL shadow registers to allow for dynamic configuration changes.

(15) Single and global loading of registers: Loading from shadow registers supports single and global loading to avoid partial loading, e.g. multi-phase applications. The loading time supports prescale. Global loading can simplify software by deleting interrupts, and ensure that all registers are loaded simultaneously.

(16) Enhancement of the trip zone submodule: Independent flags are added to reflect the tripping status of each TZ source. Changes are made to the trip zone submodule to support certain power converter switching technologies, e.g. valley switching.

(17) Enhancement of the digital compare submodule: The width of the blanking window filter register is increased from 8 bits to 16 bits. The DCCAP function has been enhanced to provide more programmability.

(18) Enhancement related to PWM synchronization: Synchronous generation of PWM based on CMPC and CMPD events. These events can also be used for PWMSYNC pulse selection.

30.5.2 Functional description of PWM pins

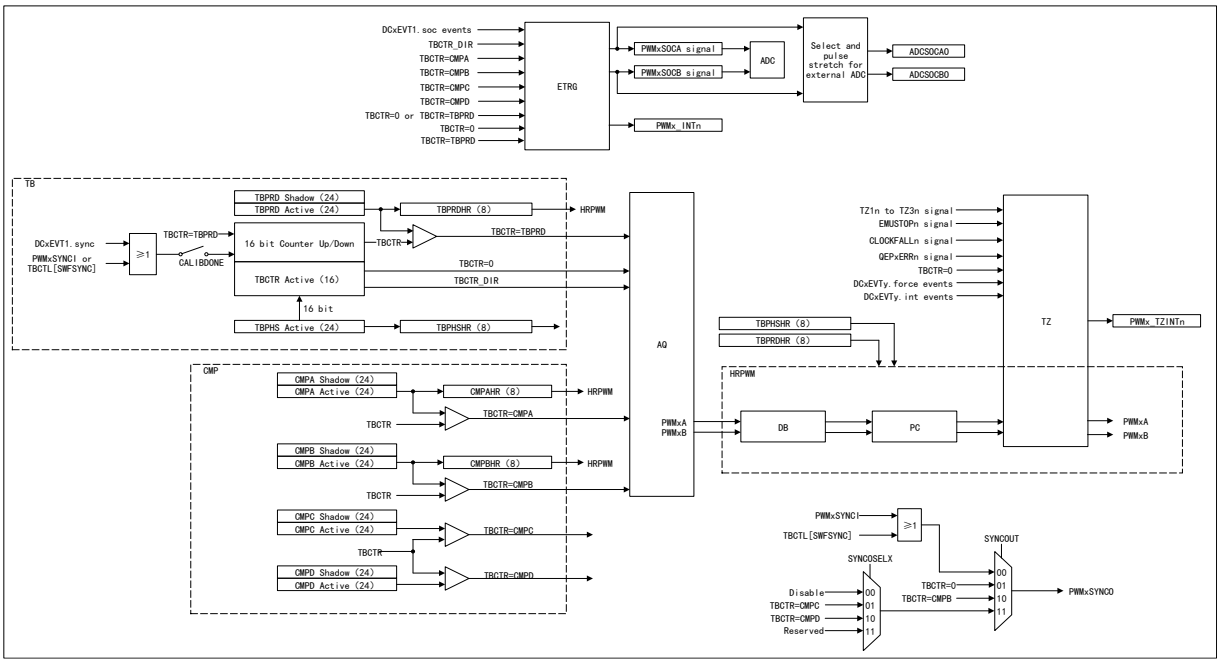
The input X-BAR and PWM XBAR are used to connect the signal on the device pin to this module. Any GPIO on this device can be configured as an input mode. The GPIO input mode needs to be set to asynchronous mode by writing 11 to the corresponding GPACSELy register. Internal pull-up can be configured in the GPxPUD register. When the GPIO mode is used, the GPxINV register can flip the signals. In addition, the signals from TRIP4 to TRIP12 (except for TRIP6) must pass through PWM X-BAR after passing through the input X-BAR.

GPIO multiplexing registers needs to be configured for PWM. You need to configure the GPACSELy fields and keep the corresponding GPACSELy fields at the default value of 0, and then write the expected value to the GPACSELy registers ,to avoid signal jitter on the pins.

For details of GPIO multiplexers, please refer to the Datasheet. For details of GPIO settings and XBAR configurations, please refer to GPIO.

30.5.2.1 **Main PWM signals**

Figure 58 Structure Block Diagram for Interconnection between PWM Module and Key Internal Signals



Note: The PWM DC submodule generates DCxEVTy events based on the input level of TRIPINPUT.

Table 119 Main PWM Signals

Signal type	Signal name	Description
PWM output signal	PWMxA PWMxB	PWM signal outputted to the outside of the device
Trip zone signal	From TZ1n to TZ6n	Each submodule on the device can be configured to use or ignore any trip zone signals, and these input signals notify the PWM module of external faults. This allows trip actions to be configured in the event of clock faults or CPU stoppage. TZ1n to TZ3n: They can be configured as asynchronous input through GPIO peripherals and X-BAR. TZ4n: Connect to the inverted QEPx error signals. TZ5n: Connect to the system clock failure logic. TZ6n: Connect to the EMUSTOPn output of the CPU.
Time base synchronization signal	PWMxSYNCI PWMxSYNCO	The synchronization signal is connected to each PWM module in a daisy chain form. Each module can be configured to use or ignore synchronous inputs. For the purpose of synchronization, the PWM module is divided into three groups:

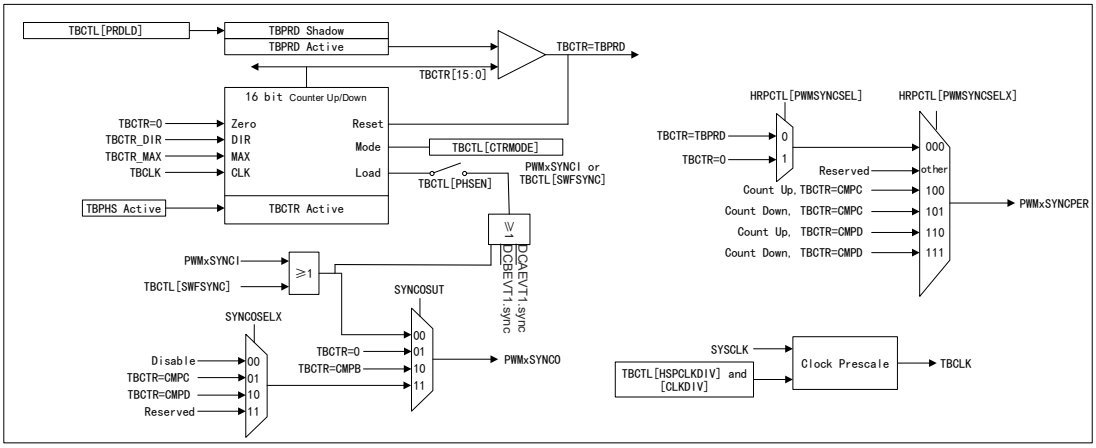
Signal type	Signal name	Description
	PWMxSYNCSEL	External SYNC signal: EXTSYNCIN1 or EXTSYNCIN2 can be used to transmit SYNC signals to the first PWM module on each chain. PWMxSYNCO signal: These modules can also transmit the PWMxSYNCO signals to GPIO; please refer to Synchronization for details. PWMxSYNCSEL signal: Each PWM module can generate this signal, and this signal is used to synchronize GPDAC and COMP. Its function is configured by the HRPCTL register, but it is unrelated to HRPWM. For details about the content of use of PWMxSYNCSEL by GPDAC and COMP, please refer to the corresponding chapter.
Signal for ADC start of conversion	PWMxSOCA	Each PWM module has PWMxSOCA and PWMxSOCB signals. Any PWM module can trigger the start of conversion, and the ETRG submodule configures which event to trigger the start of conversion.
	PWMxSOCB	
Comparator output signal	COMPxOUT	The output signal of the COMP module can be transmitted to one or all 12 trip inputs (TRIPIN1-TRIPIN12) through the input X-BAR, and when used in combination with the trip zone signal, it can generate digital compare events.
Peripheral bus	-	The peripheral bus is 32-bit wide and supports both 16-bit and 32-bit writing of PWM registers.

30.5.3 Functional description of time base submodule

The time base submodule is used to determine the timing of all events for the PWM module, so each PWM module has its own time base submodule. The internal synchronization logic allows the time base of multiple PWM modules to work together to form a system.

30.5.3.1 Structure Block Diagram of Time Base Submodule

Figure 59 Structure Block Diagram of Time Base Submodule



Key signals of time base submodule

Table 120 Key Signals of Time Base Submodule

Signal type	Signal name	Description
Time-Base Clock	TBCLK	Through the PWMCLK prescale, it can be used for all PWM submodules, and this clock determines the rate at which TBCTR increases or decreases.
Direction of time base counter	TBCTR_DIR	It indicates the current direction of the PWM time base counter. This signal is at a high level when the counter increases and at a low level when the counter decreases.
The time base counter is equal to the maximum value	TBCTR_MAX	When TBCTR reaches its maximum value, i.e. TBCTR=0xFFFF, this event is generated. This signal is only used as a status bit.
The time base counter is equal to the cycle value	TBCTR=TBPRD	When the counter value is equal to the active cycle register value, i.e. TBCTR=TBPRD, this signal is generated.
The time base counter is equal to 0	TBCTR=0	When the counter value is equal to zero, i.e. TBCTR=0, this signal is generated.
The time base counter is equal to the active counter compare B register	TBCTR=CMPB	This event is generated by the COMP submodule, and is used to synchronize the output logic.
Time base synchronization signal	PWMxSYNCI Synchronous input of time base	This signal is a pulse input, and is used to synchronize TBCTR with the previous TBCTR in the synchronization chain. The PWM module can be configured to use or ignore this signal. For the first PWM module in each synchronization chain, this signal comes from the previous PWM module or from a device pin, and is transmitted through the INPUT5 or INPUT6 of the input X-BAR. For each subsequent ePWM module in the synchronization chain, this signal comes from the previous PWM module. For example, PWM1 generates the PWM2SYNCI signal, PWM2 generates the PWM3SYNCI signal, and so on. For the synchronization sequence of specific devices, please refer to Synchronization.
	PWMxSYNCO Synchronous output of time base	This output pulse is used to synchronize the TBCTR of the next PWM module in the synchronization chain. The PWM module generates this signal through the following event sources: (1) TBCTR=0 (2) TBCTR=CMPB (3) PWMxSYNCI
	PWMxSYNCPER Synchronous output	This signal is used to synchronize GPDAC and COMP with PWM, and its function is configured by the HRPCTL register, but it is unrelated to HRPWM.

Signal type	Signal name	Description
	of time base peripherals	

30.5.3.2 PWM period and frequency calculation

The frequency of PWM events is controlled through the TBCTR register and counter mode, and the counter mode is selected through the TBCTL register. The figure below shows the relationship between the period T_{PWM} and frequency F_{PWM} in three operating modes when TBPRD is set to 4, TBCLK determines the time increment for each step.

(1) Count-up mode

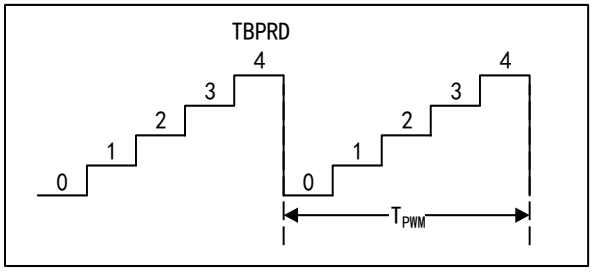
In this mode, TBCTR increases from 0 until it reaches the value in the TBPRD register. When the TBPRD value is reached, the time base counter is reset to 0 and begins to increase again.

In this mode, the calculation formulas for period T_{PWM} and frequency F_{PWM} are:

$$T_{PWM} = (TBPRD + 1) * T_{TBCLK}$$

$$F_{PWM} = \frac{1}{T_{PWM}}$$

Figure 60 Time Base Frequency and Period of Count Up Mode

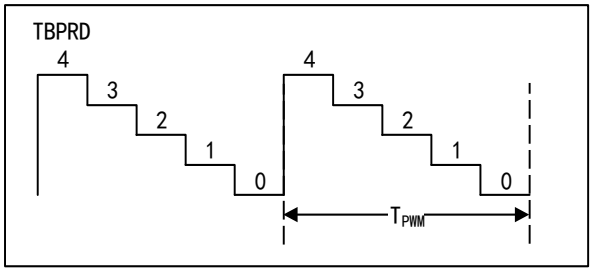


(2) Count-down mode

In this mode, TBCTR decreases from the TBPRD value until it reaches 0. When it reaches 0, TBCTR is reset to the TBPRD value and begins to decrease again.

For the calculation formulas of period T_{PWM} and frequency F_{PWM} in this mode, please refer to Count Up Mode.

Figure 61 Time Base Frequency and Period of Count Down Mode



(3) Center-aligned count mode

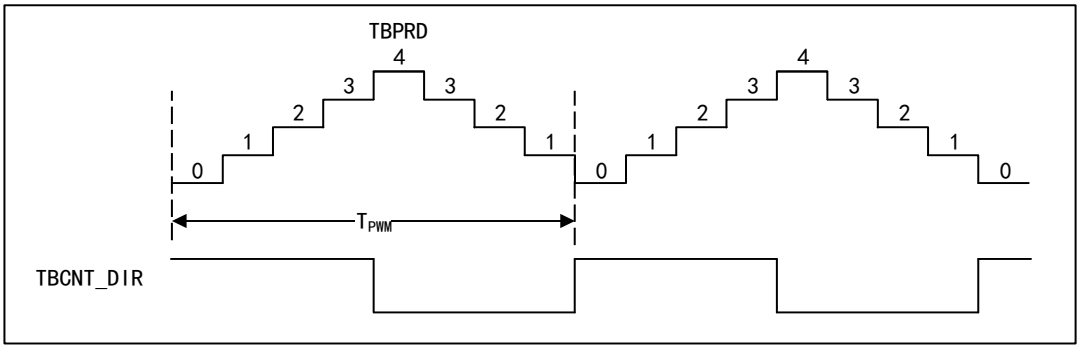
In this mode, TBCTR increases from 0 until it reaches the TBPRD value. After the TBPRD value is reached, TBCTR begins to decrease until it reaches 0. At this point, the counter repeats the pattern and begins to increase.

In this mode, the calculation formulas for period T_{PWM} and frequency F_{PWM} are:

$$T_{PWM} = 2 * TBPRD * T_{TBCLK}$$

$$F_{PWM} = \frac{1}{T_{PWM}}$$

Figure 62 Time Base Frequency and Period of Center-aligned Count Mode



TBPRD shadow register

All shadow register descriptions related to the PWM module are defined as follows:

Table 121 Key Signals of Time Base Submodule

Register classification	Description
Active register	The active register controls the hardware and is responsible for the operations caused or called by hardware.
Shadow register	The shadow register provides the active register with a buffer, that is, a temporary storage location, so it has no direct impact on any hardware control. To prevent data corruption or misoperation caused by asynchronous modification of registers by software, the shadow register will transfer its contents to the active register at a specific point.

The TBPRD register has a shadow register, and the memory mapping address of the active register and the shadow register is the same. Shadowing allows for update of registers to be synchronized with hardware. The PRDLD bit determines to write to which register or read from which register, to enable and disable the TBPRD shadow register.

Table 122 Key Signals of Time Base Submodule

Operating mode of TBPRD register	Description
Shadow mode	PRDLD=0, enable the shadow mode (default), and then both read and write operations to the TBPRD memory addresses will enter the shadow register. When TBCTR=0 or/and the synchronization event is determined by the PRDLDSYNC bit, the contents of the shadow register will be transferred to the active register. The PRDLDSYNC bit is valid only when PRDLD=0. For details of the synchronization input sources, please refer to Synchronization. By configuring the GLDCFG register, the time base period register can also adopt a global loading mechanism. If the global load mode is selected, all registers that enable this mode will undergo content transfer from the shadow registers to the active registers at the same time. The time of content transfer is determined by the GLDCTL register. For details of the global loading mechanism, please refer to Global Loading.
Immediate mode	PRDLD=1, enable the immediate load mode, and then the read or write operations to the TBPRD memory address will directly access the active register.

30.5.3.3 Operating mode of time base counter

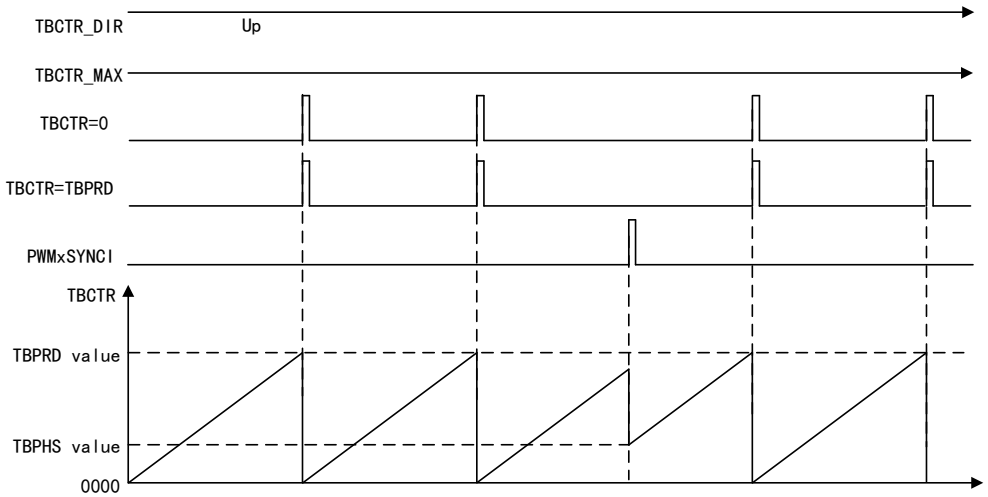
The four operating modes of TBCTR are as follows:

- Count-up mode
- Count-down mode
- Center-aligned count mode
- Freeze mode

When events occur and how the time base responds to the PWMxSYNCI signals in the count-up mode, count-down mode, and center-aligned count mode are shown in the following timing diagram.

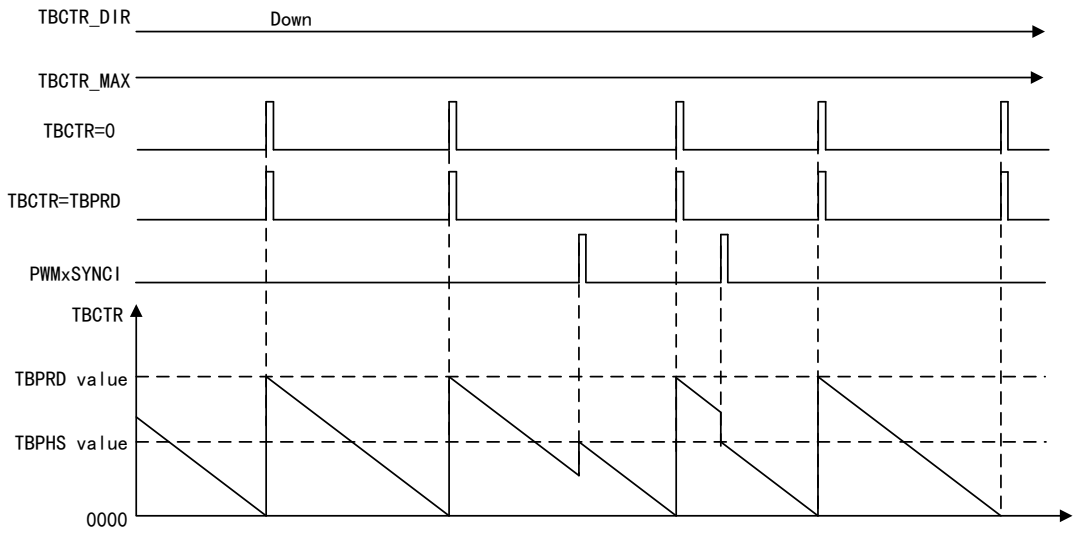
Count-up mode

Figure 63 Count-up Mode Timing Diagram



Count-down mode

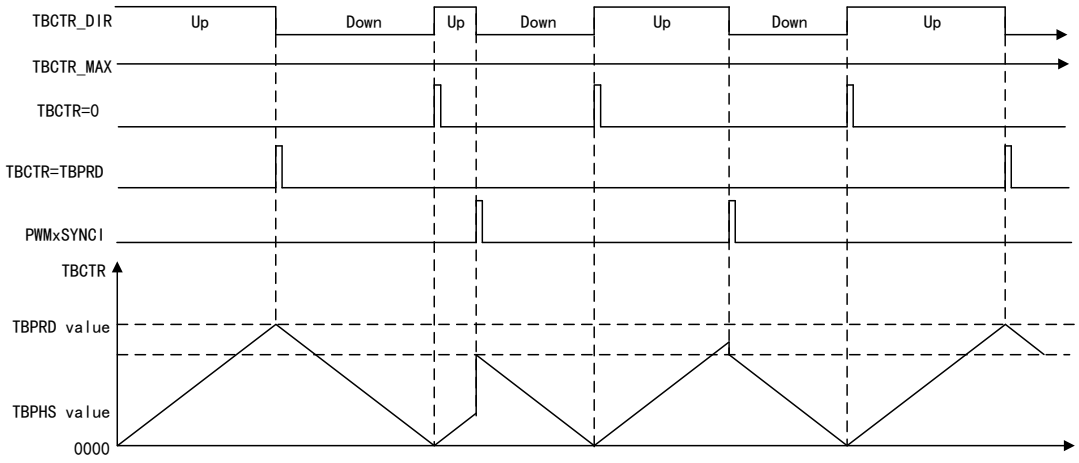
Figure 64 Count-down Mode Timing Diagram



Center-aligned count mode

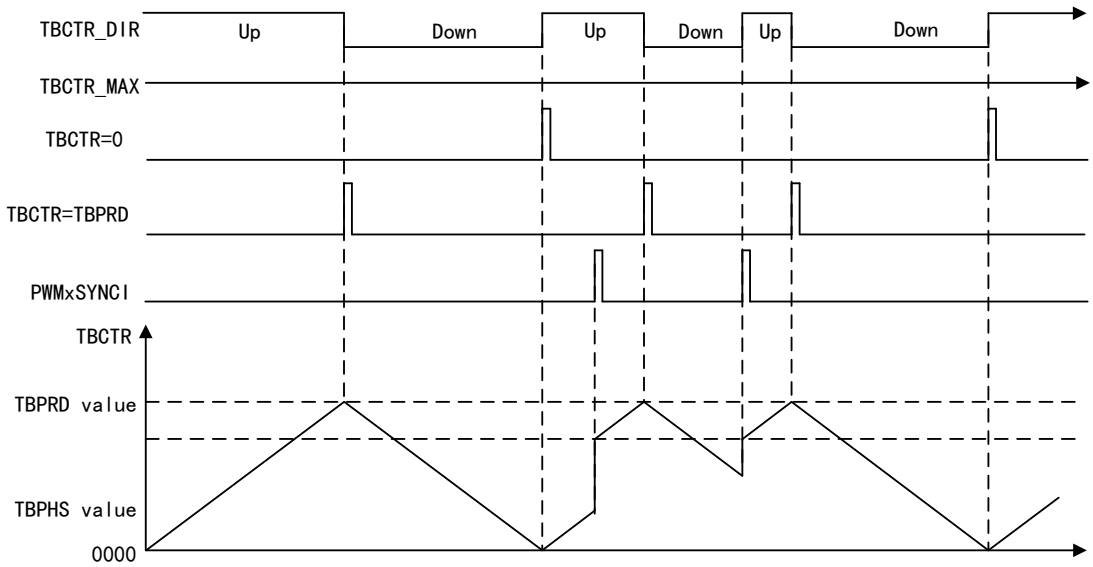
In this mode, the TBCTR counts down after the synchronization event.

Figure 65 Center-aligned Count Mode Timing Diagram (PHSDIR=0)



In this mode, the TBCTR counts up after the synchronization event.

Figure 66 Center-aligned Count Mode Timing Diagram (PHSDIR=1)



Freeze mode

In this mode, TBCTR keeps its current value unchanged.

30.5.3.4 Synchronization

Time base clock synchronization

The TBCLKSYNC bit in the peripheral clock enable register can globally synchronize all enabled PWM modules with TBCLK. When the bit is set, all enabled PWM module clocks will start simultaneously and be aligned with the first rising edge of TBCLK. In order to achieve optimal synchronization of TBCLK, the prescaler of each PWM module must be set to be the same.

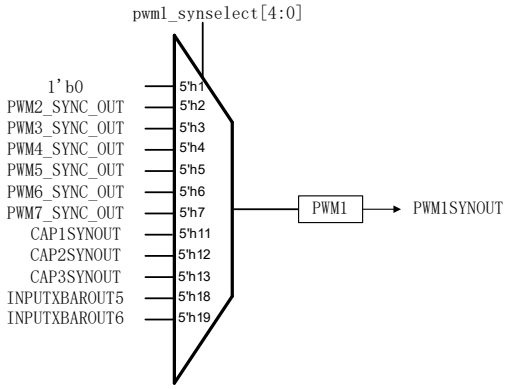
The operation steps to enable PWMCLK are as follows:

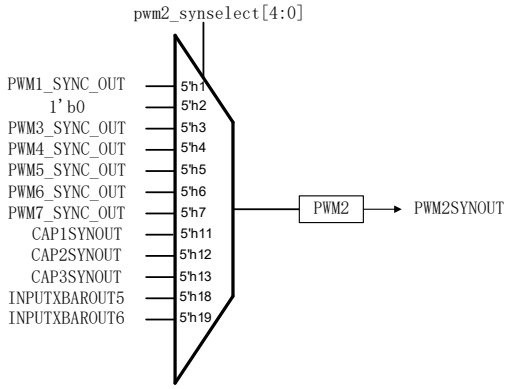
- Enable the PWMx module clocks in the PCLKCRx register
- Clear TBCLKSYNC
- Configure the PWM module
- Set TBCLKSYNC

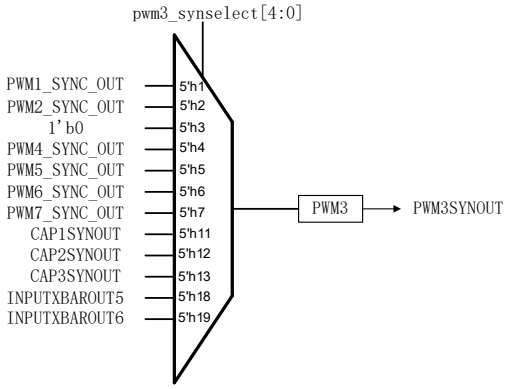
Synchronization of time base counter

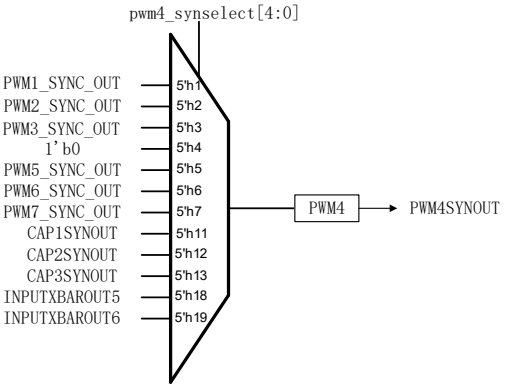
Synchronization scheme adds the flexibility of PWM module synchronization. Each PWM module has a synchronous input, output, and peripheral synchronous output. As shown in the synchronization structure block diagram of the time base counter, EXTSYNCIN1/2 is sourced from INPUTXBAR5/6 and can be select any GPIO as the synchronous input.Ensure that the longest path does not exceed 4 PWM/CAP modules to correctly configure the synchronous chain propagation path using the SYNCSELECT register.

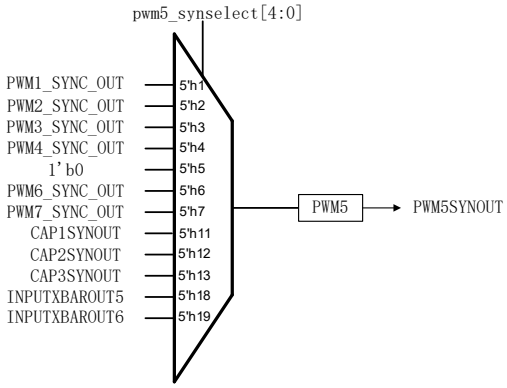
Figure 67 Structure Block Diagram for Synchronization of Time Base Counters

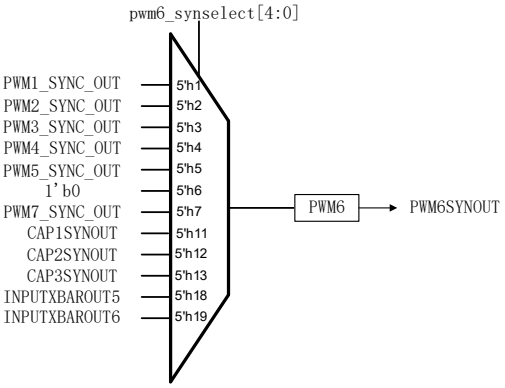


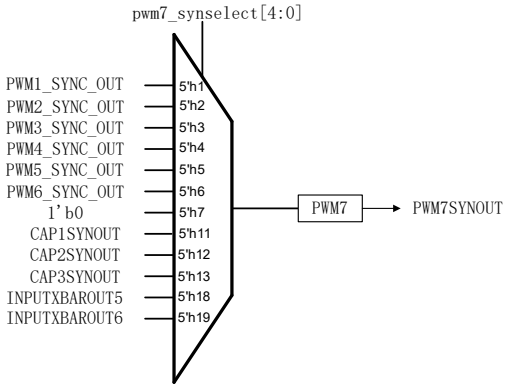












Note: Please refer to Datasheet for the number of PWM and CAP modules available on specific devices.

Each PWM module can be configured to use or ignore synchronous inputs. When the TBCTL[PHSEN] bit is set, if one of the conditions is met, the TBCTR of the PWM module will automatically load the contents of the TBPHS register.

Table 123 Conditions for TBCTR Automatically Loading the TBPHS Register Contents

Condition	Description
Input synchronization pulse	When the input synchronization pulse (PWMxSYNCI) is detected, load the value of the TBPHS register into the TBCTR register. This operation occurs on the next valid TBCLK edge.
Software forced synchronization pulse	Writing 1 to the SWFSYNC bit will call the software forced synchronization. This pulse is OR with the synchronization input signals, so it has the same effect as the input synchronization pulse.
Synchronization pulse for digital compare event	The digital compare events DCAEVT1 and DCBEVT1 can be configured to generate synchronization pulses that have the same effect as the input synchronization pulse.

Note: When the PWMxSYNCI signal remains high, synchronization will not occur continuously. The rising edge of the PWMxSYNCI signal is valid.

This function enables automatically synchronize with the time base between the two PWM modules. By synchronizing different PWM modules, leading or lagging phase control can be added to the waveforms generated by them. In the center-aligned count mode, the direction of TBCTR is configured through the PHSDIR bit after the synchronization event occurs. The new direction is independent of the direction before the synchronization event. In the count-up or count-down mode, the PHSDIR bit will be ignored, as shown in the timing diagrams of the three count modes.

Clear the TBCTL[PHSEN] bit to zero to allows the PWM to ignore synchronization input pulses, but synchronization pulses can still be transmitted to PWMxSYNCO and be used to synchronize other ePWM modules. This allows us to set up a controller time base such as PWM1 to run the other PWM of the downstream module in sync with this controller, For details, please refer to Power Topology Application.

Synchronization of PWM module clocks

Global synchronization can be implemented for the clocks of all enabled PWM modules on the device through the TBCLKSYNC bit. When TBCLKSYNC=0, stop the time base clocks of all PWM modules. When TBCLKSYNC=1, enable all PWM timebase clocks at the aligned TBCLK rising edge. In order to achieve optimal synchronization of TBCLK, the prescaler of each PWM module must be set to be the same.

The operation steps for synchronizing the PWM clocks are as follows:

- Enable the clocks of each PWM module. For details, please refer to System Control and Interrupt
- Clear TBCLKSYNC, and the time base clocks in any enabled PWM module will be stopped
- Configure the desired PWM mode and prescale value
- Set TBCLKSYNC

30.5.3.5 Simultaneous writing to multiple TBPRD and CMPx registers

In frequency conversion applications, TBPRD and CMPx registers need to be written simultaneously between PWM modules. This prevents the TBCTR=0 or TBCTR=TBPRD pulse from forcing the shadow register to load into the active register before all PWM registers are modified, so as to load some registers from the new shadow value and load other registers from the old shadow value. To support this, a link scheme for the registers between PWM modules is added to the CMPx register.

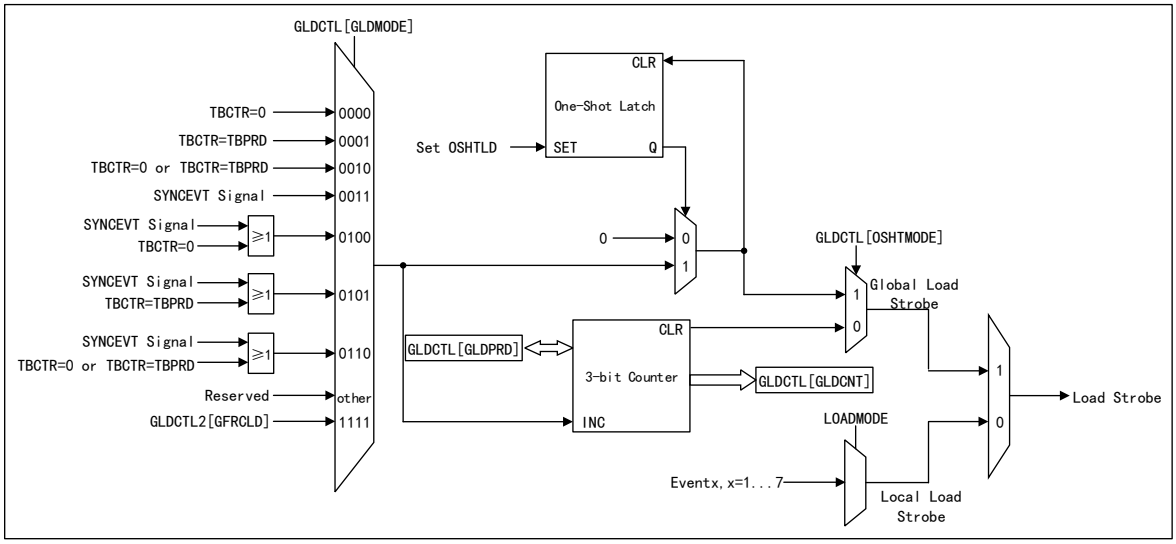
For the specific PWM module #A, the user code writes "B+1" into the PWMXLINK register, which is linked to the corresponding field of the register. "B" is the linked PWM module number, which means that TBPRD, CMPA: CMPAHR, CMPB: CMPBHR ,CMPC CMPD,GLDCTL2 written into the PWM module "B" will be simultaneously written into the corresponding register in PWM module "A". For example, if the PWM3 PWMXLINK register is configured with CMPA: CMPAHR linked to PWM1, then writing to CMPA: CMPAHR in PWM1 will simultaneously write the same value to CMPA: CMPAHR in PWM3. If PWM4 also links its CMPA: CMPAHR register to PWM1, then writing to PWM1 will write the same value to the CMPA: CMPAHR register in PWM3 and PWM4.

30.5.3.6 Global loading

If the global loading function is enabled, for all registers that enable this mode, the transfer of shadow register contents to active registers occurs at the same time, and the time is determined by the GLDCTL[GLDMODE] field. When GLDCTL[GLD]=1, ignore the selection of loading events from each register's shadow register to the active register, and configure the global loading of the corresponding register enabled by the GLDCFG register.

When GLDCTL[GLD]=1 and the corresponding bit in the GLDCFG register is 0, the corresponding register is not affected by the global load mode. The time of loading the shadow register into the active register is determined by the event selection bit of loading from each register's own shadow register into the active register.

Figure 68 Global Loading Structure Block Diagram



Note: The SYNCEVT signal can only be passed when the PHSEN bit is set.

Pulse prescale function

This characteristic offers the following configuration: For every "N" occurrences of the global load event selected by the GLDCTL[GLDMODE] field, a transfer from the shadow register to the active register will occur once. For registers that cannot be or are not configured to use the global loading mechanism, i.e. when GLDCTL[GLD]=0 or the corresponding bit in GLDCFG register is 0, this prescale function is not available.

Single loading mode

Restricts to transfer the shadow register to the active register only once. When OSHTLD=1, all registers configured to use the global loading mechanism will undergo a transfer from the shadow register to the active register when the event selected by the GLDCTL[GLDMODE]. field occurs, at the same time, the GFRCLD bit will be automatically cleared to zero. If the global loading occurs again, this bit needs to be manually set to 1.

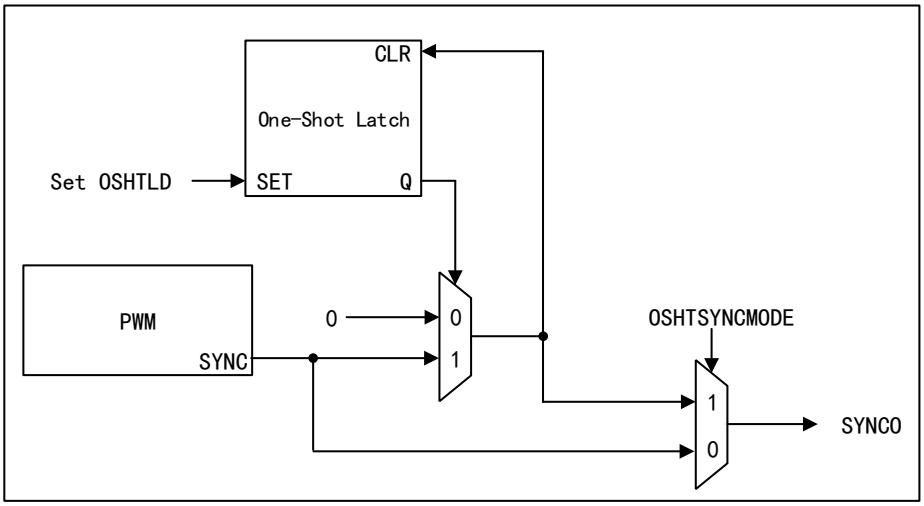
The GFRCLD bit can be used to force the transfer from the shadow register to the active register through software. The GLDCTL2 register can also be linked to multiple PWM modules through the GLDCTL2LINK bit. Based on the above single-loading characteristics, this provides a method for accurately updating multiple PWM registers within one or more PWM modules during the specific PWM event, even within a single clock cycle, and it can be used in frequency conversion applications and/or multi-phase interleaved applications.

Note: Single-loading mode is disabled in high-resolution mode.

Single synchronization mode

The OSHTSYNCMODE and OSP bits need to be configured so that the single synchronization mode can be enabled to generate synchronization output signal (SYNCOUT) pulses, as shown in the following figure.

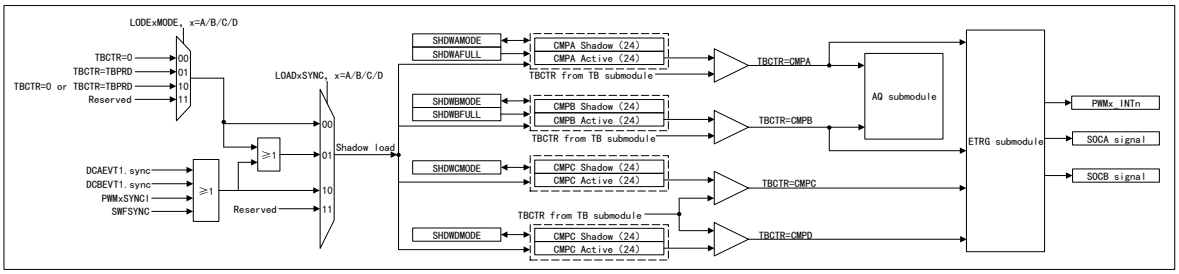
Figure 69 Single Synchronization Mode Structure Block Diagram



30.5.4 Functional description of counter compare submodule

30.5.4.1 Structure block diagram

Figure 70 Counter Compare Submodule Structure Block Diagram

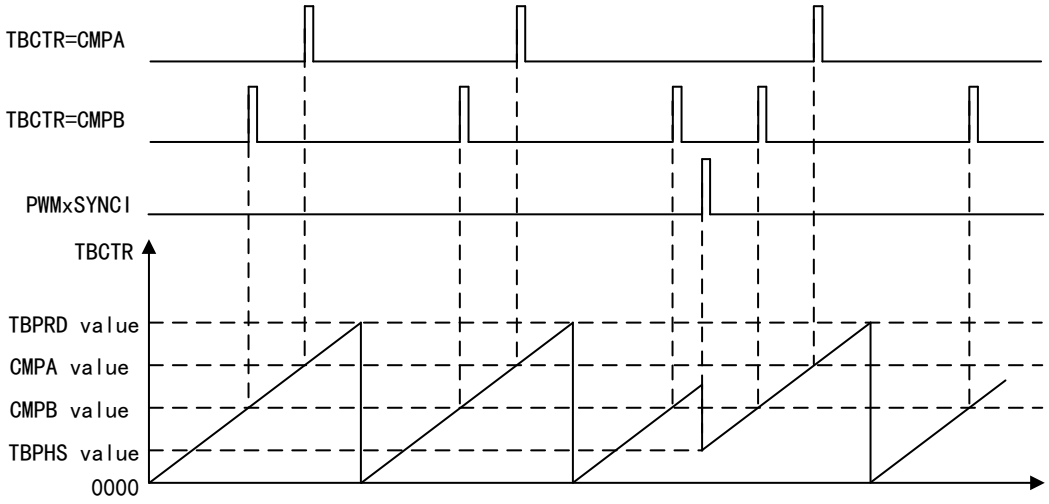


30.5.4.2 Operating mode

The CC submodule can generate comparison events in count up mode, count down mode, and center-aligned mode. The comparison event waveforms of the counters in three counting modes are shown in the following figure, including the time when the event is generated and when the PWMxSYNCI signal works.

Count-up mode

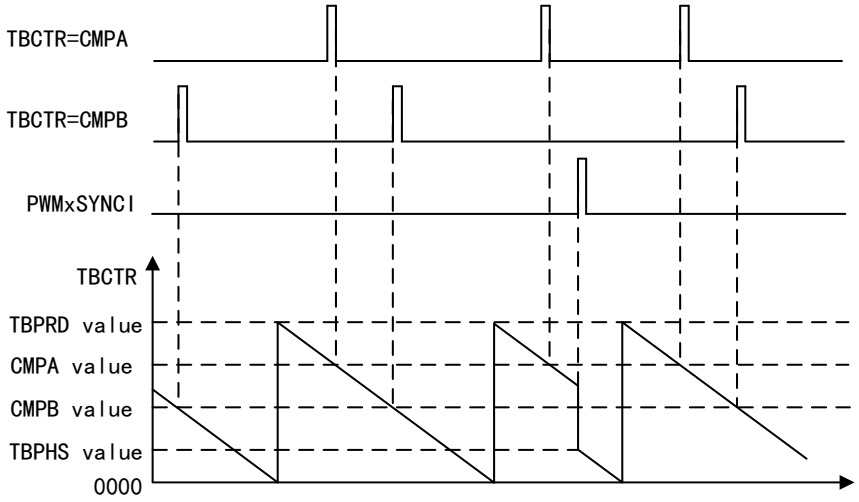
Figure 71 Count-up Mode Timing Diagram



Count-down mode

The external synchronization events of PWMxSYNCI can cause the TBCTR count values to jump. This will cause the comparison event to be skipped, as shown in the figure below, and the synchronization event will cause the TBCTR=CMPA event to be skipped. This skipping phenomenon is considered a normal function, and must be taken into consideration.

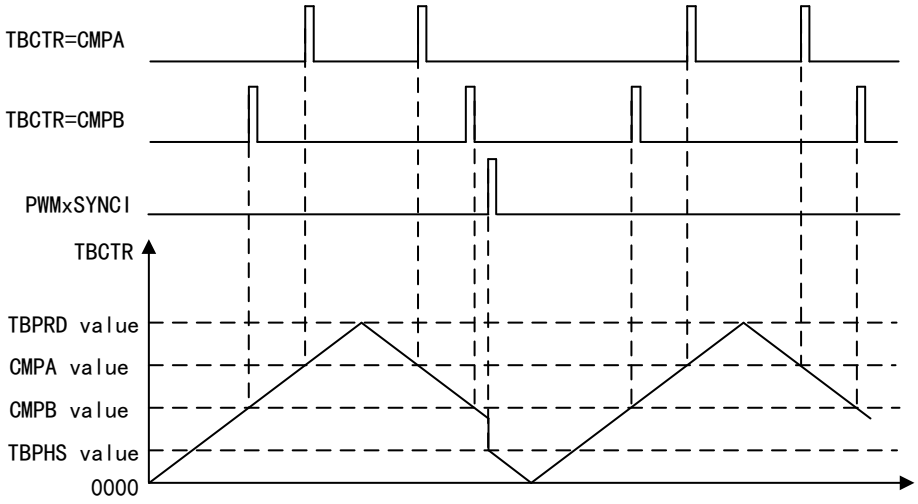
Figure 72 Count-down Mode Timing Diagram



Center-aligned count mode

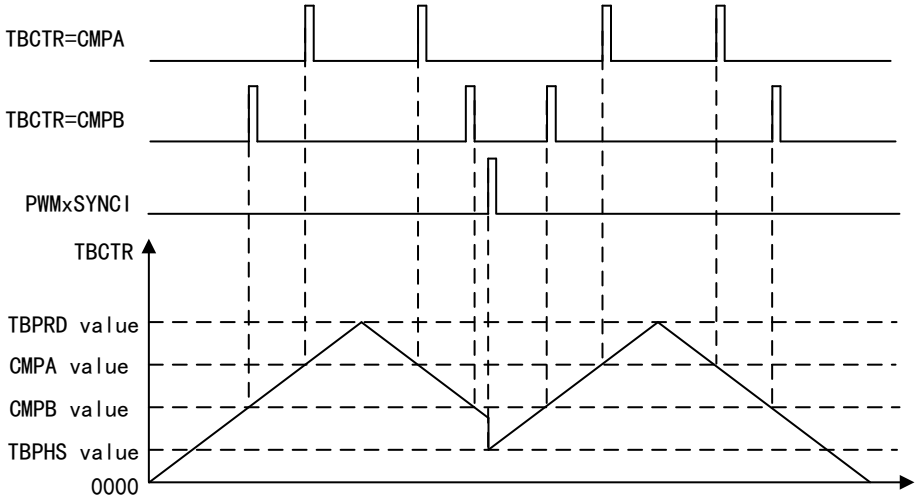
As shown in the figure below, the TBCTR counts down after the synchronization event.

Figure 73 Center-aligned Count Mode Timing Diagram (PHSDIR=0)



As shown in the figure below, the TBCTR counts up after the synchronization event.

Figure 74 Center-aligned Count Mode Timing Diagram (PHSDIR=1)



30.5.4.3 **Operation points**

The CC submodule is responsible for generating events for use by action qualifiers and/or event triggers. There are four independent comparison events:

- TBCTR=CMPA
- TBCTR=CMPB
- TBCTR=CMPC (for event triggers only)
- TBCTR=CMPD (for event triggers only)

For count up or down mode: Each event occurs only once per week

For center-aligned count mode:

- If the comparison value is equal to 0x00 or TBPRD, each event will occur once per period

- If the comparison value is between 0x00 and TBPRD, each event will occur twice per period

These events are sent to the AQ submodule. In this submodule, they are qualified according to the direction of the counter, and converted into corresponding actions when enabled. For details, please refer to Functional Description of the Action Qualifier Submodule.

The CMPA and CMPB registers each have an associated shadow register that allows register updates to be synchronized with the hardware. To prevent waveform distortion or false operations caused by asynchronous modification of registers by software, when the shadow register is used, updates to the active register only occur at the specific time points. The active register and the shadow register have the same memory address. The SHDWAMODE and SHDWBMODE bits determine the registers to write to or read. The behavior description of the shadow mode and immediate load mode is as follows:

(1) Shadow mode

Clear the SHDWAMODE bit and SHDWBMODE bit to enable the shadow mode of CMPA and CMPB registers. By default, the shadow mode of the CMPA and CMPB registers is enabled.

When the shadow register is enabled, the contents of the shadow register will be transferred to the active register in one of the following events, and these events are determined by the LOADAMODE field, LOADBMODE field, LOADASYNC field, and LOADBSYNC field:

- TBCTR=0
- TBCTR=TBPRD
- TBCTR=0 and TBCTR=TBPRD
- Synchronization event: Caused by DCAEVT1 or DCBEVT1 or PWMx synchronization input or software forced synchronization pulse
- The event or synchronization event selected by LOADAMODE/LOADBMODE

This submodule only uses the contents of the active register to generate the events that are sent to the action qualifier.

(2) Immediate mode

When SHDWAMODE=1 or SHDWBMODE=1, and the immediate load mode is selected, the read or write to the register will directly access the active register.

Additional comparator

The CMP submodule can generate two additional independent comparison events based on two compare registers, and provide them to the ETRG submodule:

- TBCTR=CMPC

- TBCTR=CMPD

The CMPC and CMPD registers each have an associated shadow register enabled by default. The active register and the shadow register have the same memory address. If the values of CMPC and CMPD active registers are equal to TBCTR, the CMP submodule will generate a TBCTR equal to the counter compare C event or counter compare D event, respectively. Enable or disable the CMPC and CMPD shadow registers through the SHDWCMODE field and SHDWDMODE field, and the behavior description of the shadow mode and immediate load mode is as follows:

(1) Shadow mode

Clear the SHDWCMODE bit and SHDWDMODE bit to enable the shadow mode of CMPC and CMPD registers. By default, the shadow mode of the CMPC and CMPD registers is enabled.

When the shadow register is enabled, the contents of the shadow register will be transferred to the active register in one of the following events, and these events are determined by the LOADCMODE field, LOADDMODE field, LOADCSYNC field, and LOADDSYNC field:

- TBCTR=0
- TBCTR=TBPRD
- TBCTR=0 and TBCTR=TBPRD
- Synchronization event: Caused by DCAEVT1 or DCBEVT1 or PWMx synchronization input or software forced synchronization pulse
- The event or synchronization event selected by LOADCMODE/LOADDMODE

This submodule only uses the contents of the active register to generate the events that are sent to the action qualifier.

(2) Immediate mode

When SHDWCMODE=1 or SHDWDMODE=1, and the immediate load mode is selected, the read or write to the register will directly access the active register.

Global loading

By configuring the corresponding bits in the GLDCFG register, all counter compare registers can use the global load control mechanism. If the global loading function is enabled, the load of shadow register contents in a register that enables this mode into an active register will occur simultaneously on the event defined by the GLDCTL register.

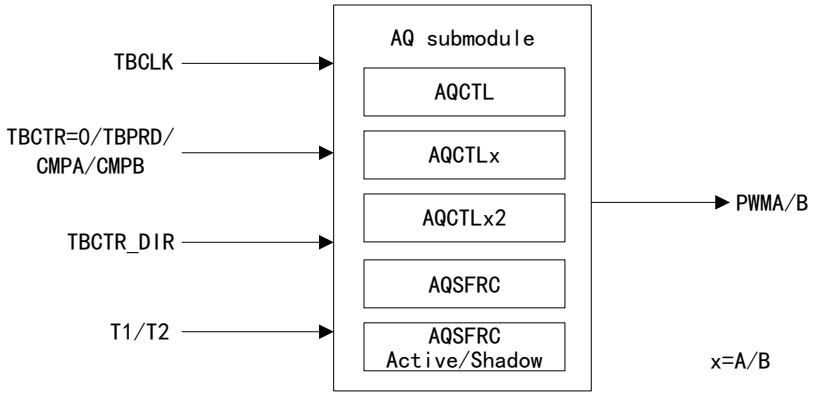
30.5.5 Functional description of action qualifier submodule

The action qualifier submodule is used for waveform construction and PWM generation, and determines which events are converted into various action

types, so as to generate the required switching waveforms on the PWMxA and PWMxB outputs.

30.5.5.1 Structure block diagram

Figure 75 Structure Block Diagram of Action Qualifier Submodule



30.5.5.2 Input and output

Input of action qualifier submodule

The AQ submodule can be used as an input event as shown below:

Table 124 Input Event

Input event	Description
TBCTR=0	The time base counter is equal to 0, i.e. the TBCTR register is equal to 0
TBCTR=TBPRD	The time base counter is equal to the cycle value, i.e. the TBCTR register is equal to the TBPRD register
TBCTR=CMPA	The time base counter is equal to the counter compare A, i.e. the TBCTR register is equal to the CMPA register
TBCTR=CMPB	The time base counter is equal to the counter compare A, i.e. the TBCTR register is equal to the CMPB register
TZ1 event	Based on the comparator, trip, or synchronous input events
TZ2 event	Based on the comparator, trip, or synchronous input events
Software forced event	Asynchronous events initiated by software, controlled by the AQSFRC and AQCSFRC registers

Note: If the CSFA field is not expected to operate in shadow mode, the RLDCSF field must be configured to disable the shadow mode.

Output of action qualifier submodule

The AQ submodule controls the behavior of the two outputs of PWMxA and PWMxB when specific events occur. The event input of this submodule is further qualified by the counter direction (up or down), which can generate independent actions at the count-up and count-down stages of the counter.

Table 125 Actions that May be Applied to PWMxA and PWMxB

Action	Description
Set to high	The output of PWMxA or PWMxB is set to a high level
Set to low	The output of PWMxA or PWMxB is set to a low level
Flip	Flip the state of the output signal of PWMxA or PWMxB. If PWMxA or PWMxB is currently high, pull it down. Otherwise, pull it up
No action	Keep the current level of PWMxA and PWMxB unchanged. Although this option prevents events from generating actions on the output of PWMxA and PWMxB, these events can still trigger interrupts or ADCSOC. For details, please refer to Functional Description of Event Trigger Submodule

The actions specified by the PWMxA or PWMxB output are independent of each other, and all events can generate actions on a given output. For example, both TBCTR=CMPA and TBCTR=CMPB can be applied to the PWMxA output. All qualifier actions can be configured through the associated PWM control register.

The AQTSRCSEL register is used to select T1 and T2 event sources. The T1/T2 selection and configuration of trip or digital compare events in the AQ submodule are independent of the configuration of this event in the TZ submodule. A specific trip event can be configured to cause or not cause a trip action in the TZ submodule, but the action qualifier can use the same event to generate T1/T2 so as to control the generation of PWM.

30.5.5.3 Operating mode

The following figure shows PWM behavior using static comparison register values. In a running system, the CMPA and CMPB active registers are updated from their respective shadow registers at the end of each cycle and can be specified to occur at TBCTR=0 or TBCTR=TBPRD. In certain cases, actions based on old values can take effect for an additional period, or actions based on new values can be delayed for an additional period. The following PWM configuration can avoid this:

Count-up mode

(1) Generate asymmetric PWM

To achieve asymmetric PWM with a duty cycle of 0%-100%, use the following configuration:

- Load CMPA or CMPB when TBCTR=TBPRD
- Use the TBCTR=0 action to pull up the PWM
- Use the comparison action when counting up to pull down the PWM
- By adjusting the comparison value between 0 and TBPRD+1, a PWM duty cycle of 0%-100% can be achieved

(2) Generate asymmetric PWM with dead band

To achieve asymmetric PWM with a duty cycle of 0%-100%, use the following configuration:

- When the CMPA value is too close to 0 or TBPRD, and CMPx is less than the dead band or greater than (TBPRD-dead band), the action specified by the AQCTL register for CMPx will not work
- To avoid this, the AQCTL register settings can only be changed when the counter counts up and is equal to CMPA or when the counter counts down and is equal to CMPA (both are set to high or low) and high pulse or low pulse is generated
- Ensure that the shadow mode is enabled and the software update is synchronized with the PWM carrier cycle

Center-aligned count mode

(1) Generate symmetric PWM

- When it is equal to zero, load CMPA or CMPB, and use the CMPA or CMPB value greater than 1
- When it is equal to the cycle, load CMPA or CMPB, and use the CMPA or CMPB value less than TBPRD-1

There is always at least one TBCLK cycle of pulses in a PWM period, and when it is very short, the system will always ignore it.

(2) Generate asymmetric PWM

To achieve asymmetric PWM with a duty cycle of 0%-50%, use the following configuration:

- Load CMPA or CMPB when TBCTR=TBPRD
- Use the TBCTR=TBPRD action to pull down the PWM
- Use the comparison action when counting up to pull up the PWM
- By adjusting the comparison value between 0 and TBPRD, a PWM duty cycle of 0%-50% can be achieved

(3) Generate asymmetric PWM with dead band

To achieve asymmetric PWM with a duty cycle of 0%-100%, use the following configuration:

- When the CMPA value is too close to 0 or TBPRD, and CMPx is less than the dead band/2 or greater than (TBPRD-dead band/2), the action specified by the AQCTL register for CMPx will not work
- To avoid this, the AQCTL register settings can only be changed when the counter counts up and is equal to CMPA or when the counter counts down and is equal to CMPA (both are set to high or low) and high pulse or low pulse is generated
- Ensure that the shadow mode is enabled and the software update is synchronized with the PWM carrier cycle

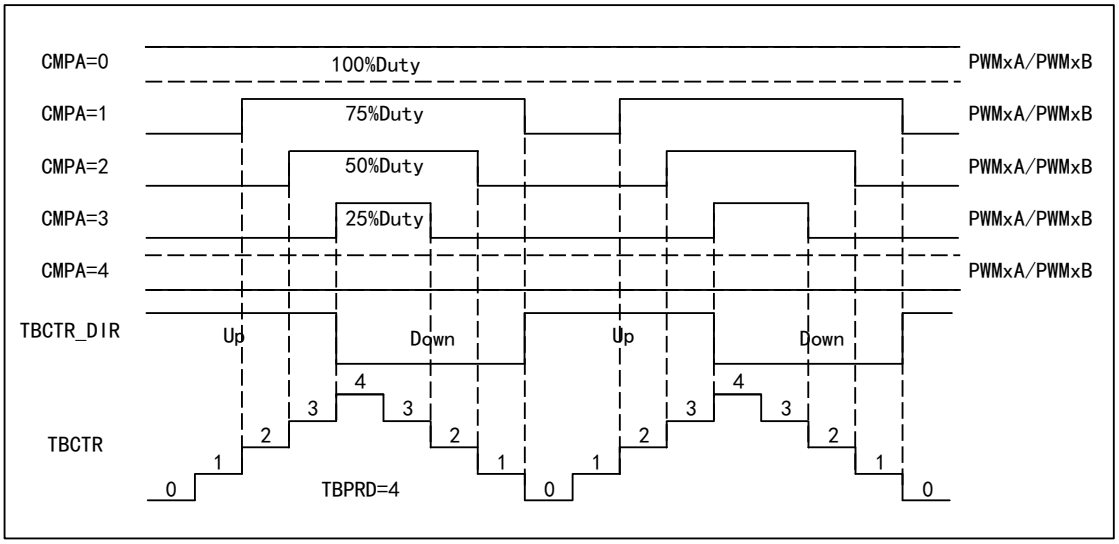
Use the center-aligned count mode of TBCTR to generate a symmetrical PWM

waveform as shown in the following figure. In this mode, 0%-100% DC modulation is achieved by using the equal matching comparison for the count-up and count-down portions of the waveform. An example of using the CMPA for comparison:

- When the counter counts up, the CMPA match will pull up the PWM output
- When the counter counts down, the comparison matching will pull down the PWM signal
- When CMPA=0, the PWM signal remains high throughout the entire cycle, forming a 100%-duty cycle waveform
- When CMPA=TBPRD, the PWM signal is low, reaching a 0% duty cycle

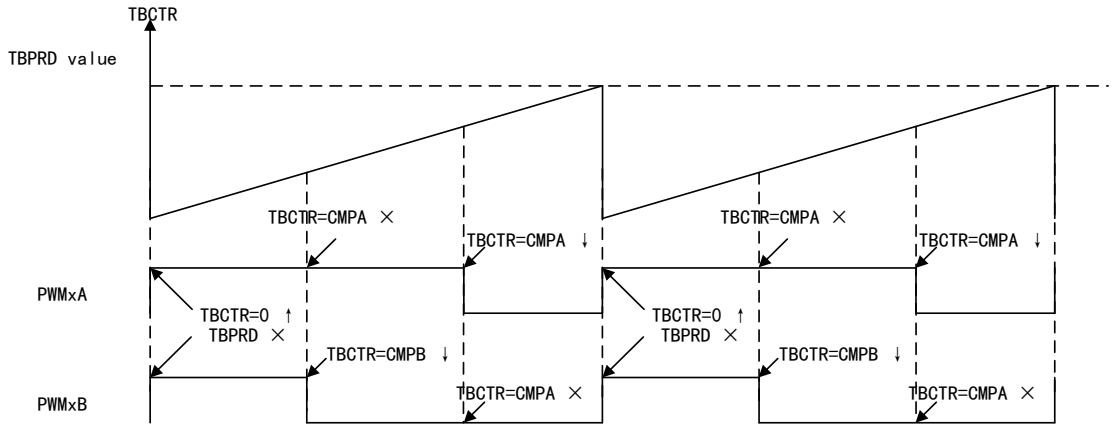
When using this configuration in practice, a CMPA/B value greater than or equal to 1 is required so that CMPA/B is allowed to load when TBCTR=0. A CMPA/B value less than or equal to TBPRD-1 is required so that CMPA/B is allowed to load when TBCTR=TBPRD. Therefore, there is at least one TBCLK cycle of pulses in a PWM period, and when it is very short, the system will always ignore it.

Figure 76 Symmetrical Timing Diagram of Center-aligned Count Mode



The PWM waveform for the common action qualifier configuration is shown in the following figure. TBPRD, CMPA, and CMPB are used in the figure and example to indicate the values written to their respective registers, and the hardware uses the active register; CMPx refers to CMPA or CMPB; PWMxA and PWMxB refer to the agreement of the output signals of PWMx.

Figure 77 Single-edge Asymmetric Waveform Diagram with Independent Modulation on PWMxA and PWMxB in Count-up Mode (High Level is Active)



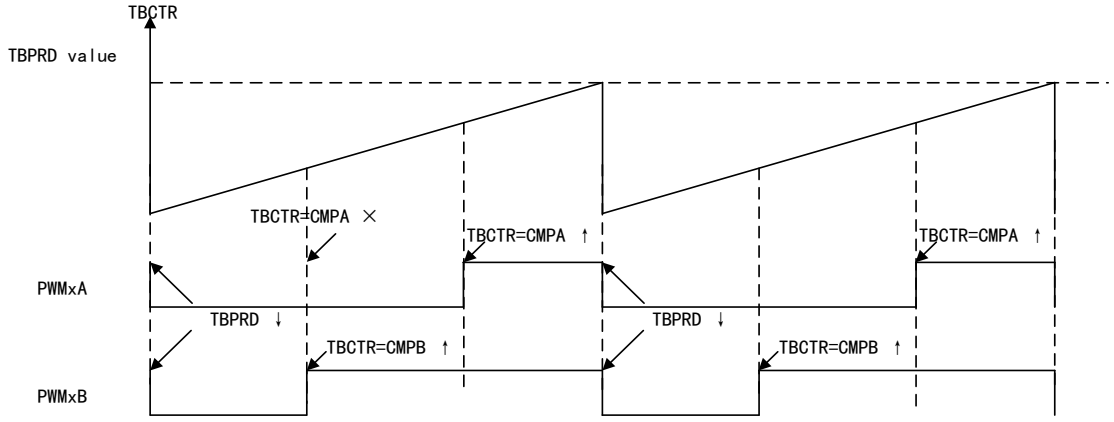
Note:

(1) PWM period = T_{TBCLK}* (TBPRD+1).

(2) Set the duty cycle modulation of PWMxA/B through CMPA/B and the high level is active.

(3) The actions of TBCTR equal to 0 and the cycle do not occur simultaneously, and they are separated by a TBCLK cycle. TBCTR is 0 at the end of the cycle.

Figure 78 Single-edge Asymmetric Waveform Diagram with Independent Modulation on PWMxA and PWMxB in Count-up Mode (Low Level is Active)



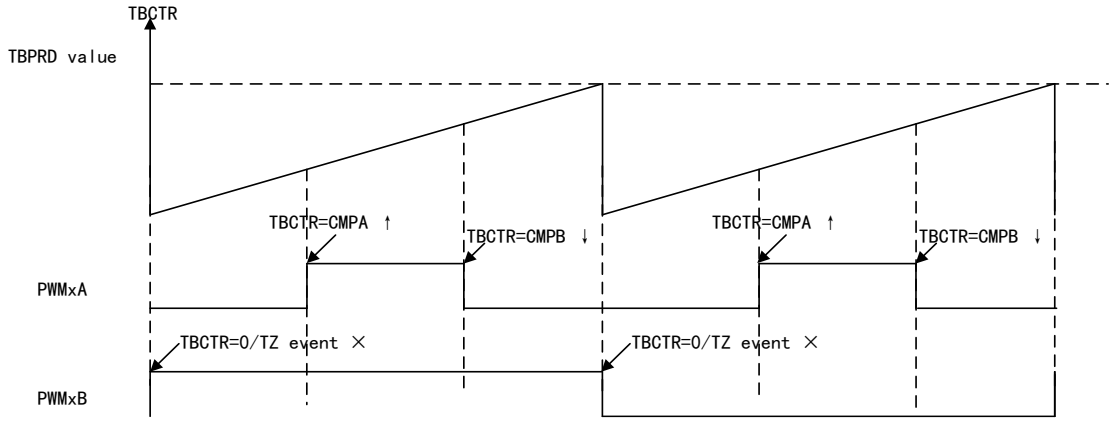
Note:

(1) PWM period = T_{TBCLK}* (TBPRD+1).

(2) Set the duty cycle modulation of PWMxA/B through CMPA/B and the low level is active.

(3) The actions of TBCTR equal to 0 and the cycle do not occur simultaneously, and they are separated by a TBCLK cycle. TBCTR is 0 at the end of the cycle.

Figure 79 Pulse Placement Asymmetric Waveform Diagram with Independent Modulation on PWMxA in Count-up Mode



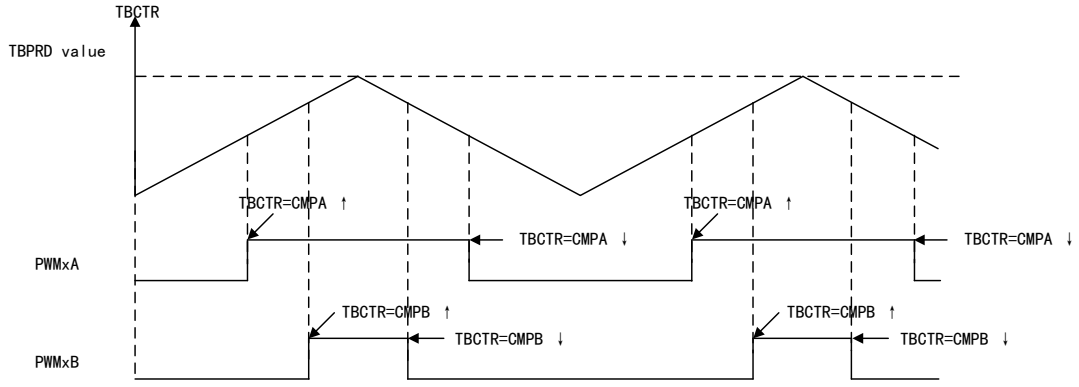
Note:

(1) PWM frequency = 1/ [T_{TBCLK}* (TBPRD+1)].

(2) Pulses can be set at any position within the PWM period (0~TBPRD).

(3) The high-level duty cycle is proportional to (CMPB-CMPA).

Figure 80 Double-edge Symmetric Waveform Diagram with Independent Modulation on PWMxA and PWMxB in Center-aligned Mode (Low Level is Active)



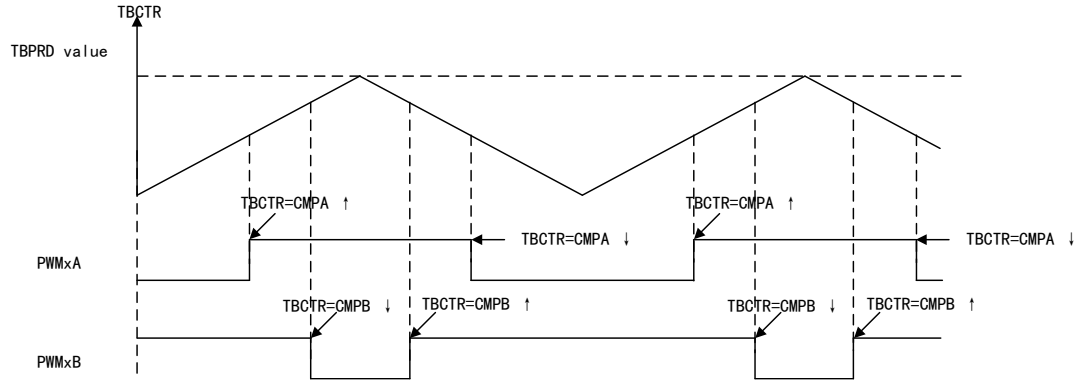
Note:

(1) PWM period = 2* T_{TBCLK} * TBPRD.

(2) Set the duty cycle modulation of PWMxA/B through CMPA/B and the low level is active.

(3) The independent power switch can be driven by the output PWMxA and PWMxB.

Figure 81 Double-edge Symmetric Waveform Diagram with Independent Modulation on PWMxA and PWMxB in Center-aligned Mode (Complementary)



Note:

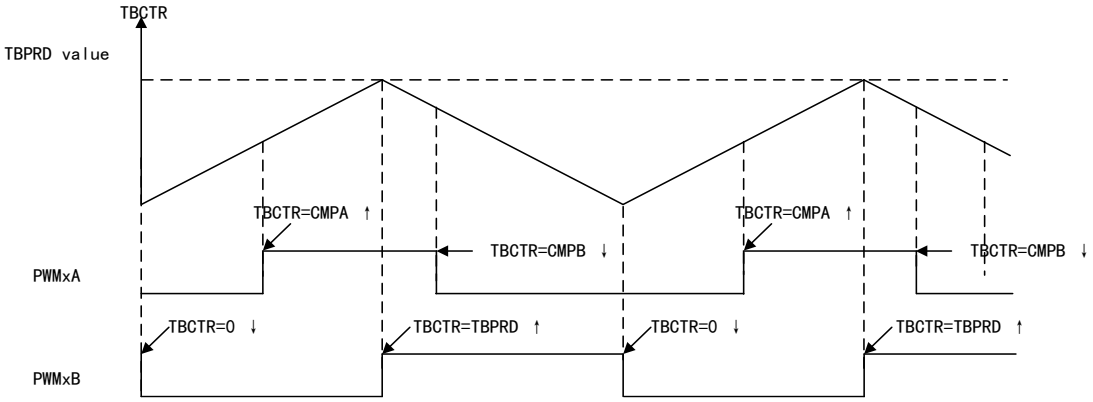
(1) PWM period = 2* T_{TBCLK} * TBPRD.

(2) Set the duty cycle modulation of PWMxA/B through CMPA/B and the low level is active.

(3) The complementary power switch can be driven by the output PWMxA and PWMxB.

(4) The edge placement dead band (dead band=CMPB-CMPA) can be programmed through software, and the typical edge delay method is also applicable to the dead band module.

Figure 82 Double-edge Asymmetric Waveform Diagram with Independent Modulation on PWMxA and PWMxB in Center-aligned Mode (Low Level is Active)



Note:

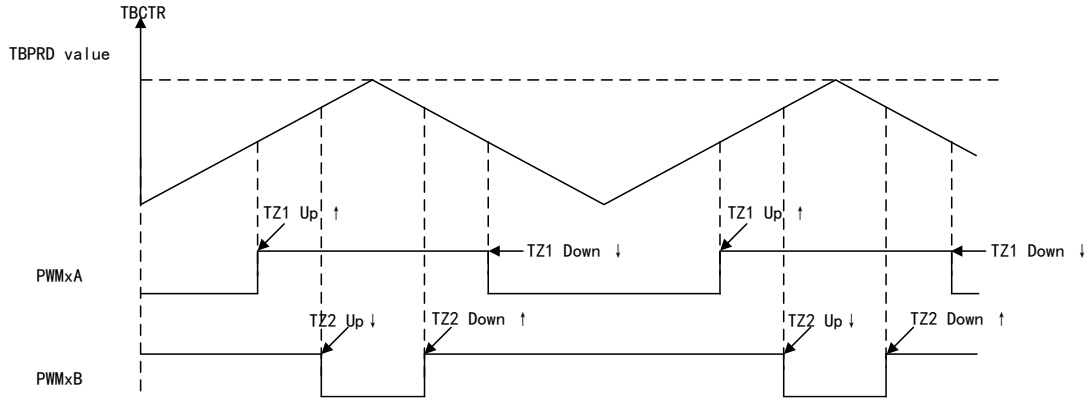
(1) PWM period = 2* T_{TBCLK} * TBPRD.

(2) The rising and falling edges can be asymmetrically located within the PWM period.

(3) The duty cycle modulation of PWMxA is set through CMPA and CMPB, and the duty cycle modulation of PWMxB is fixed at 50%. (CMPA+CMPB) is proportional to the low duty cycle of PWMxA.

(4) If this example is changed to an active high level, it needs to be set low on CMPA and high on CMPB.

Figure 83 Waveform Diagram of PWM Generated Using T1 and T2 Events in Center-aligned Count Mode



Note:

(1) PWM period = 2* T_{TBCLK} * TBPRD.

(2) When TBCTR counts up or counts down, the independent TZ1 and TZ2 event actions are used to generate PWMxA and PWMxB outputs.

30.5.5.4 **Event priority**

The PWM action qualifier can receive multiple events simultaneously. The hardware assigns a priority to these events. Events that occur later have a higher priority, and software-forced events have the highest priority.The priority of action qualification events for various modes is shown in the table below, with priority 1 having the highest priority, and the highest level has the lowest priority.

In the count-up mode, the counter direction is always upward, so it will not receive the count-down event. In the count-down mode, the counter direction is always downward, so it will not receive the count-up event. In the center-aligned count mode, the priority of TBCTR varies slightly with the change of the direction.

Table 126 Priority of Action Qualification Events for Various Modes

Count mode	Priority	Event
Count-up mode	1	Software forced event
	2	TBCTR=TBPRD
	3	TZ1 when counting up
	4	TZ2 when counting up
	5	When counting up, the counter equals CMPB
	6	When counting up, the counter equals CMPA
	7	TBCTR=0
Count-down mode	1	Software forced event
	2	TBCTR=0

Count mode	Priority	Event
	3	TZ1 when counting down
	4	TZ2 when counting down
	5	When counting up, the counter equals CMPB
	6	When counting up, the counter equals CMPA
	7	TBCTR=TBPRD
Center-aligned count mode	1	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: Software forced event
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: Software forced event
	2	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: TZ1 when counting up
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: TZ1 when counting down
	3	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: TZ2 when counting up
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: TZ2 when counting down
	4	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: When counting up, the counter equals CMPB
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: When counting down, the counter equals CMPB
	5	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: When counting up, the counter equals CMPA
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: When counting down, the counter equals CMPA
	6	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: TBCTR=0
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: TBCTR=TBPRD

Count mode	Priority	Event
	7	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: TZ1 when counting down
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: TZ1 when counting up
	8	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: TZ2 when counting down
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: TZ2 when counting up
	9	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: When counting down, the counter equals CMPB
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: When counting up, the counter equals CMPB
	10	For the event when TBCTR counts up, TBCTR=0 increases to TBCTR=TBPRD: When counting down, the counter equals CMPA
		For the event when TBCTR counts down, TBCTR=TBPRD decreases to TBCTR=1: When counting up, the counter equals CMPA

When CMPA/CMPB is greater than the cycle, the events will occur as shown in the following table:

Table 127 Actions when CMPA/CMPB is greater than the cycle

Count mode	Event
Count-up mode	Compare event when counting up: When CMPA/CMPB≤TBPRD, the TBCTR=CMPA or TBCTR=CMPA event occurs when the comparison value matches. When CMPA/CMPB>TBPRD, the event will not occur.
	Compare event when counting down: The event will not occur
Count-down mode	Compare event when counting up: The event will not occur
	Compare event when counting down: When CMPA/CMPB<TBPRD, the TBCTR=CMPA or TBCTR=CMPA event occurs when the comparison value matches. When CMPA/CMPB≥TBPRD, the TBCTR=TBPRD event occurs when the cycle matches.

Count mode	Event
Center-aligned count mode	Compare event when counting up: When CMPA/CMPB<TBPRD and the counter counts up, the TBCTR=CMPA or TBCTR=CMPA event occurs when the comparison value matches. When CMPA/CMPB≥TBPRD, the TBCTR=TBPRD event occurs when the cycle matches.
	Compare event when counting down: When CMPA/CMPB<TBPRD and the counter counts down, the TBCTR=CMPA or TBCTR=CMPA event occurs when the comparison value matches. When CMPA/CMPB≥TBPRD, the TBCTR=TBPRD event occurs when the cycle matches.

30.5.5.5 Shadow mode

The action qualifier supports the shadow mode. In addition, it also supports the loading of the shadow register to the active register when a synchronization event occurs. Enable and disable the shadow mode of this register through the SHDWAQAMODE and SHDWAQBMODE bits. The behavior description of the shadow mode and immediate load mode is as follows:

(1) Shadow mode

Enable the shadow mode of the AQCTLA and AQCTLB registers through the SHDWAQAMODE and SHDWAQBMODE bits. The shadow mode of AQCTLA and AQCTLB registers is disabled by default.

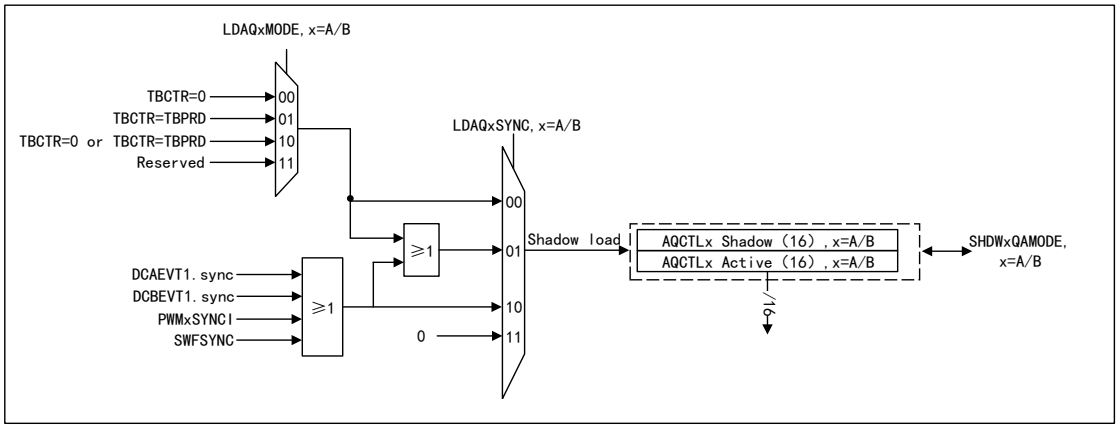
When the shadow register is enabled, the contents of the shadow register will be transferred to the active register in one of the following events, and these events are determined by the LDAQAMODE field, LDAQBMODE field, LDAQASYNC field, and LDAQBSYNC field:

- TBCTR=0
- TBCTR=TBPRD
- TBCTR=0 and TBCTR=TBPRD
- Synchronization event: Caused by DCAEVT1 or DCBEVT1 or PWMx synchronization input or software forced synchronization pulse
- The event or synchronization event selected by LDAQAMODE/LDAQBMODE

(2) Immediate mode

As shown in the figure below, when SHDWAQAMODE=0 or SHDWAQBMODE=0, and the immediate load mode is selected, the read or write to the register will directly access the active register.

Figure 84 SHDWAQAMODE/SHDWAQBMODE Structure Block Diagram



Note:

(1) The DCxEVT1.sync event is generated by the PWM DC submodule based on the TRIPIN input level

(2) When CMPA=0 or CMPB=0 boundary, the AQCTLA and AQCTLB shadow registers are loaded into the active register.

(3) When CMPA or CMPB=0 and the action qualifier action on the AQCTLA and AQCTLB registers occurs at the same time that the shadow register is loaded into the active register, that is to say, when CMPA=0 and AQCTLA loads the shadow register into the active register when TBCTR=0, the two events will conflict. Therefore, it is recommended that the AQCTLA and AQCTLB registers use non-zero comparison value of the counter when the TBCTR=0 boundary enables the shadow mode.

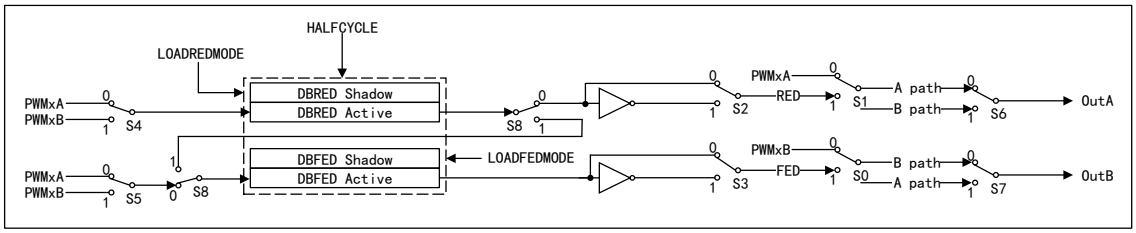
Global loading

The corresponding bits in the GLDCFG register can be configured or the global loading mechanism can be used for the AQCTLA, AQCTLA2, AQCTLB, AQCTLB2 and AQCSFRC registers. If global loading is enabled, the shadow register contents of the relevant register are loaded into the active register at the same time on the event defined by the GLDCTL register. For details of the global loading control mechanism, please refer to Global Loading.

30.5.6 Functional description of dead-band generator submodule

30.5.6.1 Structure block diagram

Figure 85 Structure Block Diagram of Dead-band Generator Submodule



Note: S0 and S1 are controlled by the OUT_MODE bit, S2 and S3 are controlled by the POLSEL bit, S4 and S5 are controlled by the IN_MODE bit, S6 and S7 are controlled by the OUTSWAP bit, and S8 is controlled by the DEDB_MODE bit.

30.5.6.2 Added operating mode

The operating mode added to the dead-band generator submodule is as follows:

- The dead-band counter is 14 bits
- Enable the high-resolution dead band rising edge delay and falling edge delay through the DBREDHR and DBFEDHR registers
- The dead-band high-resolution register and dead-band register have the shadow register

Note:

(1) When the high-resolution dead band is enabled, the PWM chopping cannot be used.

(2) The high-resolution dead band RED and FED functions require enabling a half-cycle clock, i.e. HALFCYCLE=1.

(3) RED and FED cannot simultaneously act on PWMxA and PWMxB. When RED and FED act simultaneously, they can only act on PWM output A (OutA) or PWM output B (OutB).

(4) Make the channel B generate a phase shift relative to the channel A: By using the DEDB_MODE bit, derive PWMxB based on PWMxA, and apply the corresponding phase shift of RED and FED. If the duty cycle of PWMxA is less than the phase shift, the falling edge of PWMxA is ahead of the rising edge of PWMxB. It is suggested that the required phase shift should be less than the current waveform duty cycle of the input dead band module.

Shadow mode

Enable the shadow mode of DBREDHR and DBFED registers through the SHDWDBRED and SHDWDBFED bits. The shadow mode of DBRED and DBFED registers is disabled by default.

In addition, the DBCTL register can be shadowed. Enable the shadow mode of DBCTL register through the SHDWDBCTLMODE bit.

When the shadow register is enabled, the contents of the shadow register will be transferred to the active register in one of the following events, and these events are determined by the LOADREDMODE field, LOADFEDMODE field and LOADDBCTLMODE field:

- TBCTR=0
- TBCTR=TBPRD
- TBCTR=0 or TBCTR=TBPRD

Note: The shadow mode must be enabled before programming the values of DBRED and DBFED registers. If the shadow register is enabled after these registers are programmed, these registers will be loaded to 0.

Global loading

The corresponding bit in the GLDCFG register can be configured, or the global loading mechanism can be used for the DBCTL, DBRED: DBREDHR, and DBFED: DBFEDHR registers. If the global loading function is enabled, for all registers that enable this mode, the shadow register contents will be loaded into the active register and event defined by the GLDCTL register will occur simultaneously.

Note:

(1) When the DBRED or DBFED active register loads a new shadow value while the dead-band counter is counting, the new DBRED or DBFED will not affect the current edge, but only affect the next PWMx edge.

(2) When the global loading time is set to TBCTR=0x00, the dead band value should not be 0. Similarly, when the global loading time is TBCTR=TBPRD, the dead band value should not be a period value.

(3) The TBPRDHR register cannot be used with global loading together. When the high-resolution period must be changed in an application, users must write a separate period register in the PWM interrupt service routine (ISR), the ISR must be synchronized with the PWM switching period, and the single bit of global loading is also written.

30.5.6.3 Operation points

Although the dead-band generator supports all combination modes, there are only seven typical dead band operating modes. These modes assume that the IN_MODE field has configured PWMxA In as the input source of RED and FED. By changing the input source, non-traditional or enhanced modes can be achieved. The typical dead band operating mode can be classified into the following categories:

(1) Simultaneously bypass FED and RED: Mode 1

Allow users to completely disable the TB submodule on the PWM signal path.

(2) Typical dead band polarity setting: Modes 2-5

These typical polarity configurations can solve all the active high-level and low-level modes required for the existing industrial power switch gate drive, and the timing diagram is shown in the dead band timing diagram of the typical mode with 0%<duty cycle<100%. It is important to note that the AQ submodule needs to be configured to generate the signal shown in PWMxA, so as to generate the equivalent waveform shown in the dead band timing diagram of the typical mode with 0%<duty cycle<100%.

(3) Bypass RED: Mode 6

For details about bypassing the RED combination, please refer to Classification of Dead Band Operating Modes.

(4) Bypass FED: Mode 7

For details about bypassing the FED combination, please refer to Classification of Dead Band Operating Modes.

Classification of dead band operating modes

Table 128 Dead Band Configuration

Mode		Mode description	Configuration bit						
			OUT_MODE	POLSEL	OUTSWAP	DEDB_MODE	OUT_MODE	POLSEL	OUTSWAP
			S0	S1	S2	S3	S6	S7	S8
Typical mode	1	PWMxA and PWMxB are directly connected, while bypassing RED and FED, with no delay	0	0	X	X	-	-	-
	2	Active high complement (AHC)	1	1	0	1	-	-	-
	3	Active low complement (ALC)	1	1	1	0	-	-	-
	4	Active high (AH)	1	1	0	0	-	-	-
	5	Active low (AL)	1	1	1	1	-	-	-
	6	PWMxA output=PWMxA input PWMxB output=PWMxA input with FED	1	0	0/1	0/1	-	-	-
	7	PWMxA output=PWMxA input with RED PWMxB output=PWMxB input	0	1	0/1	0/1	-	-	-
Additional modes		The PWMxA and PWMxB signals are defined by the OUT_MODE field.	-	-	-	-	0	0	0
		PWMxA=Path A defined by the OUT_MODE field. PWMxB=Path A with the rising edge delay or bypass delay defined by the OUT_MODE field.	-	-	-	-	0	1	0
		PWMxA=Path B with the falling edge delay or bypass delay defined by the OUT_MODE field.	-	-	-	-	1	0	0

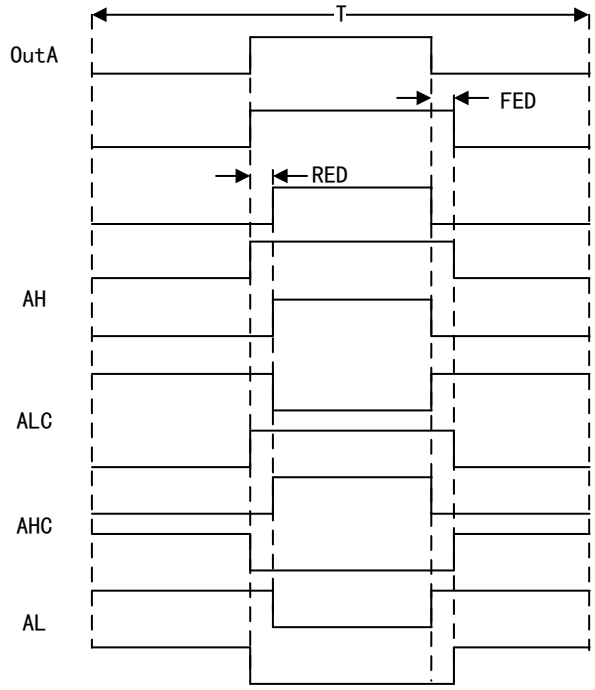
Mode	Mode description	Configuration bit						
		OUT_MODE	POLSEL	OUTSWAP	DEDB_MODE	OUT_MODE	POLSEL	OUTSWAP
		S0	S1	S2	S3	S6	S7	S8
	PWMxB=Path B defined by the OUT_MODE field.							
	PWMxA=Path B with the falling edge delay or bypass delay defined by the OUT_MODE field. PWMxB=Path A with the rising edge delay or bypass delay defined by the OUT_MODE field.	-	-	-	-	1	1	0
	RED only acts on the PWMxA or PWMxB selected by the S4 switch (IN_MODE[4] bit) on Path A. FRD only acts on the PWMxA or PWMxB selected by the S5 switch (IN_MODE[5] bit) on Path B.	-	-	-	-	X	X	0
	RED and FRD act on the signal source selected by the S4 switch (IN_MODE[4] bit) and only output to the signal path B. (1)	-	-	-	-	X	X	1

Note: (1) When this bit is set, the user can set OUT_MODE = 01 (make Path A equal to InA) or set OUTSWAP = 1x (make PWMxA output signal equal to Path B); otherwise, PWMxA has no signal.

Rising edge delay and falling edge delay

The DB submodule supports independent RED and FED values. Configure the delay amount through the DBRED and DBFED registers, which are 10-bit registers, and their value represents the TBCLK period number of the signal edge delay.

Figure 86 Dead Band Timing Diagram of Typical Mode with 0%<Duty Cycle<100%



(1) Turn on the full-cycle clock

The calculation formula for rising edge delay is: RED=DBRED*T_{TBCLK}

The calculation formula for falling edge delay is: FED=DBFED*T_{TBCLK}

(2) Turn on the half-cycle clock

(3) The calculation formula for rising edge delay is: RED=DBRED*(T_{TBCLK}/2)

(4) The calculation formula for falling edge delay is: FED=DBFED*(T_{TBCLK}/2)

Wherein, T_{TBCLK} is the period of TBCLK after it is prescaled by PWMCLK.

The delay values for various TBCLK options are shown in the table below, and the clock frequency of PWM input is 100MHz.

Table 129 Delay Values for Various TBCLK Options

Delay value (DBRED and DBFED)	TBCLK			Unit
	PWMCLK 4 frequency division	PWMCLK 2 frequency division	PWMCLK without frequency division	
1	0.04	0.02	0.01	μs

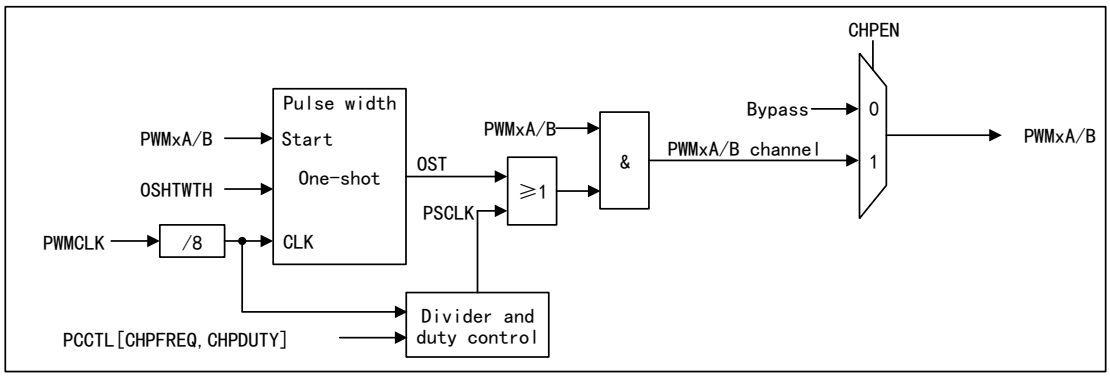
Delay value (DBRED and DBFED)	TBCLK			Unit
	PWMCLK 4 frequency division	PWMCLK 2 frequency division	PWMCLK without frequency division	
5	0.2	0.1	0.05	
10	0.4	0.2	0.1	
100	4	2	1	
200	8	4	2	
400	16	8	4	
500	20	10	5	
600	24	12	6	
700	28	14	7	
800	32	16	8	
900	36	18	9	
1000	40	20	10	

30.5.7 Functional description of PWM chopper submodule

The PWM chopper submodule allows the high-frequency carrier signals to modulate the PWM waveforms generated by AQ and DB. This function can be used when a gate drive based on the pulse transformer is needed to control the power switching elements.

30.5.7.1 Structure block diagram

Figure 87 Structure Block Diagram of Dead-band Generator Submodule



30.5.7.2 Operation points

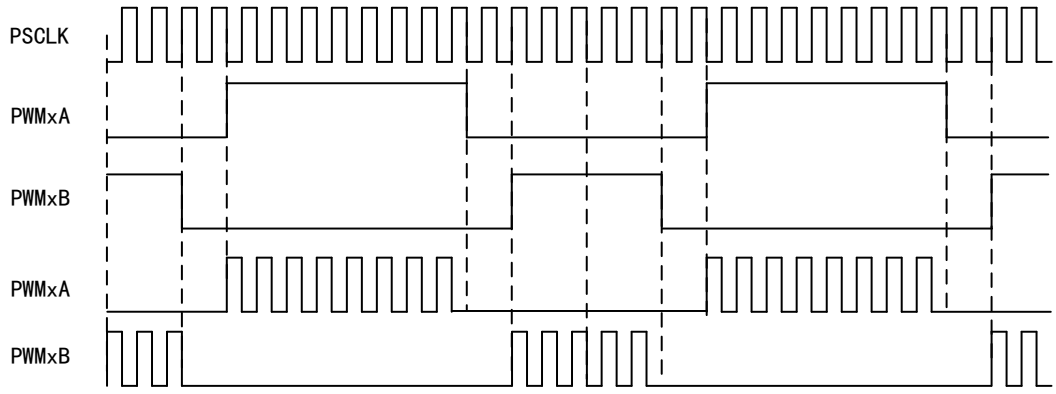
- The carrier clock is from PWMCLK
- Enable the chopper function using the CHPEN bit
- The single pulse width can be programmed through the OSHTWTH field
- The frequency and duty cycle of the chopper clock are controlled by the CHPFREQ and CHPDUTY fields, respectively
- The functions of a single block are as follows:

- Provide a high-energy first pulse to ensure that the power switch is stable and can be turned on quickly
- Subsequent continuous pulses, ensuring that the power switch remains on

30.5.7.3 Waveform

The simplified waveform with only chopping action is shown in the following figure, in which the duty cycle and single pulse control are not displayed.

Figure 88 Simple PC Submodule Timing Diagram (with Only Chopping Action)



Single pulse

The width of the first pulse can be programmed as any pulse width value (16 types in total). T_{PWMCLK} is the cycle of the system clock, the length of OSHTWTH is 4 bits, with the value ranging from 1 to 16, so the calculation formula for the first pulse width or period is: $T_{1st_pulse} = 8*T_{PWMCLK} * OSHTWTH$

Below are the PWM timing diagram of the first pulse and subsequent continuous pulses, and the pulse width value when PWMCLK is 80MHz.

Figure 89 PWM Timing Diagram of the First Pulse and Subsequent Continuous Pulses

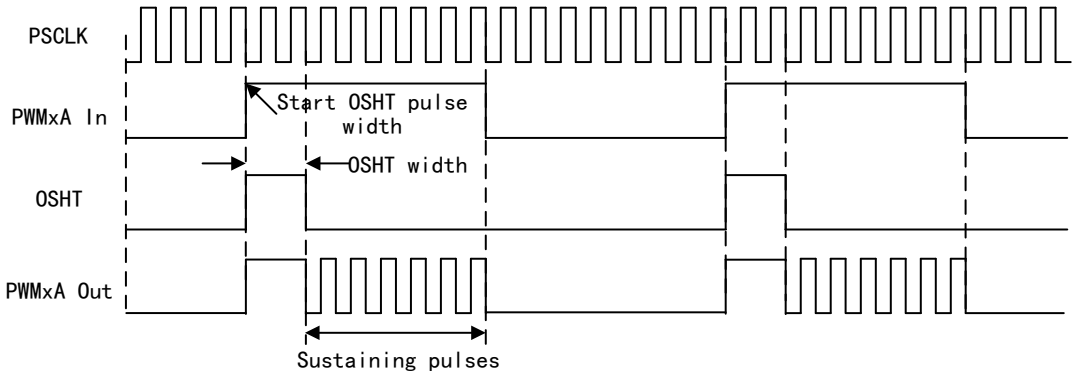


Table 130 Pulse Width Values When PWMCLK is 80MHz

PWMCLK	OSHTWTH	Pulse width value
80MHz	0x00	100 ns
	0x01	200 ns

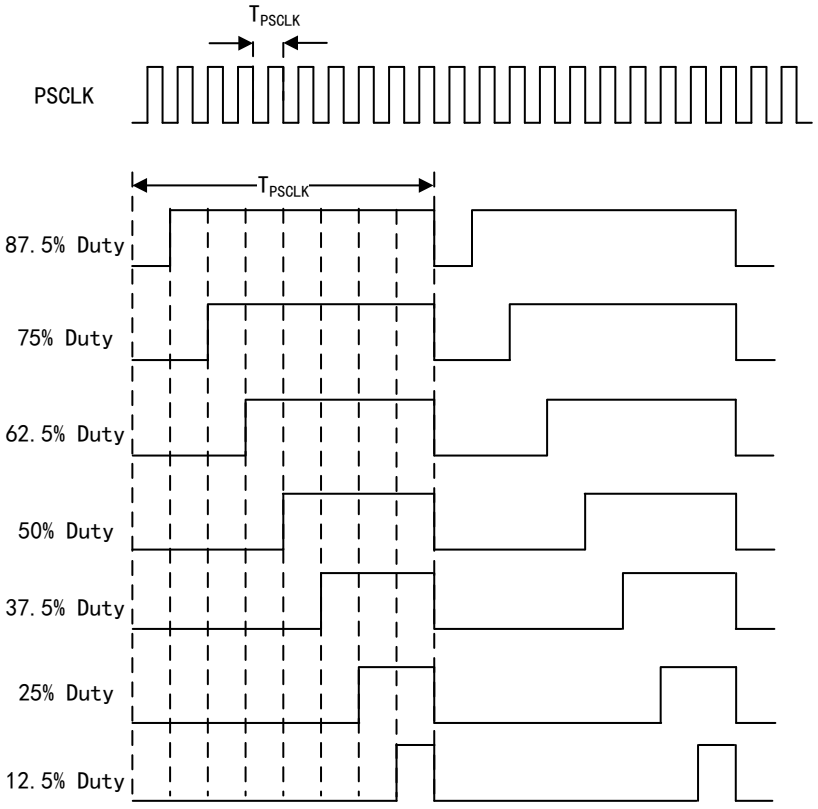
PWMCLK	OSHTWTH	Pulse width value
	0x02	300 ns
	0x03	400 ns
	0x04	500 ns
	...	...
	0x0D	1400 ns
	0x0E	1500 ns
	0x0F	1600 ns

Duty cycle control

When designing pulse transformer-based gate drives, it is necessary to understand the magnetic properties or characteristics of the transformer and associated circuits, including saturation. Programmable duty cycle of the second and subsequent pulses to assist the gate-drive designer. These sustained pulses ensure that power is maintained during opening and the opening and closing pole drive strength and polarity can be adjusted or optimized using software control to design a programmable duty cycle.

Configure any duty cycle control (7 types in total) through the CHPDUTY field, and the duty cycle range is 12.5%~87.5%.

Figure 90 PWM Timing Diagram of Duty Cycle Control



30.5.8 Functional description of trip zone submodule

Each PWM module is connected to six trip zone signals, namely TZ1n to TZ6n:

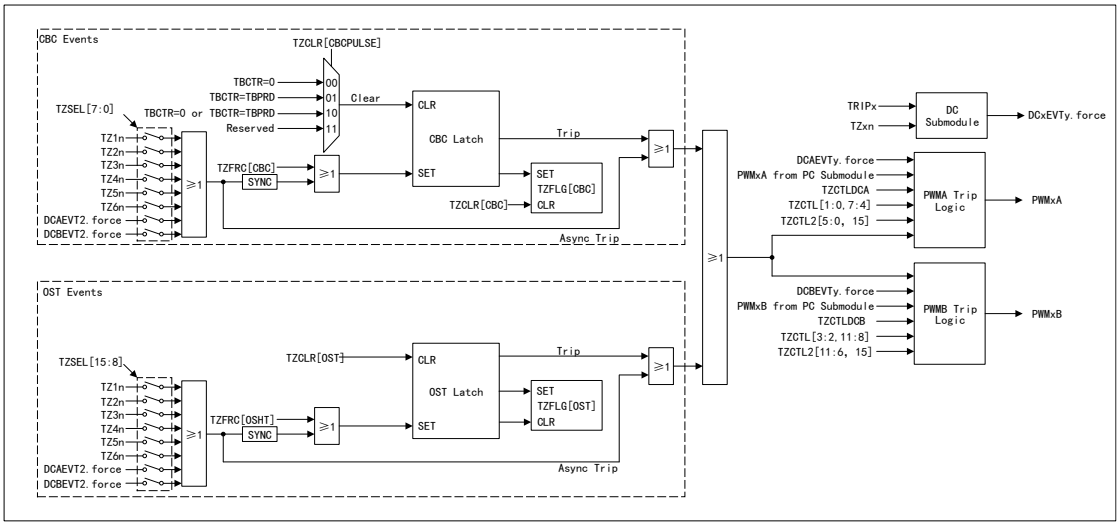
- TZ1n to TZ3n: From the GPIO multiplexers with QEP module devices
- TZ4n: From inverted QEPxERR signals
- TZ5n: Connect to the system clock fault logic (CLOCLKn signal)
- TZ6n: From EMUSTOP output from CPU

These signals indicate tripping or external error states, and the PWM output can make corresponding responses when a fault occurs.

30.5.8.1 Control logic

For details of DCxEVTy signal, please refer to DC Submodule.

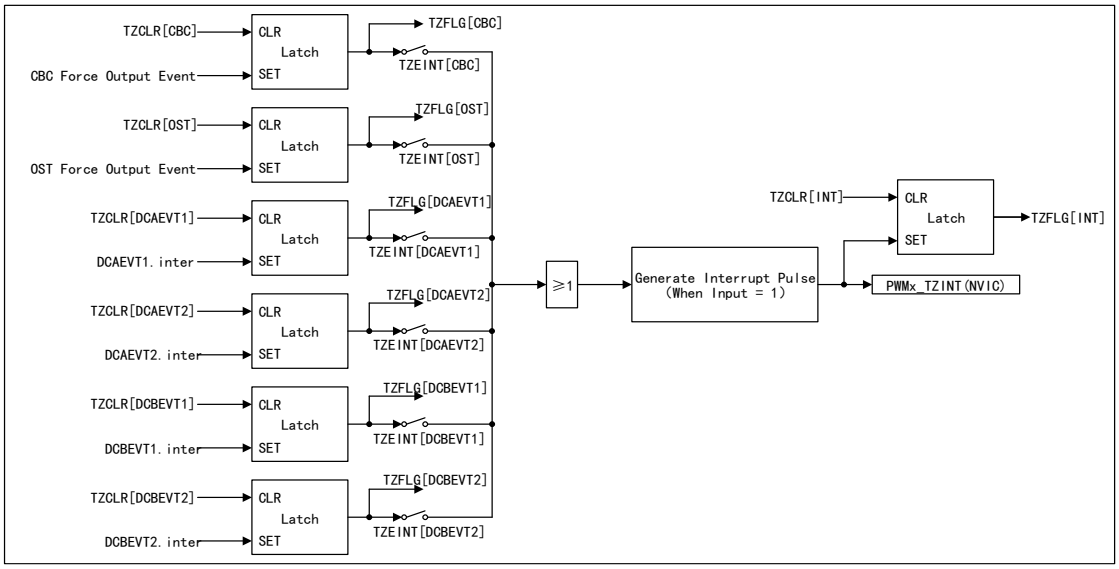
Figure 91 Control Logic Structure Block Diagram



30.5.8.2 Interrupt

For details of DCxEVTy signal, please refer to DC Submodule. Detect the source of the PWMx trip zone interrupt through the CBC, OST, and DCxEVTy flag bits. When multiple interrupt sources are used to generate PWMx trip zone interrupts, these flags can be read and cleared to identify and handle the specific events that cause the interrupts.

Figure 92 Interrupt Logic Structure Block Diagram



30.5.8.3 **Operation points**

(1) The TZ1n to TZ6n signals (TZxn signals) are active low input signals

(2) The following situations indicate that a tripping event has occurred:

- When any signal in the TZxn signal becomes low
- When the DCxEy force event based on TZDCSEL register selection occurs

(3) Each PWM module can be separately configured to use or ignore each trip zone signal or digital comparison event. The trip zone signal or digital comparison event used is determined by the TZSEL register

(4) The TZxn signal can choose whether to synchronize with PWMCLK, and perform digital filtering in the GPIO multiplexer

(5) To trigger the fault condition of the PWM module, the minimum 3*TBCLK low-level pulse width needs to be inputted into TZxn

(6) When the pulse width is less than this value, the OST or CBC latch cannot latch the trip condition

(7) Asynchronous trip: Even if the clock is missing for any reason, it can still ensure that the valid event on the TZxn input trips

(8) GPIOs or peripherals must be configured correctly. Please refer to System Control and Interrupt for details.

(9) The configuration options are as follows, and they are determined by the CBCx, OSHTx, and DCxEVTy bits of the TZSEL register:

- Each TZxn input can be separately configured to provide OST or CBC events for the PWM module
- The DCxE1 event can be configured as:

- Directly trip the PWM module
- Provide OST events for this module

- The DCxE2 event can be configured as:
  - Directly trip the PWM module
  - Provide CBC events for this module

One-shot trip (OST) Event

When an OST event occurs, immediately perform the operations specified in the TZA and TZB fields on the PWMxA and PWMxB outputs. For details, please refer to the table of actions taken when a trip event occurs. When the counter counts up or down, the independent actions taken when a trip event occurs are specified by configuring the corresponding field of the TZCTL2 register. These actions only take effect when ETZE=1.

If an OST event occurs, the OST bit will be set, and when the interrupt is enabled in the TZEINT register and NVIC peripheral, a PWMx trip zone interrupt will also be generated. In the TZCBCFLG register, the corresponding flag of the event that causes the OST event will also be set. The corresponding bit in the TZOSTCLR register can be manually written to clear the TZOSTFLG register bit to zero (if needed).

When the OST interrupt is enabled (OST=1) and the one-shot trip source is selected as DCxEVT1 through the TZSEL register, there is no need to enable the DCxEVT1 interrupt (DCAEVT1/DCBEVT1=1), which is because the digital comparison event triggers the interrupt through the OST mechanism.

Note: After confirming that the OST trip source has become inactive, the TZFLG and TZOSTFLG register flags should be cleared to zero. Otherwise, when the interrupt is enabled, an OST interrupt may occur and the OST flag is 0, depending on when the flag is cleared.

The actions taken when a trip event occurs can be independently configured for each PWM output pin through the TZCTL, TZCTL2, TZCTLDCA, and TZCTLDCB registers. The actions that can be taken are as follows:

Table 131 Actions Taken when a Trip Event Occurs

TZCTL register field settings	Actions of PWMxA and/or PWMxB	Status
00	High-impedance state	Tripped
01	Force the output to a high level	Tripped
10	Force the output to a low level	Tripped
11	The output remains in the original state	No operation

Cycle-by-cycle trip (CBC) event

When a CBC event occurs, immediately perform the operations specified in the TZA and TZB fields on the PWMxA and PWMxB outputs. For details, please

refer to the table of actions taken when a trip event occurs. When the counter counts up or down, the independent actions taken when a trip event occurs are specified by configuring the corresponding field of the TZCTL2 register. These actions only take effect when ETZE=1.

If a CBC event occurs, the TZFLG[CBC] bit will be set, and when the interrupt is enabled in the TZEINT register and NVIC peripheral, a PWMx trip zone interrupt will also be generated. In the TZCBCFLG register, the corresponding flag of the event that causes the CBC event will also be set.

When the CBC interrupt is enabled (TZEINT[CBC]=1) and the CBC trip source is selected as DCxEVT2 through the TZSEL register, there is no need to enable the DCxEVT2 interrupt (DCAEVT2/DCBEVT2=1), which is because the digital comparison event triggers the interrupt through the CBC mechanism.

When there is no trip event, the status of the CBC trip latch will be automatically cleared based on the selection of the TZCLR register. Therefore, in this mode, each PWM period will be reset or the trip event will be cleared. The TZCBCFLG register flag bit and TZCLR[CBC] bit will remain set before they are manually written to the TZCBCCLR register flag bit and the TZCLR[CBC] bit is cleared to zero. If the CBC trip event still exists after the TZCBCFLG register flag bit and TZCLR[CBC] bit are cleared, these flag bits will be set again immediately.

Digital comparison event (DCxEVTy)

The combination of DCxH and DCxL signals based on the TZDCSEL register selection can generate a digital comparison event DCxEVTy. Select the source signal of DCxH and DCxL signals as the trip zone input pin or COMPx signal through the DCTRIPSEL register. For details of the DC submodule signal, please refer to Functional Description of Digital Compare Submodule.

When DCxEVTy occurs, immediately perform the operations specified in the DCxEVTy field on the PWMxA and PWMxB outputs. For details, please refer to the table of actions taken when a trip event occurs. When the counter counts up or down, the independent actions taken when a trip event occurs are specified by configuring the corresponding fields of the TZCTLDCA and TZCTLDCB registers. These actions only take effect when ETZE=1.

If a digital comparison trip event occurs, the DCxEVTy bit will be set, and when the interrupt is enabled in the TZEINT register and NVIC peripheral, a PWMx trip zone interrupt will also be generated.

When there is no digital comparison trip event, the specified state on the pin will be automatically cleared. The TZFLG[DCAEVTy] and TZFLG[DCBEVTy] bits will remain set until they are manually written to the DCAEVTy or the DCBEVTy bit is cleared to zero. If the CBC trip event still exists after the TZFLG[DCAEVTy] and TZFLG[DCBEVTy] bits are cleared, these flag bits will be set again immediately.

Trip zone configuration

Table 132 Trip Zone Configuration Options

Condition	PWM1 register configuration	PWM2 register configuration
The OST event on TZ1n pulls down the outputs of PWM1A and PWM1B, and force the outputs of PWM2A and PWM2B to be high	(1) OSHT1=1: Select TZ1n as the one-shot trip source of PWM1 (2) TZA=10: When a trip event occurs, force the PWM1A output to a low level (3) TZB=10: When a trip event occurs, force the PWM1B output to a low level	(1) OSHT1=1: Select TZ1n as the one-shot trip source of PWM2 (2) TZA=01: When a trip event occurs, force the PWM2A output to a high level (3) TZB=01: When a trip event occurs, force the PWM2B output to a high level
(1) The CBC event on TZ5n pulls down the outputs of PWM1A and PWM1B (2) The OST event on TZ1n or TZ6n sets PWM2A to a high-impedance state	(1) CBC5 =1: Select TZ5n as the CBC trip source of PWM1 (2) TZA=10: When a trip event occurs, force the PWM1A output to a low level (3) TZB=10: When a trip event occurs, force the PWM1B output to a low level	(1) OSHT1=1: Select TZ1n as the one-shot trip source of PWM2 (2) OSHT6=1: Select TZ6n as the one-shot trip source of PWM2 (3) TZA=00: When a trip event occurs, PWM2A is set to a high-impedance state (4) TZB=11: When a trip event occurs, PWM2B remains in the original state, and the trip event will be ignored

When configuring the PWM X-BAR/Input X-BAR and GPIO options, it is important to note that changing the X-BAR input selection may cause unexpected events to occur. It is suggested that the input X-BAR and GPIO configuration should be completed before enabling the PWM trip zone. When it is necessary to change the input X-BAR/GPIO configuration after the PWM trip zone is enabled, users can turn off the trip (by clearing the TZSEL register to zero), and reconfigure the TZSEL register after the input X-BAR selection is changed.

30.5.9 Functional description of event trigger submodule

The event trigger submodule is responsible for managing the events generated by the time base submodule, counter comparison submodule, and digital compare submodule. When the selected event occurs, it will generate an interrupt signal to the CPU and transmit a pulse of start of conversion to the ADC.

30.5.9.1 Structure block diagram

As shown in the figure below, the ETRG submodule can monitor various event

conditions, and according to the prescale settings, determine to issue interrupt requests and ADCSOC at an interval of one event, two events until fifteen events.

Figure 93 Structure Block Diagram of Event Trigger Submodule

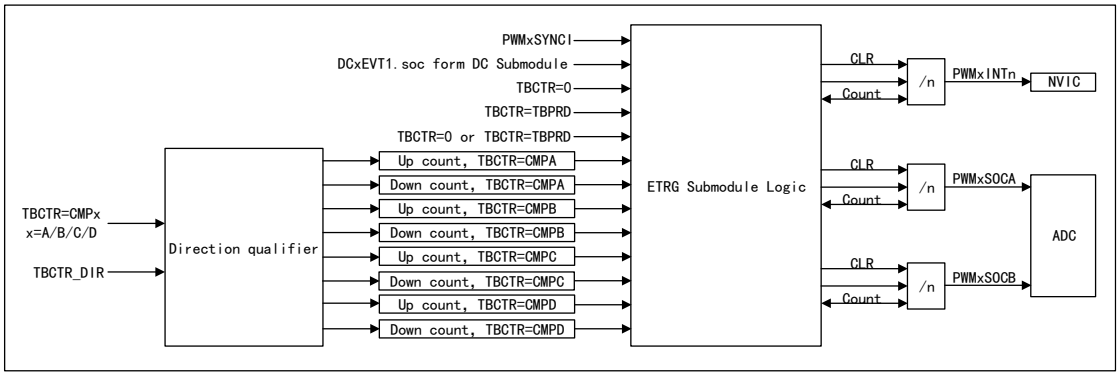


Table 133 Supplementary Instructions for ETRG Submodule

Register	Description
ETSEL	Select the event that triggers the interrupt or the start of conversion of ADC
ETPS	Configure the event frequency division options mentioned above
ETFLG	This register represents the state of the selected event and the prescale event
ETCLR	This register is used to clear the flag bits in the ETFLG register to zero through software
ETFRC	These bits are used for software to forcibly generate events, for debugging or software intervention
ETINTPS	Configure the interrupt event prescale option, and support counting of a cycle of up to 15 events
ETSOCPS	Configure SOC event prescale option, and support counting of a cycle of up to 15 events
ETCNTINITCTL	This register enables the use of a synchronization event or software forcing to enable initialization of the ETCNTINIT register
ETCNTINIT	This register allows users to initialize the interrupt, SOCA, or SOCB counters using the programming values during synchronization events or software forcing

30.5.9.2 **Event-triggered pulse generator**

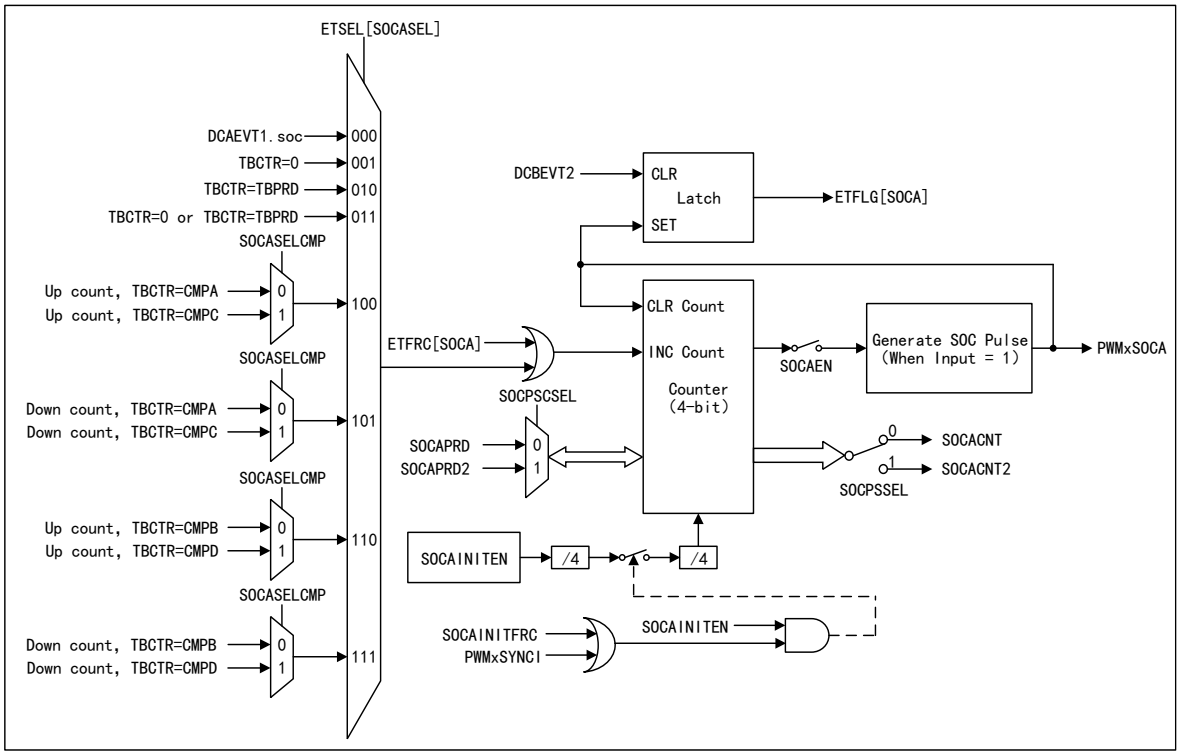
SOCA pulse trigger

The operation of an event triggering the SOCA pulse trigger is shown in the figure, and its operating mode is as follows:

- Enhanced functions: Including the SOCASELCMP and SOCBSELCMP bits to enable CMPC and CMPD events respectively, so as to initiate the start of conversion
- The SOCPSSEL bit can determine whether the SOCACNT2 bit and SOCAPRD2 bit take over the right of control

- The behavior of the SOCA event counter and SOCA cycle is similar to that of an interrupt generator, but it can continuously generate pulses. That is to say, when a pulse is generated, the ETFLG[SOCA] bit is latched, but the interrupt generator will not stop the generation of subsequent pulses
- The SOCAEN bit enables/disables the generation of pulses, and the input events are still counted unless the cycle value is reached
- Eventss that trigger SOCA and SOCB pulses can be separately configured through the SOCASEL and SOCBSEL fields, and the following events can be used:
  - The same as the events specified by the interrupt generation logic
  - DCAEVT1.soc and DCBEVT1.soc event signals from the DC submodule
- The initialization scheme of SOCA event counter 2 is similar to that of an interrupt generator, and it includes corresponding enable, value initialization, and synchronization event or software forcing options

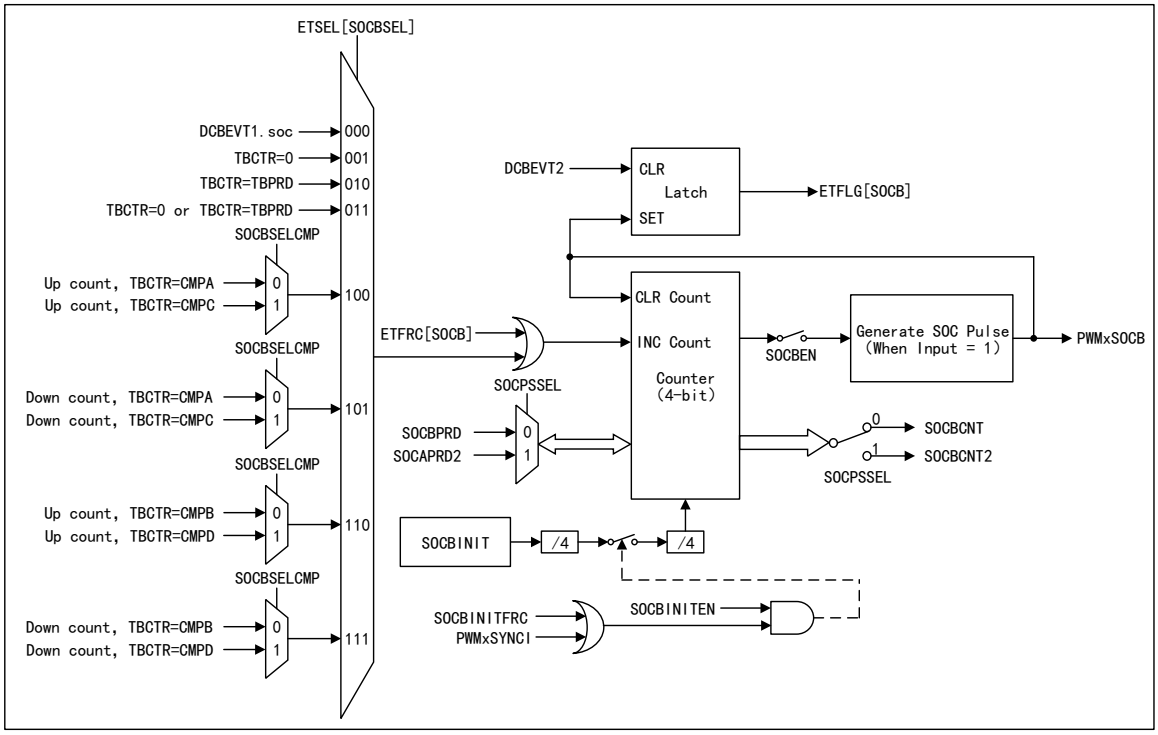
Figure 94 Structure Block Diagram of SOCA Pulse Trigger



SOCB pulse trigger

The operation of the event triggering the SOCB pulse trigger is shown in the figure. For details about its operating mode, please refer to SOCA Pulse Trigger.

Figure 95 Structure Block Diagram of SOCB Pulse Trigger



30.5.9.3 **Interrupt**

The interrupt generation logic triggered by events is as shown in the following figure. The number of events required to generate interrupt pulses through the INTPRD field selection is: no interrupt is generated, and an interrupt is generated at an interval of one event/two events/three events.

The INTPSSEL bit determines whether to use the INTCNT2 and INTPRD2 bits of the ETINTPS register to determine the event frequency of generating an interrupt to every 0... 15 events. The events that can cause interrupts through the INTSEL field and INTSELCMP bit configuration are shown in the table below.

Figure 96 Interrupt Generator Structure Block Diagram

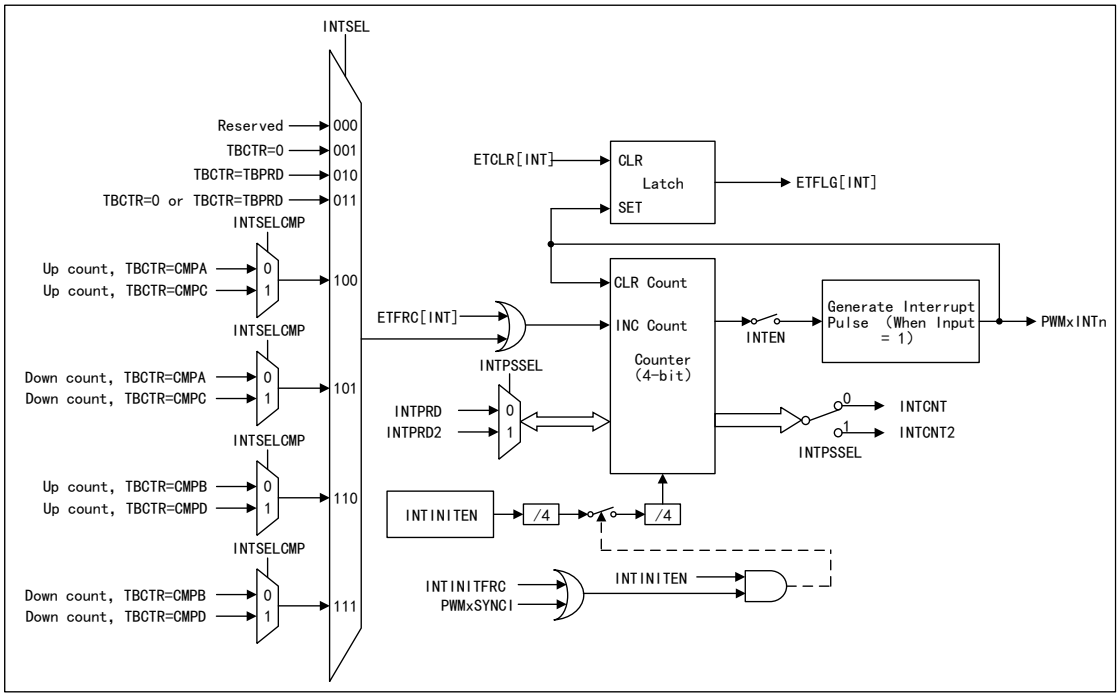


Table 134 Supplementary Instructions for Interrupt Generator

Event	Description
TBCTR=0	The time base counter is equal to 0
TBCTR=TBPRD	The time base counter is equal to the time base period
TBCTR=0 or TBCTR=TBPRD	0 or equal to the time base period
TBCTRU=CMPA	When the counter counts up, the time base counter equals CMPA
TBCTRD=CMPA	When the counter counts down, the time base counter equals CMPA
TBCTRU=CMPB	When the counter counts up, the time base counter equals CMPB
TBCTRD=CMPB	When the counter counts down, the time base counter equals CMPB
TBCTRU=CMPC	When the counter counts up, the time base counter equals CMPC
TBCTRD=CMPC	When the counter counts down, the time base counter equals CMPC
TBCTRU=CMPD	When the counter counts up, the time base counter equals CMPD
TBCTRD=CMPD	When the counter counts down, the time base counter equals CMPD

Based on the selection made by the INTPSSEL bit, the number of events that have occurred can be read through the INTCNT field or the INTCNT2 field. When a specified event occurs, based on the selection made by the INTPSSEL bit, the INTCNT field or INTCNT2 field will continuously increase until it reaches the value specified by INTPRD or INTPRD2. If INTCNT=INTPRD, the counter stops counting and generates the corresponding output (interrupt). The counter will be cleared to zero only when an interrupt is sent to NVIC.

When INTCNT reaches INTPRD or INTCNT2 reaches INTPRD2, the behavior

of the counter is shown in the table.

Table 135 Behaviors of the Counter

Condition	Behavior
INTEN=0 or INT=1	When the counter reaches the period value (INTCNT=INTPRD), the counter stops counting the events.
INTEN=1 and INT=0	Generate an interrupt pulse, INT=1, and the INTCNT field is cleared to zero. The event counter starts counting events again.
INTEN=1 and INT=1	The counter will maintain a high output level until the INT bit is cleared. This allows the next interrupt to be in a wait state while one interrupt is processed.

The following methods are applicable to INTCNT/INTCNT2 and INTPRD/INTPRD2:

- Writing 0 to the INTPRD field will automatically clear the counter to zero and reset the counter output, so no interrupt will be generated
- All operations of writing to the INTPRD field will cause INTCNT to maintain its original value
- INTCNT resets upon overflow
- ETFRC[INT]=1 will increase the event counter
- When the counter reaches the period value, the behavior of the counter is the same as the description in the table
- If INTPRD=00, the counter is disabled and the ETFRC[INT] bit is ignored, so no event will be detected

When INTCNT and INTPRD are used, an interrupt can be generated at an interval of one event/two events/three events. When INTCNT2 and INTPRD2 are used, an interrupt can be generated at an interval of one event to 15 events (at most).

INTCNT2 can be initialized based on the selection of the INTINITEN bit through the value in the INTINIT field. When INTINITEN is set, INTCNT2 will be initialized using the contents of the INTINIT field in case of a synchronization event or a software forcing event determined by the INTINITFRC field.

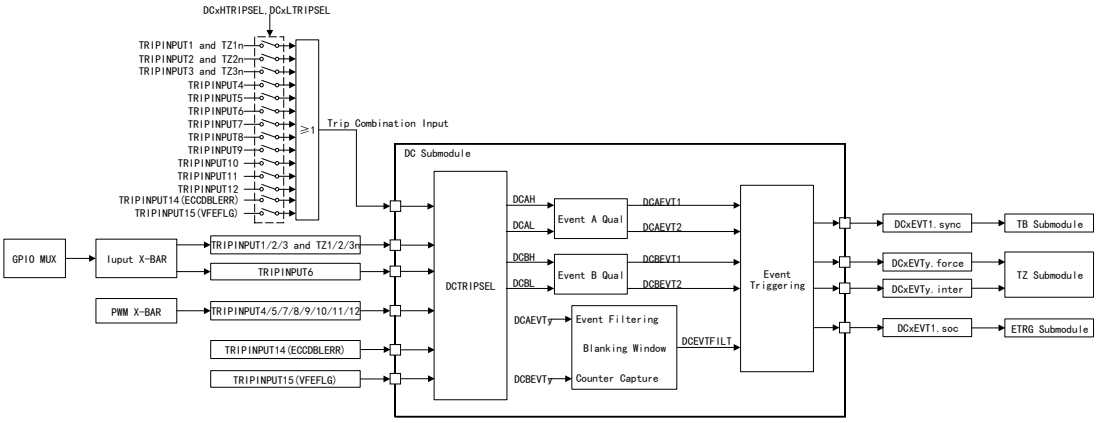
30.5.10 Functional description of digital compare submodule

In the PWM system, the signals exchanged between the digital compare submodule and other submodules are shown in the structure block diagram.

This module is used to compare the external signals (e.g. COMPx signals) with PWM modules, directly generate PWM events/actions, and feed it back to TB, TZ, and ETRG submodules. This module supports the blank window function for filtering out the noise or unnecessary pulses in DC event signals

30.5.10.1 Structure block diagram

Figure 97 Structure Block Diagram of Digital Compare Submodule

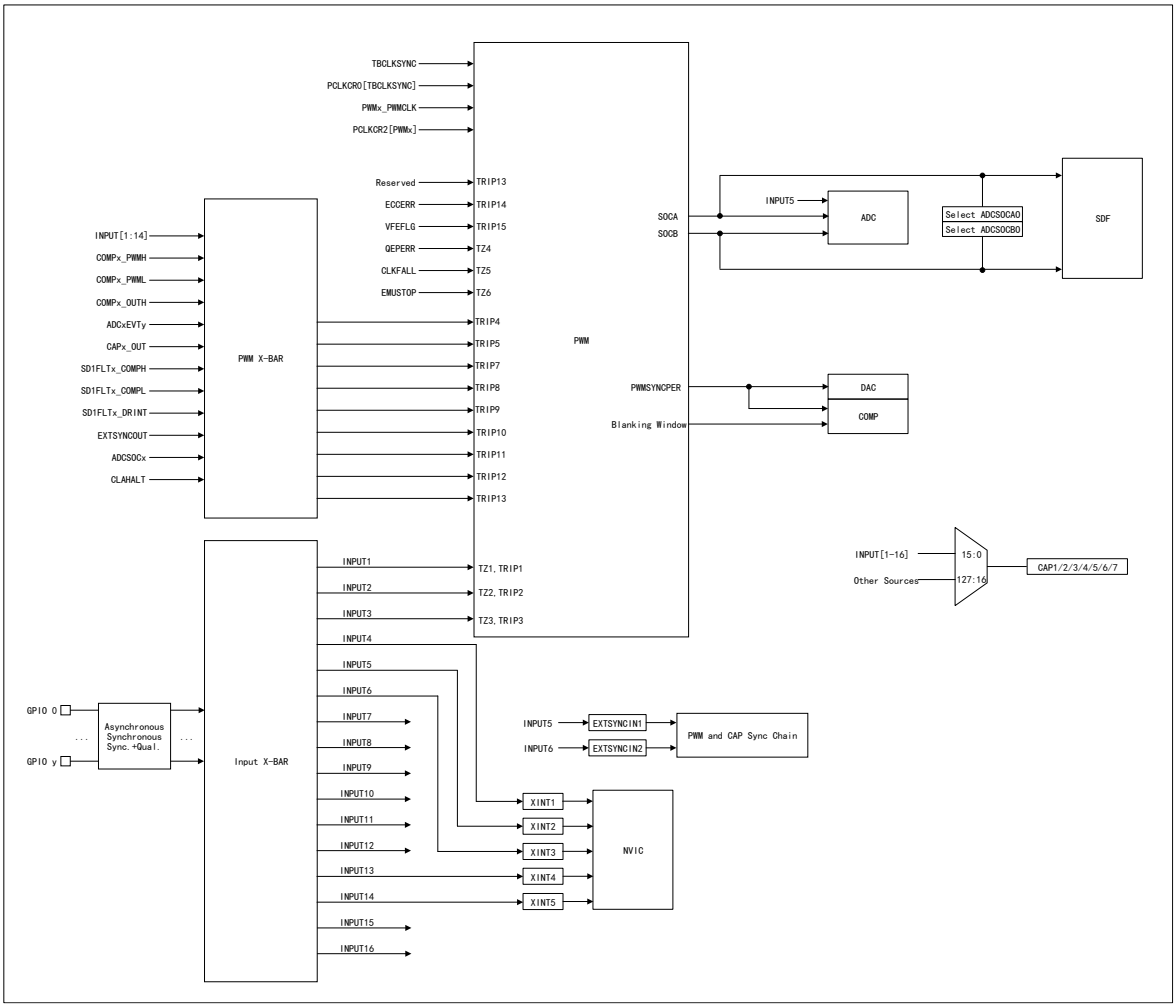


30.5.10.2 Input and output

As shown in the figure below, the CAP input signal is from the input X-BAR signal. On this device, any GPIO pin can be mapped to the trip input and/or trip zone input of the TZ submodule and DC submodule. The INPUTxSELECT register can be used to determine to select GPIO pins as the trip zone input/trip input.

Note: In order to avoid locking the trip signals falsely, users have the responsibility to drive the selected pins to the correct state before enabling the clock and configuring the trip input for the corresponding PWM peripheral.

Figure 98 GPIO Multiplexer-to-Trip Input Structure Block Diagram



30.5.10.3 Use COMP to trip PWM

Enhanced tripping action

In order to allow multiple COMP to affect the DCxEVTy event and trip action at the same time, all trip inputs from the outside of the PWM module are combined through OR logic circuits, and are selected as the trip input combination through the DCTRIPSEL register and inputted to the DCxH and DCxL.

The DCxHTRIPSEL and DCxLTRIPSEL registers can be used to select which trip inputs to be put into the combinational logic circuit to generate DCxH and DCxL signals. The inputs selected as the trip input combination will be passed to the DCTRIPSEL register.

Turn off PWM in a cycle-by-cycle manner

When using COMP to trip the PWM in a cycle-by-cycle manner, measures should be taken to prevent the comparator trip state asserted in one PWM period from being extended to the next period. Under certain conditions,

precautions need to be taken to prevent delayed or persistent tripping events that occur at the end of a PWM cycle from continuing into the next PWM cycle. The comparator can transmit the trip status signal to the downstream PWM module. For applications such as peak current mode control, it is expected that there is only one trip event during each PWM period.

When either the comparator digital filter or the PWM DC submodule is used to qualify the comparator trip signal, N clock cycles will be introduced for qualification before the PWM trip logic circuit can respond to the logic level changes of the trip signal. If the PWM trip state is qualified, the trip state will remain valid for N clock cycles after the comparator trip signal fails. When a qualitative comparator trip signal remains asserted for N clock cycles before the end of a PWM period, the trip condition will be cleared only after the next PWM period starts. Therefore, the trip condition will be detected immediately at the beginning of a new PWM period.

There are three ways an application can ensure that the qualitative trip signal seen by the PWM trip logic fails before the end of each PWM cycle. And in order to avoid the above expectations of the trip state.

Table 136 Solutions

No.	Solution
1	Design the system in such a way that the comparator trip will not assert within N clock cycles before the end of the PWM period.
2	Activate the blank of the comparator trip signal through an event filter at least two clock cycles before the PWMSYNC signal, and continue the blank for at least N clock cycles in the next PWM period.
3	When COMP passes through the COMPxLATCH path, the COMPxLATCH bit needs to be cleared at least N clock cycles before the end of the PWM period. This latch can be cleared by generating an advanced PWMSYNC signal or through the COMPSTSCLR register. The PWM module on this device can generate the PWMSYNC signal by matching the CMPC or CMPD through the HRPCTL register, and it is used to place this signal at any position during the PWM period.

30.5.10.4 Operation points

Digital comparison event

As shown by the functional characteristics of the DC submodule, the DCxH and DCxL signals can be generated by selecting the COMPx signal and trip inputs TZ1n to TZ3n of the analog comparator module through the DCTRIPSEL register. The TZDCSEL register is used to qualify the actions on the selected DCxH and DCxL signals, thereby generating the DCxEVTy events.

The DCxEVTy event can be unfiltered or it can be filtered to provide the filtered event signal DCEVTFILT. Please refer to Event Filtering for filter details. The DCxEVTy event signals or filtered DCEVTFILT signals can generate forced

events for PWM synchronization signals, ADCSOC, TZ submodule, or TZ interrupts.

When the TZxn signal is used for the digital comparison event trip function, it is considered as a normal input signal and can be defined as a low active or high active input. When either the TZxn signal or the DCxEVTy forced signal is valid, the PWM output will be asynchronously tripped. To maintain the latched state, a minimum of 3*TBCLK synchronization pulse width is required. If the pulse width is less than 3*TBCLK synchronization pulse width, the trip state may be latched or not latched by CBC or OST latch.

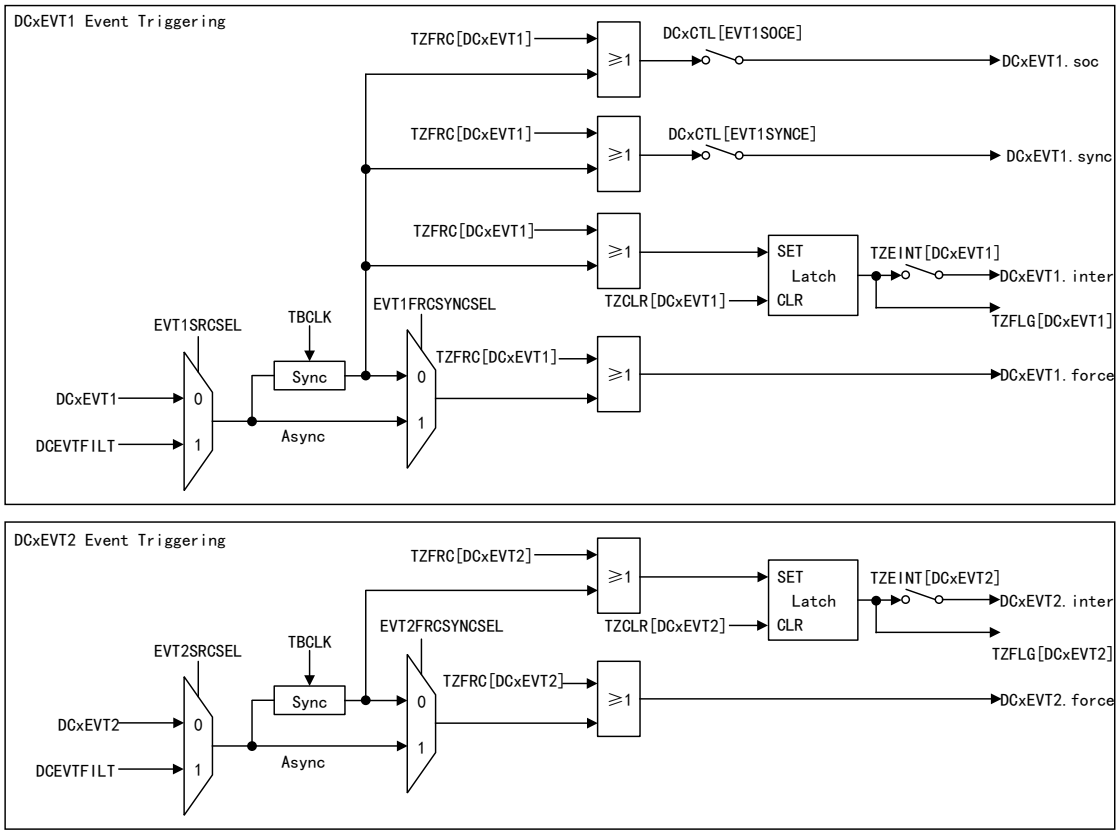
Table 137 Impact of Event Signals on Other Modules

Signal	Description
Forced signal	Configuring the DCAEVTy forced signal as a forced trip zone state through the TZCTL, TZCTLDCA and TZCTLDCB registers can directly affect the output of the PWMxA pin, or configuring the DCBEy.force signal as the one-shot trip/cycle-by-cycle trip source through the TZSEL register can use the TZCTL or TZCTL2 register configuration to affect the tripping action. The priority of conflicting operations in the TZCTL, TZCTL2, TZCTLDCA, and TZCTLDCB registers is shown in the PWMxA/PWMxB Priority Conflicting Operation Table.
Interrupt signal	The DCAEVTy interrupt signal is generated for the NVIC trip zone interrupt. The DCxEVTy bit needs to be set in the TZEINT register to enable the interrupts. If one of the events occurs, it will trigger the PWMx trip zone interrupt, and the corresponding bit in the TZCLR register must be set to clear the interrupt.
Signal for start of conversion	The DCxE1.soc signal interacts with the ETRG submodule, and the SOCASEL and SOCBSEL bits can be used to select the events that generate SOCA and SOCB pulses.
Synchronization signal	The DCxE1 synchronization signal is OR with the PWMx synchronization input signal and SPFSW bit, so as to generate the synchronization pulses for TBCTR.

Table 138 PWMxA/PWMxB Priority Conflicting Operation

Output	Priority conflicting operation
PWMxA	(1) The highest priority is TZA, followed by DCAEVT1, and the lowest priority is DCAEVT2 (2) The highest priority is TZAU, followed by DCAEVT1U, and the lowest priority is DCAEVT2U (3) The highest priority is TZAD, followed by DCAEVT1D, and the lowest priority is DCAEVT2D
PWMxB	(1) The highest priority is TZB, followed by DCBEVT1, and the lowest priority is DCBEVT2 (2) The highest priority is TZBU, followed by DCBEVT1U, and the lowest priority is DCBEVT2U (3) The highest priority is TZBD, followed by DCBEVT1D, and the lowest priority is DCBEVT2D

Figure 99 DCxEVTy Event Trigger Structure Block Diagram



Event filtering

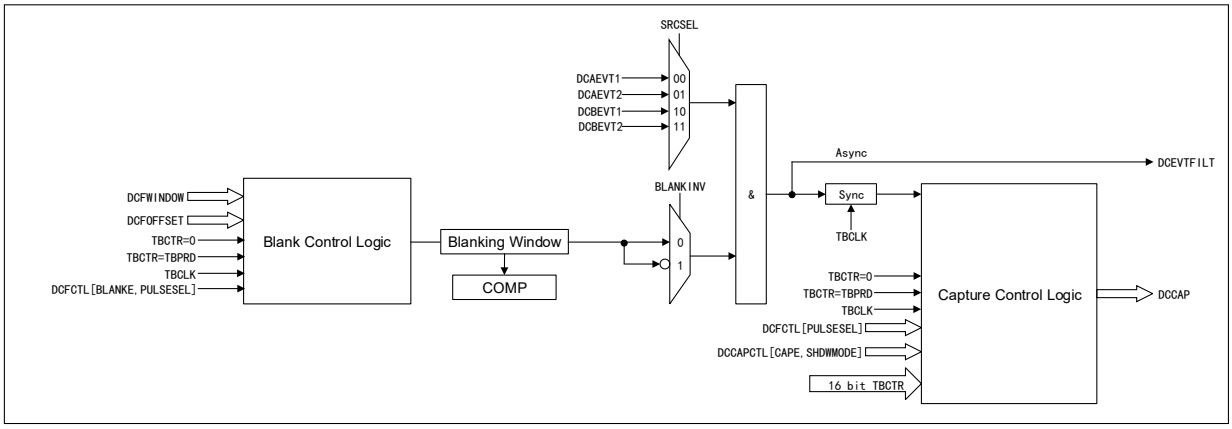
(1) Blank control logic

This logic is used to define a blank window, and all events that occur on the signal will be ignored when the window is active. Configure the blank window through the DCFCTL, DCFOFFSET, and DCFWINDOW registers. The blank window is enabled through the BLANKE bit, and is aligned with the selected pulse simultaneously according to the selection of the PULSESEL field.

To eliminate noise, the DCxEVTy event can be filtered using the event filtering logic circuits, to selectively blank the events for a period of time. This method can be used to select the analog comparator output to trigger the DCxEVTy event, and the blank control logic is used to filter out the potential noise on the signal before tripping the PWM output, and generating interrupts or ADCSOC.

Select the filtering block signal source as the DCxEVTy signal through the SRCSEL field. Program the TBCTR count offset value into the DCFOFFSET register, and this register determines the starting point of the blank window after TBCTR=0 or TBCTR=TBPRD. The duration of the blank window, in the unit of the number of TBCLK counts after the offset counter expires, is written to the DCFWINDOW register through the application. Before and after the blank window ends, the events can still generate the signals of forcing, interrupt, synchronization, and start of conversion as before.

Figure 100 Event Filtering Structure Block Diagram



(2) Capture control logic

Event filtering can also capture the count values of the DCxEVTy events selected based on the DCCAPCTL register.

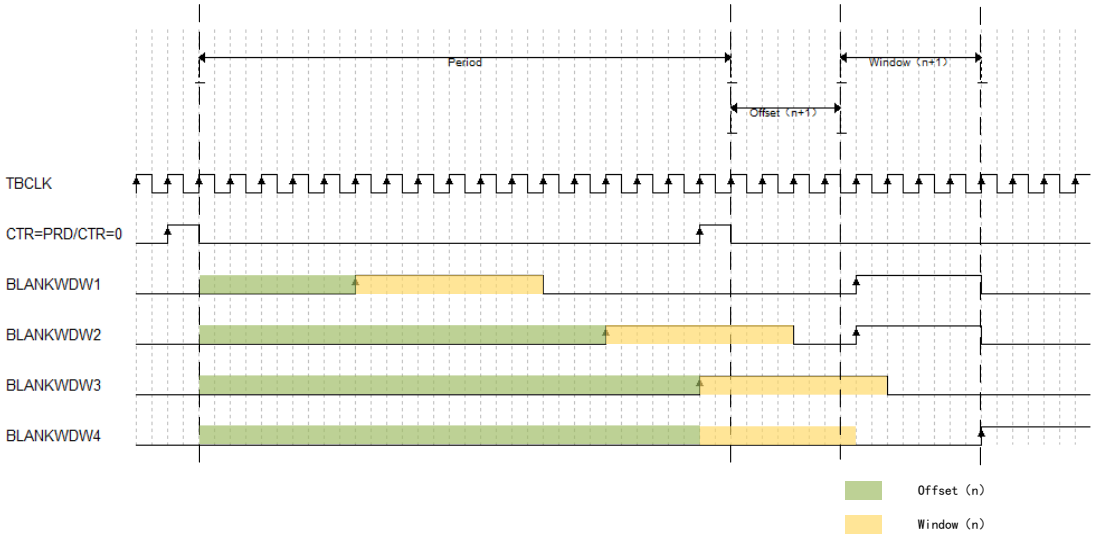
If the blank control logic is enabled, the selected DCxEVTy event will trigger the counter to be captured in the active register, and the CPU can read from the active register. If the shadow mode is enabled, the value of the active register will be copied to the shadow register in the event selected in the PULSESEL field, and the CPU will read from the shadow register.

No other capture events will occur after the selected DCxEVTy event until the event specified by the CAPMODE bit occurs. CAPMODE can be configured in the following ways:

- No other capture events will occur until the CAPSTS bit is cleared through the CAPCLR bit
- No other capture events will occur until the event selected in the PULSESEL field occurs

The several timing conditions for blank window and offset during the PWM period are shown in the following figure. When the blank window crosses the TBCTR=0 or TBCTR=TBPRD boundary, the next window still starts from the same offset value, i.e. after the TBCTR=0 or TBCTR=TBPRD pulse.

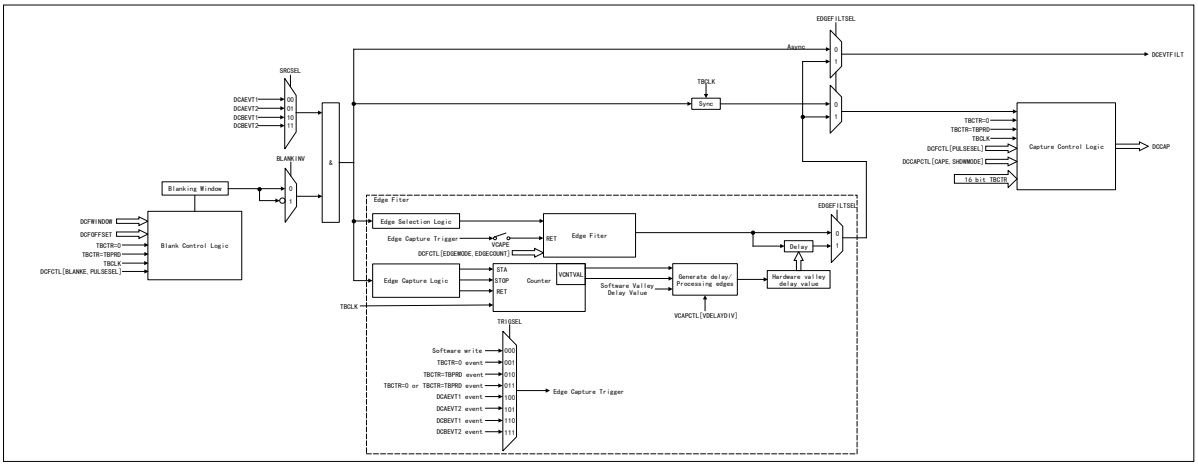
Figure 101 Blank Window Timing Diagram



Valley switching function

The event filtering describes the event filtering logic and valley switching function, and the programmable valley switching can be implemented without adding any external circuits. This function provides an on-chip hardware mechanism, which supports capture of oscillation period, and can accurately delay the PWM switching time, and allows for the number of programmable edges before the delay takes effect. In addition, it provides selection of multiple triggers and events, and optimal performance under changing system/operating conditions which are easy to adapt to changes.

Figure 102 Structure Block Diagram of Valley Switching



The DCxEVTy signal needs further processing so as to support the valley switching function. The operating steps to enable the valley switching function are as follows:

Table 139 Operating Steps to Enable Valley Switching Function

Step	**Operation**	**Description**
1	Select the filtering block signal source	By selecting the input source of the valley switching through the SRCSEL field, a blank window (blank control logic) can be added to this input. Here, the input of the valley switching is selected as the comparator output or external input.
2	Select the mode and number of edges	Capture the number of rising edge, falling edge, or double edges captured by the edge filter through the EDGEMODE and EDGECOUNT fields.
3	Select events to restart the edge filter	Select the correct event based on the TRIGSEL field to reset and restart the edge filter, and the edge capture event is triggered or sent by the selected edge.
4	VCAPE=1	Enable the valley capture logic
5	Select the edge of starting counting	Select the starting edge of measurement and capture of the oscillation period through the STARTEDGE field, e.g. the 16-bit counter starts counting
6	Select the edge of stopping counting	Select the edge of the 16-bit counter stopping counting through the STOPEDGE field. The oscillation period information is provided by the captured counter value, and the STARTEDGE value must always be less than the STOPEDGE value.
7	Configure and apply the capture delay	Configure the DCxEVTy signal after edge filtering and apply the capture delay. The value of the valley time base counter can be directly applied, and be used together with the software programming value for offset adjustment, or only a small part of the delay can be applied/not applied with the software valley delay value. This can be correctly applied to the delay of valley points.
8	Configure the edge filter output delay	Set the EDGEFILTDLYSEL bit, and apply the hardware delay according to the above captured value.

30.5.11 PWM X-BAR

30.5.11.1 PWM X-BAR structure block diagram

The module can select various trigger sources, making it any one of the 8 dedicated PWM trip inputs, namely TRIP4/5/7/8/9/10/11/12.

For PWM X-BAR structure block diagram, see X-BAR flag section.

Note: For details about X-BAR-related information and X-BAR flag bits, see the X-BAR section.

30.5.11.2 PWM X-BAR single output structure block diagram

PWM X-BAR has eight outputs, which are routed to each PWM module. For the PWM X-BAR single output structure block diagram, see the PWM XBAR section, and its output shares the same architecture as all other outputs.

First, determine the signals passed to PWM. For details, please see the Output X-BAR multiplexing configuration table in the X-BAR section. Each TRIPx output multiplexer (32 multiplexers in total) selects at most one signal. The input signal for each multiplexer is determined by the TRIPxMUX0TO15CFG and TRIPxMUX16TO31CFG registers. The multiplexers in the TRIPxMUXENABLE register need to be enabled to pass any signal through PWM. Before passing the respective TRIPx signals to PWM, all enabled multiplexers are logically ORed. The polarity of the TRIPx signals is selected via the TRIPOUTINV register.

Note: The "Reserved" signals are not usable, and all unused and reserved signals are set to 0.

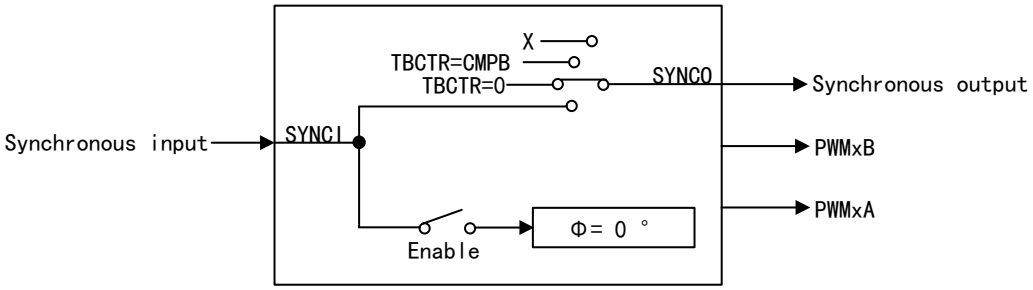
30.5.12 Power topology applications

The PWM module contains all the necessary local resources, allowing it to operate as a fully independent module or synchronize with other identical PWM modules.

30.5.12.1 Simplified PWM module structure block diagram

For better understanding of multiple modules working together in a system, the simplified PWM module and multi-switch power topology PWM X-BAR structure block diagram is as shown below, illustrating how the multi-switch power topology works with multiple PWM modules to control the required key resources.

Figure 103 Simplified PWM Module Structure Block Diagram



30.5.12.2 Synchronization input and synchronization output configuration

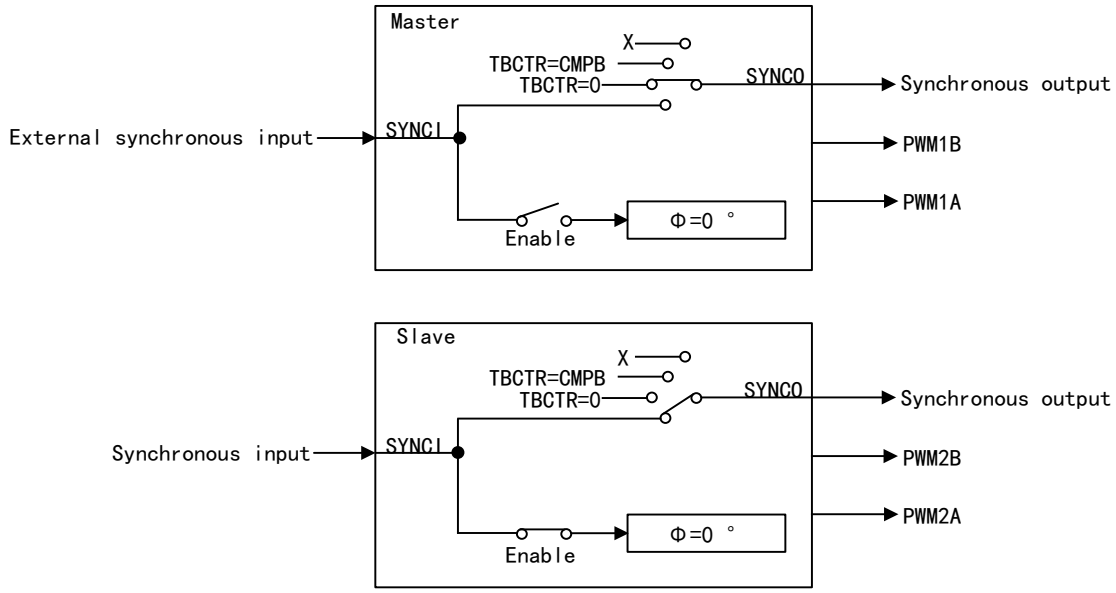
Table 140 Synchronization Input and Synchronization Output Configuration

Option	Configuration	Description
Synchronous input	Enable switch closure	When the synchronization latch signal arrives, load the phase register into its own counter
	Enable switch disconnection	There is no operation, ignoring the incoming synchronization latch signal
	Synchronization signal pass-through	Connect synchronization output to synchronization input
	Synchronization output connected to TBCTR=TBPRD	In master mode, provide synchronization signals at PWM boundaries
	Synchronization output connected to TBCTR=CMPB	In master mode, provide synchronization signals at any programmable time point
	Disable synchronization output	The module is in independent mode, not providing synchronization signals to other modules
Synchronous output	Synchronization signal pass-through	Connect synchronization output to synchronization input
	Synchronization output connected to TBCTR=TBPRD	In master mode, provide synchronization signals at PWM boundaries
	Synchronization output connected to TBCTR=CMPB	In master mode, provide synchronization signals at any programmable time point
	Disable synchronization output	The module is in independent mode, not providing synchronization signals to other modules

Mode combination

For synchronization output options, the module can ignore or choose to load a new phase value during synchronization input strobe by enabling the switch. Although multiple combinations are possible, master mode and slave mode are the two most common modes, as shown in the figure below.

Figure 104 PWM1 Master Mode and PWM2 Slave Mode Configuration Diagram



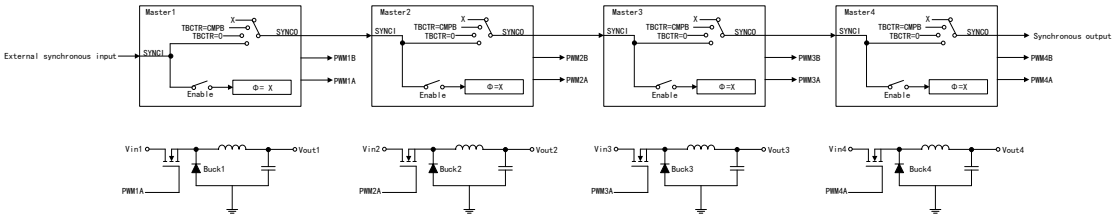
30.5.12.3 Buck converter

Buck is one of the simplest power converter topologies. When a single PWM module is configured as a master module, it can control two Buck stages of the same frequency.

Control multiple independent frequencies

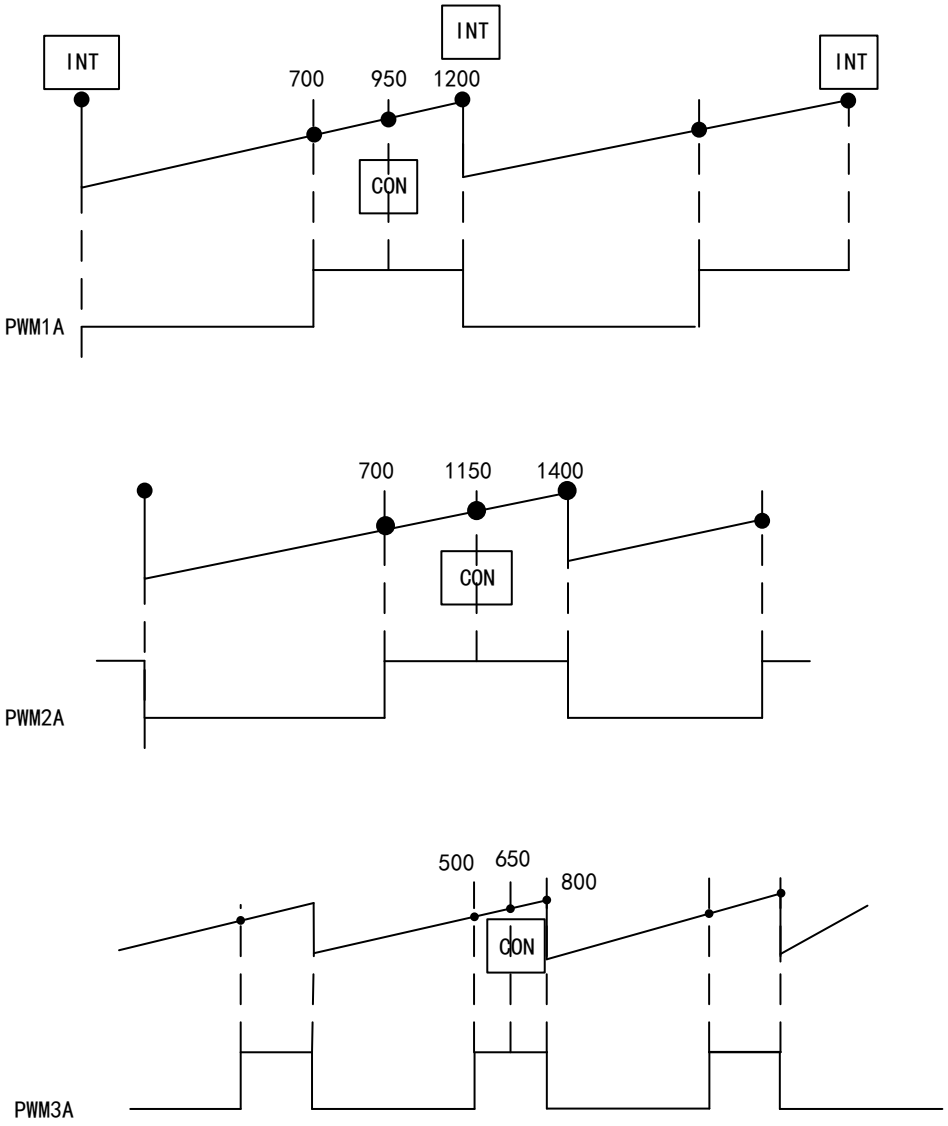
When each Buck converter needs to be controlled by independent frequency, each converter stage must be assigned a PWM module. Each stage operates at an independent frequency. In this case, all four PWM modules are configured as master modules without using synchronization.

Figure 105 Structure Block Diagram for Controlling Four Buck Stages (FPWM1≠FPWM2≠FPWM3≠FPWM4)



Note: If Φ=X, the value of the time base phase shift register can be ignored.

Figure 106 Waveform Diagram of Four Buck Stages (only three waveforms are shown)



Note:

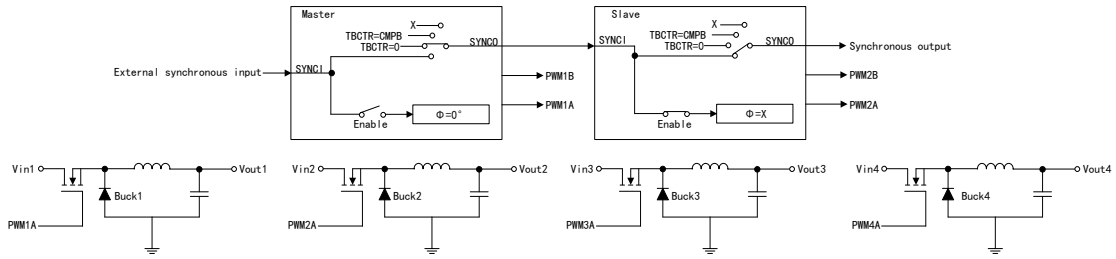
(1) INT indicates the event triggers an interrupt, while CON indicates the event triggers the start of ADC conversion.

(2) Other unspecified points indicate the start of ADC conversion.

Control multiple identical frequencies

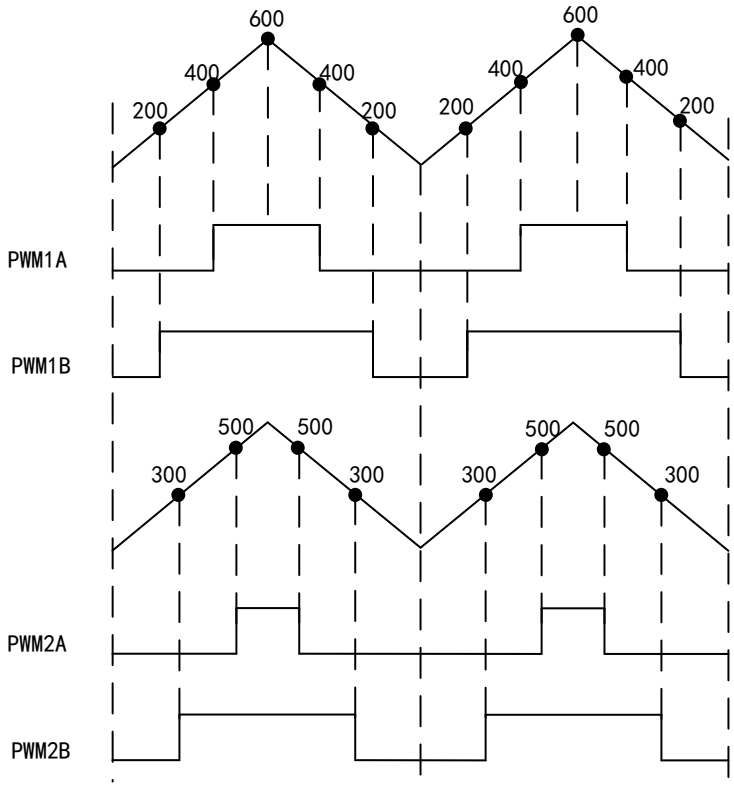
When synchronization is required, configure PWM2 as a slave module and run it at an integer multiple of PWM1's frequency, i.e., FPWM2 = N * FPWM1. The structure block diagram and waveform diagram of four Buck stages are shown below. Synchronization signals from master to slave mode ensure these modules remain locked.

Figure 107 Structure Block Diagram for Controlling Four Buck Stages



Note: If Φ=X, the value of the time base phase shift register can be ignored.

Figure 108 Buck Waveforms for Controlling Four Buck Stages (FPWM1 = FPWM2)



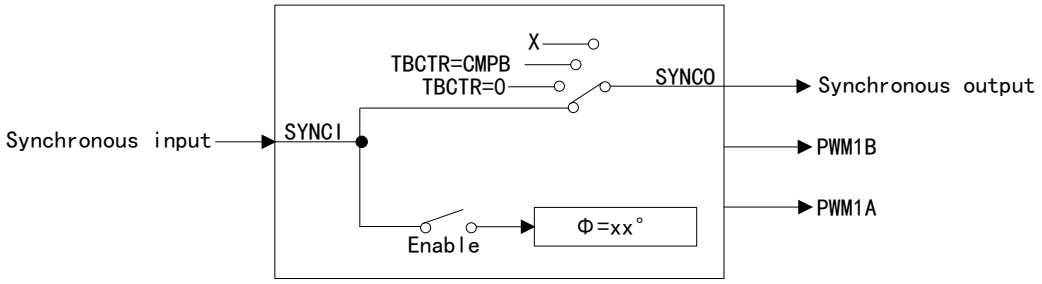
Note: Each dot represents an ADC conversion.

Control peak current control mode Buck converter

Peak current control techniques can be used for automatic overcurrent limitation, fast correction of input voltage variations, and reduction of magnetic saturation. The topology block diagram and waveform diagram of a Buck converter using PWM1A and on-chip analog comparators are shown below. Output current is detected via a current-sense resistor and fed to the non-inverting input of the on-chip comparator. An internally programmable 12-bit DAC can be connect this input to an external reference or provide a reference peak current to the comparator's inverting end. The output of the comparator

serves as the input for the DC submodule. If the detected current reaches the reference peak, the PWM module is configured to trip the PWM1A output. A cycle-by-cycle trip mechanism is used.

Figure 109 Peak Current Control Mode Buck Converter Structure Block Diagram



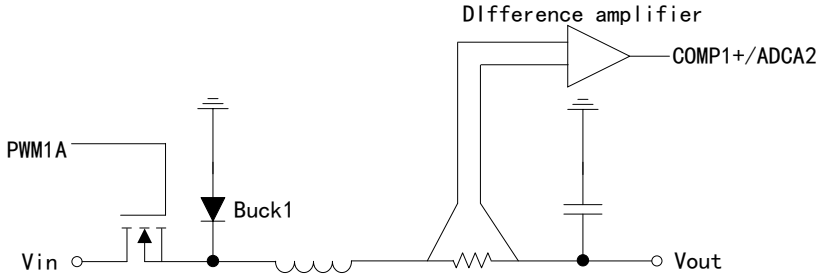
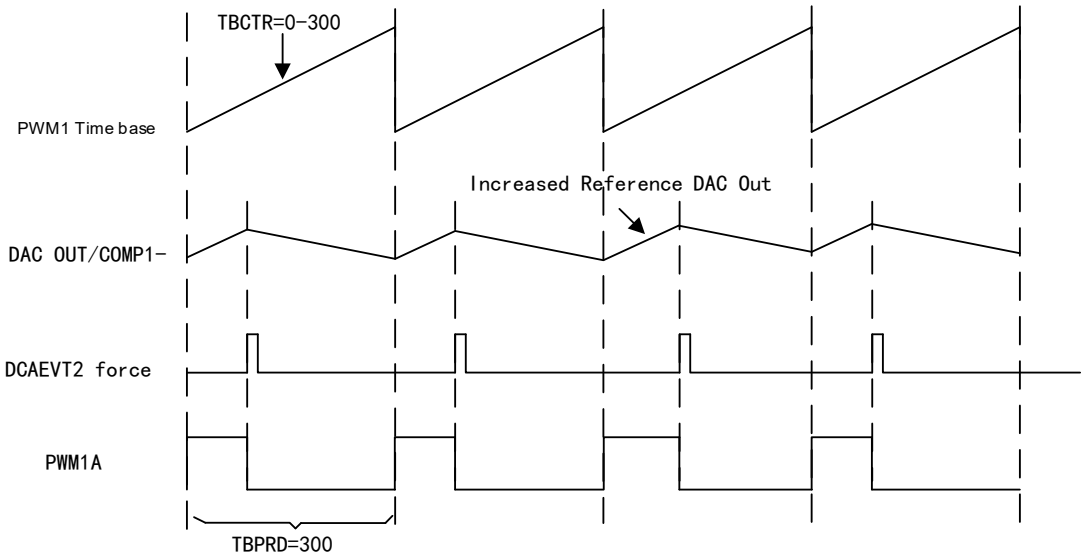


Figure 110 Peak Current Control Mode Buck Converter Waveform Diagram (only three waveforms are shown)



30.5.12.4 Half H-bridge converter

The same PWM module can be used to control the topology of multiple switching elements and the use of a single PWM module to control the half-H bridge stage can extend this control to multiple stages. The structure block diagram and waveform diagram for controlling two synchronized half H-bridges are shown below, where PWM2 operates at an integer multiple of PWM1's

frequency, i.e., FPWM2 = N * FPWM1.

PWM2 (slave mode) is configured for synchronization signal pass-through, allowing PWM3 to control the third half H-bridge while remaining synchronized with PWM1 (master mode).

Figure 111 Structure Block Diagram for Controlling Two Synchronized Half H-bridges

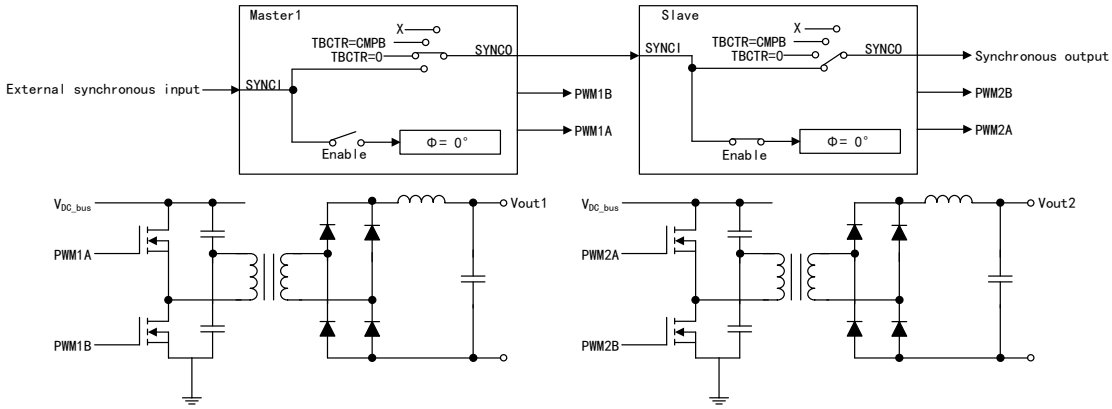
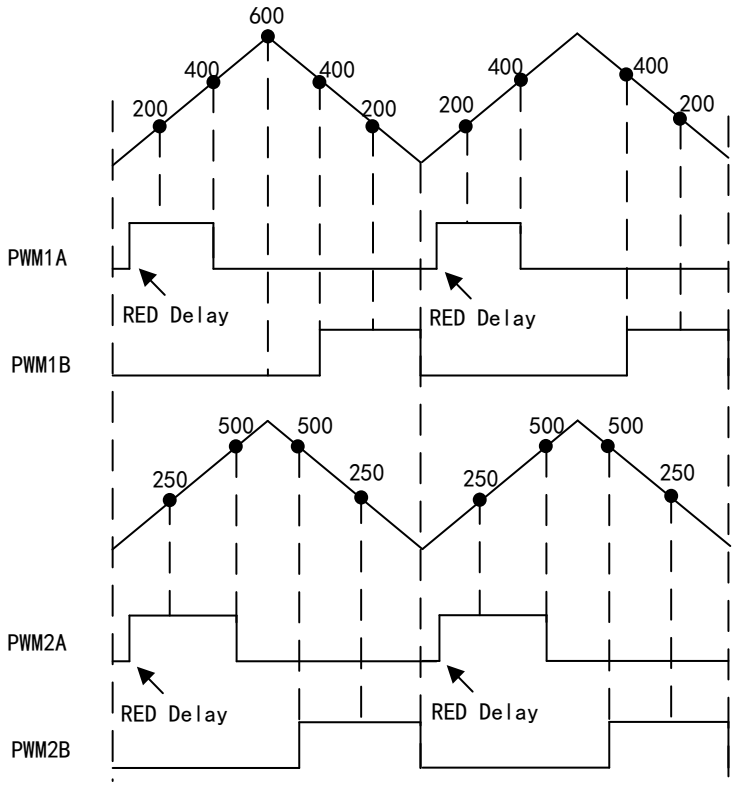


Figure 112 Waveform Diagram for Two Synchronized Half H-bridges (FPWM1 = FPWM2)



30.5.12.5 Dual three-phase inverters

When extending single power stage control to multiple modules for a three-phase inverter, three PWM modules are used to control six switching elements, with each module controlling one leg of the inverter. To ensure all legs switch at the same frequency and remain synchronized, a master-slave setup can be

used, with one master module and two slave modules. The structure block diagram and waveform diagram for six PWM modules controlling two independent three-phase inverters (each driving one motor) are shown in the figure below.

As described in the Buck converter section, the inverter can be operated in the following ways:

- Synchronize the operation of two inverters
  - PWM1 as the master module, with other modules as slave modules
- Operate each inverter at different frequencies
  - PWM1 and PWM4 as master modules, where FPWM1 = FPWM2 = FPWM3, and FPWM4 = FPWM5 = FPWM6
  - PWM4/5/6 operate at an integer multiple of PWM1/2/3's frequency, i.e., FPWM4/5/6 = N * FPWM1/2/3

Figure 113 Structure Block Diagram of Dual Three-phase Inverter Stages

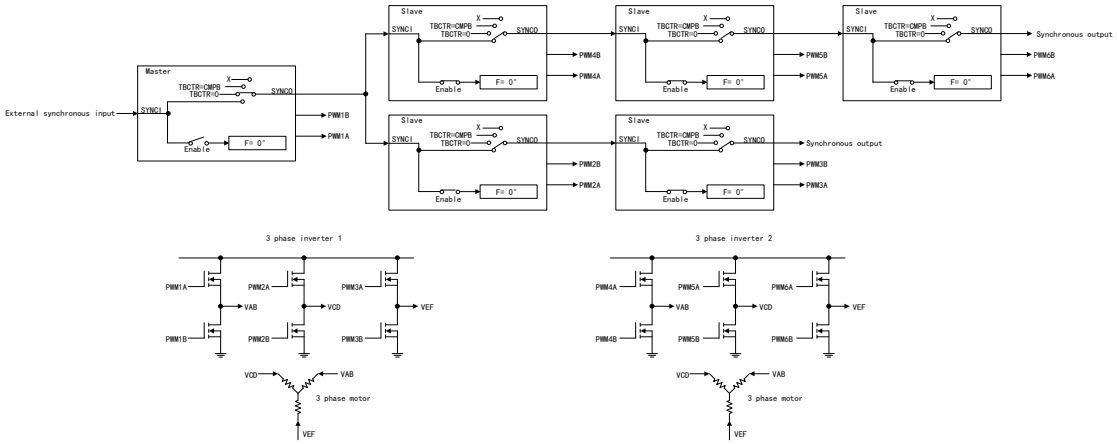
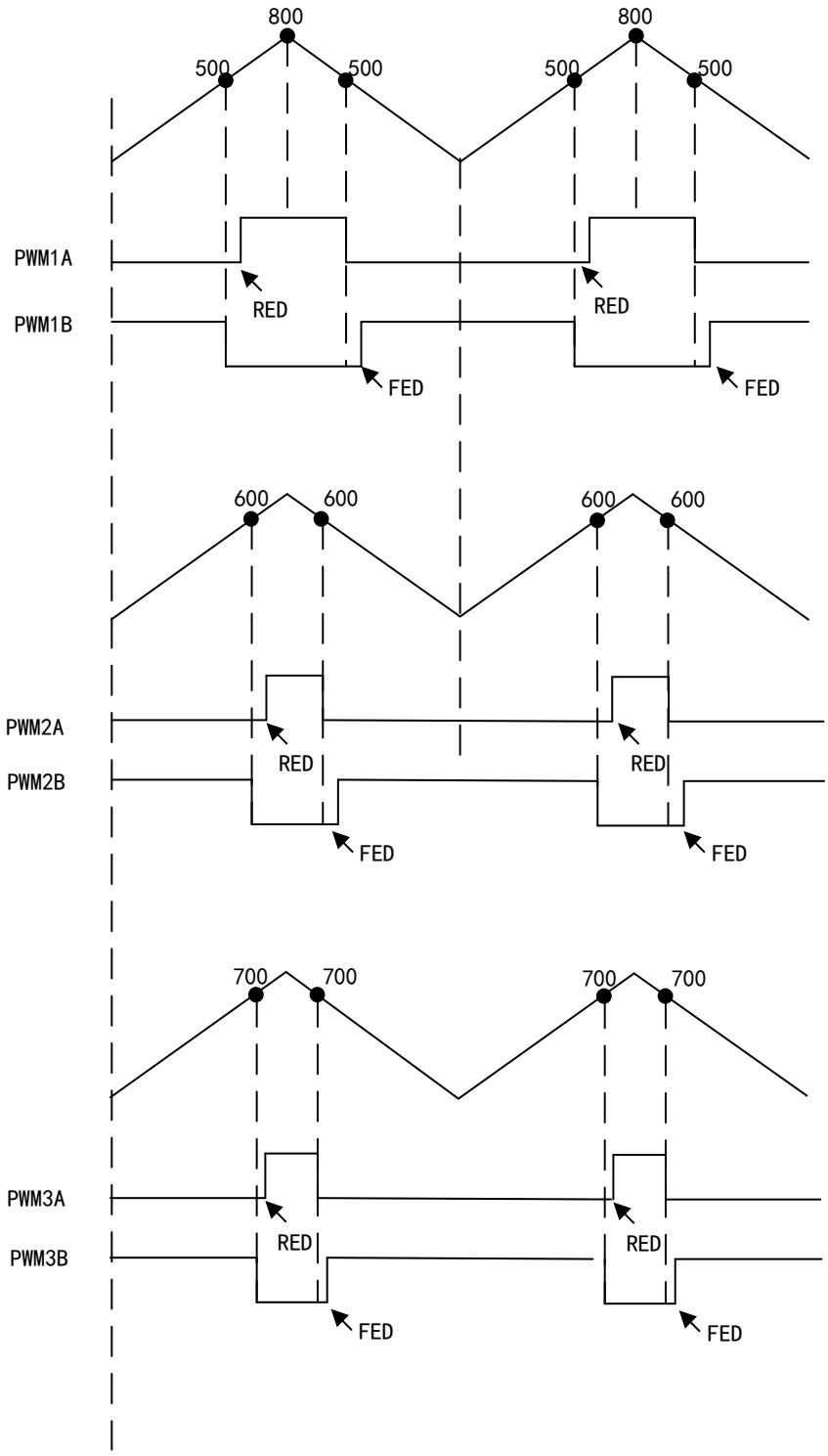


Figure 114 Inverter Waveform Diagram for Dual Three-phase Inverters (only one inverter is shown)



30.5.12.6 Three-phase interleaved DC/DC converter

The structure block diagram and waveform diagram for controlling the three-phase interleaved DC/DC converter using three PWM modules are shown below, where PWM1 acts as the master module, and other modules as slave

modules. Since the phase relationship between adjacent modules must be 120° to operate, this requires the TBPHS registers of the slave modules to be set to 1/3 and 2/3 of TBPRD.

The formula for calculating the TBPHS value for N phases is: TBPHS (N, M) = (TBPRD/N) * (M-1), where N is the number of phases, and M is the PWM module number. This concept can be extended to four or more phases by setting the TBPHS value accordingly. For example, if TBPRD = 600 and N = 3, the phase value for PWM2 is TBPHS(3, 2) = 200, and for PWM3, it is TBPHS(3, 3) = 400. Both slave modules are synchronized to the PWM1 module.

Figure 115 Structure Block Diagram for Three-phase Interleaved DC/DC Converter

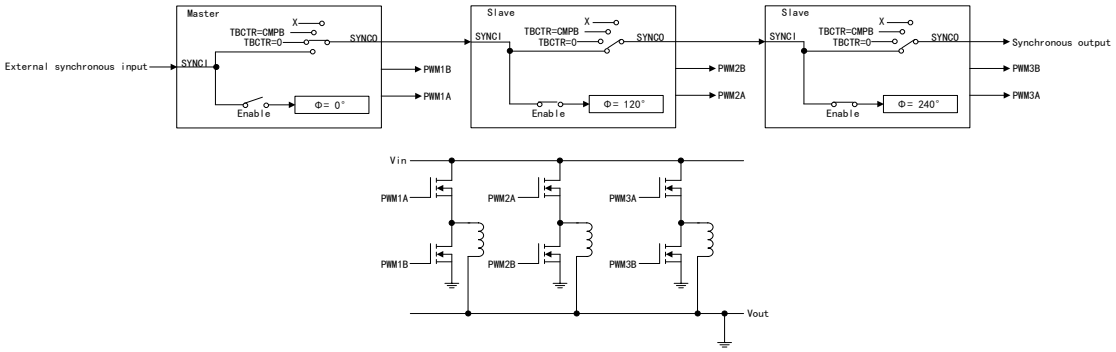
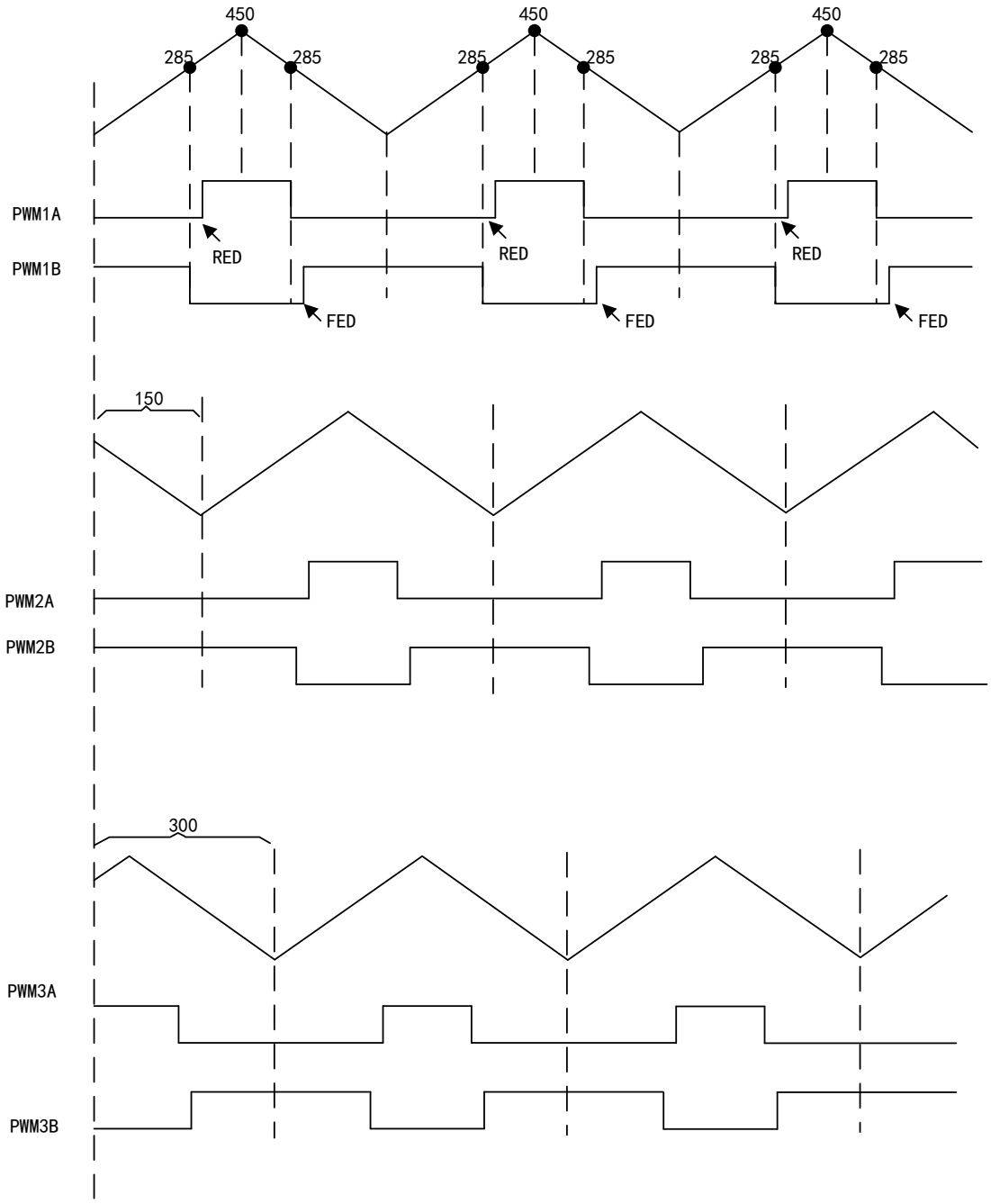


Figure 116 Waveform Diagram for Three-phase Interleaved DC/DC Converter



30.5.12.7 Phase control

To ensure proper operation, appropriate values can be set for the TBPHS register, enabling multiple PWM modules to address power topologies dependent on the phase relationship between bridge arms. In order to synchronize the input pulse, the value of the TBPHS register can be loaded into the TBCTR register, and the PWM module can be configured. For details, see the time base submodule function description section.

The slave module leads the master module with a phase relationship of 120°

between the two modules. The waveform diagram for this configuration is shown in the figure below, where the period value of the master and slave modules is 600, and that of the slave module is 200. Whenever the master module generates a TBCTR=TBPRD synchronization pulse, the phase value is loaded into the PWM_TBCTR register of the slave module, ensuring that the time base of the slave module always leads the master module by 120°.

Figure 117 Structure Block Diagram for Phase Control of Two PWM Modules

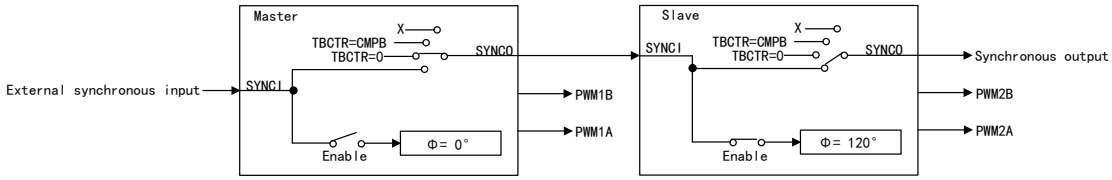
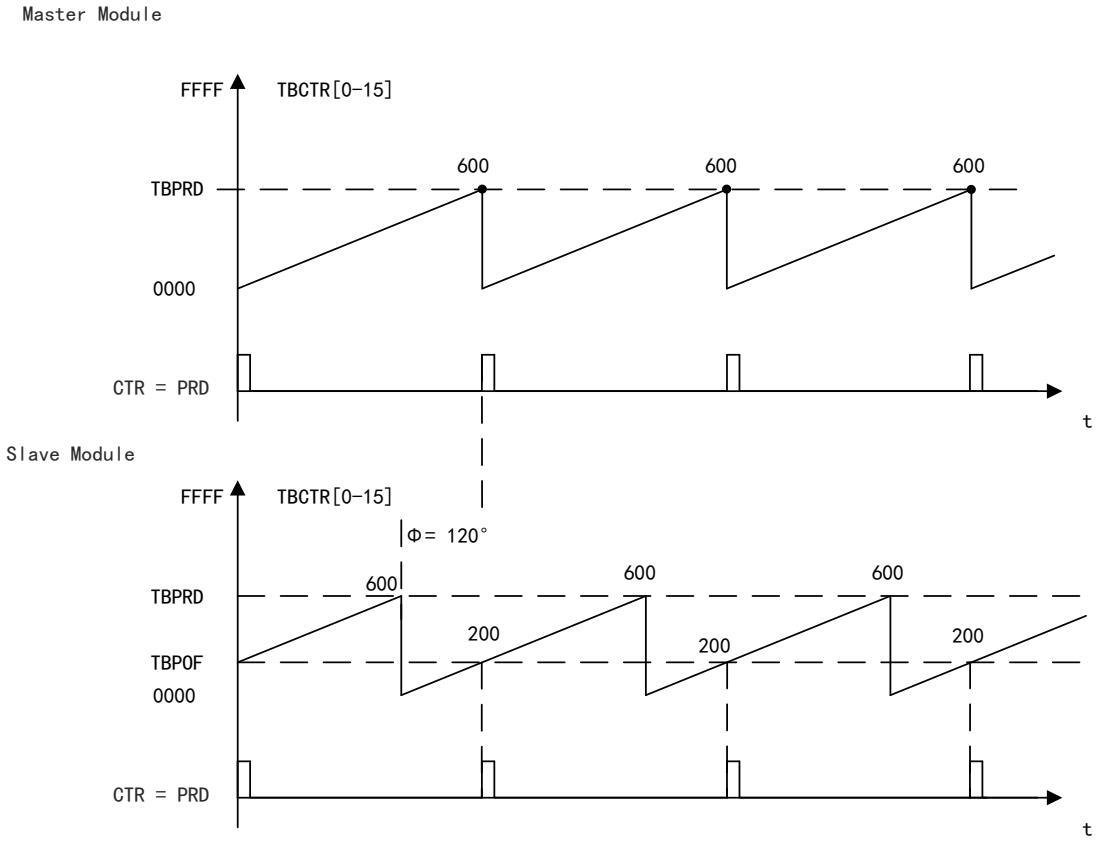


Figure 118 Waveform Diagram for Phase Synchronization between Two PWM Modules

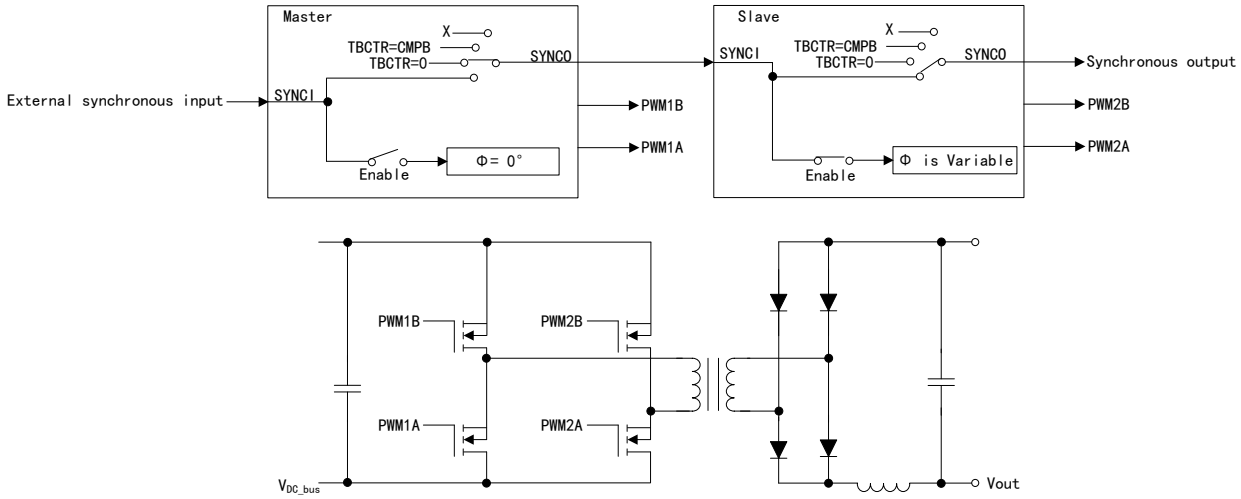


30.5.12.8 Zero-voltage switching full-bridge converter

As shown in the full H-bridge control structure block diagram, a static phase relationship exists between modules. At this time, the power topology can be controlled by adjusting the duty cycle or the phase relationship can be dynamically changed cycle by cycle. The control of zero-voltage switching full-bridge or phase-shifted full-bridge power structures conforms to this characteristic. In this system:

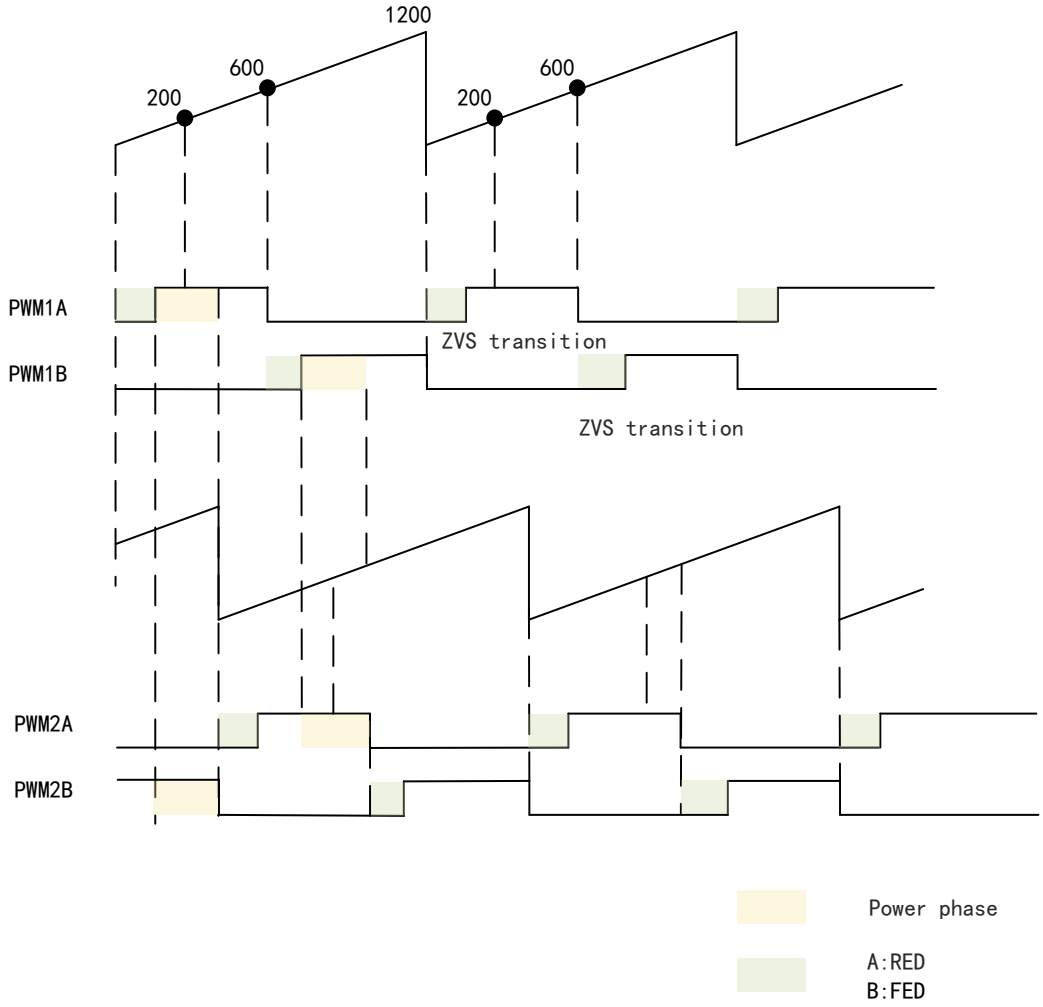
- The control parameter is the phase relationship between modules, and the duty cycle is usually maintained at approximately 50%
- Resources of two PWM modules need to be allocated to a single power stage, requiring control over four switching elements

Figure 119 Full H-bridge Control Structure Block Diagram (F_{PWM1}=F_{PWM2})



The waveform generated by synchronously controlling the full H-bridge using the master-slave module combination is shown in the figure below. At this time, both the master and slave modules need to switch at the same PWM frequency. The phase is controlled through the TBPHS register of the slave module. If the TBPHS register of the master module is not used, it is initialized to 0.

Figure 120 Full H-bridge Waveform Diagram

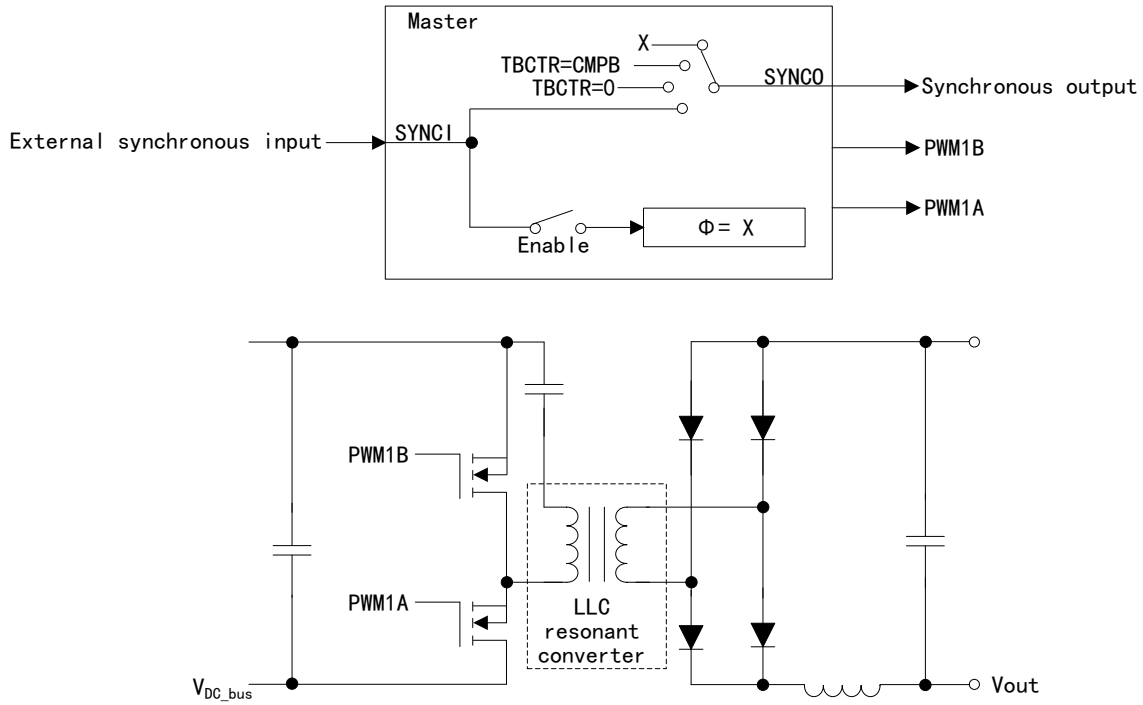


30.5.12.9 **Resonant converter**

In the field of power electronics, the H-bridge LLC resonant converter topology can also be used in consumer electronics applications requiring high efficiency and power density. The block diagrams of two resonant converters and the PWM waveform diagram are shown in the figure below. PWM1 uses a single-channel configuration and can be extended to multi-channel. At this time:

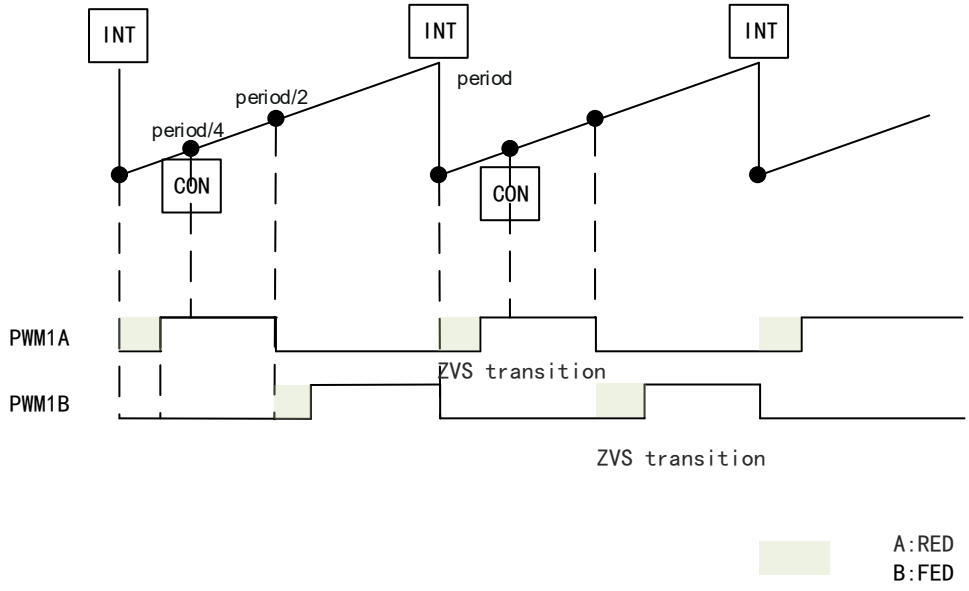
- The control parameter is the frequency, and the duty cycle is usually maintained at approximately 50%
- The dead band is not controlled and remains constant at 300 ns, but users can update the dead band in real time
- Sufficient delay can be adjusted through software switching to improve efficiency

Figure 121 Block Diagrams for Two Resonant Converters



Note: If Φ=X, the value of the time base phase shift register can be ignored.

Figure 122 Resonant Converter PWM Waveform Diagram



Note:

(1) INT indicates the event triggers an interrupt, while CON indicates the event triggers the start of ADC conversion.

(2) Other unspecified points indicate the start of ADC conversion.

30.5.13 Register latch protection mechanism

To protect critical PWM registers from accidental damage during code runaway, a register lock protection mechanism is added. During code runaway, critical PWM registers are protected from accidental write damage. When the value of the KEY field in the PWMLOCK register is set to 0xA5A5, this register can be successfully written. For details, see the description of the PWMLOCK register.

The lock bits and corresponding registers defined by the PWMLOCK register are as follows:

- HRLOCK: Lock the HRPWM register settings. The locked registers are HRCNFG, HRPWR, HRMSTEP, and HRPCTL
- GLLOCK: Lock the global load register settings. The locked registers are GLDCTL and GLDCFG
- TZCFGLOCK: Lock the trip zone register settings. The locked registers are TZSEL, TZDCSEL, TZCTL, TZCTL2, TZCTLDCx, and TZEINT
- TZCLRLOCK: Lock the trip zone clear register settings. The locked registers are TZCLR, TZCBCCLR, TZOSTCLR, and TZFRC
- DCLOCK: Lock the digital compare register settings. The locked registers are DCTRIPSEL, DCxCTL, DCFCTL, DCCAPCTL, DCxHTRIPSEL, and DCxLTRIPSEL

Note: Due to the presence of the KEY field in the same register, when the KEY matches, any 16-bit write operation to either the upper or lower 16 bits of the register will be ignored. Only 32-bit write operations will succeed.

30.6 Register bank address

Table 141 Register Bank Address

Device register	Register bank	Start address	End address
Pwm1Regs	PWM_REGS	0x5000 0000	0x5000 03FF
Pwm2Regs	PWM_REGS	0x5000 0400	0x5000 07FF
Pwm3Regs	PWM_REGS	0x5000 0800	0x5000 0BFF
Pwm4Regs	PWM_REGS	0x5000 0C00	0x5000 0FFF
Pwm5Regs	PWM_REGS	0x5000 1000	0x5000 13FF
Pwm6Regs	PWM_REGS	0x5000 1400	0x5000 17FF
Pwm7Regs	PWM_REGS	0x5000 1800	0x5000 1BFF
SyncSocRegs	SYNC_SOC_REGS	0x4003_0C80	0x4003_0FFF

30.7 Register address mapping

Table 142 PWM_REGS Offset Address

Register name	Register description	Offset address	WRPRT
TBCTL	Time base control register	0x00	-
TBCTL2	Time base control register 2	0x02	-
TBCTR	Time base counter register	0x08	-
TBSTS	Time base status register	0x0A	-
CMPCTL	Counter compare control register	0x10	-
CMPCTL2	Counter compare control register 2	0x12	-
DBCTL	Dead-band generator control register	0x18	-
DBCTL2	Dead-band generator control register 2	0x1A	-
AQCTL	Action qualifier control register	0x20	-
AQTSRCSEL	Select action qualifier trigger event source register	0x22	-
PCCTL	PWM chopper control register	0x28	-
VCAPCTL	Valley capture control register	0x30	-
VCNTCFG	Valley counter configuration register	0x32	-
HRCNFG	HRPWM configuration register	0x40	√
HRPWR	Enable HRP calibration logic register	0x42	√
HRMSTEP	HRP step count register	0x4C	√
HRCNFG2	HRPWM configuration register 2	0x4E	√
HRPCTL	High-resolution period control register	0x5A	√
TRREM	HRPWM conversion high-resolution remainder register	0x5C	√
GLDCTL	Global PWM load control register	0x68	√
GLDCFG	Global PWM load event configuration register	0x6A	√
PWMXLINK	PWMx link register	0x70	-
AQCTLA	Action qualifier control register for output A	0x80	-
AQCTLA2	Additional action qualifier control register for output A	0x82	-
AQCTLB	Action qualifier control register for output B	0x84	-
AQCTLB2	Additional action qualifier control register for output B	0x86	-
AQSFRC	Action qualifier software forced register	0x8E	-
AQCSFRC	Action qualifier continuous software forced register	0x92	-
DBREDHR	Dead-band generator rising edge delay high-resolution register	0xA0	-

Register name	Register description	Offset address	WRPRT
DBRED	Dead-band generator rising edge delay register	0xA2	-
DBFEDHR	Dead-band generator falling edge delay high-resolution register	0xA4	-
DBFED	Dead-band generator falling edge delay register	0xA6	-
TBPHS	Time base phase offset register	0xC0	-
TBPRDHR	High-resolution time base period register	0xC4	-
TBPRD	Time base period register	0xC6	-
CMPA	Counter compare module A register	0xD4	-
CMPB	Counter compare module B register	0xD8	-
CMPC	Counter compare module C register	0xDE	-
CMPD	Counter compare module D register	0xE2	-
GLDCTL2	Global PWM load control register 2	0xE8	-
SWVDELVAL	Software valley mode delay register	0xEE	-
TZSEL	Trip zone selection register	0x100	√
TZDCSEL	Trip zone digital comparator configuration register	0x104	√
TZCTL	Trip zone control register	0x108	√
TZCTL2	Trip zone control register 2	0x10A	√
TZCTLDCA	Trip zone control digital comparator A register	0x10C	√
TZCTLDCB	Trip zone control digital comparator B register	0x10E	√
TZEINT	Trip area interrupt enable register	0x11A	√
TZFLG	Trip zone flag register	0x126	-
TZCBCFLG	Trip zone CCT flag register	0x128	-
TZOSTFLG	Trip zone OST flag register	0x12A	-
TZCLR	Clear trip zone register	0x12E	√
TZCBCCLR	Clear trip zone CCT register	0x130	√
TZOSTCLR	Clear trip zone OST register	0x132	√
TZFRC	Forced trip zone flag register	0x136	√
ETSEL	Event trigger selection register	0x148	-
ETPS	Event trigger preallocated register	0x14C	-
ETFLG	Event trigger flag register	0x150	-
ETCLR	Clear event trigger register	0x154	-
ETFRC	Forced event trigger register	0x158	-
ETINTPS	Event trigger interrupt prescale register	0x15C	-

Register name	Register description	Offset address	WRPRT
ETSOCPS	Event trigger SOC prescale register	0x160	-
ETCNTINITCTL	Event trigger counter initialization control register	0x164	-
ETCNTINIT	Event trigger counter initialization register	0x168	-
DCTRIPSEL	Digital compare trip selection register	0x180	√
DCACTL	Digital compare A control register	0x186	√
DCBCTL	Digital compare B control register	0x188	√
DCFCTL	Digital compare filter control register	0x18E	√
DCCAPCTL	Digital compare capture control register	0x190	√
DCFOFFSET	Digital compare filter offset register	0x192	-
DCFOFFSETCNT	Digital compare filter offset counter register	0x194	-
DCFWINDOW	Digital compare filter window register	0x196	-
DCFWINDOWCNT	Digital compare filter window counter register	0x198	-
DCCAP	Digital compare counter capture register	0x19E	-
DCAHTRIPSEL	Digital compare A high trip selection register	0x1A4	√
DCALTRIPSEL	Digital compare A low trip selection register	0x1A6	√
DCBHTRIPSEL	Digital compare B high trip selection register	0x1A8	√
DCBLTRIPSEL	Digital compare B low trip selection register	0x1AA	√
PWMLOCK	PWM lock register	0x1F4	-
HWVDELVAL	Hardware valley mode delay register	0x1FA	-
VCNTVAL	Hardware valley counter register	0x1FC	-

Table 143 SYNC_SOC_REGS Offset Address

Register name	Register description	Offset address	WRPRT
SYNCSELECT0	Synchronization input and output selection register 0	0x00	√
ADCSOCOUTSELECT	External ADCSOC selection register	0x04	√
SYNCSOCLOCK	Synchronization input source and external ADCSOC selection lock register	0x08	√
SYNCSELECT1	Synchronization input and output selection register 1	0x0C	√
SYNCSELECT2	Synchronization input and output selection register 2	0x10	√
SYNCSELECT5	Synchronization input and output selection register 5	0x1C	√

30.8 Register functional description

30.8.1 Time Base Control Register (TBCTL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	CTRMODE	R/W	Counter Mode Select TBCTR will not change the operating mode after normal operation. After changing the TBCTR operating mode, it will take effect at the next TBCLK edge, and TBCTR will increment/decrement from the value before the change. 00: Count-up mode 01: Count-down mode 10: Center-aligned mode 11: Freeze count mode (default)	3h
2	PHSEN	R/W	Phase Enable 0: Disable 1: Enable, and allow the TBPRD register to load shadow register synchronization events (when software forced synchronization signal or PWMx synchronization input signal occurs)	0h
3	PRDLD	R/W	Time-Base Period Load Mode Select 0: Shadow mode. When TBCTR = 0 and/or PRDLDSYNC = 01, the TBPRD register loads the value of shadow register, and reading/writing the TBPRD register actually accesses the shadow register 1: Immediate mode. Immediately bypass the shadow register, and reading/writing the TBPRD register actually accesses the active register	0h
5:4	SYNCOSEL	R/W	Sync Output Source Select 00: PWMx sync input or SWFSYNC 01: TBCTR = 0 10: TBCTR = CMPB 11: PWMx sync output determined by TBCTL2[SYNCOSELX] bit	0h
6	SWFSYNC	R/S	Sync Pulse Forced by Software This bit only affects PWMx sync output when SYNCOUT = 00. 0: Invalid, reading returns 0 1: Forcibly generate a sync pulse	0h
9:7	HSPCLKDIV	R/W	High Speed Time Base Clock Pre-Scale Factor TBCLK = PWMCLK / (HSPCLKDIV * CLKDIV) 000: The frequency division factor is 1 001: The frequency division factor is 2 (default) 010: The frequency division factor is 4 011: The frequency division factor is 6	1h

Field	Name	R/W	Description	Reset value
			100: The frequency division factor is 8 101: The frequency division factor is 10 110: The frequency division factor is 12 111: The frequency division factor is 14	
12:10	CLKDIV	R/W	Timer Base Clock Pre-Scale Factor Configure TBCLK = PWMCLK / (HSPCLKDIV * CLKDIV) 000: The frequency division factor is 1 (default) 001: The frequency division factor is 2 010: The frequency division factor is 4 011: The frequency division factor is 8 100: The frequency division factor is 16 101: The frequency division factor is 32 110: The frequency division factor is 64 111: The frequency division factor is 128	0h
13	PHSDIR	R/W	Count Direction Configure After a sync event occurs, the new value of the TBPOF register is loaded into TBCTR, and the count direction of TBCTR is configured by this bit, regardless of the direction of TBCTR before the sync event. This bit only applies when TBCTR is configured as center-aligned counting mode, so this bit is ignored in count-up and count-down modes. 0: Count down (after sync event) 1: Count up (after sync event)	0h
15:14	FREE_SOFT	R/W	Emulation Mode Configure These fields configure the behavior of the PWM time base counter during simulation events. 00: The time base counter stops after the next increment or decrement 01: The time base counter stops after completing the entire cycle (1) Count up mode: Stops when TBCTR equals the period (2) Count down mode: Stops when TBCTR = 0x00 (3) Center-aligned count mode: Stops when TBCTR = 0x00 1x: Free running	0h

30.8.2 Time Base Control Register 2 (TBCTL2)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
5:0	Reserved			0h
6	OSHTSYNCMODE	R/W	Oneshot Sync Mode Enable 0: Disable	0h

Field	Name	R/W	Description	Reset value
			1: Enable	
7	OSHTSYNC	R/S	Oneshot Sync Pulse 0: Invalid, reading returns 0 1: Allow single synchronization pulse	0h
11:8	Reserved			0h
13:12	SYNCOSELX	R/W	Extended SYNCOUT Select 00: Disable PWMx sync output signal 01: Sync output signal is CMPC 10: Sync output signal is CMPD 11: Reserved	0h
15:14	PRDLDSYNC	R/W	Period Load Condition on SYNC event Select This field is valid only when PRDLD = 0. When the following conditions are met, the value of the TBPRD shadow register is loaded into the active register: 00: Only when TBCTR = 0 01: When TBCTR = 0 or a sync event occurs 10: Only when a sync event occurs 11: Reserved	0h

30.8.3 Time Base Counter Register (TBCTR)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	TBCTR	R/W	Time Base Counter	0h

30.8.4 Time Base Status Register (TBSTS)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CTRDIR	R	Count Direction Flag 0: Count down in progress 1: Count up in progress	1h
1	SYNCI	R_W1C	External Synchronization Event Flag Write 1 to clear the latched event. 0: Write 0 is invalid, and no external sync event occurred 1: An external sync event occurred	0h
2	CTRMAX	R_W1C	Time-base Counter Reached the Max Value Flag Write 1 to clear the latched event. 0: TBCTR has never reached the max value 1: TBCTR has reached the max value 0xFFFF	0h
15:3	Reserved			0h

30.8.5 Counter-Compare Control Register (CMPCTL)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	LOADAMODE	R/W	Counter-Compare A (CMPA) Load Event Select This field is invalid when SHDWAMODE = 1. When the following conditions are met, the value of the shadow register is loaded into the active CMPA register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load operation (freeze)	0h
3:2	LOADBMODE	R/W	Counter-Compare B (CMPB) Load Event Select This field is invalid when SHDWBMODE = 1. When the following conditions are met, the value of the shadow register is loaded into the active CMPB register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load operation (freeze)	0h
4	SHDWAMODE	R/W	Counter-Compare A (CMPA) Load Mode 0: Immediate mode. For immediate compare mode, all write and read operations can directly access the active register. This only applies to the active compare register A. 1: Shadow mode. In this mode, all write operations by the CPU will access the shadow register. CMPA operates in double-buffer mode.	0h
5	Reserved			0h
6	SHDWBMODE	R/W	Counter-Compare B (CMPB) Load Mode 0: Immediate mode. For immediate compare mode, all write and read operations can directly access the active register. This only applies to the active compare register B. 1: Shadow mode. In this mode, all write operations by the CPU will access the shadow register. CMPB operates in double-buffer mode.	0h
7	Reserved			0h
8	SHDWAFULL	R	CMPA Shadow Full Status This bit will be automatically cleared when a load pulse occurs. Condition for setting this bit: (1) If a 32-bit write operation or a 16-bit write operation is performed on the CMPA register, this bit will be set. (2) If a 16-bit write operation is performed on the CMPA register, this bit will not be set.	0h

Field	Name	R/W	Description	Reset value
			0: CMPA shadow register is not full 1: CMPA shadow register is full, and a CPU write operation will overwrite the current shadow value	
9	SHDWBFULL	R	CMPB Shadow Full Status This bit will be automatically cleared when a load pulse occurs. 0: CMPB shadow register is not full 1: CMPB shadow register is full, and a CPU write operation will overwrite the current shadow value	0h
11:10	LOADASYNC	R/W	CMPA Load Condition Configure This bit is only valid when SHDWAMODE = 0. When a sync event occurs, if the following conditions are met, the CMPA value can be loaded from the shadow register to the active register: 00: Load when the LOADAMODE bit is set 01: Load when the LOADAMODE bit is set and a sync event occurs 10: Load only when a sync event is received 11: Reserved	0h
13:12	LOADBSYNC	R/W	CMPB Load Condition Configure This bit is only valid when SHDWBMODE = 0. When a sync event occurs, if the following conditions are met, the CMPB value can be loaded from the shadow register to the active register: 00: Load when the LOADBMODE bit is set 01: Load when the LOADBMODE bit is set and a sync event occurs 10: Load only when a sync event is received 11: Reserved	0h
15:14	Reserved			0h

30.8.6 Counter-Compare Control Register 2 (CMPCTL2)

Offset address: 0x12

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	LOADCMODE	R/W	Counter-Compare C (CMPC) Load Event Select This field is invalid in immediate mode. When the following conditions are met, load the value of the shadow register into the active CMPC register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load operation (freeze)	0h
3:2	LOADDMODE	R/W	Counter-Compare D (CMPD) Load Event Select This field is invalid in immediate mode. When the following conditions are met, load the value of the shadow register into the active CMPD register:	0h

Field	Name	R/W	Description	Reset value
			00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load operation (freeze)	
4	SHDWCMODE	R/W	Counter-Compare C (CMPC) Load Mode 0: Immediate mode. For immediate compare mode, all write and read operations can directly access the active register. This only applies to the active compare register C. 1: Shadow mode. In this mode, all write operations by the CPU will access the shadow register. CMPC operates in double-buffer mode.	0h
5	Reserved			0h
6	SHDWDMODE	R/W	Counter-Compare D (CMPD) Load Mode 0: Immediate mode. For immediate compare mode, all write and read operations can directly access the active register. This only applies to the active compare register D. 1: Shadow mode. In this mode, all write operations by the CPU will access the shadow register. CMPD operates in double-buffer mode.	0h
9:7	Reserved			0h
11:10	LOADCSYNC	R/W	CMPC Load Condition Configure This bit is only valid when SHDWCMODE = 0. When a sync event occurs, if the following conditions are met, the CMPC value can be loaded from the shadow register to the active register: 00: Load when the LOADCMODE bit is set 01: Load when the LOADCMODE bit is set and a sync event occurs 10: Load only when a sync event is received 11: Reserved	0h
13:12	LOADDSYNC	R/W	CMPD Load Condition Configure This bit is only valid when SHDWDMODE = 0. When a sync event occurs, if the following conditions are met, the CMPD value can be loaded from the shadow register to the active register: 00: Load when the LOADDMODE bit is set 01: Load when the LOADDMODE bit is set and a sync event occurs 10: Load only when a sync event is received 11: Reserved	0h
15:14	Reserved			0h

30.8.7 Dead-Band Generator Control Register (DBCTL)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	OUT_MODE	R/W	Dead-Band Output Select [0]: Control S0 switch. [1]: Control S1 switch 00: Ignore or completely disable DBM. In this case, the POLSEL bit and IN_MODE bit are invalid. 01: Path A equals the delayed bypass of the A signal path (InA), while Path B equals the falling edge delay in the B signal path. 10: Path A equals the rising edge delay in the A signal path, while Path B equals the delayed bypass of the B signal path (InB). 11: Fully enable DBM, with rising edge delay and falling edge delay in active status.	0h
3:2	POLSEL	R/W	Invert Delayed Signal Select [2]: Control S2 switch. [3]: Control S3 switch. This field allows inversion of the PWMA delayed signal or PWMB delayed signal before it is transmitted to the dead-band generator submodule. OUT_MODE=11 and IN_MODE=00 correspond to typical up/down switch control discovered on a branch of a digital motor control inverter. Other enhanced modes are not typical usage modes. 00: High level active (AH) mode (default). No inversion is applied to PWMA delayed signal and PWMB delayed signal. 01: Low level active complementary (ALC) mode. Inversion is applied only to the PWMA delayed signal. 10: High level active complementary (AHC) mode. Inversion is applied only to the PWMB delayed signal. 11: Low level active (AL) mode. Inversion is applied to both PWMA delayed signal and PWMB delayed signal.	0h
5:4	IN_MODE	R/W	Dead-Band Input Select [4]: Control S4 switch. [5]: Control S5 switch. This field can select the input source for rising edge delay and falling edge delay. By default, the input source for rising edge delay and falling edge delay is PWMxA In to generate the typical dead-band waveform. 00: The input source for rising edge delay and falling edge delay is PWMxA In 01: The input source for falling edge delay is PWMxA In, while the input source for rising edge delay is PWMxB In 10: The input source for rising edge delay is PWMxA In, while the input source for falling edge delay is PWMxB In 11: The input source for rising edge delay and falling edge delay is PWMxB In	0h

Field	Name	R/W	Description	Reset value
7:6	LOADREDMODE	R/W	Rising Edge Load Event Select This field is invalid in immediate mode. When the following conditions are met, it will be loaded from the shadow register into the active DBRED register: 00: Load when the counter equals 0 01: Load when the counter equals the period 10: Load when the counter equals 0 or the counter equals the period 11: Disable load (freeze mode)	0h
9:8	LOADFEDMODE	R/W	Falling Edge Load Event Select This field is invalid in immediate mode. When the following conditions are met, it will be loaded from the shadow register into the active DBFED register: 00: Load when the counter equals 0 01: Load when the counter equals the period 10: Load when the counter equals 0 or the counter equals the period 11: Disable load (freeze mode)	0h
10	SHDWDBREDMODE	R/W	RED Dead-Band Load Mode 0: Immediate mode. To perform the falling edge delay dead-band operation immediately, all write or read operations by CPU can directly access the active register. This only applies to the active DBRED register. 1: Shadow mode (default). All write operations through the CPU will access the shadow register. It will operate in a double-buffer manner.	0h
11	SHDWDBFEDMODE	R/W	FED Dead-Band Load Mode 0: Immediate mode. To perform the falling edge delay dead-band operation immediately, all write or read operations by CPU can directly access the active register. This only applies to the active DBFED register. 1: Shadow mode (default). All write operations through the CPU will access the shadow register. It will operate in a double-buffer manner.	0h
13:12	OUTSWAP	R/W	Dead Band Output Swap Control [12]: Control S6 switch. [13]: Control S7 switch. 00: PWMxA and PWMxB output signals are defined by the OUT_MODE field 01: PWMxA output signal is defined by the OUT_MODE field as path A. The PWMxB output signal is defined by the OUT_MODE field as Path A, which equals the rising edge delay of the A signal path or delayed bypass. 10: The PWMxA output signal is defined by the OUT_MODE field as Path B, which equals the falling	0h

Field	Name	R/W	Description	Reset value
			edge delay of the B signal path or delayed bypass. The PWMxB output signal is defined by the OUT_MODE field as Path B. 11: The PWMxA output signal is defined by the OUT_MODE field as Path B, which equals the falling edge delay of the B signal path or delayed bypass. The PWMxB output signal is defined by the OUT_MODE field as Path A, which equals the rising edge delay of the A signal path or delayed bypass.	
14	DEDB_MODE	R/W	Dead Band Dual-Edge Delay B Mode Control This bit controls the S8 switch. When this bit is set, the user can set OUT_MODE = 01 (make Path A equal to InA) or set OUTSWAP = 1x (make PWMxA output signal equals Path B); otherwise, PWMxA is invalid. 0: The rising edge delay is used for the input source selected by the A signal path DBCTL[4] bit (S4 switch) as InA/InB, and falling edge delay is used for the input source selected by the B signal path DBCTL[5] bit (S5 switch) as InA/InB 1: The rising edge delay and falling edge delay are used for the input source selected by the DBCTL[4] bit (S4 switch) and to output to the B signal path	0h
15	HALFCYCLE	R/W	Half Cycle Clocking Enable 0: Enable full cycle clock, at which point the dead-band counter clock is TBCLK 1: Enable half cycle clock, at which point the dead-time counter clock is 2 * TBCLK	0h

30.8.8 Dead-Band Generator Control Register 2 (DBCTL2)

Offset address: 0x1A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	LOADDBCTLMODE	R/W	DBCTL Load Event Select This field is invalid in immediate mode. 00: Load when the counter equals 0 01: Load when the counter equals the period 10: Load when the counter equals 0 or the counter equals the period 11: Disable load (freeze mode)	0h
2	SHDWDBCTLMODE	R/W	DBCTL Load Mode 0: Immediate mode. All write or read operations through the CPU can directly access the active register. This only applies to the active DBCTL register. 1: Shadow mode. Except for all write and read operations to DBCTL[5:0], which are transmitted to the shadow register, other fields will still access the active registers.	0h

Field	Name	R/W	Description	Reset value
15:3	Reserved			0h

30.8.9 Action Qualifier Control Register (AQCTL)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	LDAQAMODE	R/W	Action Qualifier A Load Event Select This field is invalid in immediate mode. When the following conditions are met, it will be loaded from the shadow register into the active AQCTLA register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load (freeze mode)	0h
3:2	LDAQBMODE	R/W	Action Qualifier B Load Event Select This field is invalid in immediate mode. When the following conditions are met, it will be loaded from the shadow register into the active AQCTLB register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Disable load (freeze mode)	0h
4	SHDWAQAMODE	R/W	Action Qualifier A Load Mode 0: Immediate mode. In this mode, all writes or read operations can directly access the active registers. This only applies to the active action qualifier registers. 1: Shadow mode. All write operations are performed by the CPU through accessing the shadow register. AQCTLA register is used for double buffer operations.	0h
5	Reserved			0h
6	SHDWAQBMODE	R/W	Action Qualifier B Load Mode 0: Immediate mode. In this mode, all writes or read operations can directly access the active registers. This only applies to the active action qualifier registers. 1: Shadow mode. All write operations are performed by the CPU through accessing the shadow register. AQCTLB register is used for double buffer operations.	0h
7	Reserved			0h
9:8	LDAQASYNC	R/W	AQCTLA Load Condition Configure This bit is only valid when SHDWAQAMODE=1.	0h

Field	Name	R/W	Description	Reset value
			Under the following conditions, control when the AQCTLA register loads the shadow register to the active register operation: 00: Load when the LDAQAMODE bit is set 01: Load when the LDAQAMODE bit is set and a sync event occurs 10: Load when a sync event is received 11: Reserved	
11:10	LDAQBSYNC	R/W	AQCTLB Load Condition Configure This bit is only valid when SHDWAQBMODE=1. Under the following conditions, control when the AQCTLB register loads the shadow register to the active register operation: 00: Load when the LDAQBMODE bit is set 01: Load when the LDAQBMODE bit is set and a sync event occurs 10: Load when a sync event is received 11: Reserved	0h
15:12	Reserved			0h

30.8.10 Select Action Qualifier Trigger Event Source Register (AQTSRCSEL)

Offset address: 0x22

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	T1SEL	R/W	T1 Event Source Configure 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: PWMx synchronization input 1000: DCEVTFILT Others: Reserved	0h
7:4	T2SEL	R/W	T2 Event Source Configure 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: PWMx synchronization input 1000: DCEVTFILT	0h

Field	Name	R/W	Description	Reset value
			Others: Reserved	
15:8	Reserved			0h

30.8.11 PWM Chopper Control Register (PCCTL)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CHPEN	R/W	Chopping Function Enable 0: Disable 1: Enable	0h
4:1	OSHTWTH	R/W	One-Shot Pulse Width Configure 0000: PWMCLK / 8 width* 1; when PWMCLK = 100MHz, the one-shot pulse width is 80ns 0001: PWMCLK / 8 width* 2; when PWMCLK = 100MHz, the one-shot pulse width is 160ns 0010: PWMCLK / 8 width* 3; when PWMCLK = 100MHz, the one-shot pulse width is 240ns 0011: PWMCLK / 8 width* 4; when PWMCLK = 100MHz, the one-shot pulse width is 320ns 0100: PWMCLK / 8 width* 5; when PWMCLK = 100MHz, the one-shot pulse width is 400ns 0101: PWMCLK / 8 width* 6; when PWMCLK = 100MHz, the one-shot pulse width is 480ns 0110: PWMCLK / 8 width* 7; when PWMCLK = 100MHz, the one-shot pulse width is 560ns 0111: PWMCLK / 8 width* 8; when PWMCLK = 100MHz, the one-shot pulse width is 640ns 1000: PWMCLK / 8 width* 9; when PWMCLK = 100MHz, the one-shot pulse width is 720ns 1001: PWMCLK / 8 width* 10; when PWMCLK = 100MHz, the one-shot pulse width is 800ns 1010: PWMCLK / 8 width* 11; when PWMCLK = 100MHz, the one-shot pulse width is 880ns 1011: PWMCLK / 8 width* 12; when PWMCLK = 100MHz, the one-shot pulse width is 960ns 1100: PWMCLK / 8 width* 13; when PWMCLK = 100MHz, the one-shot pulse width is 1040ns 1101: PWMCLK / 8 width* 14; when PWMCLK = 100MHz, the one-shot pulse width is 1120ns 1110: PWMCLK / 8 width* 15; when PWMCLK = 100MHz, the one-shot pulse width is 1200ns 1111: PWMCLK / 8 width* 16; when PWMCLK = 100MHz, the one-shot pulse width is 1280ns	0h
7:5	CHPFREQ	R/W	Chopping Clock Prescaler 000: The frequency division factor is 1; when TBCLK = 100MHz, the chopping clock is 12.5MHz 001: The frequency division factor is 2; when TBCLK = 100MHz, the chopping clock is 6.25MHz	0h

Field	Name	R/W	Description	Reset value
			010: The frequency division factor is 3; when TBCLK = 100MHz, the chopping clock is 4.16MHz 011: The frequency division factor is 4; when TBCLK = 100MHz, the chopping clock is 3.12MHz 100: The frequency division factor is 5; when TBCLK = 100MHz, the chopping clock is 2.50MHz 101: The frequency division factor is 6; when TBCLK = 100MHz, the chopping clock is 2.08MHz 110: The frequency division factor is 7; when TBCLK = 100MHz, the chopping clock is 1.78MHz 111: The frequency division factor is 8; when TBCLK = 100MHz, the chopping clock is 1.56MHz	
10:8	CHPDUTY	R/W	Chopping Clock Duty Cycle Configure 000: The duty cycle is 12.5% 001: The duty cycle is 25% 010: The duty cycle is 37.5% 011: The duty cycle is 50% 100: The duty cycle is 62.5% 101: The duty cycle is 75% 110: The duty cycle is 87.5% 111: Reserved	0h
15:11	Reserved			0h

30.8.12 Valley Capture Control Register (VCAPCTL)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	VCAPE	R/W	Valley Capture Enable 0: Disable 1: Enable	0h
1	VCAPSTART	R/W	Valley Capture Start When the TRIGSEL bit is selected under software trigger conditions, writing 1 to this bit can trigger a capture sequence via the software. 0: Invalid 1: When TRIGSEL = 000, trigger a capture sequence once	0h
4:2	TRIGSEL	R/W	Capture Sequence Trigger Event Select The valley capture sequence is triggered through the event selected by these fields. If the selected event occurs, the capture sequence is triggered immediately, and retriggered regardless of the current capture state. Capture is performed based on the event selected by the DCFCTL[SRCSEL] field. 000: Trigger the capture sequence through software writing to the TRIGSEL bit 001: TBCTR=0 event	0h

Field	Name	R/W	Description	Reset value
			010: TBCTR=TBPRD event 011: TBCTR=0 or TBCTR=TBPRD event 100: DCAEVT1 event 101: DCAEVT2 event 110: DCBEVT1 event 111: DCBEVT2 event Note: The TRIGSEL bit and SRCSEL bit cannot select the same event simultaneously.	
6:5	Reserved			0h
9:7	VDELAYDIV	R/W	Valley Delay Mode Divide Configure These fields can be used to adjust the delay value between consecutive edge captures. 000: Hardware valley delay value = Software valley delay value 001: Hardware valley delay value = Valley base counter + Software valley delay value 010: Hardware valley delay value = Valley base counter shifted right by 1 bit + Software valley delay value 011: Hardware valley delay value = Valley base counter shifted right by 2 bits + Software valley delay value 100: Hardware valley delay value = Valley base counter shifted right by 4 bits + Software valley delay value Others: Reserved	0h
10	EDGEFILTDLYSEL	R/W	Edge Filter Output Delay 0: No delay 1: Edge filter output HWVDELVAL delay	0h
15:11	Reserved			0h

30.8.13 Valley Counter Configuration Register (VCNTCFG)

Offset address: 0x32

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	STARTEDGE	R/W	Start Count Edge Select When the counter operation is activated and the trigger event selected by the TRIGSEL bit occurs, these fields select the edge at which the valley counter begins counting. 0000: Counting has not started 0001: Start counting at the 1st edge 0010: Start counting at the 2nd edge 0011: Start counting at the 3rd edge ... 1111: Start counting at the 15th edge	0h
6:4	Reserved			0h

Field	Name	R/W	Description	Reset value
7	STARTEDGESTS	R	Start Count Edge Status When the trigger event selected by the TRIGSEL bit occurs, this bit is reset. If a capture sequence is triggered and the edge to start counting occurs, this bit is set. 0: The edge to start counting has not occurred 1: The edge to start counting has occurred	0h
11:8	STOPEDGE	R/W	Stop Count Edge Select When the counter operation is activated and the trigger event selected by the TRIGSEL bit occurs, these fields select the edge at which the valley counter stops counting. 0000: Counting has not stopped 0001: Stop counting at the 1st edge 0010: Stop counting at the 2nd edge 0011: Stop counting at the 3rd edge ... 1111: Stop counting at the 15th edge	0h
14:12	Reserved			0h
15	STOPEDGESTS	R	Stop Count Edge Status When the trigger event selected by the TRIGSEL bit occurs, this bit is reset. If a capture sequence is triggered and the edge to stop counting occurs, this bit is set. 0: The edge to stop counting has not occurred 1: The edge to stop counting occurred	0h

30.8.14 HRPWM Configuration Register (HRCNFG)

Offset address: 0x40

Reset type: SYSRSn

This register only applies to PWM modules with HRPWM function.

Field	Name	R/W	Description	Reset value
1:0	EDGMODE	R/W	PWMxA Edge Mode Select This field is used to select the edge of path A signal controlled by HRP logic. 00: Disable HRPWM function (default) 01: HRP controls the rising edge of path A signal, and this rising edge is controlled by CMPAHR 10: HRP controls the falling edge of path A signal, and this falling edge is controlled by CMPAHR 11: HRP controls both the rising and falling edges of path A signal, and both edges are controlled by TBPRDHR register or TBPHSHR, respectively	0h
2	CTLMODE	R/W	PWMxA Control Mode Select This bit is used to select the register that controls the edge position of path A signal	0h

Field	Name	R/W	Description	Reset value
			0: Period or duty cycle control mode (default), where the edge position is controlled by TBPRDHR register or CMPAHR 1: Phase control mode, where the edge position is controlled by TBPHSHR	
4:3	HRLOAD	R/W	PWMxA Shadow Value Load Event Select This field is used to select the time event to load the value of CMPAHR to the active register. When the following conditions are met, load the value of CMPAHR to the active register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Reserved	0h
5	SELOUTB	R/W	PWMxB Output Invert Configure This bit configures the output signal type of the PWMxB channel. When an inversion operation is performed, high-resolution mode is considered, and the inverted PWMxB signal will include any changes made in high-resolution mode. 0: Path B outputs the normal signal 1: Path B outputs the signal opposite to path A	0h
6	AUTOCONV	R/W	Auto Convert Enable This bit is used to select whether the period, fraction duty cycle or phase in the TBPHSHR, TBPRDHR, or CMPAHR registers will be scaled manually through software calculation or scaled automatically using the HRP scaling factor in the HRMSTEP register. The SFO library function will calculate the appropriate HRP scaling factor based on actual needs and automatically update it to the HRMSTEP register. When manually scaling the fraction duty cycle or phase in the application software, automatic conversion must be disabled. 0: Disable 1: Enable	0h
7	SWAPAB	R/W	PWM A & B Output Signals Swap This bit can exchange the PWMxA and PWMxB output signals. 0: The PWMxA and PWMxB output signals remain unchanged 1: Exchange the PWMxA and PWMxB output signals	0h
9:8	EDGMODEB	R/W	PWMxB Edge Mode Select This field is used to select the edge of path B signal controlled by HRP logic. 00: Disable HRPWM function (default) 01: HRP controls the rising edge of path B signal, and this rising edge is controlled by CMPBHR register	0h

Field	Name	R/W	Description	Reset value
			10: HRP controls the falling edge of path B signal, and this falling edge is controlled by CMPBHR register 11: HRP controls both the rising and falling edges of path B signal, and both edges are controlled by TBPRDHR register or TBPHSHR register	
10	CTLMODEB	R/W	PWMxB Control Mode Select This bit is used to select the register that controls the edge position of path B signal 0: Period or duty cycle control mode (default), where the edge position is controlled by TBPRDHR register or CMPBHR register 1: Phase control mode, where the edge position is controlled by TBPHSHR register	0h
12:11	HRLOADB	R/W	PWMxB Shadow Value Load Event Select This field is used to select the time event to load the value of CMPBHR to the active register. When the following conditions are met, load the value of CMPBHR to the active register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Reserved	0h
15:13	Reserved			0h

30.8.15 Enable HRP Calibration Logic Register (HRPWR)

Offset address: 0x42

Reset type: SYSRSn

This register only applies to PWM modules with HRPWM function.

Field	Name	R/W	Description	Reset value
14:0	Reserved			0h
15	CALPWRON	R/W	HRP Calibration Logic Enable This bit only applies to PWM1. Power consumption can be reduced by disabling HRP calibration logic. 0: Disable 1: Enable	0h

30.8.16 HRP Step Count Register (HRMSTEP)

Offset address: 0x4C

Reset type: SYSRSn

This register allows only 16-bit access and only applies to PWM modules with HRPWM function. An error with the debugger will be caused in 32-bit access mode.

Field	Name	R/W	Description	Reset value
7:0	HRMSTEP	R/W	Number of HRP Steps	0h

Field	Name	R/W	Description	Reset value
			Under the condition of enabling automatic conversion, this field automatically converts the values in the CMPAHR, CMPBHR, DBFEDHR, DBREDHR, TBPRDHR, or TBPHSHR registers into scaled micro-edge delays on the HRPWM output. At the end of each calibration, the SFO calibration software will write the appropriate value to this field.	
15:8	Reserved			0h

30.8.17 HRPWM configuration register 2 (HRCNFG2)

Offset address: 0x4E

Reset type: SYSRSn

This register allows only 16-bit access and only applies to PWM modules with HRPWM function. An error with the debugger will be caused in 32-bit access mode.

Field	Name	R/W	Description	Reset value
1:0	EDGMODEDB	R/W	Dead Band Edge Mode Select This field is used to select the dead-band edge controlled by HRP logic. 00: Disable HRPWM function (default) 01: HRP controls the rising edge of the dead band, and this rising edge is controlled by DBREDHR register 10: HRP controls the falling edge of the dead band, and this falling edge is controlled by PWM_DBREDHR register 11: HRP controls both the rising and falling edges of the dead band, and both edges are controlled by DBREDHR register and DBFEDHR register, respectively.	0h
3:2	CTLMODEDBRED	R/W	DBREDHR Load Event Select This field is used to select the time event to load the value of DBREDHR to the active register. When the following conditions are met, load the value of DBREDHR to the active register: 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Reserved	0h
5:4	CTLMODEDBFED	R/W	DBFEDHR Load Event Select This field is used to select the time event to load the value of DBFEDHR to the active register. When the following conditions are met, load the value of DBFEDHR to the active register: 00: Load when TBCTR = 0	0h

Field	Name	R/W	Description	Reset value
			01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Reserved	
15:6	Reserved			0h

30.8.18 High-Resolution Period Control Register (HRPCTL)

Offset address: 0x5A

Reset type: SYSRSn

This register only applies to PWM modules with HRPWM function.

Field	Name	R/W	Description	Reset value
0	HRPE	R/W	High Resolution Period Enable To use the HRPWM period function, the CTRMODE bit must be set to a mode other than the count-down mode. 0: Disabled; In this case, the PWM type is similar to type 4 PWM 1: Enabled; HRPWM can simultaneously control the high-resolution duty cycle and frequency	0h
1	PWMSYNCSEL	R/W	PWMSYNCPER Source Select This bit selects the PWMSYNCPER signal source used to drive the COMP and GPDAC modules. 0: TBCTR = TBPRD as the PWMSYNCPER signal source 1: TBCTR = 0 as the PWMSYNCPER signal source	0h
2	TBPHSHRLOADE	R/W	High Resolution Phase Synchronization Enable This bit allows high-resolution phase synchronization of PWM module through a sync input, software forced sync pulse, or digital compare event, enabling multiple PWM modules of the same frequency to be phase-aligned at high resolution. TBCTL[PHSEN] bit and TBPHSHRLOADE bit can work independently. To use high-resolution phase and period control function, both TBCTL[PHSEN] bit and TBPHSHRLOADE bit must be set. When high-resolution period is in center-aligned count mode, if TBPHSHR is 0, both TBCTL[PHSEN] bit and TBPHSHRLOADE bit must be set. If only high-resolution duty cycle is enabled, this bit does not need to be set. 0: Disable 1: Enabled; synchronization occurs using the contents of the TBPHSHR register.	0h
3	Reserved			0h
6:4	PWMSYNCSELX	R/W	Extended Signal Select	0h

Field	Name	R/W	Description	Reset value
			000: PWMSYNCSEL bit defines the PWMSYNCPER signal (default) 100: The value of the counter equals the value of CMPC register and counts up 101: The value of the counter equals the value of CMPC register and counts down 110: The value of the counter equals the value of CMPD register and counts up Others: Reserved	
15:7	Reserved			0h

30.8.19 HRPWM Conversion High-resolution Remainder Register (TRREM)

Offset address: 0x5C

Reset type: SYSRSn

This register only applies to PWM modules with HRPWM function. When synchronization input, software forced sync pulse or digital compare events are enabled, the low 8 bits of this register can be automatically initialized with TBPHSHR value, or users can also write a value through CPU. In the asymmetrical mode, if TRREM[10] is not used, it can be forced to 0.

Priority for updating TRREM register:

(3) Transmit the TBPHSHR value to the TRREM register through software or hardware synchronization

(4) Update the TRREM register through HRPWM hardware

(5) Write into the TRREM register through CPU

Field	Name	R/W	Description	Reset value
10:0	TRREM	R/W	Translator Remainder Portion These fields are used to continue tracking the remainder portion of the HRPWM algorithm calculation, which is used to continue tracking the remainder portion of the hardware calculation. In the count-up mode, TRREM is initialized to 0. In the center-aligned count mode, TRREM is initialized to 256. In the asymmetrical mode, TRREM[7:0] = TBPHSHR[15:8], TRREM[10:8] = 000. In the symmetrical mode, TRREM[7:0] = TBPHSHR[15:8], TRREM[10:8] = 001.	0h
15:11	Reserved			0h

30.8.20 Global PWM Load Control Register (GLDCTL)

Offset address: 0x68

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GLD	R/W	Global Load Event Select This bit is used to control the loading behavior from shadow registers to active registers. 0: All shadow register loading behaviors occur according to their respective specified reload control bits 1: All shadow register loading behaviors are defined by the GLDMODE bit	0h
4:1	GLDMODE	R/W	Global Load Pulse Select This field is used to select when to perform the global load pulse operation during the shadow register to active register reload mode. 0000: Load when the counter equals 0 0001: Load when the counter equals the period 0010: Load when the counter equals 0 or the counter equals the period 0011: Load on SYNCEVT signal, where SYNCEVT signal is the logical OR operation of PWMx synchronization input, software forced synchronization pulse, DCAEVT1.sync, and DCBEVT1.sync 0100: Load on SYNCEVT signal or when the counter equals 0 0101: Load on SYNCEVT signal or when the counter equals the period 0110: Load on SYNCEVT signal, when the counter equals 0 or the counter equals the period 1111: Load when the GFRCLD bit is written Others: Reserved	0h
5	OSHTMODE	R/W	One Shot Load Mode Enable Single-shot mode is only used under the condition of GLD = 1. 0: Disabled, all selected global load pulses will continuously load from shadow registers to active registers 1: Enabled, all global load pulses are blocked until OSHTLD = 1	0h
6	Reserved			0h
9:7	GLDPRD	R/W	Global Load Strobe Event Select This field is used to select when to generate the global load pulse after a specific number of events occur 000: Disable counter, no pulse generated 001: Generate pulse when GLDCNT = 001 010: Generate pulse when GLDCNT = 010 011: Generate pulse when GLDCNT=011 100: Generate pulse when GLDCNT=100 101: Generate pulse when GLDCNT=101 110: Generate pulse when GLDCNT=110	0h

Field	Name	R/W	Description	Reset value
			111: Generate pulse when GLDCNT=111	
12:10	GLDCNT	R	The Number of Global Load Events Select This field selects the number of selected events that have occurred. 000: No event occurred 001: 1 event occurred 010: 2 events occurred 011: 3 events occurred 100: 4 events occurred 101: 5 events occurred 110: 6 events occurred 111: 7 events occurred	0h
15:13	Reserved			0h

30.8.21 Global PWM Load Configuration Register (GLDCFG)

Offset address: 0x6A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TBPRD_TBPRDHR	R/W	TBPRD_TBPRDHR Global Load Event Configure (TBPRD_TBPRDHR Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
1	CMPA_CMPAHR	R/W	CMPA_CMPAHR Global Load Event Configure (CMPA_CMPAHR Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
2	CMPB_CMPBHR	R/W	CMPB_CMPBHR Global Load Event Configure (CMPB_CMPBHR Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
3	CMPC	R/W	CMPC Global Load Event Configure (CMPC Global Load Event Configure)	0h

Field	Name	R/W	Description	Reset value
			0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	
4	CMPD	R/W	CMPD Global Load Event Configure (CMPD Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
5	DBRED_DBREDHR	R/W	DBRED_DBREDHR Global Load Event Configure (DBRED_DBREDHR Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
6	DBFED_DBFEDHR	R/W	DBFED_DBFEDHR Global Load Event Configure (DBFED_DBFEDHR Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
7	DBCTL	R/W	DBCTL Global Load Event Configure (DBCTL Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
8	AQCTLA_AQCTLA2	R/W	AQCTLA_AQCTLA2 Global Load Event Configure (AQCTLA_AQCTLA2 Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
9	AQCTLB_AQCTLB2	R/W	AQCTLB_AQCTLB2 Global Load Event Configure (AQCTLB_AQCTLB2 Global Load Event Configure)	0h

Field	Name	R/W	Description	Reset value
			0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	
10	AQCSFRC	R/W	AQCSFRC Global Load Event Configure (AQCSFRC Global Load Event Configure) 0: Even when GLDCTL[GLD] = 1, the register will use the local reload configuration 1: When GLDCTL[GLD] = 1 and this bit is set, the register will use the global load configuration	0h
15:11	Reserved			0h

30.8.22 PWMx Link Register (PWMXLINK)

Offset address: 0x70

Reset type: SYSRSn

This register is used to configure the link relationship between PWMx modules, and the default reset value for each PWMx module is different. To prevent unintentional linking of modules, the reset value links each PWMx module to itself.

Field	Name	R/W	Description	Reset value
3:0	TBPRDLINK	R/W	TBPRD and TBPRDHR Link (TBPRD and TBPRDHR Link) This field is used to perform write operations on the selected PWMx module, and selected based on the following bits: Write to the TBPRD: TBPRDHR register and also simultaneously write to the TBPRD_TBPRDHR register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	X
7:4	CMPALINK	R/W	CMPA and CMPAHR Link (CMPA and CMPAHR Link) This field is used to perform write operations on the selected PWMx module, and selected based on the following: Write to the CMPA_CMPAHR register and also simultaneously write to the CMPA and CMPAHR register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	X
11:8	CMPBLINK	R/W	CMPB and CMPBHR Link	X

Field	Name	R/W	Description	Reset value
			(CMPB and CMPBHR Link) This field is used to perform write operations on the selected PWMx module, and selected based on the following: Write to the CMPB_CMPBHR register and also simultaneously write to the CMPB and CMPBHR register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	
15:12	CMPCLINK	R/W	CMPC Link This field is used to perform write operations on the selected PWMx module, and selected based on the following: Write to the CMPC register and also simultaneously write to the CMPC register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	X
19:16	CMPDLINK	R/W	CMPD Link This field is used to perform write operations on the selected PWMx module, and selected based on the following: Write to the CMPD register and also simultaneously write to the CMPD register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	X
27:20	Reserved			0h
31:28	GLDCTL2LINK	R/W	GLDCTL2 Link This field is used to perform write operations on the selected PWMx module, and selected based on the following: Write to the GLDCTL2 register and also simultaneously write to the GLDCTL2 register of the current PWMx. 0000: PWM1 0001: PWM2 ... Until the last PWM instance Others: Reserved	0h

30.8.23 Output A Action Qualifier Control Register (AQCTLA)

Offset address: 0x80

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	ZRO	R/W	Action When TBCTR = 0 Configure 00: No operation; PWMxA output remains unchanged 01: Clear; PWMxA is forced to output low level 10: Set; PWMxA is forced to output high level 11: Flip; PWMxA low level output signal is forced to switch to high level, and high level output signal is forced to switch to low level Note: In the center-aligned count mode, when the counter equals 0, the counter direction is defined as count up	0h
3:2	PRD	R/W	Action When TBCTR =TBPRD Configure Refer to ZRO description	0h
5:4	CAU	R/W	Action When TBCTR = CMPA On Up Count Configure Refer to ZRO description	0h
7:6	CAD	R/W	Action When TBCTR = CMPA On Down Count Configure Refer to ZRO description	0h
9:8	CBU	R/W	Action When TBCTR= CMPB On Up Count Configure Refer to ZRO description	0h
11:10	CBD	R/W	Action When TBCTR = CMPB On Down Count Configure Refer to ZRO description	0h
15:12	Reserved			0h

30.8.24 Output A Additional Action Qualifier Control Register (AQCTLA2)

Offset address: 0x82

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	T1U	R/W	Action when event occurs on T1 in UP-Count Configure 00: No operation; PWMxA output remains unchanged 01: Clear; PWMxA is forced to output low level 10: Set; PWMxA is forced to output high level 11: Flip; PWMxA low level output signal is forced to switch to high level, and high level output signal is forced to switch to low level Note: In the center-aligned count mode, when the counter equals 0, the counter direction is defined as count up	0h
3:2	T1D	R/W	Action when event occurs on T1 in DOWN-Count Configure Refer to T1U description	0h
5:4	T2U	R/W	Action when event occurs on T2 in UP-Count Configure Refer to T1U description	0h
7:6	T2D	R/W	Action when event occurs on T2 in DOWN-Count Configure Refer to T1U description	0h
15:8	Reserved			0h

30.8.25 Output B Action Qualifier Control Register (AQCTLB)

Offset address: 0x84

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	ZRO	R/W	Action When TBCTR = 0 Configure 00: No operation; PWMxB output remains unchanged 01: Clear; PWMxB is forced to output low level 10: Set; PWMxB is forced to output high level 11: Flip; PWMxB low level output signal is forced to switch to high level, and high level output signal is forced to switch to low level Note: In the center-aligned count mode, when the counter equals 0, the counter direction is defined as count up	0h
3:2	PRD	R/W	Action When TBCTR =TBPRD Configure Refer to ZRO description	0h
5:4	CAU	R/W	Action When TBCTR = CMPA On Up Count Configure Refer to ZRO description	0h
7:6	CAD	R/W	Action When TBCTR = CMPA On Down Count Configure Refer to ZRO description	0h
9:8	CBU	R/W	Action When TBCTR = CMPB On Up Count Configure Refer to ZRO description	0h
11:10	CBD	R/W	Action When TBCTR = CMPB On Down Count Configure Refer to ZRO description	0h
15:12	Reserved			0h

30.8.26 Output B Additional Action Qualifier Control Register (AQCTLB2)

Offset address: 0x86

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	T1U	R/W	Action when event occurs on T1 in UP-Count Configure 00: No operation; PWMxB output remains unchanged 01: Clear; PWMxB is forced to output low level 10: Set; PWMxB is forced to output high level 11: Flip; PWMxB low level output signal is forced to switch to high level, and high level output signal is forced to switch to low level Note: In the center-aligned count mode, when the counter equals 0, the counter direction is defined as count up	0h
3:2	T1D	R/W	Action when event occurs on T1 in DOWN-Count Configure Refer to T1U description	0h
5:4	T2U	R/W	Action when event occurs on T2 in UP-Count Configure Refer to T1U description	0h
7:6	T2D	R/W	Action when event occurs on T2 in DOWN-Count Configure Refer to T1U description	0h
15:8	Reserved			0h

30.8.27 Action Qualifier Software Forced Register (AQSFRC)

Offset address: 0x8E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	ACTSFA	R/W	Action When One-Time Software Force A Is Invoked Configure 00: No operation; Path A output remains unchanged 01: Clear; set Path A output to low level 10: Set; set Path A output to high level 11: Flip; switch path A low level output signal to high level, or high level output signal to low level Note: These actions are not affected by the counter direction.	0h
2	OTSFA	R/S	One-Time Software Forced Event on Output A Enable After enabling a one-time software forced event on Path A output, this field will be automatically cleared. The forced event is one-time, and subsequent events on Path A output can override this forced event. 0: Invalid, reading returns 0. 1: Enable	0h
4:3	ACTSFB	R/W	Action When One-Time Software Force B Is Invoked Configure 00: No operation; Path B output remains unchanged 01: Clear; set Path B output to low level 10: Set; set Path B output to high level 11: Flip; switch Path B low level output signal to high level, or high level output signal to low level Note: These actions are not affected by the counter direction.	0h
5	OTSFB	R/S	One-Time Software Forced Event on Output B Enable After enabling a one-time software forced event on Path B output, this field will be automatically cleared. The forced event is one-time, and subsequent events on Path B output can override this forced event. 0: Invalid, reading returns 0 1: Enable	0h
7:6	RLDCSF	R/W	AQCSFRC Reload Mode Select This field is used to select when to load the value of the AQCSFRC shadow register into the active register. 00: Load when TBCTR = 0 01: Load when TBCTR = TBPRD 10: Load when TBCTR = 0 or TBCTR = TBPRD 11: Immediate load, do not use the shadow register, and CPU directly accesses the active register	0h
15:8	Reserved			0h

30.8.28 Action Qualifier Continuous Software Forced Register (AQCSFRC)

Offset address: 0x92

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	CSFA	R/W	Continuous Software Force on Output A Configure Immediate Mode: In this mode, continuous software force takes effect at the next TBCLK edge. Shadow Mode: Configured shadow mode using the RLDCSF bit. In this mode, continuous software force takes effect at the next TBCLK edge after the shadow is loaded into the active register. 00: Disable software force with no effect on Path A output 01: Set Path A output continuous software force to low level 10: Set Path A output continuous software force to high level 11: Disable software force with no effect on Path A output	0h
3:2	CSFB	R/W	Continuous Software Force on Output B Configure Immediate Mode: In this mode, continuous software force takes effect at the next TBCLK edge. Shadow Mode: Configured shadow mode using the RLDCSF bit. In this mode, continuous software force takes effect at the next TBCLK edge after the shadow is loaded into the active register. 00: Disable software force with no effect on Path B output 01: Set Path B output continuous software force to low level 10: Set Path B output continuous software force to high level 11: Disable software force with no effect on Path B output	0h
15:4	Reserved			0h

30.8.29 Dead-Band Generator Rising Edge Delay High-Resolution Register (DBREDHR)

Offset address: 0xA0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
8:0	Reserved			0h
15:9	DBREDHR	R/W	Dead Band Rising Edge Delay High Resolution	0h

30.8.30 Dead-Band Generator Rising Edge Delay Register (DBRED)

Offset address: 0xA2

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
13:0	DBRED	R/W	Dead Band Rising Edge Delay Value	0h
15:14	Reserved			0h

30.8.31 Dead-Band Generator Falling Edge Delay High-Resolution Register (DBFEDHR)

Offset address: 0xA4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
8:0	Reserved			0h

Field	Name	R/W	Description	Reset value
15:9	DBFEDHR	R/W	Dead Band Falling Edge Delay High Resolution	0h

30.8.32 Dead-Band Generator Falling Edge Delay Register (DBFED)

Offset address: 0xA6

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
13:0	DBFED	R/W	Dead Band Falling Edge Delay Value	0h
15:14	Reserved			0h

30.8.33 Time Base Phase Shift Register (TBPHS)

Offset address: 0xC0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	TBPHSHR	R/W	Time Base Phase Offset High Resolution Ignore the lower 8 bits of the TBPHSHR register, ignore write operations, and read operations return 0. The TBPHSHR function must be simulated through the TRREM register, rather than TBPHSHR.	0h
31:16	TBPHS	R/W	Time Base Phase Offset This register is used to adjust the PWM phase, i.e., to set the phase offset between the selected PWM and the time base counter that provides the synchronization input signal. If TBCTL[PHSEN] = 0, TBCTR will not load the phase value and ignore the sync event. If TBCTL[PHSEN] = 1, TBCTR will load the phase value when a sync event occurs (which can be generated by software forced sync or PWMx sync input signal).	0h

30.8.34 High-Resolution Time Base Period Register (TBPRDHR)

Offset address: 0xC4

Reset type: SYSRSn

Ignore the lower 8 bits of this register, ignore write operations, and read operations return 0. This register only applies to PWM modules controlled by high-resolution time base periods. This register is used only when the high-resolution time base period function is enabled.

Field	Name	R/W	Description	Reset value
15:0	TBPRDHR	R/W	Time Base Period High Resolution This register is used to store the upper 8 bits of the high-resolution time base period value, and is not affected by the PRDLD bit. Read and write operations on TBPRDHR are performed on the shadow register.	0h

30.8.35 Time Base Period Register (TBPRD)

Offset address: 0xC6

Reset type: SYSRSn

The shadow register of this register can be enabled/disabled via TBCTL[PRDLD] bit. By default, this register is used to enable the shadow register. .

Field	Name	R/W	Description	Reset value
15:0	TBPRD	R/W	Time Base Period This register determines the TBCTR period, thereby setting the PWM frequency. If TBCTL[PRDLD]=0, the shadow register is enabled, and any read/write will automatically enter into the shadow register. At this time, when TBCTR = 0, the active register is loaded from the shadow register. If TBCTL[PRDLD] = 1, the shadow register is disabled, and any read/write operations will automatically go to the active register. The memory mapping addresses of the active register and shadow register are the same.	0h

30.8.36 Counter Compare Module A Register (CMPA)

Offset address: 0xD4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CMPAHR	R/W	Counter Compare A High Resolution Extension Ignore the lower 8 bits of the CMPAHR register. The upper 8 bits of this register are used to store and operate the high-resolution portion of the counter compare A value. The CMPA register can be accessed via a single 32-bit read/write operation. Like the CMPA register, the shadow register can be enabled via the SHDWAMODE bit. For details, refer to the CMPA register description.	0h
31:16	CMPA	R/W	Counter Compare A The CMPA register is used for continuous comparison with TBCTR. If the two values are equal, the counter compare module will generate an event with TBCTR equal to count compare module A. This event is sent to the action qualifier for qualification and conversion into one or more actions. The configuration of the AQCTLA and AQCTLB registers determines whether these actions can be applied to PWMxA or PWMxB output signals. The actions that can be defined by the AQCTLA and AQCTLB registers are as follows: No operation: Ignore the event with TBCTR equal counter compare module A Clear: Set PWMxA or PWMxB output signal to low level Set: Set PWMxA or PWMxB output signal to high level Flip: Flip the state of the output signal of PWMxA or PWMxB Enable the CMPA shadow register via the SHDWAMODE bit, with the shadow mode enabled by default. When SHDWAMODE = 0, enable the CMPA shadow register; in this case, any write or read operation will automatically enter the shadow register. In this case, the	0h

Field	Name	R/W	Description	Reset value
			LOADAMODE bit can determine when the shadow register is loaded into the active register. When SHDWAMODE = 1, the CMPA shadow register is disabled; in this case, any write or read operation will directly enter the active register. Before writing, the current status of the CMPA shadow register can be checked using the SHDWAFULL bit to determine if it is full. In both modes, the memory mapping addresses of the active register and shadow register are the same.	

30.8.37 Counter Compare Module B Register (CMPB)

Offset address: 0xD8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CMPBHR	R/W	Counter Compare B High Resolution Extension Ignore the lower 8 bits of the CMPBHR register.	0h
31:16	CMPB	R/W	Counter Compare B The CMPB register is used for continuous comparison with TBCTR. If the two values are equal, the counter compare module will generate an event with TBCTR equal to count compare module B. This event is sent to the action qualifier for qualification and conversion into one or more actions. The configuration of the AQCTLA and AQCTLB registers determines whether these actions can be applied to PWMxA or PWMxB output signals. The actions that can be defined by the AQCTLA and AQCTLB registers are as follows: No operation: Ignore the event with TBCTR equal counter compare module A Clear: Set PWMxA or PWMxB output signal to low level Set: Set PWMxA or PWMxB output signal to high level Flip: Flip the state of the output signal of PWMxA or PWMxB Enable the CMPB shadow register via the SHDWBMODE bit, with the shadow mode enabled by default. When SHDWBMODE = 0, enable the CMPB shadow register; in this case, any write or read operation will automatically enter the shadow register. In this case, the LOADBMODE bit can determine when the shadow register is loaded into the active register. When SHDWBMODE = 1, the CMPB shadow register is disabled; in this case, any write or read operation will directly enter the active register. Before writing, the current status of the CMPB shadow register can be checked using the SHDWBFULL bit to determine if it is full. In both modes, the memory mapping addresses of the active register and shadow register are the same.	0h

30.8.38 Counter Compare Module C Register (CMPC)

Offset address: 0xDE

Reset type: SYSRSn

The linking function of this register is always 16-bit access.

Field	Name	R/W	Description	Reset value
15:0	CMPC	R/W	Counter Compare C The CMPC register is used for continuous comparison with TBCTR. If the two values are equal, the counter compare module will generate an event with TBCTR equal to count compare C. Enable the CMPC shadow register via the SHDWCMODE bit, with the shadow mode enabled by default. When SHDWCMODE = 0, enable the CMPC shadow register; in this case, any write or read operation will automatically enter the shadow register. In this case, the LOADCMODE bit can determine when the shadow register is loaded into the active register. When SHDWCMODE = 1, disable the CMPC shadow register; in this case, any write or read operation will directly enter the active register. In both modes, the memory mapping addresses of the active register and shadow register are the same.	0h

30.8.39 Counter Compare Module D Register (CMPD)

Offset address: 0xE2

Reset type: SYSRSn

The linking function of this register is always 16-bit access.

Field	Name	R/W	Description	Reset value
15:0	CMPD	R/W	Counter Compare D The CMPD register is used for continuous comparison with TBCTR. If the two values are equal, the counter compare module will generate an event with TBCTR equal to count compare D. Enable the CMPD shadow register via the SHDWDCMODE bit, with the shadow mode enabled by default. When SHDWDMODE = 0, enable the CMPD shadow register; in this case, any write or read operation will automatically enter the shadow register. In this case, the LOADDMODE bit can determine when the shadow register is loaded into the active register. When SHDWDMODE = 1, disable the CMPD shadow register; in this case, any write or read operation will directly enter the active register. In both modes, the memory mapping addresses of the active register and shadow register are the same.	0h

30.8.40 Global PWM Load Control Register 2 (GLDCTL2)

Offset address: 0xE8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	OSHTLD	R/S	Reload Event in One Shot Mode Enable (Reload Event in One Shot Mode Enable) This bit allows a load pulse event and prevents subsequent events. 0: Invalid, reading returns 0 1: Enable one-shot latch condition. When the selected load pulse occurs, the shadow-to-activity reload will occur and the latch will be cleared.	0h
1	GFRCLD	R/S	Force Load Event in One Shot Mode Enable (Force Load Event in One Shot Mode Enable) This bit is used to load events in global load mode through testing or software force. 0: Invalid, reading returns 0 1: Force a load event at the input position of the event prescale counter.	0h
15:2	Reserved			0h

30.8.41 Software Valley Mode Delay Register (SWVDELVAL)

Offset address: 0xEE

Reset type: SYSRSn

By setting the value of this register, users can define the hardware calculated delay value according to the VDELAYDIV bit.

Field	Name	R/W	Description	Reset value
15:0	SWVDELVAL	R/W	Software Valley Delay Value (Software Valley Delay Value)	0h

30.8.42 Trip Zone Select Register (TZSEL)

Offset address: 0x100

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CBC1	R/W	TZ1 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ1. 0: No 1: Yes	0h
1	CBC2	R/W	TZ2 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ2. 0: No 1: Yes	0h
2	CBC3	R/W	TZ3 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ3. 0: No 1: Yes	0h

Field	Name	R/W	Description	Reset value
3	CBC4	R/W	TZ4 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ4. 0: No 1: Yes	0h
4	CBC5	R/W	TZ5 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ5. 0: No 1: Yes	0h
5	CBC6	R/W	TZ6 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is TZ6. 0: No 1: Yes	0h
6	DCAEVT2	R/W	DCAEVT2 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is DCAEVT2. 0: No 1: Yes	0h
7	DCBEVT2	R/W	DCBE2 as a CBC source Select This bit is used to select whether the cycle-by-cycle trip source is DCBE2. 0: No 1: Yes	0h
8	OSHT1	R/W	TZ1 as a OST source Select This bit is used to select whether the one-shot trip source is TZ1. 0: No 1: Yes	0h
9	OSHT2	R/W	TZ2 as a OST source Select This bit is used to select whether the one-shot trip source is TZ2. 0: No 1: Yes	0h
10	OSHT3	R/W	TZ3 as a OST source Select This bit is used to select whether the one-shot trip source is TZ3. 0: No 1: Yes	0h
11	OSHT4	R/W	TZ4 as a OST source Select This bit is used to select whether the one-shot trip source is TZ4. 0: No 1: Yes	0h
12	OSHT5	R/W	TZ5 as a OST source Select	0h

Field	Name	R/W	Description	Reset value
			This bit is used to select whether the one-shot trip source is TZ5. 0: No 1: Yes	
13	OSHT6	R/W	TZ6 as a OST source Select This bit is used to select whether the one-shot trip source is TZ6. 0: No 1: Yes	0h
14	DCAEVT1	R/W	DCAEVT1 as a OSTsource Select This bit is used to select whether the one-shot trip source is DCAEVT1. 0: No 1: Yes	0h
15	DCBEVT1	R/W	DCBE1 as a OST source Select This bit is used to select whether the one-shot trip source is DCBE1. 0: No 1: Yes	0h

30.8.43 Trip Zone Digital Comparator Configuration Register (TZDCSEL)

Offset address: 0x104

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	DCAEVT1	R/W	Digital Compare Output A Event 1 Configure 000: Disable digital comparator A output event 1 001: Trigger when the DCBH signal is low level 010: Trigger when the DCBH signal is high level 011: Trigger when the DCBL signal is low level 100: Trigger when the DCBL signal is high level 101: Trigger when the DCBL signal is high level, and when the DCBH signal is low level 110: Reserved 111: Reserved	0h
5:3	DCAEVT2	R/W	Digital Compare Output A Event 2 Configure Refer to DCAEVT1 description	0h
8:6	DCBEVT1	R/W	Digital Compare Output B Event 1 Configure Refer to DCAEVT1 description	0h
11:9	DCBEVT2	R/W	Digital Compare Output B Event 2 Configure Refer to DCAEVT1 description	0h
15:12	Reserved			0h

30.8.44 Trip Zone Control Register (TZCTL)

Offset address: 0x108

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	TZA	R/W	Trip Event Occurs PWMxA Action Configure This field defines the action taken on PWMxA output when a trip event occurs. The TZSEL register can determine which trip zone pins can trigger the event. 00: PWMxA is high impedance 01: PWMxA is forced to output high level 010: PWMxA is forced to output low level 11: PWMxA remains in its original state	0h
3:2	TZB	R/W	Trip Event Occurs PWMxB Action Configure This field defines the action taken on PWMxB output when a trip event occurs. The TZSEL register can determine which trip zone pins can trigger the event. 00: PWMxB is high impedance 01: PWMxB is forced to output high level 10: PWMxB is forced to output low level 11: PWMxB remains in its original state	0h
5:4	DCAEVT1	R/W	Digital Compare Output A Event 1 Action On PWMxA Configure 00: PWMxA is high impedance 01: PWMxA is forced to output high level 010: PWMxA is forced to output low level 11: Disable trip action	0h
7:6	DCAEVT2	R/W	Digital Compare Output A Event 2 Action On PWMxA Configure Refer to DCAEVT1 description	0h
9:8	DCBEVT1	R/W	Digital Compare Output B Event 1 Action On PWMxA Configure 00: PWMxB is high impedance 01: PWMxB is forced to output high level 10: PWMxB is forced to output low level 11: Disable trip action	0h
11:10	DCBEVT2	R/W	Digital Compare Output B Event 1 Action On PWMxA Configure Refer to DCBEVT1 description	0h
15:12	Reserved			0h

30.8.45 Trip Zone Control Register 2 (TZCTL2)

Offset address: 0x10A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	TZAU	R/W	Trip Event Occurs PWMxA Action while Count direction is UP Configure 000: PWMxA outputs high impedance 001: PWMxA is forced to output high level 010: PWMxA is forced to output low level 011: Flip PWMxA output state	0h

Field	Name	R/W	Description	Reset value
			100: Reserved 101: Reserved 110: Reserved 111: Disable trip action	
5:3	TZAD	R/W	Trip Event Occurs PWMxA Action while Count direction is Down Configure Refer to TZAU description	0h
8:6	TZBU	R/W	Trip Event Occurs PWMxB Action while Count direction is UP Configure 000: PWMxB outputs high impedance 001: PWMxB is forced to output high level 010: PWMxB is forced to output low level 011: Flip PWMxB output state 100: Reserved 101: Reserved 110: Reserved 111: Disable trip action	0h
11:9	TZBD	R/W	Trip Event Occurs PWMxB Action while Count direction is Down Configure Refer to TZBU description	0h
14:12	Reserved			0h
15	ETZE	R/W	Enable TZCTL2 (TZCTL2 Enable) 0: Trip action defined by TZCTL register 1: Trip action defined by TZCTL2, TZCTLDCA, and TZCTLDCB registers, ignoring TZCTL register settings.	0h

30.8.46 Trip Zone Control Digital Comparator A Register (TZCTLDCA)

Offset address: 0x10C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	DCAEVT1U	R/W	Digital Compare Output A Event 1 Action On PWMxA Configure while Count direction is UP Configure 000: PWMxA outputs high impedance 001: PWMxA is forced to output high level 010: PWMxA is forced to output low level 011: Flip PWMxA output state 100: Reserved 101: Reserved 110: Reserved 111: Disable trip action	0h
5:3	DCAEVT1D	R/W	Digital Compare Output A Event 1 Action On PWMxA Configure while Count direction is Down Configure Refer to DCAEVT1U description	0h
8:6	DCAEVT2U	R/W	Digital Compare Output A Event 1 Action On PWMxA Configure while Count direction is UP Configure Refer to DCAEVT1U description	0h

Field	Name	R/W	Description	Reset value
11:9	DCAEVT2D	R/W	Digital Compare Output A Event 2 Action On PWMxA Configure while Count direction is Down Configure Refer to DCAEVT1U description	0h
15:12	Reserved			0h

30.8.47 Trip Zone Control Digital Comparator B Register (TZCTLDCB)

Offset address: 0x10E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	DCBEVT1U	R/W	Digital Compare Output B Event 1 Action On PWMxB Configure while Count direction is UP Configure 000: PWMxB outputs high impedance 001: PWMxB is forced to output high level 010: PWMxB is forced to output low level 011: Flip PWMxB output state 100: Reserved 101: Reserved 110: Reserved 111: Disable trip action	0h
5:3	DCBEVT1D	R/W	Digital Compare Output B Event 1 Action On PWMxB Configure while Count direction is Down Configure Refer to DCBEVT1U description	0h
8:6	DCBEVT2U	R/W	Digital Compare Output B Event 1 Action On PWMxB Configure while Count direction is UP Configure Refer to DCBEVT1U description	0h
11:9	DCBEVT2D	R/W	Digital Compare Output B Event 2 Action On PWMxB Configure while Count direction is Down Configure Refer to DCBEVT1U description	0h
15:12	Reserved			0h

30.8.48 Trip Zone Interrupt Enable Register (TZEINT)

Offset address: 0x11A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	CBC	R/W	Trip-zone Cycle-by-Cycle Interrupt Enable A cycle-by-cycle trip event will cause an NVIC interrupt. 0: Disable 1: Enable	0h
2	OST	R/W	Trip-zone One-Shot Interrupt Enable A one-shot trip event will cause an NVIC interrupt. 0: Disable 1: Enable	0h

Field	Name	R/W	Description	Reset value
3	DCAEVT1	R/W	Digital Compare Output A Event 1 Interrupt Enable 0: Disable 1: Enable	0h
4	DCAEVT2	R/W	Digital Compare Output A Event 2 Interrupt Enable 0: Disable 1: Enable	0h
5	DCBEVT1	R/W	Digital Compare Output B Event 1 Interrupt Enable 0: Disable 1: Enable	0h
6	DCBEVT2	R/W	Digital Compare Output B Event 2 Interrupt Enable 0: Disable 1: Enable	0h
15:7	Reserved			0h

30.8.49 Trip Zone Flag Register (TZFLG)

Offset address: 0x126

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R	Trip Interrupt Latched Status An NVIC interrupt will not be generated again until this bit is cleared. Setting this bit before setting the CBC or OST bit will generate another interrupt pulse. Clearing all status flags will prevent further interrupts. This bit can be cleared by writing the appropriate value to the TZCLR register. 0: No interrupt occurs 1: NVIC interrupt is generated due to trip condition	0h
1	CBC	R	Cycle-By-Cycle Trip Event Latched Status If this bit is set, it will remain set until it is manually cleared. When this bit is cleared but a cycle-by-cycle trip event still exists, this bit will be immediately set again. When TBCTR = 0, if the trip condition no longer exists, the signal condition will be automatically cleared. Signal conditions can only be cleared when TBCTR = 0, regardless of the position within the period. This bit can be cleared by writing the appropriate value to the TZCLR register. 0: No cycle-by-cycle trip event occurred 1: A cycle-by-cycle trip event occurred, caused by selecting a signal as the cycle-by-cycle trip source	0h
2	OST	R	A One-Shot Trip Event Latched Status This bit can be cleared by writing the appropriate value to the TZCLR register. 0: No shot trip event occurred 1: A one-shot trip event occurred on the one-shot trip source pin	0h
3	DCAEVT1	R	Digital Compare Output A Event 1 Latched Status	0h

Field	Name	R/W	Description	Reset value
			0: No trip event occurred in DCAEVT1 1: A trip event occurred in DCAEVT1	
4	DCAEVT2	R	Digital Compare Output A Event 2 Latched Status 0: No trip event occurred in DCAEVT2 1: A trip event occurred in DCAEVT2	0h
5	DCBEVT1	R	Digital Compare Output B Event 1 Latched Status 0: No trip event occurred in DCBEVT1 1: A trip event occurred in DCBEVT1	0h
6	DCBEVT2	R	Digital Compare Output B Event 2 Latched Status 0: No trip event occurred in DCBEVT2 1: A trip event occurred in DCBEVT2	0h
15:7	Reserved			0h

30.8.50 Trip Zone CBC Flag Register (TZCBCFLG)

Offset address: 0x128

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CBC1	R	CBC1 Latch Flag 0: No trip event occurred in CBC1 1: A trip event occurred in CBC1	0h
1	CBC2	R	CBC2 Latch Flag 0: No trip event occurred in CBC2 1: A trip event occurred in CBC2	0h
2	CBC3	R	CBC3 Latch Flag 0: No trip event occurred in CBC3 1: A trip event occurred in CBC3	0h
3	CBC4	R	CBC4 Latch Flag 0: No trip event occurred in CBC4 1: A trip event occurred in CBC4	0h
4	CBC5	R	CBC5 Latch Flag 0: No trip event occurred in CBC5 1: A trip event occurred in CBC5	0h
5	CBC6	R	CBC6 Latch Flag 0: No trip event occurred in CBC6 1: A trip event occurred in CBC6	0h
6	DCAEVT2	R	Digital Compare A Output Event 2 Trip Latch Flag 0: No trip event occurred in DCAEVT2 1: A trip event occurred in DCAEVT2	0h
7	DCBEVT2	R	Digital Compare B Output Event 2 Trip Latch Flag 0: No trip event occurred in DCBEVT2 1: A trip event occurred in DCBEVT2	0h
15:8	Reserved			0h

30.8.51 Trip Zone OST Flag Register (TZOSTFLG)

Offset address: 0x12A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	OST1	R	OST1 Latch Flag 0: No trip event occurred in OST1 1: A trip event occurred in OST1	0h
1	OST2	R	OST2 Latch Flag 0: No trip event occurred in OST2 1: A trip event occurred in OST2	0h
2	OST3	R	OST3 Latch Flag 0: No trip event occurred in OST3 1: A trip event occurred in OST3	0h
3	OST4	R	OST4 Latch Flag 0: No trip event occurred in OST4 1: A trip event occurred in OST4	0h
4	OST5	R	OST5 Latch Flag 0: No trip event occurred in OST5 1: A trip event occurred in OST5	0h
5	OST6	R	OST6 Latch Flag 0: No trip event occurred in OST6 1: A trip event occurred in OST6	0h
6	DCAEVT1	R	Digital Compare A Output Event 1 Latch Flag 0: No trip event occurred in DCAEVT1 1: A trip event occurred in DCAEVT1	0h
7	DCBEVT1	R	Digital Compare B Output Event 1 Latch Flag 0: No trip event occurred in DCBEVT1 1: A trip event occurred in DCBEVT1	0h
15:8	Reserved			0h

30.8.52 Clear Trip Zone Register (TZCLR)

Offset address: 0x12E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R/S	Global Interrupt Flag Clear An NVIC interrupt will not be generated again until the INT bit is cleared. After clearing the INT bit, setting any flag bit will generate another interrupt pulse. Clearing all status flags will prevent further interrupts. 0: Invalid, reading returns 0 1: Clear the INT bit	0h
1	CBC	R/S	Cycle-By-Cycle Trip Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC trip condition	0h

Field	Name	R/W	Description	Reset value
2	OST	R/S	One-Shot Trip Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST trip condition	0h
3	DCAEVT1	R/S	Digital Compare A Output Event 1 Latch Flag Clear 0: Invalid, reading returns 0 1. Clear the DCAEVT1 trip condition	0h
4	DCAEVT2	R/S	Digital Compare A Output Event 2 Latch Flag Clear 0: Invalid, reading returns 0 1. Clear the DCAEVT2 trip condition	0h
5	DCBEVT1	R/S	Digital Compare B Output Event 1 Latch Flag Clear 0: Invalid, reading returns 0 1. Clear the DCBEVT1 trip condition	0h
6	DCBEVT2	R/S	Digital Compare B Output Event 2 Latch Flag Clear 0: Invalid, reading returns 0 1. Clear the DCBEVT2 trip condition	0h
13:7	Reserved			0h
15:14	CBCPULSE	R/W	Cycle-By-Cycle Trip Latch Pulse Clear This field is used to choose to clear the CBC latch pulse. 00: Pulse when the counter equals 0 01: Pulse when the counter equals the period 10: Pulse when the counter equals 0 or the counter equals the period 11: Do not clear the CBC latch	0h

30.8.53 Clear Trip Zone CBC Register (TZCBCCLR)

Offset address: 0x130

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CBC1	R/S	Cycle-By-Cycle Trip (CBC1) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC1 bit	0h
1	CBC2	R/S	Cycle-By-Cycle Trip (CBC2) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC2 bit	0h
2	CBC3	R/S	Cycle-By-Cycle Trip (CBC3) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC3 bit	0h
3	CBC4	R/S	Cycle-By-Cycle Trip (CBC4) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC4 bit	0h
4	CBC5	R/S	Cycle-By-Cycle Trip (CBC5) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC5 bit	0h

Field	Name	R/W	Description	Reset value
5	CBC6	R/S	Cycle-By-Cycle Trip (CBC6) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the CBC6 bit	0h
6	DCAEVT2	R/S	Digital Compare Output A Event 2 selected for CBC Flag Clear 0: Invalid, reading returns 0 1: Clear the DCAEVT2 bit	0h
7	DCBEVT2	R/S	Digital Compare Output B Event 2 selected for CBC Flag Clear 0: Invalid, reading returns 0 1: Clear the DCBEVT2 bit	0h
15:8	Reserved			0h

30.8.54 Clear Trip Zone OST Register (TZOSTCLR)

Offset address: 0x132

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	OST1	R/S	One-Shot Trip (OST1) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST1 bit	0h
1	OST2	R/S	One-Shot Trip (OST2) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST2 bit	0h
2	OST3	R/S	One-Shot Trip (OST3) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST3 bit	0h
3	OST4	R/S	One-Shot Trip (OST4) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST4 bit	0h
4	OST5	R/S	One-Shot Trip (OST5) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST5 bit	0h
5	OST6	R/S	One-Shot Trip (OST6) Latch Flag Clear 0: Invalid, reading returns 0 1: Clear the OST6 bit	0h
6	DCAEVT1	R/S	Digital Compare Output A Event 1 selected for OST Flag Clear 0: Invalid, reading returns 0 1: Clear the DCAEVT1 bit	0h
7	DCBEVT1	R/S	Digital Compare Output B Event 1 selected for OST Flag Clear 0: Invalid, reading returns 0 1: Clear the DCBEVT1 bit	0h
15:8	Reserved			0h

30.8.55 Force Trip Zone Flag Register (TZFLG)

Offset address: 0x136

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	CBC	R/S	Force a Cycle-by-Cycle Trip Event via Software 0: Invalid, reading returns 0 1: Force to trigger a cycle-by-cycle trip event and set the CBC bit	0h
2	OST	R/S	Force a One-Shot Trip Event via Software 0: Invalid, reading returns 0 1: Force to trigger a one-shot trip event and set the OST bit	0h
3	DCAEVT1	R/S	Digital Compare Output A Event 1 Forced Flag 0: Invalid, reading returns 0 1: Force to trigger the DCAEVT1 trip condition and set the DCAEVT1 bit	0h
4	DCAEVT2	R/S	Digital Compare Output A Event 2 Forced Flag 0: Invalid, reading returns 0 1: Force to trigger the DCAEVT2 trip condition and set the DCAEVT2 bit	0h
5	DCBEVT1	R/S	Digital Compare Output B Event 1 Forced Flag 0: Invalid, reading returns 0 1: Force to trigger the DCBE1 trip condition and set the DCBEVT1 bit	0h
6	DCBEVT2	R/S	Digital Compare Output B Event 2 Forced Flag 0: Invalid, reading returns 0 1: Force to trigger the DCBE2 trip condition and set the DCBEVT2 bit	0h
15:7	Reserved			0h

30.8.56 Event Trigger Select Register (ETSEL)

Offset address: 0x148

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	INTSEL	R/W	PWM Interrupt Select This field is used to select the conditions for generating the PWM interrupt event. 000: Reserved 001: Generate when TBCTR = 0 010: Generate when TBCTR = TBPRD 011: Generate when TBCTR = 0 or TBCTR = TBPRD (used for center-aligned count mode) 100: Generate when the counter counts up, and TBCTR = CMPA or TBCTR = CMPC 101: Generate when the counter counts down, and TBCTR = CMPA or TBCTR = CMPC	0h

Field	Name	R/W	Description	Reset value
			110: Generate when the counter counts up, and TBCTR = CMPB or TBCTR = CMPD 111: Generate when the counter counts down, and TBCTR=CMPB or TBCTR=CMPD	
3	INTEN	R/W	PWM Interrupt Generation Enable 0: Disable 1: Enable	0h
4	SOCASELCMP	R/W	PWMxSOCA Compare Select The event selected for this bit will be processed through the SSOCASEL bit. 0: Enable the event when the counter counts up, and TBCTR = CMPA or TBCTR = CMPB. Enable the event when the counter counts down, and TBCTR = CMPA or TBCTR = CMPB 1: Enable the event when the counter counts up, and TBCTR = CMPC or TBCTR = CMPD. Enable the event when the counter counts down, and TBCTR = CMPC or TBCTR = CMPD	0h
5	SOCBSELCMP	R/W	PWMxSOCB Compare Select The event selected for this bit will be processed through the SOCBSEL bit. 0: Enable the event when the counter counts up, and TBCTR = CMPA or TBCTR = CMPB. Enable the event when the counter counts down, and TBCTR = CMPA or TBCTR = CMPB 1: Enable the event when the counter counts up, and TBCTR = CMPC or TBCTR = CMPD. Enable the event when the counter counts down, and TBCTR = CMPC or TBCTR = CMPD	0h
6	INTSELCMP	R/W	PWMxINT Compare Select The event selected for this bit will be processed through the INTSEL bit. 0: Enable the event when the counter counts up, and TBCTR = CMPA or TBCTR = CMPB. Enable the event when the counter counts down, and TBCTR = CMPA or TBCTR = CMPB 1: Enable the event when the counter counts up, and TBCTR = CMPC or TBCTR = CMPD. Enable the event when the counter counts down, and TBCTR = CMPC or TBCTR = CMPD	0h
7	Reserved			0h
10:8	SOCASEL	R/W	Generate PWMxSOCA Event Select This field is used to select the conditions for generating the PWMxSOCA event. Select the PWMxSOCA comparator event based on the SOCASELCMP bit. 000: Generate when the DCAEVT1.soc event occurs 001: Generate when TBCTR = 0 010: Generate when TBCTR = TBPRD	0h

Field	Name	R/W	Description	Reset value
			011: Generate when TBCTR = 0 or TBCTR = TBPRD (used for center-aligned count mode) 100: Generate when the counter counts up, and TBCTR = CMPA or TBCTR = CMPC 101: Generate when the counter counts down, and TBCTR = CMPA or TBCTR = CMPC 110: Generate when the counter counts up, and TBCTR = CMPB or TBCTR = CMPD 111: Generate when the counter counts down, and TBCTR=CMPB or TBCTR=CMPD	
11	SOCAEN	R/W	Path A Trigger ADC Conversion Pulse Enable (PWMxSOCA) 0: Disable 1: Enable	0h
14:12	SOCBSEL	R/W	Generate PWMxSOCB Event Select This field is used to select the conditions for generating the PWMxSOCB event. Select the PWMxSOCB comparator event based on the SOCBSELCMP bit. 000: Generate when the DCBE1.soc event occurs 001: Generate when TBCTR = 0 010: Generate when TBCTR = TBPRD 011: Generate when TBCTR = 0 or TBCTR = TBPRD (used for center-aligned count mode) 100: Generate when the counter counts up, and TBCTR = CMPA or TBCTR = CMPC 101: Generate when the counter counts down, and TBCTR = CMPA or TBCTR = CMPC 110: Generate when the counter counts up, and TBCTR = CMPB or TBCTR = CMPD 111: Generate when the counter counts down, and TBCTR=CMPB or TBCTR=CMPD	0h
15	SOCBEN	R/W	Path B Trigger ADC Conversion Pulse Enable (PWMxSOCB) 0: Disable 1: Enable	0h

30.8.57 Event Trigger Preallocated Register (ETPS)

Offset address: 0x14C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	INTPRD	R/W	PWM Interrupt Period Select The field is used to determine the conditions before generating an interrupt. ETSEL[INT] must be set to 1 to generate an interrupt, and no interrupt will be generated before the ETFLG[INT] flag bit is cleared without using the ETSEL[INT] bit. If an interrupt is generated, the INTECNT bit will be automatically cleared.	0h

Field	Name	R/W	Description	Reset value
			00: When INTECNT=00, no interrupt is generated, and the ETFRC[INT] bit is ignored 01: When INTECNT=01, an interrupt is generated on the 1st event 10: When INTECNT=10, an interrupt is generated on the 2nd event 11: When INTECNT=11, an interrupt is generated on the 3rd event	
3:2	INTCNT	R	PWM Interrupt Event Counter The field is used to select the number of INTSEL events that have occurred. The field is automatically cleared when an interrupt pulse is generated. When ETSEL[INT]=0 or ETFLG[INT]=1, the counter stops counting when ETPS[INTCNT] = ETPS[INTPRD]. 00: No INTSEL event occurred 01: 1 INTSEL event occurred 10: 2 INTSEL events occurred 11: 3 INTSEL events occurred	0h
4	INTPSSEL	R/W	PWMxINTn Pre-Scale Select 0: The event frequency is determined by ETPS [INTCNT and INTPRD] bits, generating one pulse every 0...3 events 1: The event frequency is determined by ETINTPS [INTCNT2 and INTPRD2] bits, generating one pulse every 0...15 events	0h
5	SOCPSSEL	R/W	PWMxSOC A/B Pre-Scale Select 0: The event frequency is determined by SOCACNT/SOCBCNT bits and SOCAPRD/SOCBPRD bits, generating one pulse every 0...3 events 1: The event frequency is determined by SOCACNT2/SOCBCNT2 bits and SOCAPRD2/SOCBPRD2 bits, generating one pulse every 0...15 events	0h
7:6	Reserved			0h
9:8	SOCAPRD	R/W	PWM ADC Start-of-Conversion A Event (PWMxSOCA) Period Select This field is used to determine the conditions before generating the PWMxSOCA pulse. ETSEL[SOCAEN] must be set to 1 to generate a pulse. Even if the ETFLG[SOCA] bit is set, a PWMxSOCA pulse will be generated. At this time, the ETPS[SOCACNT] bit will be automatically cleared. 00: When SOCACNT=00, no pulse is generated 01: When SOCACNT=01, a pulse is generated on the 1st event 10: When SOCACNT=10, a pulse is generated on the 2nd event 11: When SOCACNT=11, a pulse is generated on the 3rd event	0h

Field	Name	R/W	Description	Reset value
11:10	SOCACNT	R	PWM ADC Start-of-Conversion A Event (PWMxSOCA) Counter This field is used to select the number of ETSEL[SOCASEL] events that have occurred. 00: No SOCASEL event occurred 01: 1 SOCASEL event occurred 10: 2 SOCASEL events occurred 11: 3 SOCASEL events occurred	0h
13:12	SOCBPRD	R/W	PWM ADC Start-of-Conversion B Event (PWMxSOCB) Period Select This field is used to determine the conditions before generating the PWMxSOCB pulse. ETSEL[SOCBEN] must be set to 1 to generate a pulse. Even if the ETFLG[SOCB] bit is set, a PWMxSOCB pulse will be generated. At this time, the SOCBCNT bit will be automatically cleared. 00: When SOCBCNT=00, no pulse is generated 01: When SOCBCNT=01, a pulse is generated on the 1st event 10: When SOCBCNT=10, a pulse is generated on the 2nd event 11: When SOCBCNT=11, a pulse is generated on the 3rd event	0h
15:14	SOCBCNT	R	PWM ADC Start-of-Conversion B Event (PWMxSOCB) Counter This field is used to select the number of ETSEL[SOCBSEL] events that have occurred. 00: No SOCBSEL event occurred 01: 1 SOCBSEL event occurred 10: 2 SOCBSEL events occurred 11: 3 SOCBSEL events occurred	0h

30.8.58 Event Trigger Flag Register (ETFLG)

Offset address: 0x150

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R	PWM Interrupt (PWMx_INT) Latch Flag The bit must be cleared to generate a new interrupt, and at most one interrupt can be pending when the INT bit is still in set state. If an interrupt is pending, it will only be generated after the INT bit is cleared. 0: No interrupt occurs 1: Generate interrupt	0h
1	Reserved			0h

Field	Name	R/W	Description	Reset value
2	SOCA	R	PWM ADC Start-of-Conversion A (PWMxSOCA) Flag When this bit is set, the PWMxSOCA output continues to generate conversion start pulses. 0: No pulse occurs 1: Pulse occurs	0h
3	SOCB	R	PWM ADC Start-of-Conversion B (PWMxSOCB) Flag When this bit is set, the PWMxSOCB output continues to generate conversion start pulses. 0: No pulse occurs 1: Pulse occurs	0h
15:4	Reserved			0h

30.8.59 Clear Event Trigger Register (ETCLR)

Offset address: 0x154

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R/S	PWM Interrupt (PWMx_INT) Flag Clear 0: Invalid, reading returns 0 1: Clear the INT bit	0h
1	Reserved			0h
2	SOCA	R/S	PWM ADC Start-of-Conversion A (PWMxSOCA) Flag Clear 0: Invalid, reading returns 0 1: Clear the SOCA bit	0h
3	SOCB	R/S	PWM ADC Start-of-Conversion B (PWMxSOCB) Flag Clear 0: Invalid, reading returns 0 1: Clear the SOCB bit	0h
15:4	Reserved			0h

30.8.60 Force Event Trigger Register (ETFRC)

Offset address: 0x158

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R/S	Generate INT Force If the event is enabled in the ETSEL register, an interrupt will be generated. 0: Invalid, reading returns 0 1: Generate an interrupt and set the INT bit	0h
1	Reserved			0h
2	SOCA	R/S	Generates a Pulse on PWMxSOCA Force This bit is used for testing. If the event is enabled in the ETSEL register, a PWMxSOCA pulse will be generated. To determine whether a PWMxSOCA event is generated, the SOCA bit will be set regardless of whether a pulse is generated.	0h

Field	Name	R/W	Description	Reset value
			0: Invalid, reading returns 0 1: Generate a pulse and set the SOCA bit	
3	SOCB	R/S	Generates a Pulse on PWMxSOCB Force This bit is used for testing. If the event is enabled in the ETSEL register, a PWMxSOCB pulse will be generated. To determine whether a PWMxSOCB event is generated, the SOCB bit will be set regardless of whether a pulse is generated. 0: Invalid, reading returns 0 1: Generate a pulse and set the SOCB bit	0h
15:4	Reserved			0h

30.8.61 Event Trigger Interrupt Prescale Register (ETINTPS)

Offset address: 0x15C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	INTPRD2	R/W	PWM Interrupt Period 2 Select When INTPSSEL=1, this field is used to determine the conditions before generating an interrupt. 0000: INTCNT2=0000, no interrupt is generated 0001: INTCNT2=0001, an interrupt is generated on the 1st event 0010: INTCNT2=0010, an interrupt is generated on the 2nd event 0011: INTCNT2=0011, an interrupt is generated on the 3rd event ... 1111: INTCNT2=1111, an interrupt is generated on the 15th event	0h
7:4	INTCNT2	R	PWM Interrupt Event Counter 2 When INTPSSEL=1, this bit is used to select the number of INTSEL events that have occurred. 0000: No INTSEL event occurred 0001: 1 INTSEL event occurred 0010: 2 INTSEL events occurred 0011: 3 INTSEL events occurred ... 1111: 15 INTSEL events occurred	0h
15:8	Reserved			0h

30.8.62 Event Trigger SOC Prescale Register (ETSOCPS)

Offset address: 0x160

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	SOCAPRD2	R/W	PWM ADC Start-of-Conversion A Event (PWMxSOCA) Period 2 Select When SOCPSSEL=1, this field is used to determine the conditions before generating a SOCA pulse. 0000: SOCACNT2=0000, no pulse is generated 0001: SOCACNT2=0001, a SOCA pulse is generated on the 1st event 0010: SOCACNT2=0010, a SOCA pulse is generated on the 2nd event 0011: SOCACNT2=0011, a SOCA pulse is generated on the 3rd event ... 1111: SOCACNT2=1111, a SOCA pulse is generated on the 15th event	0h
7:4	SOCACNT2	R	PWM ADC Start-of-Conversion A Event (PWMxSOCA) Counter 2 When SOCPSSEL=1, this bit is used to select the number of SOCASEL events that have occurred. 0000: No SOCASEL event occurred 0001: 1 SOCASEL event occurred 0010: 2 SOCASEL events occurred 0011: 3 SOCASEL events occurred ... 1111: 15 SOCASEL events occurred	0h
11:8	SOCBPRD2	R/W	PWM ADC Start-of-Conversion B Event (PWMxSOCB) Counter 2 When SOCPSSEL=1, this field is used to determine the conditions before generating a SOCB pulse. 0000: SOCBCNT2=0000, no interrupt is generated 0001: SOCBCNT2=0001, a SOCB pulse is generated on the 1st event 0010: SOCBCNT2=0010, a SOCB pulse is generated on the 2nd event 0011: SOCBCNT2=0011, a SOCB pulse is generated on the 3rd event ... 1111: SOCBCNT2=1111, a SOCB pulse is generated on the 15th event	0h
15:12	SOCBCNT2	R	PWM ADC Start-of-Conversion B Event (PWMxSOCB) Counter 2 When SOCPSSEL=1, this bit is used to select the number of SOCASEL events that have occurred. 0000: No SOCASEL event occurred 0001: 1 SOCASEL event occurred 0010: 2 SOCASEL events occurred 0011: 3 SOCASEL events occurred ... 1111: 15 SOCASEL events occurred	0h

30.8.63 Event Trigger Counter Initialization Control Register (ETCNTINITCTL)

Offset address: 0x164

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	Reserved			0h
10	INTINITFRC	R/S	PWMxINT Counter 2 Initialization Force 0: Invalid 1: Force the PWMxSOCB counter to initialize with the INTINIT content	0h
11	SOCAINITFRC	R/S	PWMxSOCA Counter 2 Initialization Force 0: Invalid 1: Force the PWMxSOCB counter to initialize with the SOCAINIT content	0h
12	SOCBINITFRC	R/S	PWMxSOCB Counter 2 Initialization Force 0: Invalid 1: Force the PWMxSOCB counter to initialize with the SOCBINIT content	0h
13	INTINITEN	R/W	PWMxINT Counter 2 Initialization Enable 0: Invalid 1: Initialize the PWM interrupt counter with INTINITFRC content during a synchronization event or software force	0h
14	SOCAINITEN	R/W	PWMxSOCA Counter 2 Initialization Enable 0: Invalid 1: Initialize the PWMxSOCA counter with SOCAINITFRC content during a synchronization event or software force	0h
15	SOCBINITEN	R/W	PWMxSOCB Counter 2 Initialization Enable 0: Invalid 1: Initialize the PWMxSOCB counter with SOCBINITFRC content during a synchronization event or software force	0h

30.8.64 Event Trigger Counter Initialization Register (ETCNTINIT)

Offset address: 0x168

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	INTINIT	R/W	PWMxINT Counter 2 Initialization When a synchronization event or software force occurs, the content of this register is used to initialize the PWM interrupt counter 2.	0h
7:4	SOCAINIT	R/W	PWMxSOCA Counter 2 Initialization When a synchronization event or software force occurs, the content of this register is used to initialize the PWMxSOCA counter 2.	0h

Field	Name	R/W	Description	Reset value
11:8	SOCBINIT	R/W	PWMxSOCB Counter 2 Initialization When a synchronization event or software force occurs, the content of this register is used to initialize the PWMxSOCB counter 2.	0h
15:12	Reserved			0h

30.8.65 Digital Compare Trip Select Register (DCTRIPSEL)

Offset address: 0x180

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	DCAHCOMPSEL	R/W	Digital Compare A High Input Select 0000: Trip input 1 0001: Trip input 2 0010: Trip input 3 0011: Trip input 4 ... 1100: Reserved 1101: Trip input 14 1110: Reserved 1111: Trip input combination. Logical OR operation is performed on all trip inputs selected through the DCAHTRIPSEL register.	0h
7:4	DCALCOMPSEL	R/W	Digital Compare A Low Input Select Refer to DCAHCOMPSEL description	0h
11:8	DCBHCOMPSEL	R/W	Digital Compare B High Input Select Refer to DCAHCOMPSEL description	0h
15:12	DCBLCOMPSEL	R/W	Digital Compare B Low Input Select Refer to DCAHCOMPSEL description	0h

30.8.66 Digital Compare A Control Register (DCACTL)

Offset address: 0x186

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	EVT1SRCSEL	R/W	DCAEVT1 Source Signal Select 0: DCAEVT1 signal 1: DCEVTFILT signal	0h
1	EVT1FRCSYNCSEL	R/W	DCAE1 Force Synchronization Signal Select 0: Sync with PWMCLK 1: Asynchronous transfer	0h
2	EVT1SOCE	R/W	DCAE1 Generate SOC Enabled 0: Disable 1: Enable	0h
3	EVT1SYNCE	R/W	DCAE1 Generate SYNC Enabled 0: Disable 1: Enable	0h

Field	Name	R/W	Description	Reset value
7:4	Reserved			0h
8	EVT2SRCSEL	R/W	DCAEVT2 Source Signal Select 0: DCAEVT2 signal 1: DCEVTFILT signal	0h
9	EVT2FRCSYNCSEL	R/W	DCAEVT2 Force Synchronization Signal Select 0: Sync with PWMCLK 1: Asynchronous transfer	0h
15:10	Reserved			0h

30.8.67 Digital Compare B Control Register (DCBACTL)

Offset address: 0x188

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	EVT1SRCSEL	R/W	DCBE1 Source Signal Select 0: DCBE1 signal 1: DCEVTFILT signal	0h
1	EVT1FRCSYNCSEL	R/W	DCBE1 Force Synchronization Signal Select 0: Sync with PWMCLK 1: Asynchronous transfer	0h
2	EVT1SOCE	R/W	DCBE1 Generate SOC Enabled 0: Disable 1: Enable	0h
3	EVT1SYNCE	R/W	DCBE1 Generate SYNC Enabled 0: Disable 1: Enable	0h
7:4	Reserved			0h
8	EVT2SRCSEL	R/W	DCBE2 Source Signal Select 0: DCBE2 signal 1: DCEVTFILT signal	0h
9	EVT2FRCSYNCSEL	R/W	DCBE2 Force Synchronization Signal Select 0: Sync with PWMCLK 1: Asynchronous transfer	0h
15:10	Reserved			0h

30.8.68 Digital Compare Filter Control Register (DCFCTL)

Offset address: 0x18E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	SRCSEL	R/W	Filter Block Signal Source Select 00: DCAEVT1 signal 01: DCAEVT2 signal 10: DCBEVT1 signal 11: DCBEVT2 signal	0h

Field	Name	R/W	Description	Reset value
2	BLANKE	R/W	Blanking Window Enable 0: Disable 1: Enable	0h
3	BLANKINV	R/W	Blanking Window Inversion 0: Not reverse 1: Reverse	0h
5:4	PULSESEL	R/W	Blanking & Capture Alignment Pulse Select 00: TBCTR=TBPRD 01: TBCTR=0 10: TBCTR=0 or TBCTR=TBPRD 11: Reserved	0h
6	EDGEFILTSEL	R/W	Edge Filter Select 0: Not select 1: Select	0h
7	Reserved			0h
9:8	EDGEMODE	R/W	Edge Mode Select 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved	0h
12:10	EDGECOUNT	R/W	Edge Count Select This field is used to select the number of edges to be counted before generating a wide pulse on the TBCLK for the DCEVTFILT signal. 000: Not edge is counted 001: 1 edge is counted 010: 2 edges are counted 011: 3 edges are counted 100: 4 edges are counted 101: 5 edges are counted 110: 6 edges are counted 111: 7 edges are counted	0h
15:13	EDGESTATUS	R	Edge Status This field reflects the total number of edges currently captured. When EDGESTATUS matches EDGECOUNT, this bit is set to 0, and a TBCLK wide pulse is generated on the DCEVTFILT signal. The edge counter can be reset by writing 000 to the EDGECOUNT value.	0h

30.8.69 Digital Compare Capture Control Register (DCCAPCTL)

Offset address: 0x190

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	CAPE	R/W	TBCTL Counter Capture Enable 0: Disable	0h

Field	Name	R/W	Description	Reset value
			1: Enable	
1	SHDWMODE	R/W	TBCTR Counter Capture Shadow Mode Select 0: Shadow mode. When TBCTR = 0 or TBCTR = TBPRD (defined by the PULSESEL bit), the value of the DCCAP active register will be copied to the shadow register. At this point, the value read from the DCCAP register will be the shadow register value. 1: Active mode. In this mode, the shadow register is disabled, and the value read from the DCCAP register will always be the active register value.	0h
12:2	Reserved			0h
13	CAPSTS	R	DC Capture Event Latched Flag 0: Not occurred 1: Occurred	0h
14	CAPCLR	R/S	DC Capture Latched Flag Clear 0: Invalid 1: Clear the CAPSTS bit	0h
15	CAPMODE	R/W	Counter Capture Mode 0: When CAPE = 1 and a DCEVTFILT event occurs, the current counter value is captured into the active register. If a trigger event occurs, further trip/capture events will be ignored until the next TBCTR = 0 or TBCTR = TBPRD event (defined by the PULSESEL bit) re-triggers the capture mechanism. Shadow mode: If SHDWMODE = 0, when a TBCTR = 0 or TBCTR = TBPRD event (defined by the PULSESEL bit) occurs, the active register value will be copied to the shadow register, and the CPU will read the value of the register that will return to the shadow register. Active mode: If SHDWMODE = 1, the CPU will read the value of the register that will return to the shadow register. 0: When CAPE = 1 and a DCEVTFILT event occurs, the current counter value is captured into the active register. If a trip event occurs, the CAPSTS bit is set and further trip/capture events will be ignored until this flag bit is cleared. The flag bit can be cleared via the CAPCLR bit, which will re-trigger the capture mechanism. Shadow mode: SHDWMODE = 0; when a TBCTR = 0 or TBCTR = TBPRD event (defined by the PULSESEL bit) occurs, the active register value will be copied to the shadow register, and the CPU will read the value of the register that will return to the shadow register. Active mode: SHDWMODE = 1; the CPU will read the value of the register that will return to the shadow register.	0h

30.8.70 Digital Compare Filter Offset Register (DCFOFFSET)

Offset address: 0x192

Reset type: SYSRSn

This register is the shadow register, and the active register is loaded at the reference point defined by the PULSESEL bit. When the active register is loaded, the blank offset counter is initialized and begins counting down. If the offset counter stops, the blank window is applied. When the blank window is active, the blank window counter is re-enabled to count.

Field	Name	R/W	Description	Reset value
15:0	DCFOFFSET	R/W	Blanking Window Offset This field is used to specify the number of TBCLK cycles from the blank window reference point (TBCTR = 0 or TBCTR = TBPRD, as defined by the PULSESEL bit) to the blank window application point.	0h

30.8.71 Digital Compare Filter Offset Counter Register (DCFOFFSETCNT)

Offset address: 0x194

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DCFOFFSETCNT	R	Blanking Offset Counter These 16-bit fields are used to indicate the current value of the offset counter. The counter will stop counting down when it reaches 0, until the next TBCTR = 0 or TBCTR = TBPRD event (as defined by the PULSESEL bit) reloads the counter. Even if the device is paused due to simulation stop, the counter will continue counting down; therefore, the offset counter is not affected by the free or software simulation bits.	0h

30.8.72 Digital Compare Filter Window Register (DCFWINDOW)

Offset address: 0x196

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DCFWINDOW	R/W	Blanking Window Width 0000: No blank window generated 0001...FFFF: Specifies the blank window width, in TBCLK cycles. The blank window begins when the offset counter reaches the specified value. At this point, the blank window counter is loaded and starts counting down. It is important to avoid the situation where the blank window counter will not restart, and the blank window will end prematurely when the offset counter reaches the specified value and the blank window is currently active. The blank window can temporarily disable the signal during a certain period of the PWM cycle.	0h

30.8.73 Digital Compare Filter Window Counter Register (DCFWINDOWCNT)

Offset address: 0x198

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DCFWINDOWCNT	R	Blanking Window Counter These 16-bit fields are used to indicate the value of the current window counter. The counter stops counting down when it reaches 0, and will be reloaded when the offset counter reaches 0 again.	0h

30.8.74 Digital Compare Counter Capture Register (DCCAP)

Offset address: 0x19E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	DCCAP	R	Digital Compare Time-Base Counter Capture CAPE must be set to 1 to enable TBCTL counter capture. When this feature is enabled, the value of TBCTR will be read on the rising edge of the DCEVTFLT event. Subsequent capture events will be ignored until the next TBCTR = 0 or TBCTR = TBPRD event occurs, as defined by the PULSESEL bit. The shadow mode of the DCCAP register is enabled via the SHDWMODE bit (default). Shadow mode: SHDWMODE = 0; when a TBCTR = 0 or TBCTR = TBPRD event (defined by the PULSESEL bit) occurs, the active register value will be copied to the shadow register, and the CPU will read the value of the register that will return to the shadow register. Active mode: SHDWMODE = 1; the CPU will read the value of the register that will return to the shadow register. The memory mapping addresses of the active register and shadow register are the same.	0h

30.8.75 Digital Compare A High Trip Select Register (DCAHTRIPSEL)

Offset address: 0x1A4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TRIPINPUT1	R/W	TRIP Input 1 0: Not select 1: Select as the input to the DCAH multiplexer for a logic OR operation	0h
1	TRIPINPUT2	R/W	TRIP Input 2 Refer to TRIPINPUT1 description	0h
2	TRIPINPUT3	R/W	TRIP Input 3 Refer to TRIPINPUT1 description	0h
3	TRIPINPUT4	R/W	TRIP Input 4 Refer to TRIPINPUT1 description	0h
4	TRIPINPUT5	R/W	TRIP Input 5 Refer to TRIPINPUT1 description	0h
5	TRIPINPUT6	R/W	TRIP Input 6 Refer to TRIPINPUT1 description	0h

Field	Name	R/W	Description	Reset value
6	TRIPINPUT7	R/W	TRIP Input 7 Refer to TRIPINPUT1 description	0h
7	TRIPINPUT8	R/W	TRIP Input 8 Refer to TRIPINPUT1 description	0h
8	TRIPINPUT9	R/W	TRIP Input 9 Refer to TRIPINPUT1 description	0h
9	TRIPINPUT10	R/W	TRIP Input 10 Refer to TRIPINPUT1 description	0h
10	TRIPINPUT11	R/W	TRIP Input 11 Refer to TRIPINPUT1 description	0h
11	TRIPINPUT12	R/W	TRIP Input 12 Refer to TRIPINPUT1 description	0h
12	Reserved			0h
13	TRIPINPUT14	R/W	TRIP Input 14 Refer to TRIPINPUT1 description	0h
15:14	Reserved			0h

30.8.76 Digital Compare A Low Trip Select Register (DCALTRIPSEL)

Offset address: 0x1A6

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TRIPINPUT1	R/W	TRIP Input 1 0: Not select 1: Select as the input to the DCAL multiplexer for a logic OR operation	0h
1	TRIPINPUT2	R/W	TRIP Input 2 Refer to TRIPINPUT1 description	0h
2	TRIPINPUT3	R/W	TRIP Input 3 Refer to TRIPINPUT1 description	0h
3	TRIPINPUT4	R/W	TRIP Input 4 Refer to TRIPINPUT1 description	0h
4	TRIPINPUT5	R/W	TRIP Input 5 Refer to TRIPINPUT1 description	0h
5	TRIPINPUT6	R/W	TRIP Input 6 Refer to TRIPINPUT1 description	0h
6	TRIPINPUT7	R/W	TRIP Input 7 Refer to TRIPINPUT1 description	0h
7	TRIPINPUT8	R/W	TRIP Input 8 Refer to TRIPINPUT1 description	0h
8	TRIPINPUT9	R/W	TRIP Input 9 Refer to TRIPINPUT1 description	0h
9	TRIPINPUT10	R/W	TRIP Input 10 Refer to TRIPINPUT1 description	0h

Field	Name	R/W	Description	Reset value
10	TRIPINPUT11	R/W	TRIP Input 11 Refer to TRIPINPUT1 description	0h
11	TRIPINPUT12	R/W	TRIP Input 12 Refer to TRIPINPUT1 description	0h
12	Reserved			0h
13	TRIPINPUT14	R/W	TRIP Input 14 Refer to TRIPINPUT1 description	0h
15:14	Reserved			0h

30.8.77 Digital Compare B High Trip Select Register (DCBHTRIPSEL)

Offset address: 0x1A8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TRIPINPUT1	R/W	TRIP Input 1 0: Not select 1: Select as the input to the DCBH multiplexer for a logic OR operation	0h
1	TRIPINPUT2	R/W	TRIP Input 2 Refer to TRIPINPUT1 description	0h
2	TRIPINPUT3	R/W	TRIP Input 3 Refer to TRIPINPUT1 description	0h
3	TRIPINPUT4	R/W	TRIP Input 4 Refer to TRIPINPUT1 description	0h
4	TRIPINPUT5	R/W	TRIP Input 5 Refer to TRIPINPUT1 description	0h
5	TRIPINPUT6	R/W	TRIP Input 6 Refer to TRIPINPUT1 description	0h
6	TRIPINPUT7	R/W	TRIP Input 7 Refer to TRIPINPUT1 description	0h
7	TRIPINPUT8	R/W	TRIP Input 8 Refer to TRIPINPUT1 description	0h
8	TRIPINPUT9	R/W	TRIP Input 9 Refer to TRIPINPUT1 description	0h
9	TRIPINPUT10	R/W	TRIP Input 10 Refer to TRIPINPUT1 description	0h
10	TRIPINPUT11	R/W	TRIP Input 11 Refer to TRIPINPUT1 description	0h
11	TRIPINPUT12	R/W	TRIP Input 12 Refer to TRIPINPUT1 description	0h
12	Reserved			0h
13	TRIPINPUT14	R/W	TRIP Input 14 Refer to TRIPINPUT1 description	0h

Field	Name	R/W	Description	Reset value
15:14	Reserved			0h

30.8.78 Digital Compare B Low Trip Select Register (DCBLTRIPSEL)

Offset address: 0x1AA

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TRIPINPUT1	R/W	TRIP Input 1 0: Not select 1: Select as the input to the DCBL multiplexer for a logic OR operation	0h
1	TRIPINPUT2	R/W	TRIP Input 2 Refer to TRIPINPUT1 description	0h
2	TRIPINPUT3	R/W	TRIP Input 3 Refer to TRIPINPUT1 description	0h
3	TRIPINPUT4	R/W	TRIP Input 4 Refer to TRIPINPUT1 description	0h
4	TRIPINPUT5	R/W	TRIP Input 5 Refer to TRIPINPUT1 description	0h
5	TRIPINPUT6	R/W	TRIP Input 6 Refer to TRIPINPUT1 description	0h
6	TRIPINPUT7	R/W	TRIP Input 7 Refer to TRIPINPUT1 description	0h
7	TRIPINPUT8	R/W	TRIP Input 8 Refer to TRIPINPUT1 description	0h
8	TRIPINPUT9	R/W	TRIP Input 9 Refer to TRIPINPUT1 description	0h
9	TRIPINPUT10	R/W	TRIP Input 10 Refer to TRIPINPUT1 description	0h
10	TRIPINPUT11	R/W	TRIP Input 11 Refer to TRIPINPUT1 description	0h
11	TRIPINPUT12	R/W	TRIP Input 12 Refer to TRIPINPUT1 description	0h
12	Reserved			0h
13	TRIPINPUT14	R/W	TRIP Input 14 Refer to TRIPINPUT1 description	0h
15:14	Reserved			0h

30.8.79 PWM Lock Register (PWMLOCK)

Offset address: 0x1F4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	HRLOCK	R/WO	High Resolution PWM LOCK	0h

Field	Name	R/W	Description	Reset value
			The HRPWM register address mapping range is from 0x40 to 0x5A. 0: HRPWM registers are protected by WRPRT 1: HRPWM registers are locked and cannot be written	
1	GLLOCK	R/WO	Global PWM Load LOCK The global PWM loading register address mapping range is from 0x68 to 0x6A. 0: Global PWM loading registers are protected by WRPRT 1: Global PWM loading registers are locked and cannot be written	0h
2	TZCFGLOCK	R/WO	Trip-zone Configure LOCK The trip zone configuration register address mapping range is from 0x100 to 0x11A. 0: The trip zone configuration registers are protected by WRPRT 1: The trip zone configuration registers are locked and cannot be written	0h
3	TZCLRLOCK	R/WO	Trip-zone Configure LOCK The trip zone clear register address mapping range is from 0x12E to 0x136 0: The trip zone clear registers are protected by WRPRT 1: The trip zone clear registers are locked and cannot be written	0h
4	DCLOCK	R/WO	Digital Compare LOCK The digital comparator register address mapping range is from 0x180 to 0x1B2. 0: The digital compare registers are protected by WRPRT 1: The digital compare registers are locked and cannot be written	0h
15:5	Reserved			0h
31:16	KEY	R0/W	KEY The value of this field must be set to 0xA5A5 in order to be successfully written to this register. When the KEY matches, if a 16-bit write operation is performed on this register, either the [15:0] or [31:16] of the register will be ignored, and only a 32-bit write operation will succeed.	0h

30.8.80 Hardware Valley Mode Delay Register (HWVDELVAL)

Offset address: 0x1FA

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	HWVDELVAL	R	Hardware Valley Delay Value These fields are used to indicate the hardware valley delay value calculated in the VDELAYDIV bit. When the	0h

Field	Name	R/W	Description	Reset value
			valley capture sequence is triggered and the valley capture 1/2 values are updated, the hardware valley delay value will also be updated simultaneously.	

30.8.81 Hardware Valley Counter Register (VCNTVAL)

Offset address: 0x1FC

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	VCNTVAL	R	Valley Time Base Counter These fields are used to indicate that this register records and stores the captured valley counter value when the STOPEDGE bit is selected in the VCNTCFG register.	0h

30.8.82 Synchronous Input and Output Select Register (SYNCSELECT)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	PWM4SYNCIN	R/W	PWM4 Sync Input Source Select 000: PWM1 synchronous output 001: PWM4 synchronous output (reserved) 010: PWM7 synchronous output (reserved) 011: Reserved 100: CAP1 synchronous output (reserved) 101: External synchronous input 1 110: External synchronous input 2 111: Reserved Note: If a reserved position is selected, the output of the sync multiplexer will be driven to a low level.	7h
5:3	PWM7SYNCIN	R/W	PWM7 Sync Input Source Select 000: PWM1 synchronous output 001: PWM4 synchronous output 010: PWM7 synchronous output (reserved) 011: Reserved 100: CAP1 synchronous output (reserved) 101: External synchronous input 1 110: External synchronous input 2 111: CAP4 synchronous output (reserved) Note: If a reserved position is selected, the output of the sync multiplexer will be driven to a low level.	7h
2:0	Reserved			7h
5:3	Reserved			7h
8:6	Reserved			7h
11:9	Reserved			7h
14:12	Reserved			7h

Field	Name	R/W	Description	Reset value
17:15	Reserved			7h
23:18	Reserved			0h
28:24	SYNCOUT	R/W	Sync Output Source Select 00000: PWM1 synchronous output signal 00001: PWM2 synchronous output signal 00010: PWM3 synchronous output signal 00011: PWM4 synchronous output signal 00100: PWM5 synchronous output signal 00101: PWM6 synchronous output signal 00110: PWM7 synchronous output signal 11000: CAP1 synchronous output signal 11001: CAP2 synchronous output signal 11010: CAP3 synchronous output signal Reserved; the default setting is the PWM1 synchronous output signal	0h
31:29	Reserved			7h

30.8.83 External ADCSOC Select Register (ADCSOCOUTSELECT)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PWM1SOCAEN	R/W	PWM1 SOCA Output Source Select 0: Not select 1: Select	0h
1	PWM2SOCAEN	R/W	PWM2 SOCA Output Source Select 0: Not select 1: Select	0h
2	PWM3SOCAEN	R/W	PWM3 SOCA Output Source Select 0: Not select 1: Select	0h
3	PWM4SOCAEN	R/W	PWM4 SOCA Output Source Select 0: Not select 1: Select	0h
4	PWM5SOCAEN	R/W	PWM5 SOCA Output Source Select 0: Not select 1: Select	0h
5	PWM6SOCAEN	R/W	PWM6 SOCA Output Source Select 0: Not select 1: Select	0h
6	PWM7SOCAEN	R/W	PWM7 SOCA Output Source Select 0: Not select 1: Select	0h
15:7	Reserved			0h

Field	Name	R/W	Description	Reset value
16	PWM1SOCBEN	R/W	PWM1 SOCB Output Source Select 0: Not select 1: Select	0h
17	PWM2SOCBEN	R/W	PWM2 SOCB Output Source Select 0: Not select 1: Select	0h
18	PWM3SOCBEN	R/W	PWM3 SOCB Output Source Select 0: Not select 1: Select	0h
19	PWM4SOCBEN	R/W	PWM4 SOCB Output Source Select 0: Not select 1: Select	0h
20	PWM5SOCBEN	R/W	PWM5 SOCB Output Source Select 0: Not select 1: Select	0h
21	PWM6SOCBEN	R/W	PWM6 SOCB Output Source Select 0: Not select 1: Select	0h
22	PWM7SOCBEN	R/W	PWM7 SOCB Output Source Select 0: Not select 1: Select	0h
31:23	Reserved			0h

30.8.84 Synchronous Input Source and External ADCSOC Select Lock Register (SYNCSOCLOCK)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SYNCSELECT0	R/WO	SYNCSELECT0 Lock This bit is used to protect write operations to the SYNCSELECT0 register. If any bit of the SYNCSELECT0 register is set to 1, it can only be cleared via SYSRSn, and writing 0 to any bit of this register will be invalid. The latch mechanism is only used for write operations, but read operations are still allowed. 1: Unlocked 1: Locked	0h
1	ADCSOCOUTSELECT	R/WO	ADCSOCOUTSELECT Lock This bit is used to protect write operations to the ADCSOCOUTSELECT register. If any bit of the ADCSOCOUTSELECT register is set to 1, it can only be cleared via SYSRSn, and writing 0 to any bit of this register will be invalid.	0h

Field	Name	R/W	Description	Reset value
			The latch mechanism is only used for write operations, but read operations are still allowed. 1: Unlocked 1: Locked	
2	SYNCSELECT1	R/WO	SYNCSELECT1 Lock This bit is used to protect write operations to the SYNCSELECT1 register. If any bit of the SYNCSELECT1 register is set to 1, it can only be cleared via SYSRSn, and writing 0 to any bit of this register will be invalid. The latch mechanism is only used for write operations, but read operations are still allowed. 1: Unlocked 1: Locked	0h
3	SYNCSELECT2	R/WO	SYNCSELECT2 Lock This bit is used to protect write operations to the SYNCSELECT2 register. If any bit of the SYNCSELECT2 register is set to 1, it can only be cleared via SYSRSn, and writing 0 to any bit of this register will be invalid. The latch mechanism is only used for write operations, but read operations are still allowed. 1: Unlocked 1: Locked	0h
5:4	Reserved			0h
6	SYNCSELECT5	R/WO	SYNCSELECT5 Lock This bit is used to protect write operations to the SYNCSELECT5 register. If any bit of the SYNCSELECT5 register is set to 1, it can only be cleared via SYSRSn, and writing 0 to any bit of this register will be invalid. The latch mechanism is only used for write operations, but read operations are still allowed. 1: Unlocked 1: Locked	0h
15:7	Reserved			0h

30.8.85 Synchronous Input Select Register 1 (SYNCSELECT1)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	PWM1SYNCIN	R/W	PWM1 Sync Input Source Select 0h: Disabled	0h

Field	Name	R/W	Description	Reset value
			1h: PWM1 SYNCOUT 2h: PWM2 SYNCOUT 3h: PWM3 SYNCOUT 4h: PWM4 SYNCOUT 5h: PWM5 SYNCOUT 6h: PWM6 SYNCOUT 7h: PWM7 SYNCOUT 11h: CAP1 SYNCOUT 12h: CAP2 SYNCOUT 13h: CAP3 SYNCOUT 18h: INPUTXBAROUT5 19h: INPUTXBAROUT6 Configuration retained as for 0h	
7:5	Reserved			0h
12:8	PWM2SYNCIN	R/W	PWM2 Sync Input Source Select See the description for PWM1SYNCIN.	0h
15:13	Reserved			0h
20:16	PWM3SYNCIN	R/W	PWM3 Sync Input Source Select Refer to the description for PWM1SYNCIN.	0h
23:21	Reserved			0h
28:24	PWM4SYNCIN	R/W	PWM4 Sync Input Source Select See the description for PWM1SYNCIN.	0h
31:29	Reserved			0h

30.8.86 Synchronisation Input Select Register 2 (SYNCSELECT2)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	PWM5SYNCIN	R/W	PWM5 Sync Input Source Select See the description for PWM1SYNCIN.	0h
7:5	Reserved			0h
12:8	PWM6SYNCIN	R/W	PWM6 Sync Input Source Select See the description for PWM1SYNCIN.	0h
15:13	Reserved			0h
20:16	PWM7SYNCIN	R/W	PWM7 Sync Input Source Select Refer to the description for PWM1SYNCIN.	0h
31:21	Reserved			0h

30.8.87 Synchronisation Input Select Register 5 (SYNCSELECT5)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	CAP1SYNCIN	R/W	CAP1 Sync Input Source Select 0h: Disabled 1h: PWM1 SYNCOUT 2h: PWM2 SYNCOUT 3h: PWM3 SYNCOUT 4h: PWM4 SYNCOUT 5h: PWM5 SYNCOUT 6h: PWM6 SYNCOUT 7h: PWM7 SYNCOUT 11h: CAP1 SYNCOUT 12h: CAP2 SYNCOUT 13h: CAP3 SYNCOUT 18h: INPUTXBAROUT5 19h: INPUTXBAROUT9 Configuration retained as for 0h	0h
7:5	Reserved			0h
12:8	CAP2SYNCIN	R/W	CAP2 Sync Input Source Select See the description for CAP1SYNCIN.	0h
15:13	Reserved			0h
20:16	CAP3SYNCIN	R/W	CAP3 Sync Input Source Select Refer to the description for CAP1SYNCIN.	0h
31:21	Reserved			0h

31 High-resolution pulse width modulator (HRPWM)

31.1 Full Name and Abbreviation Description of Terms

Table 144 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
High Resolution Positioner	HRP
Rising Edge	RE
Falling Edge	FE
Both Edges	BE
Scale Factor Optimizing	SFO

31.2 Introduction

HRPWM is a technique used to extend the time resolution capability of PWM, typically used when the PWM resolution is lower than approximately 9-10 bits.

HRPWM functionality is best suited for power conversion topologies that require high-frequency PWM, such as single (multi) phase Buck, Boost, and Flyback converters, phase-shifted full bridges, and direct modulation of class D power amplifiers.

31.3 Main characteristics

(1) HRPWM allows a minimum HRPWMCLK frequency of 60 MHz, with a typical micro-edge positioning (HRP) step size of 150 ps and a maximum of 310 ps.

(2) It can be used to control the duty cycle and phase of PWM signals

(3) Capable of extending the time resolution

(4) Dead band high-resolution control
- Allow control of RED and FED during half-cycle clock operation

(5) Allow enabling high-resolution switching for PWM output

(6) High-resolution control of the PWMxB signal output can be enabled by flipping the PWMxA signal output

(7) Applied to the outputs of PWMxA and PWMxB (signal paths A and B of PWM)

(8) Self-check the software mode

- This module is used to check the operation status of HRP logic, that is, check whether it is running as designed

(9) Implement fine edge positioning or time granularity control over PWM signals by extending CMPA, CMPB, and TBPHS registers

(10) Allow high-resolution control over the PWM output signal's high-resolution period, duty cycle, and phase on the devices with PWM modules

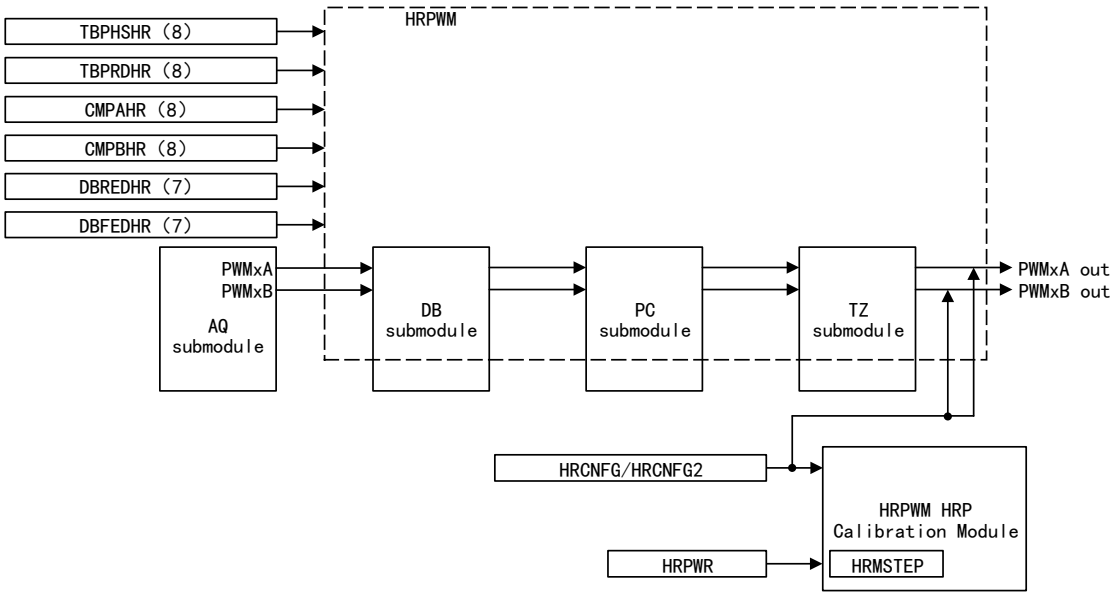
(11) PWM1 to PWM7 are equipped with HRPWM functionality. c

31.4 HRPWM structure block diagram

The High-Resolution PWM (HRPWM) module primarily enhances the temporal resolution of the PWM waveform. PWM resolution, expressed in terms of the number of resolution bits, is given by PWM Resolution = $\log_2$(TBPRD/PWMCLK). When the PWM resolution is lower than 9–10 bits, the high-resolution function must be enabled for specific applications (such as high-frequency switching power supplies). The key characteristics of this function are as follows:

- Enhanced time resolution, achieving a resolution of 150 ps (typical) via an analogue delay chain.
- Supports high-precision duty cycle control mode, high-precision period control mode and high-precision phase-shift control mode
- Supports high-precision control of the PWM A and B channel signals separately to generate PWMxA and PWMxB; supports swapping of the A and B channel outputs; supports the B channel output generating a high-precision inverted signal of the A channel.
- Supports high-precision dead-time control, capable of controlling the rising edge, falling edge or both edges.
- Depending on the control mode, different registers are selected to control the analogue delay chain.

Figure 123Structure Block Diagram of Multiple PWM Modules



Note: TBPHSHR and TBPRDHR come from the PWM TB time-base submodule, CMPAHR and CMPBHR come from the PWM CC counter comparison submodule, and DBREDHR and DBFEDHR come from the PWM DB dead-band generator submodule.

31.5 Functional description

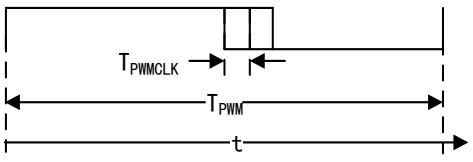
31.5.1 PWM resolution calculation

PWM peripherals are used to implement the functions mathematically equivalent to a Digital-to-Analog Converter (DAC). The timing diagram of a conventionally generated PWM is shown in the figure, and its effective resolution is related to the PWM frequency or period and the system clock frequency. Resolution calculation for conventionally generated PWM:

$$\text{PWM Resolution} = \frac{F_{PWM}}{F_{PWMCLK}} * 100\%$$

$$\text{PWM Resolution} = log_2 \frac{T_{PWM}}{T_{PWMCLK}}$$

Figure 124 Timing Diagram of a Conventionally Generated PWM



If the required PWM operating frequency cannot provide sufficient resolution in PWM mode, HRPWM shall be considered. Assuming conventional resolution

under the condition of PWMCLK being 100 MHz and an HRP step size of 180 ps, the resolution in bits for different PWM frequencies is shown in the following table. For typical and maximum performance indicators for HRP, refer to the datasheet.

Table 145 PWM and HRPWM Resolution at Different Frequencies

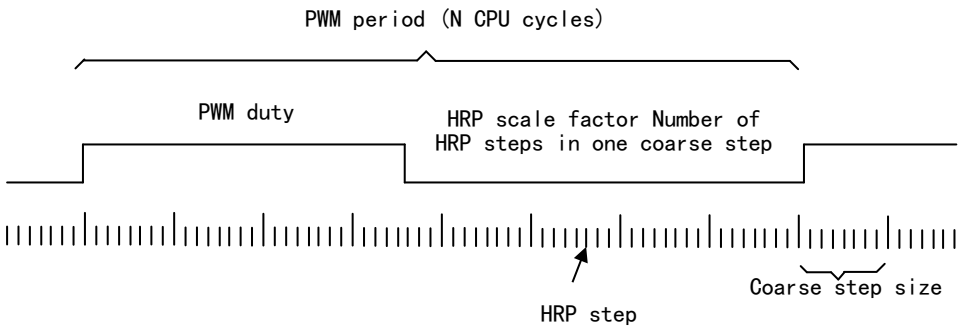
PWM frequency	PWM bits	PWM percentage	HRPWM bits	HRPWM percentage
20 kHz	12.3	0.020 %	18.1	0.000 %
50 kHz	11	0.050 %	16.8	0.001 %
100kHz	10	0.100 %	15.8	0.002 %
150 kHz	9.4	0.150 %	15.2	0.003 %
200 kHz	9	0.200 %	14.8	0.004 %
250 kHz	8.6	0.250 %	14.4	0.005 %
1000 kHz	6.6	1.000 %	12.4	0.018 %
1500 kHz	6.1	1.500 %	11.9	0.027 %
2000 kHz	5.6	2.00 %	11.4	0.036 %

31.5.2 HRP Technology

HRPWM is based on HRP technology, and the HRP logic can precisely place edges by subdividing the coarse system clock of the conventional PWM generator, with a time step precision on the order of 150 ps. For typical HRP steps on specific devices, refer to the datasheet. The self-check diagnostic software mode can check whether the HRP logic runs properly according to the design. For detailed information about software diagnostics and functions, refer to the SDK Description Document.

The HRP logic diagram is shown below. The HRP step is controlled by the high 8 bits of the CMPAHR register, and the same operational logic applies to the CMPBHR register. The control of the CMPA register value equals the number of coarse tuning steps, while the CMPAHR register value equals the number of fine tuning steps shifted 8 bits to the left, and then the result is stored in the high 8 bits of this register.

Figure 125 HRP Logic Diagram



Coarse step and HRP step calculation formula:

- Coarse step = int (PWM period * PWM duty cycle)
- HRP step = fraction (PWM period * PWM duty cycle) * HRP scale factor+0.5. Wherein, 0.5 is used for range and rounding adjustments (considering the Q8 format of 0x0080). PWM registers need to be configured to generate a conventional PWM with a given frequency and polarity to produce the HRPWM waveform. To extend the edge resolution, the PWM registers can work together with HRPWM. In various programming combinations, only a few are required and practical. For details, refer to the SDK Description Document.

31.5.3 Control of HRPWM Function

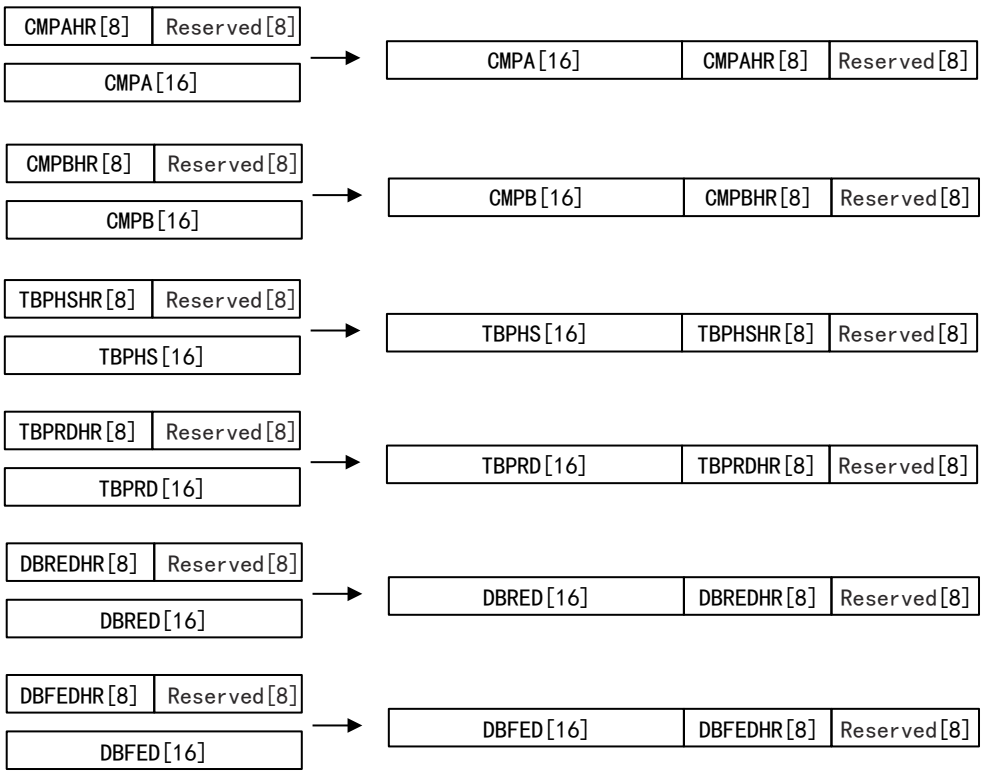
The HRP of HRPWM is controlled through the following 6 extended registers, which are concatenated with the CMPA, CMPB, TBPHS, TBPRD, DBRED, and DBFED registers used for controlling PWM operation.

Table 146 HRPWM Extended Registers

Register name	Description
CMPAHR	Counter compare A high-resolution extended register: This register is used for the output of action qualifier channel A, independent of the CMPA register
CMPBHR	Counter compare B high-resolution extended register: This register is used for the output of action qualifier channel B, independent of the CMPB register
TBPHSHR	High-resolution time-base phase offset register
TBPRDHR	High-resolution time-base period register, only applicable to certain devices
DBREDHR	Dead-band rising edge delay high-resolution register
DBFEDHR	Dead-band falling edge delay high-resolution register

Note: The TBPHSHR register cannot be used, and the TBPHSHR functions must be simulated via the TRREM register.

Figure 126 HRPWM Extended Registers and Built-in Configuration



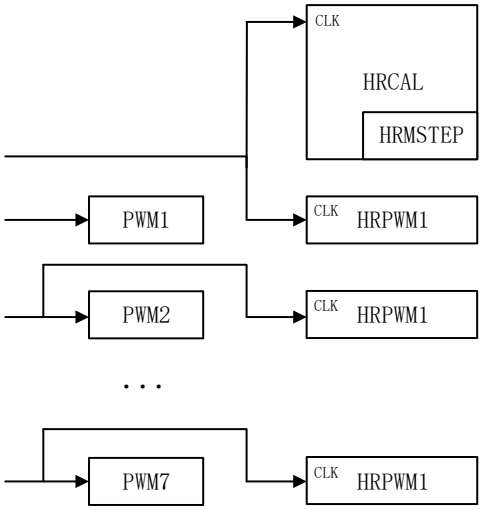
The HRPWM capability for dead-band rising and falling edge delays is only available during dead-band half-cycle clock operation. The number of HRP step bits [15:9] is half the size of the duty cycle and phase high-resolution registers.

HRPWM function is controlled via the PWM signal paths of channels A and B. By correctly configuring the HRCNFG2 register, HRPWM can be supported on the dead-band signal path. For the interface structure block diagram of HRPWM and 8-bit extended registers, refer to the "Main PWM Signals" section.

31.5.4 Clock source

The HRPWM module and HRCAL module are driven by the PWM1CLK clock. Therefore, PWM1CLK must first be enabled to enable HRCAL or any HRPWM module.

Figure 127 HRPWM and HRCAL Clock Source



31.5.5 Configuration HRPWM

If the PWM is configured as a regular PWM (providing a given frequency and polarity), you can configure the HRPWM by programming the HRCNFG register. The HRCNFG register provides the following configuration options.

Edge mode

Programmable HRP provides precise position control for rising edge (RE), falling edge (FE) or double-sided edge (BE). Where FE and RE are controlled by CMPAHR or CMPBHR registers for power topologies that require duty cycle control, and BE for topologies that require phase shift, such as phase shift full Bridges controlled by TBPRDHR or TBPHSHR registers.

Control mode

HRP can be programmed with the following two options:

- In duty cycle control mode, it is controlled by the CMPAHR or CMPBHR registers
- Controlled by the TBPHSHR register.

CMPAHR or CMPBHR registers can be used in conjunction with RE or FE control modes. The TBPHSHR register can BE used in conjunction with the BE control mode. While HRP is controlled with periodic control of the TBPHSHR register, the duty cycle and phase can also be controlled with the respective high-resolution registers.

Shadow mode

Provides the same dual buffering/shadowing options as regular PWM mode, only valid when operated by the CMPxHR and TBPHSHR registers and can be selected with the same regular loading options as the CMPA/CMPB registers.

Invalid when using the TBPHSHR register.

Control high-resolution B signal

The B signal of the PWM channel can be outputted by the PWMxB pin to achieve a high resolution output of the PWMxA signal. The HRPWM module supports high-resolution functionality on the B signal path independent of the A signal.

Swap high-resolution output

This mode allows swapping of high-resolution A and B outputs. The mode can be selected as follows:

- Allow A and B output signals to remain unchanged
- Swap A and B output signals, i.e., output A to B, and output B to A

Automatic conversion mode

The automatic conversion mode can only be used with SFO software.

Enable automatic conversion: CMPAHR = [frac(PWM period * PWM duty cycle) * 256] << 8

- In the background code, the SFO software will calculate the HRP scale factor and automatically update the HRP step count for each coarse step into the HRMSTEP register.
- The HRP calibration module automatically calculates the appropriate HRP step count (represented as a fractional duty cycle) using the values from the HRMSTEP and CMPAHR registers and moves the high-resolution PWM signal edges accordingly.

Disable automatic conversion: CMPAHR = [frac(PWM period * PWM duty cycle) * HRP scale factor+0.5] << 8

- In this mode, all calculations must be performed by the user's code, and the HRMSTEP register will be ignored.
- The automatic conversion behavior of the high-resolution period and high-resolution duty cycle is the same.
- If high-resolution period mode is enabled, automatic conversion must always be enabled.

Note: When the HRPWM module is configured for center-aligned count mode, the shadow mode of the HRPWM registers must be set to load when TBCTR=0 and TBCTR=TBPRD. User's new values will only be loaded into the shadow registers when TBCTR=0 (the shadow mode of the register must be set to TBCTR=0 and TBCTR=TBPRD). The TBCTR=TBPRD event is used for specific internal logic of the HRPWM module.

The automatic conversion mode performs calculations for CMPBHR, DBREDHR, and DBFEDHR:

Enable automatic conversion: CMPAHR = [frac(PWM period * PWM duty cycle) * 256] << 8

- In the background code, the SFO software will calculate the HRP scale factor and automatically update the HRP step count for each coarse step into the HRMSTEP register.
- The HRP calibration module automatically calculates the appropriate HRP step count (represented as a fractional duty cycle) using the values from the HRMSTEP and CMPBHR / DBREDHR/DBFEDHR registers and moves the high-resolution PWM signal edges accordingly.

Disable automatic conversion: CMPBHR behaves the same as CMPAHR, i.e., CMPBHR = [frac(PWM period * PWM duty cycle) * HRP scale factor+0.5] << 8

31.5.6 Working principle

The HRP logic is able to place edges on one of 255 discrete time points (8-bit). For typical HRP steps, refer to the datasheet. To ensure the application time step and edge placement accuracy under various conditions (such as a wide range of PWM frequencies and system clock frequencies), HRP works in conjunction with the TBM and CCM registers. The typical frequency range supported by HRPWM is shown below.

Table 147 Typical Frequency Range Supported by HRPWM

System clock (MHz)	Maximum resolution (Bits)(1)	HRP Steps per PWMCLK(2) (3)	PWM frequency	
			Minimum (Hz)(4)	Maximum (MHz)
60.0	10.9	93.0	916.0	3.0
70.0	10.6	79.0	1068.0	3.5
80.0	10.4	69.0	1221.0	4.0
90.0	10.3	62.0	1373.0	4.5
100.0	10.1	56.0	1526.0	5.0

Note:

(1) The resolution in bits provided is for the specified maximum PWM frequency

(2) TBCLK=PWMCLK

(3) The data in the table is based on a 180ps HRP time resolution, where HRP step = T_{PWMCLK}/180ps. For the HRP range, refer to the datasheet.

(4) The PWM is in asymmetric upcount mode and the PWM minimum frequency is based on a maximum period value of 65535.

Edge placement

A digital controller issues a duty cycle command expressed in units or

percentages in a typical power control loop. If an operating point requires a duty cycle of 40.5%, the converter PWM frequency is required to be 1.25 MHz. In conventional PWM generation (the system clock is 100MHz), the choice of duty cycle is about 40.5%. As shown in the table below, the duty cycle closest to the 40.5% value is the counting value 32, when the duty cycle is 40%, which is equivalent to the edge placed at 320ns.

Table 148 CMPA and Duty Cycle (left three columns) and [CMPA:CMPAHR] and Duty Cycle (right four columns)

CMPA count value(1)(2)(3)	High-level time	Duty cycle	CCOMPACMPA count value	CCOMPAHRCMPAHR count value	High-level time	Duty cycle
28	280 ns	35.0 %	32	18	323.24 ns	40.405 %
29	290 ns	36.3 %	32	19	323.42 ns	40.428 %
30	300 ns	37.5 %	32	20	323.60 ns	40.450 %
31	310 ns	38.8 %	32	21	323.78 ns	40.473 %
32	320 ns	40.0 %	32	22	323.96 ns	40.495 %
33	330 ns	41.3 %	32	23	324.14 ns	40.518 %
34	340 ns	42.5 %	32	24	324.32 ns	40.540 %
...	...	...	32	25	324.50 ns	40.563 %
32.4	324 ns	40.5 %	32	26	324.68 ns	40.585 %

Note:

(1) TBCLK is 100 MHz, 10ns.

(2) The HRP step resolution is 180ps. For typical and maximum HRP values, refer to the datasheet.

(3) For a PWM period register value of 80, T_{PWM} = 80 × 10ns = 800ns, F_{PWM} = 1/T_{PWM} = 1.25 MHz

Assuming HRP step resolution is 180ps, HRP can achieve an edge placement closer to the desired 324ns. Beyond the CMPA count value, the 22 HRP steps will place the edge at 323.96ns, resulting in almost zero error.

When using [CMPB:CMPBHR] for duty cycle control, its operation and formula are the same as those for the [CMPA:CMPAHR] register combination.

Scaling considerations

The CMPA and CMPAHR register resources are used to demonstrate the mechanism of precisely positioning edges in time. In practical applications, the fractional duty cycle needs to be written to the final integer representation ([CMPA:CMPAHR] register combination) and seamlessly provide this mapping function to the CPU.

To achieve this, the mapping or scaling steps involved must be checked. In control software, duty cycle is typically expressed in units or percentages. This approach ensures that all mathematical calculations are performed without concerning for the absolute duty cycle expressed in clock counts or ns for high-level time. Moreover, it makes the code more compatible with different converter types operating at various PWM frequencies.

Assumption conditions:

- TBCLK=10ns (100MHz)
- FPWM=1.25MHz
- PWM Duty Cycle = 40.5%
- The coarse step PWM period is 80
- For each coarse step, when the HRP step is 180ps, the HRP scale factor is 55
- The constant used to keep CMPAHR between 1 and 255 and for fractional rounding is 0.5 (default)

The following two-step scaling process is needed to facilitate the mapping scheme:

- Conversion of the integer portion of the duty cycle percentage in the CMPA register
  - CMPA register value = int(PWM period * PWM duty cycle) = 32
  - CMPAHR register value = 32 = 0x20
- Conversion of fractional portion of the CMPAHR register
  - CMPAHR register value = [frac(PWM period * PWM duty cycle) * HRP scale factor+0.5] << 8 = 22.5 * 256 = 5760, i.e., 0x1680
  - CMPAHR register value = 0x1680, i.e., 0x1600, because the hardware ignores the lower eight bits

When AUTOCONV is set and the HRP scale factor is in the HRMSTEP register, CMPxHR register value = frac(PWM period * PWM duty cycle). The rest of the conversion calculations are made automatically by the hardware, and the PWM channel will output the correct HRP-scaled signal edges. When AUTOCONV is not set, the above calculations must be performed by software.

The HRP scale factor is influenced by the system clock and the DSP's operating conditions. The HRP scaling factor C function provided by Geehy uses HRPWM's built-in diagnostics to return the optimal scaling factor, which changes slowly and the optimization process can be carried out in the background loop.

CPU ability of 32-bit data in memory can be configured in CMPx and CMPxHR register values to write as a single stitching, namely [CMPA: CMPAHR], [CMPB: CMPBHR], etc

Note:

(1) The actual implementation uses Arm's 32-bit CPU architecture, which differs from the above steps.

(2) For time-critical control loops where even one cycle is to be considered, it is recommended to use functions with optimized cycles (assembly versions). It takes a Q15 duty cycle as input and writes a single [CMPA:CMPAHR] value.

Limit HRP Duty Cycle

In high-resolution mode, HRP is not active throughout the entire 100% PWM period. HRP can operate under the following conditions:

(1) High-resolution PWM period control is not enabled
- HRP operates three PWMCLK periods after the start of the period

(2) High-resolution PWM period control is enabled using the HRPCTL register
- Count-up mode
  - HRP works between three PWMCLK cycles at the beginning of a period and three PWMCLK cycles at the end of a period
- Center-aligned count mode
  - When counting up, HRP starts working three periods after TBCTR=0, and continues until three periods before TBCTR=TBPRD
  - When counting down, HRP starts working three periods after TBCTR=TBPRD, and continues until three periods before TBCTR=0

(3) Using DBREDHR or DBFEDHR
- The registers corresponding to edge high-resolution placement in DBRED or DBFED must be greater than or equal to 7

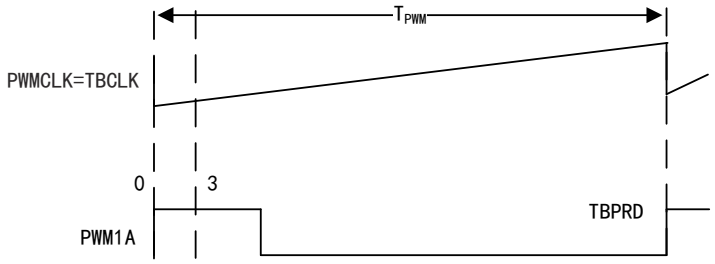
The duty cycle range is limited as shown in the figure below, it limits the HRP duty cycle. For example, when the duty cycle is 0%, precise edge control cannot be used. If high-resolution PWM period control is disabled, the conventional PWM duty cycle control can operate fully down to 0% duty cycle, even if the HRPWM function is unavailable in the first three periods. However, in most applications, the controller adjustment points are typically not designed to operate close to 0% duty cycle, so this issue can be avoided. If high-resolution PWM period control is enabled, the duty cycle must not fall within the restricted range; otherwise, undefined behavior may occur on the PWMxA output.

When the application requires HRPWM to work below the minimum duty cycle limit, HRPWM can be configured to operate in count-down mode. In this case, when HRPE=0, the rising edge position is controlled by HRP, as shown in the

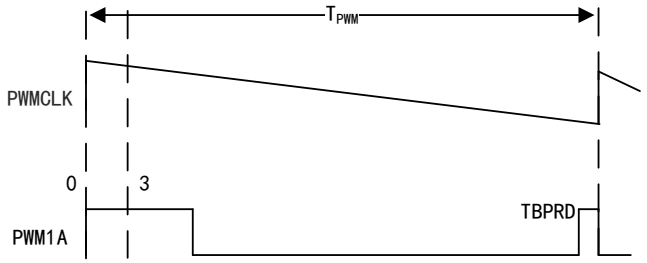
figure below. This configuration can resolve the minimum duty cycle limit issue, but there is a maximum duty cycle limit at the same percentage. The duty cycle range limit for three periods is shown in the table below.

Figure 128 Duty Cycle Range Limit

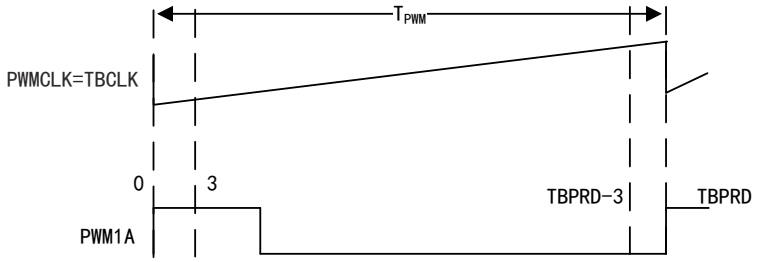
Low % duty cycle range limitation (HRPE=0)



High % duty cycle range limitation (HRPE=0)



Duty cycle range limitation of up counter (HRPE=1)



Duty cycle range limitation of up and down counters (HRPE=1)

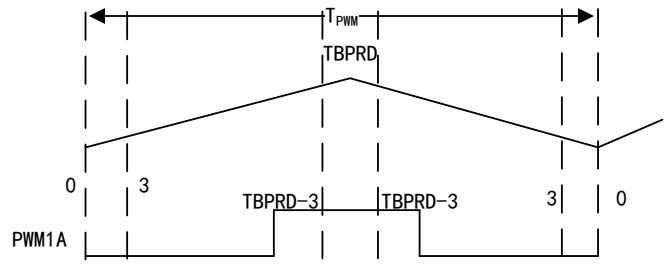


Table 149 Duty Cycle Range Limit for Three Periods

PWM period (1)	Duty cycle	
	Minimum value	Maximum value (2)
200 kHz	0.6 %	99.4 %
400 kHz	1.2 %	98.8 %
600 kHz	1.8 %	98.2 %
800 kHz	2.4 %	97.6 %
1000 kHz	3%	97 %
1200 kHz	3.6 %	96.4 %
1400 kHz	4.2 %	95.8 %
1600 kHz	4.8 %	95.2 %
1800 kHz	5.4 %	94.6 %
2000 kHz	6 %	94 %

Note:

(1) TBCLK=PWMCLK=100MHz

(2) This limitation only applies when high-resolution PWM period control is enabled

High-resolution period

If high-resolution PWM period is required, the PWM module must be initialized in a specific order.

The steps for high-resolution operation on PWMxA using the CMPA and its shadow register, along with the corresponding HRCNFG field, are shown below. To perform high-resolution operation on PWMxB, simply replace the corresponding register of channel A with that of the channel B.

Table 150 High-resolution PWM Period Operation Steps

Step	Description
1	Enable PWM clock
2	Enable HRPWM clock
3	Disable synchronous PWM clock (PWMCLKSYNC)
4	Configure PWMx registers: (1) PWMx can only be configured in count-up or center-aligned count mode. High-resolution PWM period is not compatible with the count-down mode. (2) TBPRD and CCOMP registers must be configured for shadow load. (3) CMPCTL [LOADAMODE] Count-up mode: LOADAMODE = 01; load when TBCTR = TBPRD Center-aligned count mode: LOADAMODE = 10; load when TBCTR = 0 or TBCTR = TBPRD

Step	Description
5	Configure the HRCNFG register: (1) HRLOAD = 10; load when TBCTR = 0 or TBCTR = TBPRD (2) Enable automatic conversion (AUTOCONV = 1) (3) HRP logic simultaneously controls both edges of the A path signal (AEMSEL = 11)
6	TBPOF for high-resolution PWM period: TBPHSHR synchronization, setting the LDTBCNTVEN bit and HRPHASYNCEN bit. In center-aligned count mode, these bits must be set, regardless of the contents in the TBPHSHR register.
7	Enable high-resolution PWM period control (HRPE = 1)
8	Enable synchronous PWM clock
9	Force a synchronization pulse by software (SPFSW = 1)
10	When enabling automatic conversion, HRMSTEP must include the exact HRP scale factor, which is the number of HRP steps per PWMCLK coarse step. This scale factor can be obtained using the SFO() function described in the SFO Library Software section.
11	Writing to the PWM_TBPRDHR register can control the high-resolution PWM period

If high-resolution period mode is enabled, the PWMx synchronization pulse causes jitter of ±1 TBCLK cycle in count-up mode, and ±2 TBCLK cycles in center-aligned count mode. Therefore, you cannot set TBCTR=0 or TBCTR=CMPB as the synchronous output signal source. When TBCTL=0, the software synchronization pulse can only be sent once during the high resolution cycle initialization. Applying a synchronization pulse while PWM is running can cause output jitter.

If high resolution PWM period control is enabled only on PWMxA or PWMxB, the non-high resolution output has a jitter of ±1 TBCLK cycles in up count mode and ±2 TBCLK cycles in center aligned count mode.

The duty ratio conversion procedure described in section Limiting HRP Duty ratios is also used with high-resolution PWM cycles.

Assumption conditions:

- TBCLK=10ns (100MHz)
- FPWM=1.75MHz
- Period = 571.482
- For each coarse step, when the HRP step is 180ps, the HRP scale factor is 55
- The constant used to keep TBPRDHR between 1 and 255 and for fractional rounding is 0.5 (default)

Table 151 Issue

Mode	TBPRD	FPWM	TPWM
Count-up mode	570	175.13kHz	$571*T_{TBCLK}$

Mode	TBPRD	FPWM	TPWM
	571	174.82kHz	$572*T_{TBCLK}$
Center-aligned count mode	285	175.44kHz	$285*2*T_{TBCLK}$
	286	174.82kHz	$286*2*T_{TBCLK}$

Solution (with 55 HRP steps per coarse step at 180ps):

(1) Conversion of TBPRD register's percentage integer period value

- Integer period value = int(PWM period) * $_{TTBCLK}$ = 571 * $_{TTBCLK}$
- Count-up mode: TBPRD = Period value - 1 = 570 = 0x23A
- Center-aligned count mode: TBPRD = Period value / 2 = 285 = 0x11D

(2) Conversion of TBPRDHR register fractional value

- Center-aligned count mode: TBPRDHR register value = frac(PWM period * HRP scale factor+0.5)
  - When automatic conversion is enabled and HRMSTEP = HRP scale factor = 55, TBPRDHR register value = frac(PWM period) << 8, moving the value to the high byte of TBPRDHR, i.e., TBPRDHR register value = frac(571.428) << 8 = 0x6D
  - Automatic conversion will automatically perform the calculation, so TBPRDHR HRP delay is scaled by hardware to: [(TBPRDHR[15:0] >> 8)* HRMSTEP + 0x80] << 8 =0x6D * 55+ 0x80=0x17EB >> 8
  - Period HRP delay = 0x17 HRP steps
- Center-aligned count mode:TBPRD = frac(PWM period * HRP scale factor+0.5)
  - When automatic conversion is enabled and HRMSTEP = HRP scale factor = 55, TBPRDHR register value = frac(PWM period/2) << 8, moving the value to the high byte of TBPRDHR, i.e., TBPRDHR register value = frac(285.714) << 8 = 0xB6
  - Automatic conversion will automatically perform the calculation, so TBPRDHR HRP delay is scaled by hardware to: [(TBPRDHR[15:0] >> 8)* HRMSTEP + 0x80] << 8 =0xB6 * 55+ 0x80=0x279A >> 8
  - Period HRP delay = 0x27 HRP steps

31.5.7 High-resolution dead band

Configure the high-resolution dead band rising and falling edge delays

If the PWM is configured to provide a conventional PWM with a given frequency and polarity, and the dead band is enabled in half-cycle clock mode, the high-resolution operations of DBRED and DBFED can be enabled by programming the HRCNFG2 register in the PWM module register space. This register provides the following two configuration options:

- Control mode
  - In high-resolution mode, select the time event when DBRED and DBFED load shadow values into the active register
  - Select pulses to match the selection in DBCTL[LOADREDMODE, LOADFEDMODE] bits
- Edge mode
  - HRP can achieve precise position control of DBRED, DBFED, or both dead-band rising edge of DBRED signal and falling edge of DBFED signal through programming

High-resolution dead-band operation

In up mode, the dead-band module cannot be used when any high-resolution mode is enabled.

Assumption conditions:

- TBCLK=10ns (100MHz)
- In half-cycle mode, enable dead band: TBCLK = PWMCLK
- FPWM=1.33MHz
- The duty cycle is 50%
- The dead-band rising edge delay is 5% beyond the duty cycle
- The dead-band rising edge delay in ns is: 0.05 * 375 = 18.75ns

Note: Similar to the duty cycle limit when using HRPWM; high-resolution dead band must be used when the values of DBRED and DBFED are greater than 3.

The dead-band delay value is a function of DBRED and DBFED:

- When enabling half-cycle clock, the RED and FED calculation formulas become:
  - RED= TBCLK/2 * DBRED
  - FED=TBCLK/2 * DBFED

DBRED and DBFED count values:

- In ns, DBRED = 18.75ns
- Required DBRED = RED / (TBCLK / 2) = 3.75ns

Solution (when the HRP step of each coarse step is 180ps):

- Integer dead-band value conversion for DBRED register
  - Integer DBRED value = int[RED / (TBCLK / 2)] = 3

- Decimal value conversion for DBREDHR register
  - DBREDHR value = frac[(required DBRED) * HRP scale factor+0.5] << 8 = frac(3.75) * 55+0.5 << 8 = 41.75 * 256 = 0x29C. The hardware will ignore the lower 9 bits of the DBREDHR value computed above

Note: When AUTOCONV is set and the HRP scale factor is in the HRMSTEP register, DBREDHR: DBRED = frac(required DBRED) << 8. The remaining conversion calculations are automatically performed by the hardware, and the correct HRP-scaled signal edges will appear on the PWM output channels. When AUTOCONV is not set, the above calculations must be performed by software.

31.5.8 Scale factor optimization software

HRP logic can place edges at one of 255 discrete time points. These step sizes are approximately on the order of 150ps. For typical HRP steps for the device, refer to the datasheet. HRP step size is affected by worst-case process parameters, operating temperature, and voltage; the HRP step decreases as voltage increases and temperature decreases. Otherwise, it increases. HRPWM applications can use the HRP scaling factor software function provided by Geehy. While HRPWM is running, the SFO function facilitates the step calculation for each PWMCLK cycle.

The HRPWM module has built-in self-test and diagnostic functions to find the best HRP scaling factor. Geehy also provides an SFO library that C can call to help determine the optimal scaling factor.

For detailed description of the SFO library software (SFO_Geehy_Build_V8.Lib), refer to the SFO Library Software section.

31.5.9 SFO library software

The characteristics of the SFO library are as follows:

- The SFO_Geehy_Build_V8.Lib completes the check, with its unit being the function return value
- Under repeated calls with no interruptions, the typical period required for SFO() to update the HRP scale factor is 130,000 PWMCLK cycles.

Scale factor optimization function

The program runs SFO diagnostics by driving the HRP calibration module and is able to determine the appropriate HRP scaling factor for the device at any time, that is, HRP step for each PWMCLK coarse step. Assuming both PWMCLK and TBCLK are 100 MHZ and the HRP step is 150ps, the typical scaling factor at 100 MHZ is 66 HRP steps per 10ns (TBCLK units).

This function returns the HRP scale factor value: HRP scale factor = HRP step per PWMCLK.

(1) Use restrictions of int SFO()

SFO() can be used with a minimum PWMCLK and TBCLK all is 50 MHz. Since the HRP diagnostic logic uses PWMCLK, SFO() is restricted by PWMCLK. Due to variations in the device process, the HRP step may decrease under low temperature and high core voltage conditions if it is below 50 MHz, causing 255 HRP steps to fail to cover the entire PWMCLK period.

SFO() can be called at any time on the HRP calibration module to run the SFO diagnostics.

(2) Usage of int SFO()

If the PWM channel is operating in HRPWM mode, the function can be called in the background at any time. Its function is to utilize the diagnostic logic in the HRP calibration module, and the resulting scale factor can then be applied to all PWM channels running in HRPWM mode.

This function returns values to indicate different situations:

- The program returns 00: Indicates that calibration is still running
- The program returns 01: Indicates that calibration is complete and a new scale factor has been calculated
- The program returns 10: Indicates an error, where the HRP scale factor is greater than the maximum of 255 fine steps per coarse step PWMCLK cycle. In this case, the HRMSTEP register will maintain the last HRP scale factor value less than 256 for automatic conversion

If high-resolution PWM period control is not enabled, the minimum duty cycle limit for all PWM modules operating in HRPWM mode is 3 PWMCLK cycles. If this control is enabled, an additional duty cycle limit of 3 PWMCLK cycles is imposed before the PWM period ends. For details, refer to the Limit HRP Duty Cycle section.

The scale coefficient results can be used by SFO() to update the HRMSTEP register. When AUTOCONV is set, the application software is responsible for setting CMPAHR = [frac(PWM period * PWM duty cycle) * 256]<<8 or TBPRDHR = frac(PWM period) only when the function is running in the background. The HRP calibration module automatically calculates the appropriate HRP step based on the value in the HRMSTEP and CMPxHR/TBPRDHR registers, which is represented by fractional duty cycle or period, and adjusts the edge position of the HRPWM signal accordingly.

When the AUTOCONV bit is cleared, the HRMSTEP register is ignored. The application needs to perform the following calculations manually:

- CMPAHR = frac(PWM period * PWM duty cycle) * HRP scale factor << 8 + 0x80
- The operating mode of CMPBHR, DBREDHR/DBFEDHR, and TBPHSHR are similar, and when using TBPRDHR, automatic conversion function must be enabled

(3) Program using HRPWM duty cycle based on driverlib functions

Figure 129 HRPWM Duty Cycle Program

```
float32_t dutyFine = 85.62;
float32_t count = (dutyFine * (float32_t)(EPWM_TIMER_TBPRD << 8))/100;
uint32_t compCount = (count);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_A, compCount);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_B, compCount);
```

This program can run as a background task in a slower loop, with the required CPU cycles being negligible. The repetition frequency of executing SFO functions depends on the working environment of the application. Like all digital CMOS devices, the operation of HRP is affected by changes in temperature and supply voltage. However, these parameters change slowly in most applications, and the SFO function cans usually be executed once every 5 to 10 seconds. If faster changes are required, they need to be executed more frequently to suit the application. It is important to note that there is no upper limit to the repetition frequency of the SFO function, so it can be executed quickly based on the speed of the background loop.

When using the HRPWM characteristics, the HRPWM logic is inactive during the first 3 PWMCLK cycles of the PWM period. When using TBPRDHR, the HRPWM logic is inactive during the last 3 PWMCLK cycles of the PWM period. When running an application with this configuration, if HRPE=0 and the CMPx register value is less than 3 cycles, the CMPx register must be cleared. When HRPE=1, to avoid any unexpected jumps on the PWM signal, the CMPA register value must not be less than 3 or greater than TBPRD - 3.

31.6 Register

For HRPWM registers, refer to the PWM Registers section.

32 Capture (CAP)

32.1 Full Name and Abbreviation Description of Terms

Table 152 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Auxiliary Pulse-width Modulator	APWM

32.2 Introduction

The capture is used in systems for accurately timing the external events. The capturer can decode the voltage or current amplitude from the duty cycle code current/voltage sensor, and measure the duty cycle and period of the pulse sequence signal, and the elapsed time between position sensor pulses. It supports measurement of the speed of rotating machinery through the toothed chain wheels sensed by Hall sensors.

32.3 Main characteristics

(1) 128:1 input multiplexer

(2) Event filter prescale

(3) 4 32-bit event timestamp registers

(4) Capture up to four timestamp events at a time

(5) Capture the timestamps in a four-depth circular buffer through continuous mode

(6) Used in edge polarity selection for up to four sequence timestamp capture events

(7) Can respond to any of the four events and generate interrupts

(8) Differential (Delta) mode timestamp capture

(9) Absolute timestamp capture

(10) Independent DMA trigger

(11) CAP can be configured as a single-channel PWM output (when CAP is not used for input capture)

(12) Type 1 CAP adds the following features to Type 0 CAP:
- WRPRT protection (added to key registers)
- Input multiplexer: Selects the capture input source based on the ECCTL0[INPUTSEL] bit.

- Modulus counter status bit: In a Type 0 CAP, the current status of the modulus counter cannot be determined. However, in a Type 1 CAP, the ECCTL2[MODCNTRSTS] bit can be used to indicate which capture register will be loaded during the next capture event.
- Event filter reset: The reset can be used for initialisation and debugging. Setting ECCTL2[CTRFILTRESET] clears the modulus counter, event filter and any pending interrupt flags.
- DMA Trigger Source: CAPxDMA acts as a DMA trigger. The ECCTL2[DMAEVTSEL] bit can be used to select CEVTx (x=1...4) as the DMA interrupt source.

(13) Each CAP channel has the following functions:

- 128:1 input multiplexer
- Connect the capture input using the input X-BAR
- In APWM mode, configure the output X-BAR as output
- Input capture signal prescale
- Select the independent edge polarity (rising edge/falling edge) for four capture events
- Four 32-bit timestamp registers
- 32-bit counter
- A four-order sequencer (modulo-4 counter) synchronized with external events, rising/falling edge of the CAP pin
- WRPRT protection
- Independent DMA trigger
- Interrupt function for any of the four capture events
- Able to reset the modulo counters, event filters, and interrupt flags
- Control the continuous timestamp capture in a four-depth ring buffer
- The status register of the modulo counter is used to indicate the current status of the modulo counter
- Trigger the compare registers (two bits) once, and freeze 1-4 timestamp events after capture

(14) Function description of APWM mode:

- Two 32-bit digital comparators can be used to compare the timestamp counter buses
- Capture register 1/2 can be used as a period and comparison value in APWM mode (when capture register 1/2 is not used in CAP mode)
- Double buffering is implemented using capture register 3/4 (APRD and ACMP shadow registers), and the contents can be transmitted to capture register 1/2 when the counter is triggered or immediately transmitted to capture register 1/2 when written.
- During initialization, write two period active registers, compare them, and automatically copy the initial value to the shadow value. The shadow registers can be used for subsequent comparison and updating in operation process.

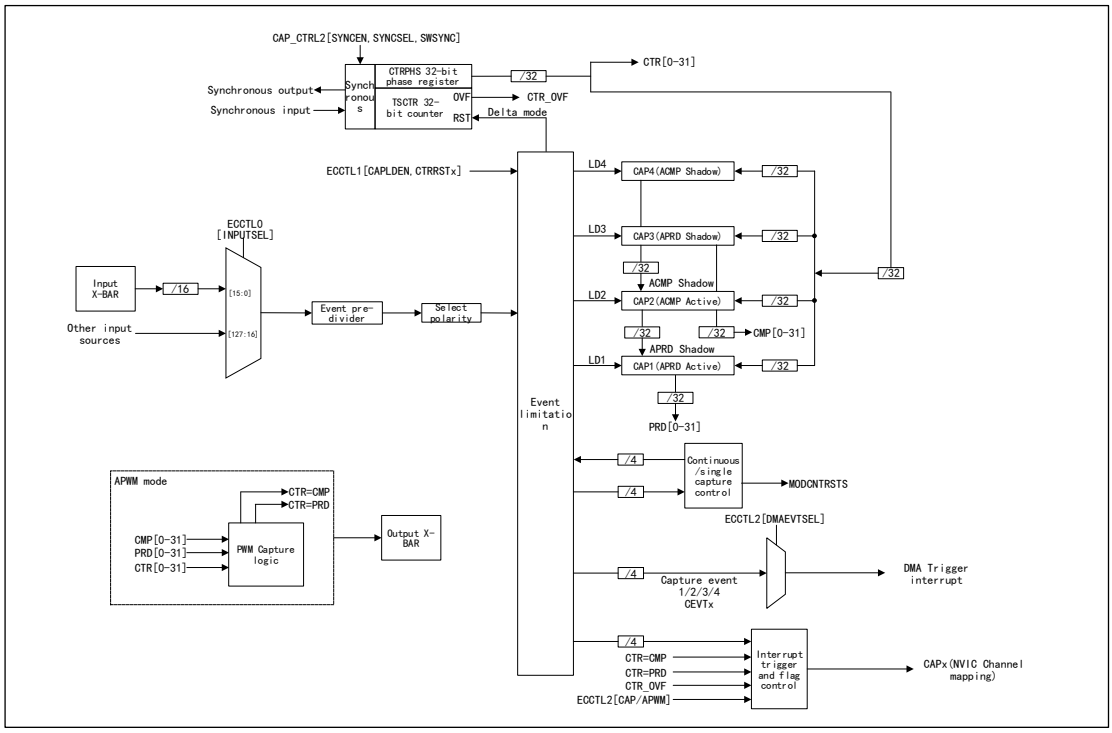
- Instant mode: In APWM mode, when writing to the active registers (capture register 1/2), the same value will also be written to the shadow register (capture register 3/4). Write to the capture register 3/4 and call the shadow mode

(15) Interrupt event:

- CTR=PRD
- CTR=CMP
- CEVTx (x=1...4)
- CTROVF

32.4 Structure block diagram

Figure 130 CAP Structure Block Diagram



32.5 Function description of CAP pins

Input capture

The input X-BAR connects the device pin to the CAP module as an input, and any GPIO on the CPU can be selected as the input source. Internal pull-up can be configured in the GPxPUD register. The GPIO input can be configured for synchronous or asynchronous mode by setting the GPxQSELy register. While synchronous inputs are noise-resistant, they can affect the accuracy of CAP by ±2 cycles. When using GPIO mode, the signal can be inverted via the GPxINV register.

Note: For the CAP module to correctly capture, CAPxIN must have at least 2 APBCLK cycle widths; otherwise, APBCLK sampling will miss the input pulse.

Multiplexer configuration

Type 1 CAP modules must be provided with a 128:1 input multiplexer, as shown in the CAP structure block diagram. The CAP input source can be configured via ECCTL0[INPUTSEL]. For details, refer to the table below.

Table 153 Multiplexer Configuration

Selection of CAP Input	CAP1 INDEX	CAP2 INDEX	CAP3 INDEX
INPUTXBAR1	0	0	0
INPUTXBAR2	1	1	1
INPUTXBAR3	2	2	2
INPUTXBAR4	3	3	3
INPUTXBAR5	4	4	4
INPUTXBAR6	5	5	5
INPUTXBAR7	6	6	6
INPUTXBAR8	7	7	7
INPUTXBAR9	8	8	8
INPUTXBAR10	9	9	9
INPUTXBAR11	10	10	10
INPUTXBAR12	11	11	11
INPUTXBAR13	12	12	12
INPUTXBAR14	13	13	13
INPUTXBAR15	14	14	14
INPUTXBAR16	15	15	15
FLB1_OUT14	16	16	Reserved
FLB1_OUT15	17	17	Reserved
Reserved	18-19	18-19	18-19
CANA_INT0	20	20	20
Reserved	21-23	21-23	21-23
OUTPUTXBAR1	24	24	24
OUTPUTXBAR2	25	25	25
OUTPUTXBAR3	26	26	26
OUTPUTXBAR4	27	27	27
OUTPUTXBAR5	28	28	28
OUTPUTXBAR6	29	29	29
OUTPUTXBAR7	30	30	30
OUTPUTXBAR8	31	31	31
Reserved	32-35	32-35	32-35
ADCCEVT1	36	36	36
ADCCEVT2	37	37	37
ADCCEVT3	38	38	38

Selection of CAP Input	CAP1 INDEX	CAP2 INDEX	CAP3 INDEX
ADCCEVT4	39	39	39
Reserved	40-43	40-43	40-43
ADCAEVT1	44	44	44
ADCAEVT2	45	45	45
ADCAEVT3	46	46	46
ADCAEVT4	47	47	47
Reserved	48-95	48-95	48-95
COMP1_CTRIPL	96	96	96
COMP2_CTRIPL	97	97	97
COMP3_CTRIPL	98	98	98
COMP4_CTRIPL	99	99	99
Reserved	100-107	100-107	100-107
COMP1_CTRIPH	108	108	108
COMP2_CTRIPH	109	109	109
COMP3_CTRIPH	110	110	110
COMP4_CTRIPH	111	111	111
Reserved	112-114	112-114	112-114
GPIO8	115	115	115
GPIO9	116	116	116
GPIO22	117	117	117
GPIO23	118	118	118
Reserved	119	119	119
COMP1_CTRIPH_OR_CTRIPL	120	120	120
COMP2_CTRIPH_OR_CTRIPL	121	121	121
COMP3_CTRIPH_OR_CTRIPL	122	122	122
COMP4_CTRIPH_OR_CTRIPL	123	123	123
Reserved	124-126	124-126	124-126
INPUTXBAR7	127	Reserved	Reserved
INPUTXBAR8	Reserved	127	Reserved
INPUTXBAR9	Reserved	Reserved	127

Output signals

The output X-BAR must connect the output signals to the OUTPUTXBARx output configuration. The GPIO multiplexer must be configured to connect OUTPUTXBARx to any GPIO pin of the GPIO multiplexer. It is required to configure the GPyGMUX register first, while keeping the corresponding field of the GPyMUX register as 0, and then write the desired value to the GPyGMUX register.

For details on the GPIO multiplexer, refer to the datasheet. For details on GPIO settings and X-BAR configuration, refer to the GPIO section.

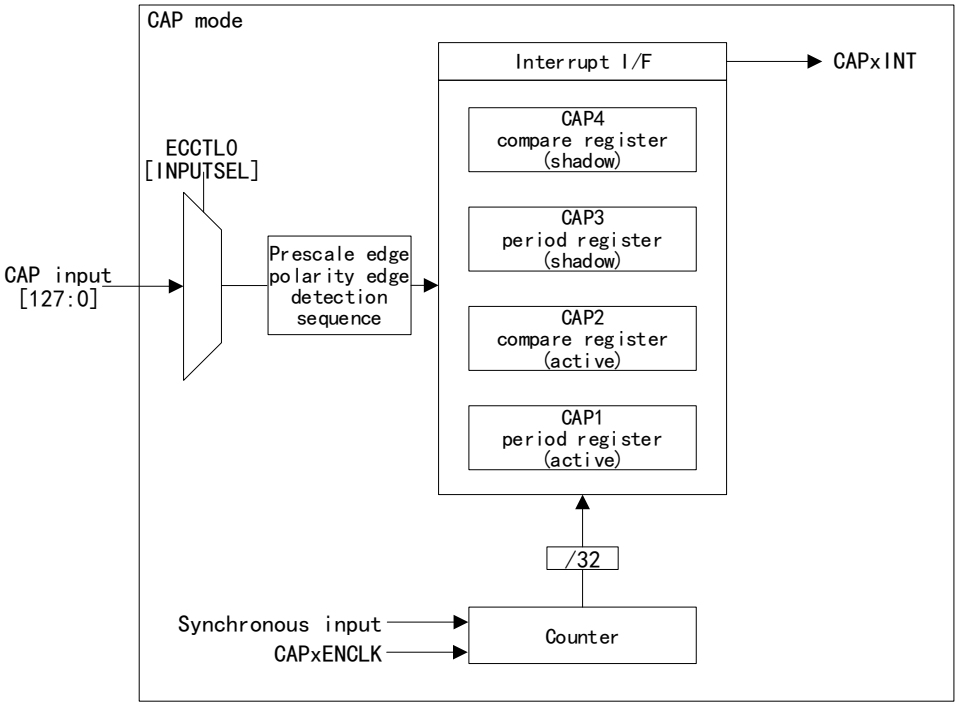
32.6 Capture description function description

CAP1 register becomes the active cycle register, and CAP3 register becomes its shadow register. CAP2 register becomes the active compare register, and CAP4 register becomes its shadow register.

Note: CAP mode and APWM mode share a pin. This pin is an input pin in capture mode.

32.6.1 CAP mode structure block diagram

Figure 131 CAP Mode Structure Block Diagram



32.6.2 Event filter prescale

The input capture signal can bypass the prescaler or be prescaled by setting N = 2-62 (multiples of 2). The structure block diagram and the timing diagram of the prescaler function are shown below, with the event filter prescaler being reset by the CTRFILTRESET bit . The prescaler plays a significant role if the input signal is of high frequency.

Figure 132 Event Filter Prescaler Structure Block Diagram

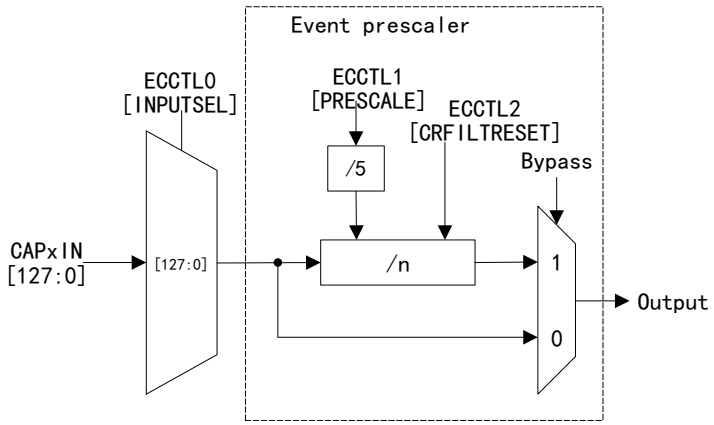
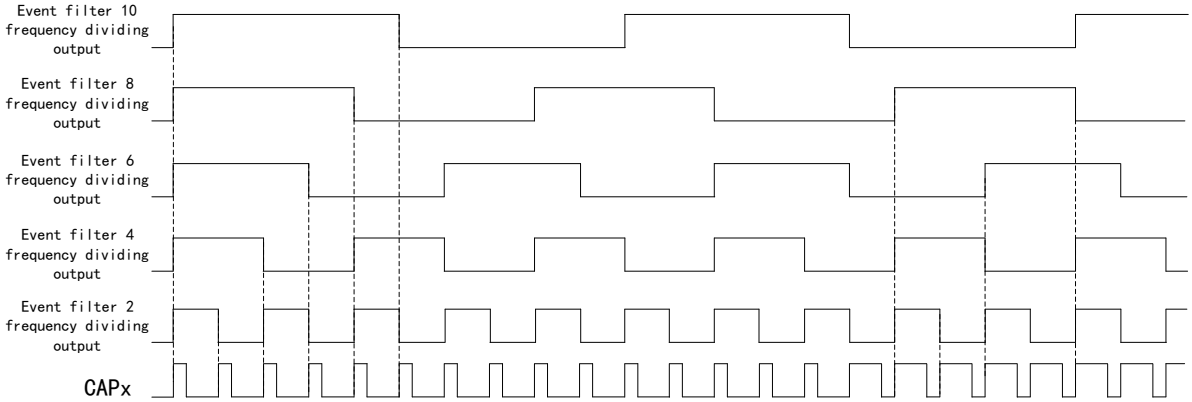


Figure 133 Prescaler Function Timing Diagram



If PRESCALE=00000, the input capture signal will bypass the event filter prescaler. If the event filter prescaler configuration is changed, the first rising edge after the change will bypass the prescaler, and the second rising edge after the change will take effect.

32.6.3 Select edge polarity and event qualification

The multiplexer provides four independent edge polarity (rising/ falling edge) selections for four capture events. On the falling edge, the capture register can be loaded, and edge events are gated into their respective capture registers via the modulo-4 counter.

The modulo-4 sequencer can qualify events for up to four edges.

32.6.4 Select continuous/single mode

The operation of CAP in continuous/single mode is as follows:

- Unless stopped, the modulo-4 counter continues to count in a 0->1->2->3->0 sequence.
- The 2-bit modulo-4 counter increments using edge qualification events CEVTx (x=1...4)

- STOP_WRAP is used during single-shot mode operation to compare the output of the modulo-4 counter. If the output matches, the modulo-4 counter is stopped, and further loading of the capture register is disabled. In this mode, the timestamp counter can be configured to reset upon capture event x using the CTRRSTx bit. If the counter equals STOP_WRAP and the REARM bit is not set, the counter will still reset at capture event x.

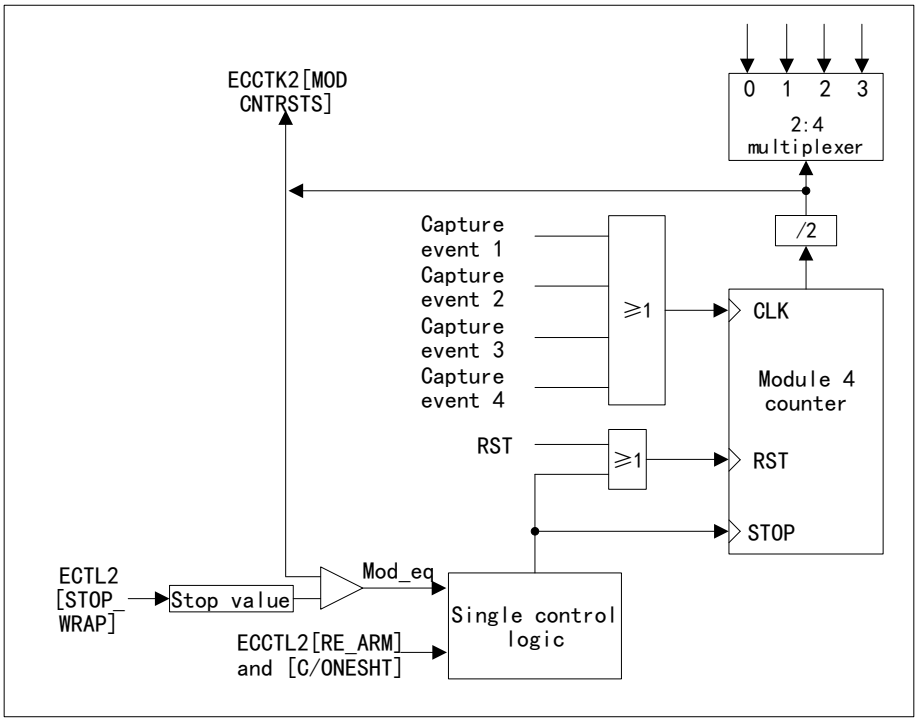
The continuous/single block uses a mono-shot type action to control the reset, start, and stop functions of the modulo-4 counter. This action can be triggered by the stop value comparator, with software controlling the reload process. When the action is triggered, the CAP module will wait for capture events 1/2/3/4 defined by the stop value, and freeze the contents of the capture register and the modulo-4 counter.

The CAP module is prepared for the capture sequence by reloading it. When the CAPLDEN bit is set, reloading will reset the modulo-4 counter to 0 and allow the capture register to be reloaded on capture events.

In continuous mode, the modulo-4 counter continues to operate in a 0->1->2->3->0 sequence, ignoring single-shot mode operation, and the capture values are continuously written to the capture register in a cyclic buffer sequence.

Note: When the user performs a reArm operation after enabling the counter during the CAP initialization stage, if there is an external signal input at that time, reading data via interrupts may result in one additional capture than expected. This phenomenon is related to the system clock and the frequency of external input signals. It is recommended that users avoid performing the reArm operation during the CAP initialization phase.

Figure 134 Continuous/Single Block Structure Block Diagram



32.6.5 Counter and phase registers

The counter uses the APB clock for timing and provides time base for capture events.

The phase register enables both hardware and software forced synchronization, allowing synchronization with other counters.

When APWM mode is used, if phase offset between modules is required, this function can be utilized.

At any of the four event loads, the counter can be reset, and time-difference capture can make use of this function. This is achieved by first capturing the counter value and then using any LDx signal to reset the counter value to 0.

32.6.6 Synchronization

The CAP module achieves synchronization between modules by selecting a common synchronization input source. The synchronization input source can be selected from external synchronization input signals from PWM, CAP, or X-Bar, or from a software synchronization input signal. As shown in the counter and synchronization function structure block diagram, the synchronization signal for the CAP module needs to undergo a logical "OR" operation with SWSYNC. The synchronization signal can be selected through the CAPSYNCINSEL [SEL] .

Note: The CAPxSYNCOUT pins entering the CAPSYNCIN multiplexer are disabled. For example, CAP1SYNCOUT cannot synchronise with CAP1, but CAP1SYNCOUT can synchronise with CAP2, CAP3, and so on.

Figure 135 Counter and Synchronization Function Structure Block Diagram

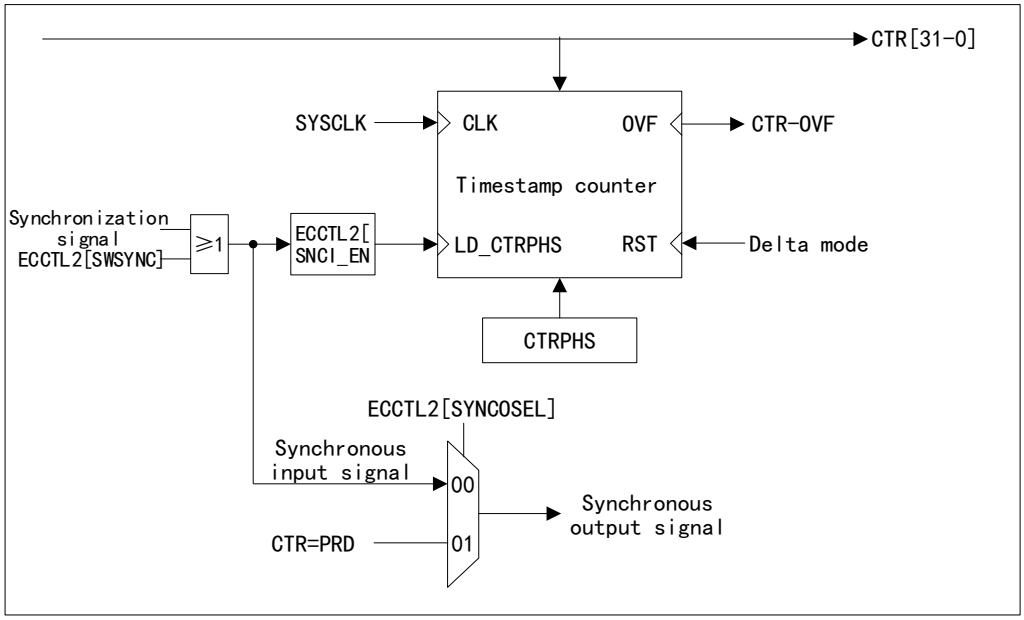
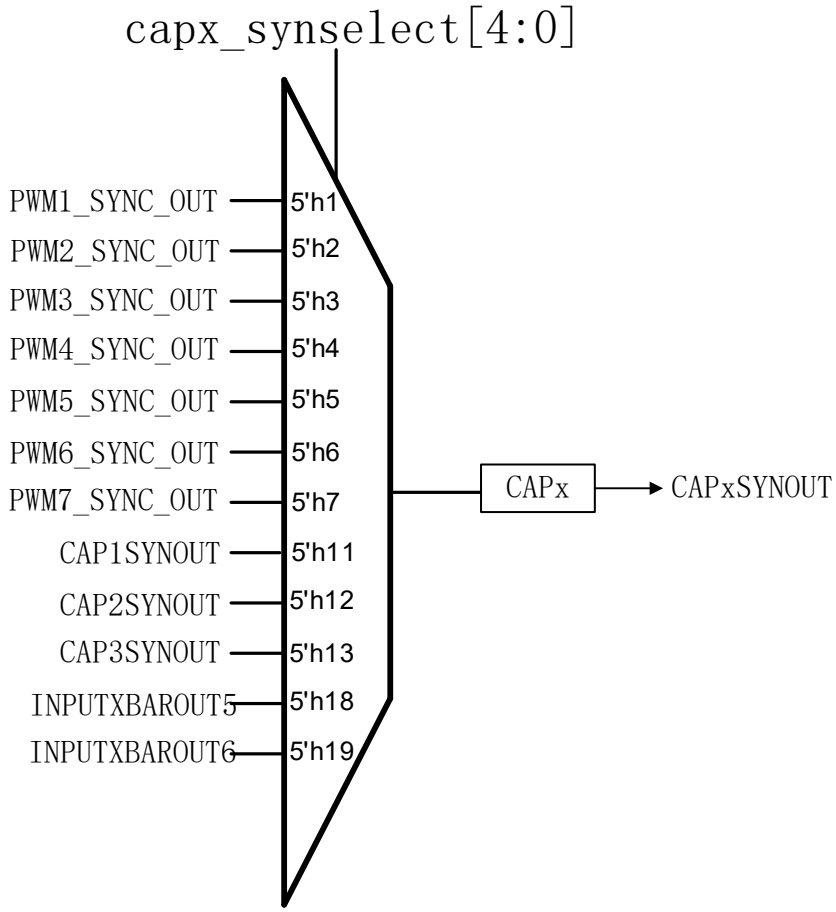


Figure136 : Timing Counter Synchronisation Scheme



Note: The CAPxSYNOUT corresponding to CAPx (x=1, 2, 3) is not fed back into the Mux; the corresponding input is 0. For example, input 5'h11 for CAP1 is 1'b0, rather than CAP1SYNOUT.

Each CAP has a corresponding CAPxSYNCINSEL register bit field (please refer

to the description of the SYNCSELECT5 register), used to select an external synchronisation input. By selecting a common synchronisation input source, the various CAP modules can be synchronised with one another. The CAP's synchronisation input source can be a software synchronisation input or an external synchronisation input. External synchronisation input signals may originate from PWM, CAP or X-Bar.

32.6.7 Timestamp capture registers

When the LD input of the timestamp capture registers is strobed, the capture register is provided by the 32-bit counter timer bus, CTR[0-31], and the loaded capture timestamp.

When CAPLDEN=0, loading of the capture registers is disabled. If a stop condition occurs during single-shot mode operation, the CAPLDEN bit is automatically cleared, preventing loading of the capture registers on a capture event, and the stop value is set to Mod4.

32.6.8 Shadow register

In APWM mode, when shadow loading is active, the following options are allowed:

- When the period is equal, PRD[31:0] = CTR[31:0]
- The new values written to the APRD or ACMP registers are immediately transferred to Capture Register 1 or Capture Register 2

In CAP mode, this logic locks any shadow loading from the APRD and ACMP registers into Capture Register 1 or Capture Register 2.

32.6.9 Interrupt

Events generating interrupt:

- Counter overflow event: FFFFFFFF -> 00000000
- APWM event: CTR = PRD, CTR = CMP
- Capture event: CEVTx, CTROVF

Due to the interrupt handling characteristics of the CPU used by G32R502, it is not recommended to use the system-level Clear Pending statement in CAP and HRCAP interrupt programs. If this statement is used, it may be impossible to enter the interrupt again during the program execution in high-frequency capture. If it is not used, there will be no such problem.

Interrupt control operations

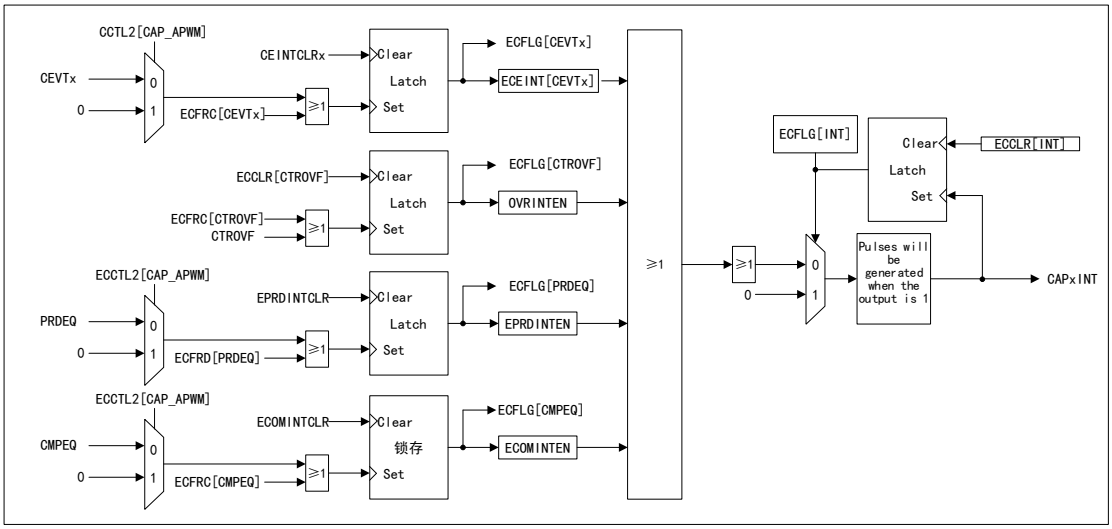
Capture events are edge- and sequencer-limited through modulo-4 gating and polarity selection, sorted by time. Any event from the CAPx module can be selected as an interrupt source to enter the NVIC.

An interrupt pulse will only be generated by the NVIC if any interrupt event is enabled, with its INT set to 0 and the corresponding interrupt flag bit set to 1.

The interrupt service routine must clear the service event and global interrupt flag bit using the ECCLR register before generating any other interrupt pulses. In the case of event filter reset, all interrupt flags can be cleared by setting the CTRFILTRESET. Interrupt events can be forcibly generated using the ECFRC register.

Note: CEINTFLG is valid only in CAP mode (CAP_APWM=0). CTR_PRD and CTR_CMP are valid only in APWM mode (CAP_APWM=1). CTROVF is valid in both CAP mode and APWM mode.

Figure 137 Interrupt



DMA

Table 154 Difference of DMA between Type 0 CAP and Type 1 CAP

CAP type	Functional description	Difference
Type 0 CAP	The CPU needs to use DMA to start data transfer.	Type 1 CAP adds a separate DMA trigger
Type 1 CAP	The DMA trigger allows continuous transfer of capture data from the CAP register to the on-chip memory. Based on DMAEVTSEL, any of the four interrupt events (CEVTx, x=1...4) can be selected as the trigger source for the DMA trigger.	

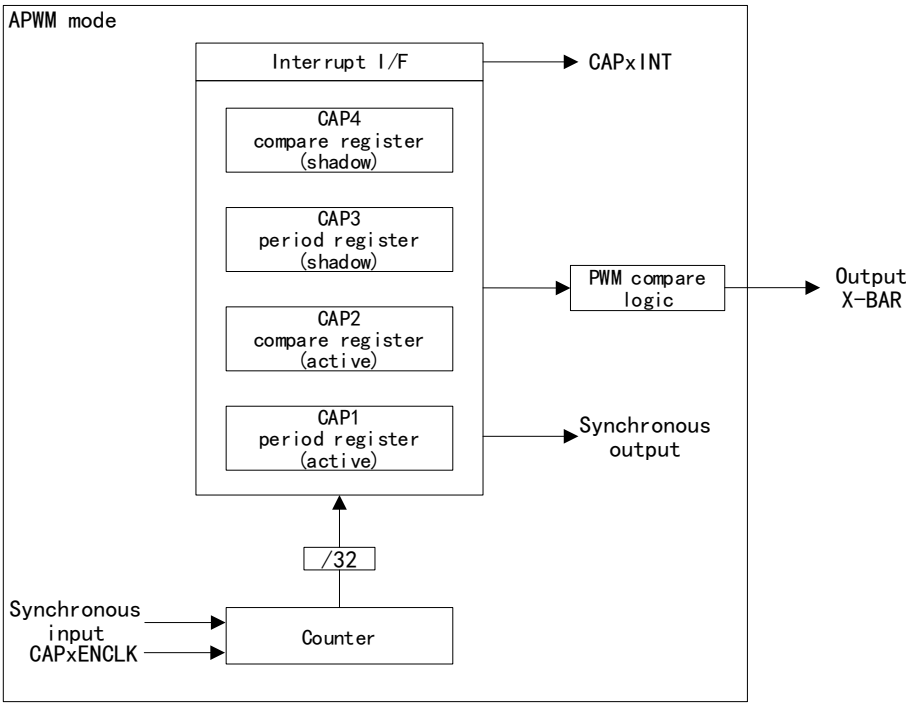
32.7 Function description of APWM mode

32.7.1 APWM structure block diagram

If the CAP module is not used for input capture, it can generate a single-channel PWM. When the counter is in count-up mode, it provides a time base for asymmetric PWM waveforms. Writing to the capture register 3/4 can call the shadow mode.

Note: CAP mode and APWM mode share a pin. This pin is an output pin in APWM mode.

Figure 138 APWM Structure Block Diagram



32.7.2 APWM mode operation

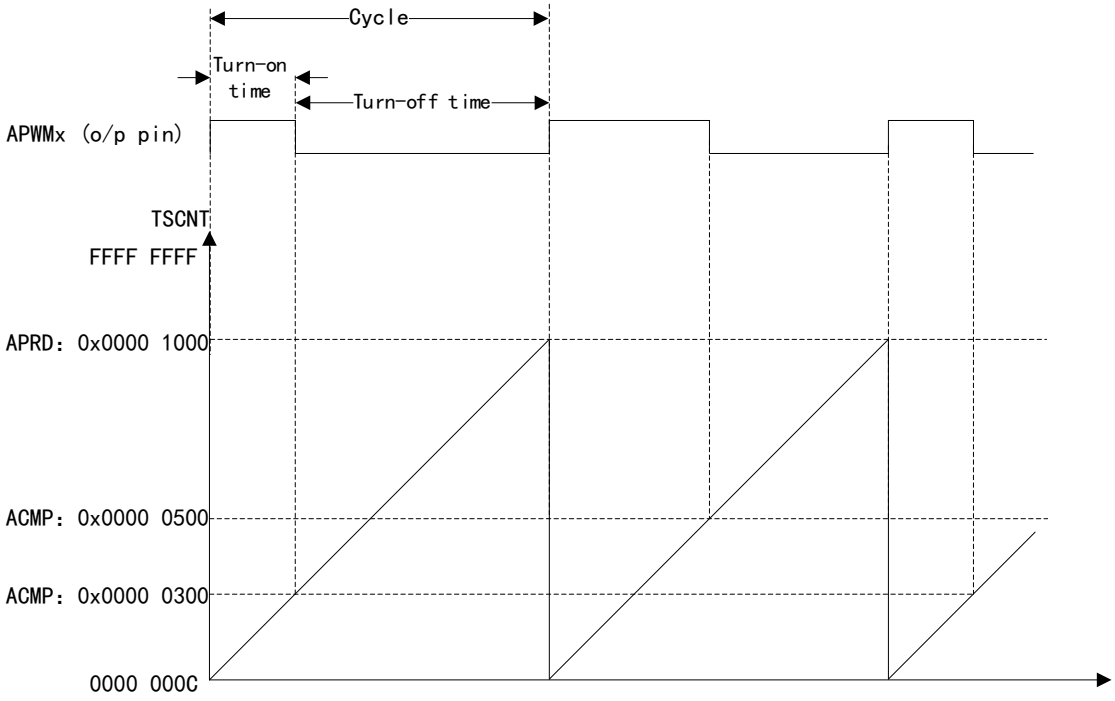
When the CAP module is configured as a PWM generator, the APWMx output pin generates a single-channel PWM waveform. When APWMPOL=0 (PWM polarity is active at high level), the comparison value in the CAP2 comparison register represents the high level of the cycle. When APWMPOL=1, the comparison value represents the low level of the cycle.

Table 155 Difference of DMA between Type 0 CAP and Type 1 CAP

Select PWM output polarity	Operation method
APWMPOL=0	CMP = 0x0000 0000, output low level for one cycle; at this point, duty cycle is 0%. CMP = 0x0000 0001, output high level for 1 cycle. CMP = 0x0000 0002, output high level for 2 cycles CMP = Period, except outputting high level for one cycle, duty cycle is less than 100%. CMP = Period + 1, output high level for the entire period, duty cycle is 100% CMP > Period + 1, output high level for the entire period
APWMPOL=1	CMP = 0x0000 0000, output high level for one cycle; at this point, duty cycle is 0%. CMP = 0x0000 0001, output low level for 1 cycle. CMP = 0x0000 0002, output low level for 2 cycles CMP = Period, except outputting low level for one cycle, duty cycle is less than 100%. CMP = Period + 1, output low level for the entire period, duty cycle is 100% CMP > Period + 1, output low level for the entire period

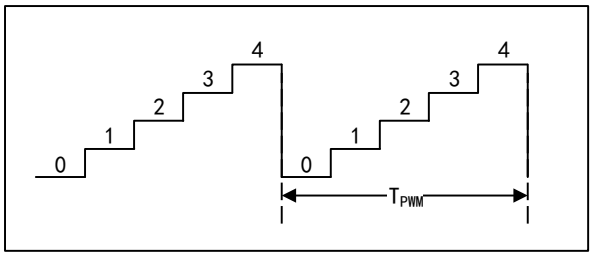
The PWM waveform for APWM mode operations is shown below.

Figure 139 PWM Generated Waveform



Based on the time base period diagram, the calculation formula for the time base frequency and period is:

Figure 140 Time Base Period Diagram



$$T_{PWM} = (CAP1 = 1) * T_{TSCNT}$$

$$F_{PWM} = \frac{1}{T_{PWM}}$$

32.7.3 Impact of Counter Comparison and PRD on CAP Output in APWM Mode

Based on the counter comparison and PRD structure block diagram, the CAP output waveform in APWM mode is shown below.

Figure 141 Counter Compasion and PRD Structure Block Diagram

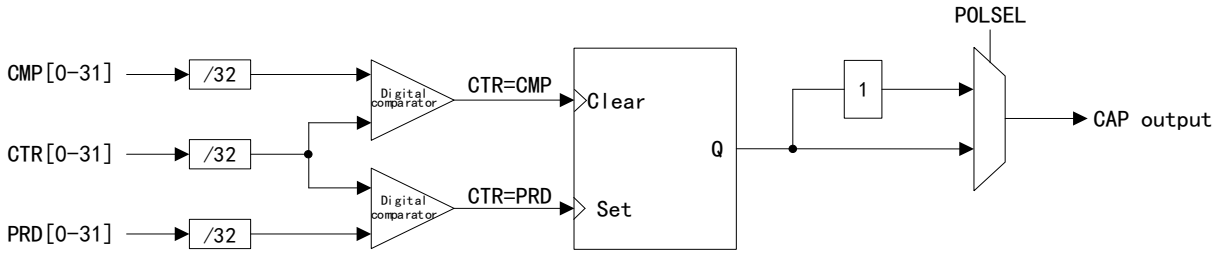
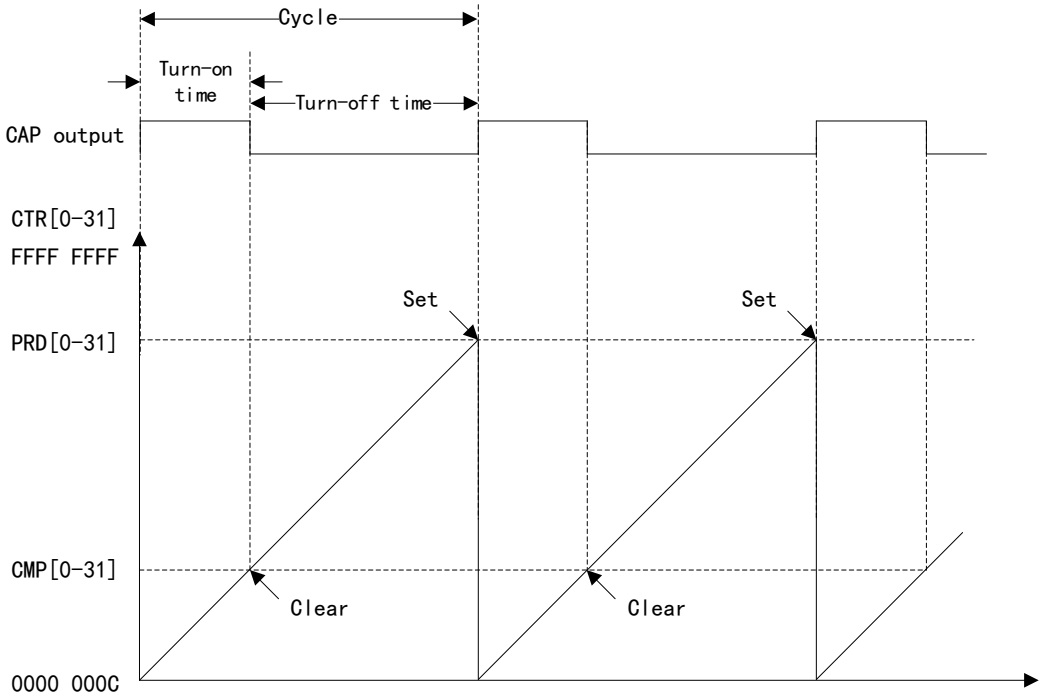


Figure 142 CAP Output Waveform in APWM Mode



32.8 CAP module application

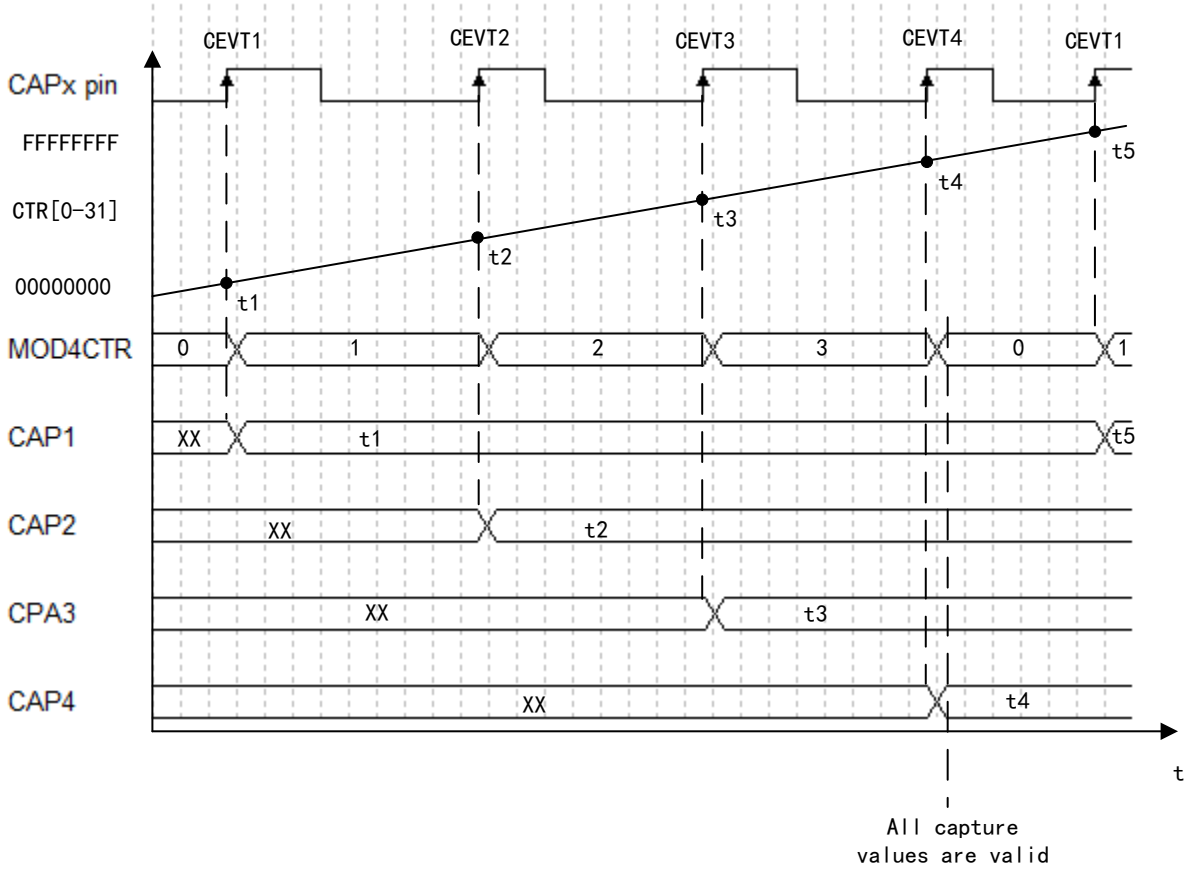
32.8.1 Absolute timestamp operation

Example of continuous capture operation in modulo-4 counter loop is as follows.

When the TSCTR is counting up (without reset), and the capture event is limited to the rising edge, it provides frequency and period information.

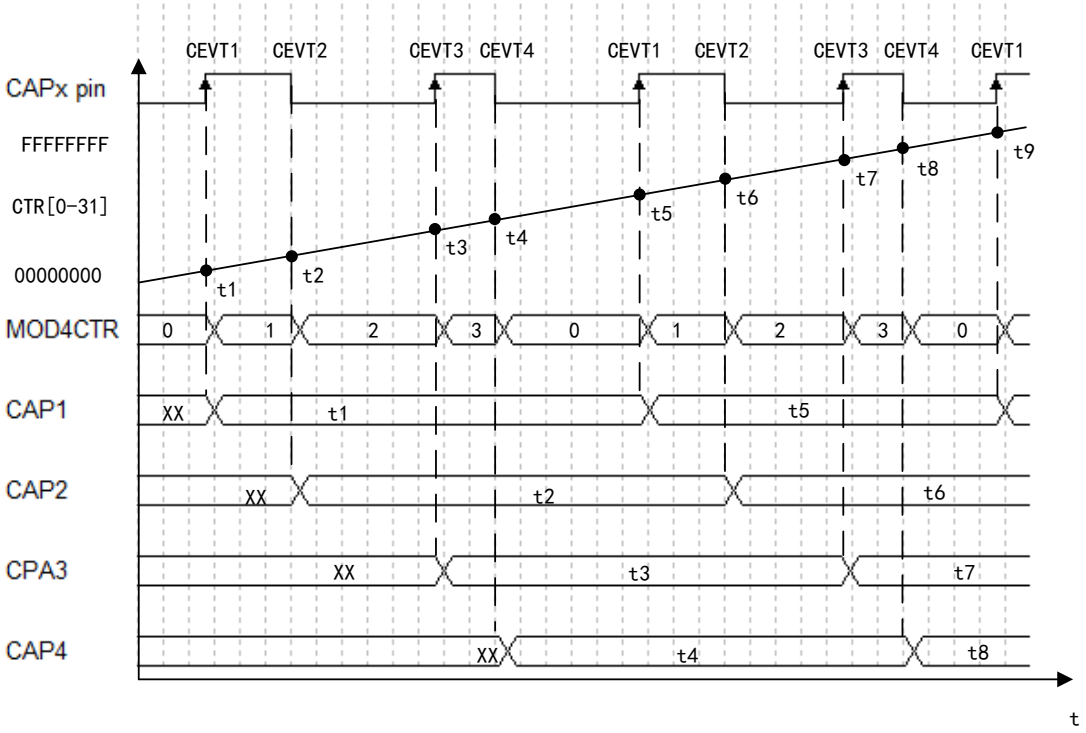
In a capture event, the timestamp counter value is captured first, then the modulo-4 counter increments to the next state. If the TSCTR reaches the maximum value FFFFFF, it will roll over to 00000000, and the CTROVF flag will be set, generating an interrupt (when TSCTR is enabled). The captured timestamp is valid after the fourth capture event (as shown in the figure below), so capture event 4 can be used to trigger an interrupt, allowing the CPU to read data from the capture register.

Figure 143 Rising Edge Detection and Absolute Timestamp Capture Sequence



When the TSCTR is counting up (without reset), and the capture event is limited to either the rising edge or the falling edge, providing duty cycle and period information. For example, Period 1 = t3 - t1, Duty cycle 1 (on-time %) = (t2 - t1) / Period 1 * 100%, Duty cycle 1 (off-time %) = (t3 - t2) / Period 1 * 100%, and so on.

Figure 144 Absolute Timestamp Capture Sequence with Rising and Falling Edge Detection



32.8.2 Time difference operation

The CAP module can be used to collect Delta timing data in a pulse train waveform.

When capture mode is used continuously, the TSCTR is counting up (without reset), and the modulo-4 counter is cycling. The capture event is limited to the rising edge. In Delta-time mode, the TSCTR is reset to 0 at each valid event.

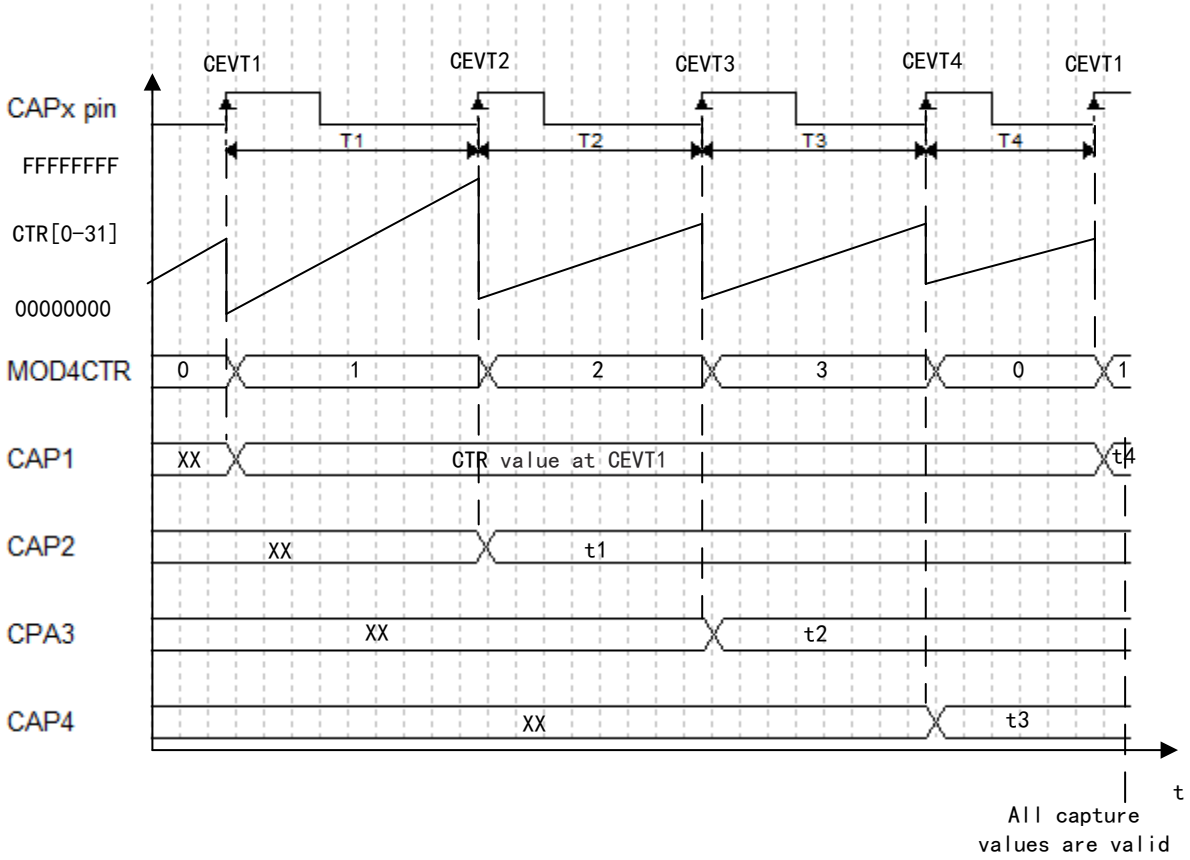
The sequence of a capture event is:

- Capture the timestamp counter value
- Reset the TSCTR
- Increment the modulo-4 counter to the next state.

If the TSCTR reaches the maximum value FFFFFF before the next event, it will roll over to 00000000 and continue counting. At this point, the CTROVF flag bit will be set, and an interrupt will be generated when the TSCTR is enabled.

Based on T1 = Period 1, T2 = Period 2, etc., the Delta-time mode allows the CAPx content to directly provide timing data without CPU calculation, as shown in the figure below. The trigger point for reading the timing data is capture event 1, where periods 1 to 4 are valid.

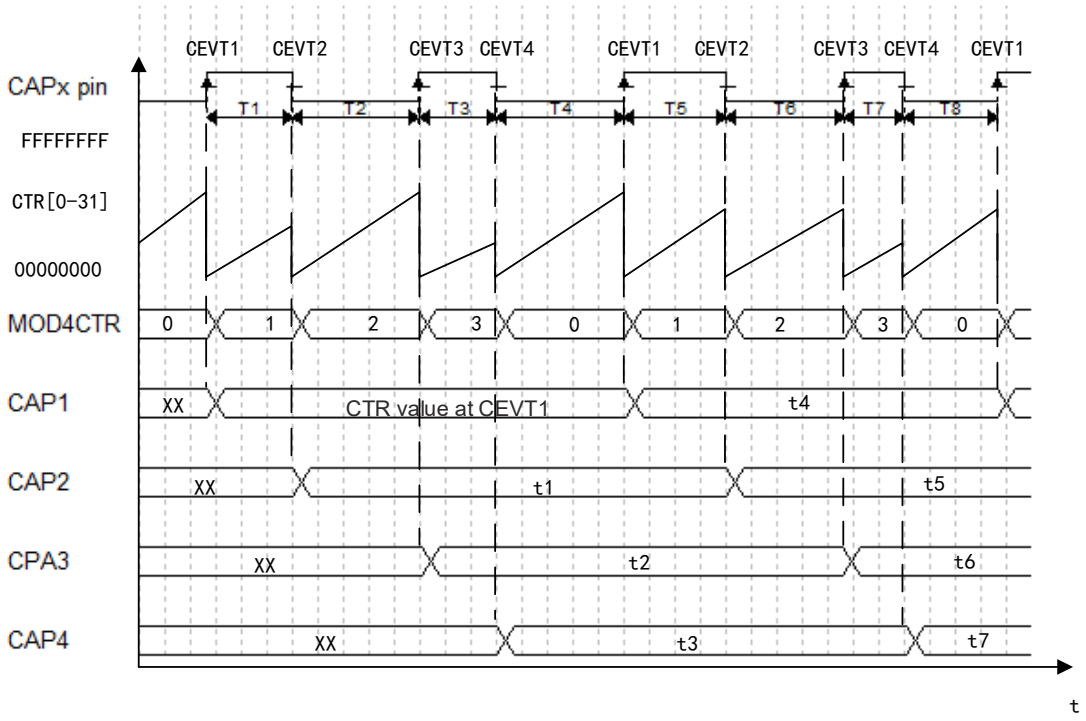
Figure 145 Rising Edge Detection and Delta-mode Timestamp Capture Sequence



When the capture event is limited to either the rising edge or the falling edge, it provides duty cycle and period information, i.e., T1 + T2 = Period 1, Duty cycle 1 (on-time %) = T1 / Period 1 * 100%, Duty cycle 1 (off-time %) = T2 / Period 1 * 100%, and so on.

During initialization, it is required to write active registers within two periods and compare them; at this time, it will automatically copy the initial value to the shadow value. For subsequent comparisons and updates during runtime, shadow registers must be used.

Figure 146 Delta-mode Timestamp Capture Sequence with Rising and Falling Edge Detection



32.9 Register bank address

Table 156 Register Bank Address

Device register	Register bank	Start address	End address
CAP1Regs	CAP_REGS	0x4002_0400	0x4002_07FF
CAP2Regs	CAP_REGS	0x4002_0800	0x4002_0BFF
CAP3Regs	CAP_REGS	0x4002_0C00	0x4002_0FFF

32.10 Register address mapping

Table 157 CAP_REGS Register Bank Address Mapping

Register name	Description	Offset address	WRPRT
TSCTR	Timestamp counter register	0x00	-
CTRPHS	Counter phase offset register	0x04	-
CAP1	Capture register 1	0x08	-
CAP2	Capture register 2	0x0C	-
CAP3	Capture register 3	0x10	-
CAP4	Capture register 4	0x14	-
ECCTL0	Capture control register 0	0x24	√

Register name	Description	Offset address	WRPRT
ECCTL1	Capture control register 1	0x28	√
ECCTL2	Capture control register 2	0x2A	√
ECEINT	Enable capture interrupt register	0x2C	√
ECFLG	Enable interrupt flag register	0x2E	-
ECCLR	Clear capture interrupt register	0x30	-
ECFRC	Force capture interrupt register	0x32	√
CAPSYNCINSEL	Synchronisation Source Selection Register	0x3C	√

32.11 Register functional description

32.11.1 Timestamp counter register (TSCTR)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	TSCTR	R/W	Time Stamp Counter (1) This register is not writable; the HR counter value can be read (2) It can be reset using the CTRFILTRESET bit (3) If not synchronized to the APBCLK domain, inaccurate readings will be caused	0h

32.11.2 Counter phase offset register (CTRPHS)

Offset address: 0x04

Reset type: SYSRSn

The counter phase offset register can be programmed for phase lead/lag. This register is loaded into the TSCTR register upon a synchronization input event or by software forcing through a control bit, enabling phase-synchronized control with the PWM time base and other CAP instances. CTRPHS is not applicable in HR mode

Field	Name	R/W	Description	Reset value
31:0	CTRPHS	R/W	Counter Phase Offset Value	0h

32.11.3 Capture register 1 (CAP1)

Offset address: 0x08

Reset type: SYSRSn

In APWM mode, this register becomes the active period register.

Field	Name	R/W	Description	Reset value
31:0	CAP1	R/W	Capture 1 This bit field can be written (loaded) in the following ways: (1) Software: Able to be used for initialization or testing purposes (2) CAP3 register used in APWM mode (ARPD shadow register) (3) TSCTR during capture events	0h

32.11.4 Capture register 2 (CAP2)

Offset address: 0x0C

Reset value: 0x0000 0000

Reset type: SYSRSn

In APWM mode, this register becomes the active compare register.

Field	Name	R/W	Description	Reset value
31:0	CAP2	R/W	Capture 2 This bit field can be written (loaded) in the following ways: (1) Software: Able to be used for testing purposes (2) CAP4 register used in APWM mode (ACMP shadow register) (3) TSCTR during capture events	0h

32.11.5 Capture register 3 (CAP3)

Offset address: 0x10

Reset type: SYSRSn

In CMP mode, this register is the timestamp capture register. In APWM mode, this register is the APRD (CAP1 period shadow) register, which can be used to update the PWM period value.

Field	Name	R/W	Description	Reset value
31:0	CAP3	R/W	Capture 3 In APWM mode, CAP3 can shadow CAP1.	0h

32.11.6 Capture register 4 (CAP4)

Offset address: 0x14

Reset type: SYSRSn

In CMP mode, this register is the timestamp capture register. In APWM mode, this register is the ACMP (CAP2 compare shadow) register, which can be used to update the PWM compare value.

Field	Name	R/W	Description	Reset value
31:0	CAP4	R/W	Capture 4 In APWM mode, CAP4 can shadow CAP2.	0h

32.11.7 Capture control register 0 (ECCTL0)

Offset address: 0x24

Reset type: CPU1. SYSRSn

Field	Name	R/W	Description	Reset value
6:0	INPUTSEL	R/W	Capture input source select 0000000: CAP input [0] 0000001: CAP input [1] 0000010: CAP input [2] … 11111111: CAP input [127]	7Fh
31:7	Reserved			0h

32.11.8 Capture control register 1 (ECCTL1)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2 (x-1)	CAPxPOL	R/W	Capture event x polarity select (x = 1...4) 0: Capture event x triggered by rising edge 1: Capture event x triggered by falling edge	0h
2x-1	CTRRSTx	R/W	Capture event x counter reset (x = 1...4) 0: Not reset counter of capture event x (absolute timestamp) 1: Reset counter after timestamp of capture event x (for delta mode operation)	0h
8	CAPLDEN	R/W	Enable load CAP1/2/3/4 register on capture event This bit does not disable the generation of CEVTx events. 0: Disable 1: Enable	0h
13:9	PRESCALE	R/W	Event filter prescale select 00000: No frequency division 00001: 2 frequency division 00010: 4 frequency division 00011: 6 frequency division 00100: 8 frequency division 00101: 10 frequency division 11110: 60 frequency division 11111: 62 frequency division	0h
15:14	FREE_SOFT	R/W	TSCTR counter run mode select 00: When simulation stops, TSCTR counter immediately stops 01: TSCTR counter continues running until TSCTR reaches 0 Others: TSCTR counter runs freely, unaffected by simulation suspend	0h

32.11.9 Capture control register 2 (ECCTL2)

Offset address: 0x2A

Reset type: Except for CTRFILTRESET, DMAEVTSEL, and MODCNTRSTS bit fields, reset type is CPU1. SYSRSn, and the other bit fields reset type is SYSRSn

Field	Name	R/W	Description	Reset value
0	CONT_ONESHT	R/W	Operate mode select This bit only applies to CAP mode. 0: Continuous mode 1: One-shot mode	0h
2:1	STOP_WRAP	R/W	Select how to run after capture events in capture / one-shot mode The number of capture registers is between 1 and 4, with the circular buffer looping and restarting as the loop value in continuous mode. Before the capture sequence of CAP1/2/3/4 registers stops (when the registers are frozen), the capture count of 1-4 is the stop value for one-shot mode. When STOP_WRAP equals the modulo-4 counter, the following actions will occur: (1) Disable loading of CAP1/2/3/4 registers (2) Stop (freeze) the modulo-4 counter Prevent interrupt events in one-shot mode until reloading. 00: In continuous mode, loop after capture event 1; in one-shot mode, stop after capture event 1 01: In continuous mode, loop after capture event 2; in one-shot mode, stop after capture event 2 10: In continuous mode, loop after capture event 3; in one-shot mode, stop after capture event 3 11: In continuous mode, loop after capture event 4; in one-shot mode, stop after capture event 4	3h
3	REARM	R/S	Re-Arming control Re-control is valid in continuous or one-shot mode. 0: Invalid, reading returns 0 1: Enable one-shot sequence as follows: (1) Enable loading of CAP1/2/3/4 registers (2) Reset the modulo-4 counter to 0 (3) Unfreeze the modulo-4 counter	0h
4	TSCTRSTOP	R/W	TSCTR run mode 0: Stop 1: Free run	0h
5	SYNCI_EN	R/W	Sync-in mode enable 0: Disable 1: Enable TSCTR loading from the CTRPHS register based on a software forced event or synchronization signal	0h
7:6	SYNCO_SEL	R/W	Sync-out signal select	0h

Field	Name	R/W	Description	Reset value
			00: Select the synchronization output signal as synchronization input event 01: Select the synchronization output signal as CTR=PRD event Others: Disable the synchronization output signal	
8	SWSYNC	R/S	Software-forced TSCTR synchronizer This bit provides a method for the user to generate software-generated synchronization pulses. In APWM mode, synchronization pulses can be generated through the CTR=PRD event. 0: Invalid, reading returns 0 1: Force the current CAP module and any CAP modules downstream that conform to SYNCO_SEL=00 to execute TSCTR shadow load operation. After writing 1, this bit will return to 0	0h
9	CAP_APWM	R/W	CAP/APWM operating mode select 0: CAP mode. This mode will have the following configurations by force: (1) CAPx/APWMx pins as capture input (2) User can enable capture register load (3) Suppress shadow load of CAP1 and CAP2 registers (4) Suppress timestamp counter reset via CTR=PRD event 1: APWM mode. This mode will have the following configurations by force: (1) CAPx/APWMx pins as APWM output (2) Disable timestamp load into capture registers (3) CAP1 and CAP2 registers can undergo shadow load (4) Timestamp counter reset via CTR=PRD event (period boundary)	0h
10	APWMPOL	R/W	APWM output polarity select This bit only applies to APWM mode. 0: High level, the comparison value defines the duration of the high level 1: Low level, the comparison value defines the duration of the low level	0h
11	CTRFILTRESET	RC_W1	Reset This bit can avoid capturing false inputs when configuring CAP and enables the capture module from a known state. 0: No effect 1: Reset the modulo counter, TSCTR, event filter, CEINTFLG, CTROVF, and HR error flags	0h
13:12	DMAEVTSEL	R/W	DMA interrupt source select 00: Capture event 1 01: Capture event 2 10: Capture event 3	0h

Field	Name	R/W	Description	Reset value
			11: Capture event 4	
15:14	MODCNTRSTS	R	Reads modulo counter status This field indicates the current status of the modulo counter. 00: CAP1 is loaded in the next capture event 01: CAP2 is loaded in the next capture event 10: CAP3 is loaded in the next capture event 11: CAP4 is loaded in the next capture event	0h

32.11.10 Enable Capture Interrupt Register (ECEINT)

Offset address: 0x2C

Reset type: SYSRSn

This register is used to enable/disable individual interrupt event sources.

Field	Name	R/W	Description	Reset value
0	Reserved			0h
x	CEVTx	R/W	Capture Event x Interrupt Enable (x=1...4) 0: Disable 1: Enable	0h
5	CTROVF	R/W	Counter Overflow Interrupt Enable 0: Disable 1: Enable	0h
6	CTR_EQ_PRD	R/W	Counter Equal Period Interrupt Enable 0: Disable 1: Enable	0h
7	CTR_EQ_CMP	R/W	Counter Equal Compare Interrupt Enable 0: Disable 1: Enable	0h
15:8	Reserved			0h

32.11.11 Capture Interrupt Flag Register (ECFLG)

Offset address: 0x2E

Reset type: SYSRSn

This register indicates whether any interrupt events occurred, including the global interrupt flag bit.

Field	Name	R/W	Description	Reset value
0	INT	R	Global Interrupt Flag 0: No interrupt occurs 1: Generate interrupt	0h
x	CEVTx	R	Capture Event x Interrupt Flag (x=1...4) This flag bit is active only in capture mode. 0: No interrupt occurs 1: CAPx pin generates capture event x interrupt, x=1...4	0h

Field	Name	R/W	Description	Reset value
5	CTROVF	R	Counter Overflow Interrupt Flag This flag bit is active in both APWM mode and capture mode. 0: No interrupt occurs 1: TSCTR changes from FFFFFFFF to 00000000	0h
6	CTR_PRD	R	Counter Equal Period Interrupt Flag This flag bit is active only in APWM mode. 0: No interrupt occurs 1: TSCTR reaches APRD (period register value) and resets	0h
7	CTR_CMP	R	Counter Equal Compare Interrupt Flag This flag bit is active only in APWM mode. 0: No interrupt occurs 1: TSCTR reaches ACMP (compare register value)	0h
15:8	Reserved			0h

32.11.12 Clear Capture Interrupt Register (ECCLR)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	RC_W1	Global Interrupt Flag Clear 0: Invalid, reading returns 0 1: Clear INT, enabling further interrupt generation when any event flag is set.	0h
x	CEVTx	RC_W1	Capture Event x Interrupt Flag Clear (x=1...4) 0: Invalid, reading returns 0 1: Clear CEINTFLG flag	0h
5	CTROVF	RC_W1	Counter Overflow Interrupt Flag Clear 0: Invalid, reading returns 0 1: Clear CTROVF flag	0h
6	CTR_PRD	RC_W1	Counter Equal Period Interrupt Flag Clear 0: Invalid, reading returns 0 1: Clear CTR_PRD flag	0h
7	CTR_CMP	RC_W1	Counter Equal Compare Interrupt Flag Clear 0: Invalid, reading returns 0 1: Clear CTR_CMP flag	0h
31:8	Reserved			0h

32.11.13 Force Capture Interrupt Register (ECFRC)

Offset address: 0x32

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h

Field	Name	R/W	Description	Reset value
x	CEVTx	R-0/W1S	Force Capture Event x Interrupt (x=1...4) 0: Invalid, reading returns 0 1: Set CEINTFLG	0h
5	CTROVF	R	Force Counter Overflow Interrupt 0: Invalid, reading returns 0 1: Set CTROVF	0h
6	CTR_PRD	R	Force Counter Equal Period Interrupt This flag bit is active only in APWM mode. 0: Invalid, reading returns 0 1: Set CTR_PRD	0h
7	CTR_CMP	R	Force Counter Equal Compare Interrupt This flag bit is active only in APWM mode. 0: Invalid, reading returns 0 1: Set CTR_CMP	0h
31:8	Reserved			0h

32.11.14 Synchronisation Source Selection Register (CAPSYNCINSEL)

Offset address: 0x3C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	SEL	R/W	SYNCIN signal source select 0h: Disable synchronisation with CAP 1h: PWM1SYNCOUT 2h: PWM2SYNCOUT 3h: PWM3SYNCOUT 4h: PWM4SYNCOUT 5h: PWM5SYNCOUT 6h: PWM6SYNCOUT 7h: PWM7SYNCOUT 8h – 10h: Reserved 11h: CAP1SYNCOUT 12h: CAP2SYNCOUT 13h: CAP3SYNCOUT 14h – 17h: Reserved 18h: INPUTXBAROUT5 19h: INPUTXBAROUT6 1Ah–1Fh: Reserved	1h
31:5	Reserved			0h

33 Quadrature encoder pulse (QEP)

33.1 Full Name and Abbreviation Description of Terms

Table 158 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Quadrature Mode Adapter	QMA
Quadrature decoder unit	QDU
Position Counter and Control Unit	PCCU
Quadrature Edge-capture Unit	QCAP
Quadrature-clock	QCLK
Unit Position Event	UPE

33.2 Introduction

Quadrature Encoder Pulse (QEP) is used with dedicated interfaces for rotary or linear incremental encoders to obtain direction, speed, and position information from the rotor, for position control and high-performance motion systems.

33.3 Main characteristics

(1) Main functional units include:

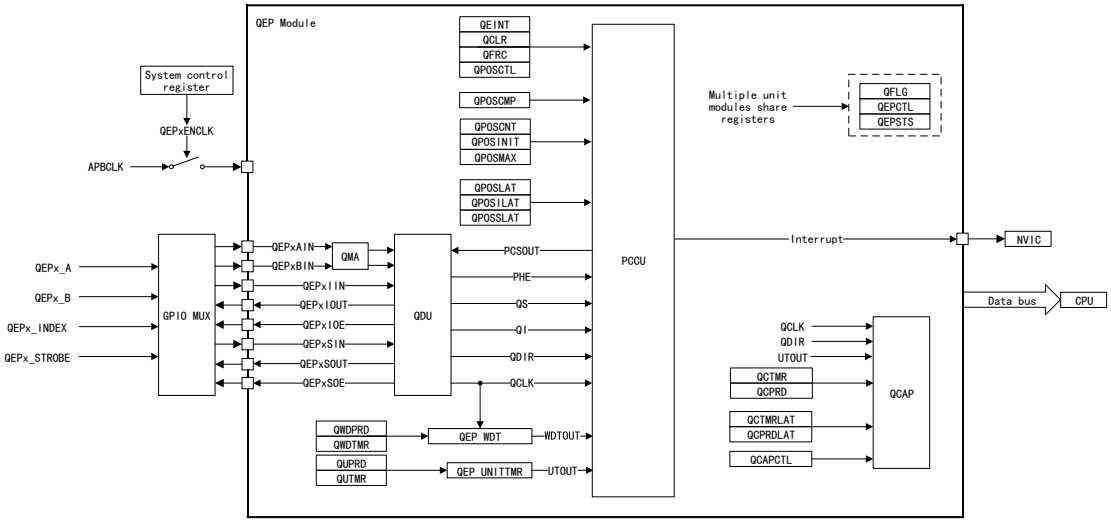
- QMA module
- Quadrature Decoder unit
- Position counter and control unit (for position measurement)
- Quadrature edge capture unit (for low-speed measurements)
- Watchdog timer (for stall detection)
- Unit timebase (for measuring speed/frequency)
- Programmable input per pin defined

(2) 11 interrupt events:

- PCE
- PHE
- QDC
- WTO
- PCU
- PCO
- PCR
- PCM
- SEL
- IEL
- UTO

33.4 Structure block diagram

Figure 147 QEP Structure Block Diagram



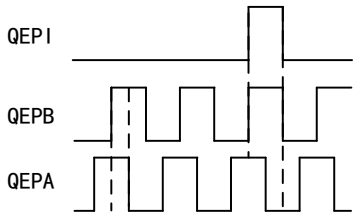
33.5 Functional description

33.5.1 Incremental encoder

Encoder disk

As shown in the encoder disk plane diagram, the incremental encoder disk has a series of channels along its perimeter, forming an alternating light-dark pattern. Disk count refers to the number of light-dark line pairs appearing per rotation. A QEPI index signal, typically generated by adding a second track, produces a signal once per rotation, as shown in the QEPI waveform diagram. The QEPI index signal can be used to indicate the absolute position. Encoder manufacturers identify the index pulse by using terms such as starting position, zero reference, index and markers.

Figure 148 QEP Waveform Diagram



Direction information can be obtained by reading the lines on the disk using two different photoelectric elements. The mechanical displacement between the two photoelectric elements is set to 1/4 of the line pair spacing, which allows the disk pattern to be observed. Mechanical displacement is measured using a

scale or mask that restricts the photoelectric element's view to the required portion of the disk lines. When the disk is rotating, the quadrature QEPA and QEPB signals generated by the two photoelectric elements are electrically phase-shifted by 90 degrees. The QEPA channel is typically defined to be in the clockwise direction relative to the QEPB channel, and vice versa, as shown in the figure below.

Figure 149 QEP Encoder Output Signal in Clockwise Rotation

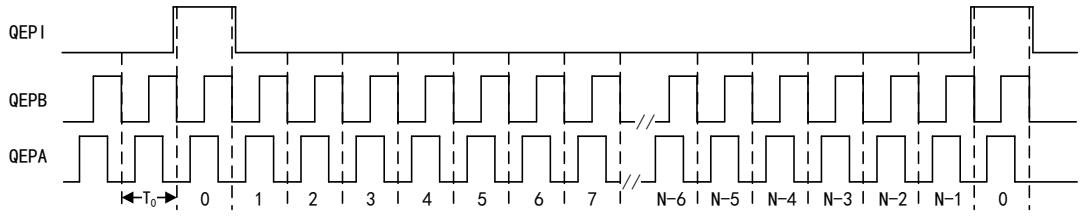
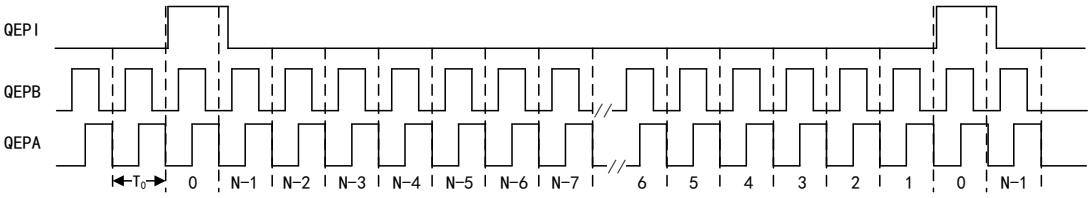


Figure 150 QEP Encoder Output Signal in Counterclockwise Rotation



Typically, each time the motor completes one rotation, the encoder wheel also completes one rotation, or the encoder wheel can be in a gear transmission ratio (relative to the motor). Thus, the speed of the motor is proportional to the frequency of the QEPA and QEPB output digital signals. The CPU can determine the motor speed by measuring the frequency of the QEPA or QEPB output. For example, a 2000-line encoder directly coupled to a motor running at 5000 RPM generates a frequency of 166.6 kHz.

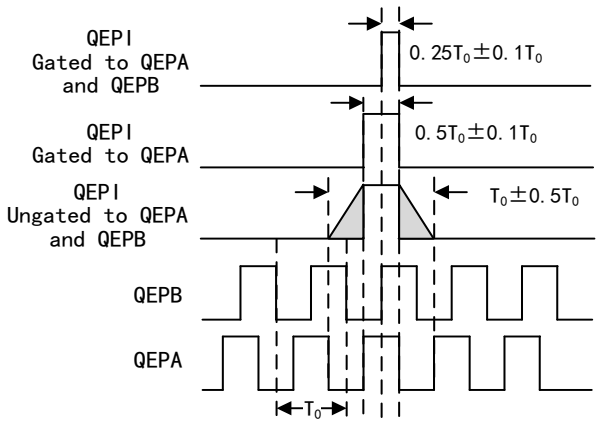
Index pulse

Among different manufacturers, the index pulse of a quadrature encoder can be classified into:

- Gated index pulse
- Non-gated index pulse

Non-standard form of the index pulse is the non-gated index pulse. When configured as non-gated, the index edge may not necessarily align with the QEPA and QEPB signals. The width of the index pulse can be equal to the full period, half-period, or quarter-period of the quadrature signal, and the strobed index pulse aligns with one of the four quadrature edges.

Figure 151 Index Pulse Waveform Diagram



Shaft encoder

Typical applications for shaft encoders:

- Robot
- Computer input in the form of a mouse

The position of the left/right and up/down axes of the mouse ball's rotation can be seen inside the mouse. Since these axes are connected to an optical shaft encoder, the computer can effectively determine the speed and direction of the mouse movement.

Different approximations of speed

In motor control, there is a cost-effective strategy to estimate the speed from digital position sensors. When speed estimation is required at both low and high speeds, one approach is to use an accurate estimation formula at low speeds and, once the motor speed rises to a certain threshold, switch to the traditional speed estimation formula via software.

(1) The traditional speed estimation formula is:

$$v(k) \approx \frac{\mathrm{x(k)} - \mathrm{x(k-1)}}{\mathrm{T}} = \frac{\Delta X}{T}$$

Table 159 Supplementary Description

Parameter	Description
v (k)	Speed at time k
x (k)	Position at time k
x (k-1)	Position at time k-1
ΔX	Incremental position movement per unit time
T	Reciprocal of rate calculated with the fixed unit time or speed

In order to provide a unit time event for speed calculation, the traditional estimation method requires a time base. Typically, the unit time is the reciprocal of the rate calculated with speed. In each unit time event, the encoder count is read once. By subtracting the previous reading from the current reading, we get x(k)−x(k−1), and then calculate the speed by multiplying the reciprocal of the constant time T between known unit time events.

The traditional speed estimation method has inherent precision limitations directly related to the following factors:

- Unit time period T
- Resolution of position sensor

For example, considering a 500-line quadrature encoder, its rate calculated with the speed is 400Hz. When this encoder is used for positioning, its resolution is increased fourfold, meaning there are 2,000 counts per revolution. Thus, the minimum detectable rotation is 0.0005 revolutions, and with a sampling rate of 400Hz, the speed resolution would be 12 rpm. However, because this resolution introduces errors equal to zero for speeds lower than 12 rpm, this method is only suitable for medium or high-speed ranges.

(2) The precise speed estimation formula is:

$$v(k) \approx \frac{\mathrm{X}}{t(k) - t(k-1)} = \frac{X}{\Delta T}$$

Table 160 Supplementary Description

Parameter	Description
v (k)	Speed at time k
t (k)	Time at time k
t (k-1)	Time at time k-1
X	Fixed unit position
ΔT	Incremental time elapsed during unit position movement

At low speeds, the speed can be estimated more accurately using the above formula. However, it requires a position sensor (e.g. quadrature encoder) to provide output with fixed interval pulse trains. The width of each pulse is defined by the motor speed for a given sensor resolution. The precise speed estimation method is to calculate motor speed by measuring the elapsed time between consecutive edges of the quadrature pulse. However, this method has opposite limitations compared with traditional speed estimation methods. The combination of high sensor resolution and relatively high motor speed leads to small time intervals ΔT, which are more affected by timer resolution. Therefore, at high speeds, the error increases when using the precise speed estimation method.

33.5.2 Pin configuration

GPIO multiplexer registers must be configured to connect the QEP to the device pins. It is required to configure the GPxGMUX register first, while ensuring the corresponding field of the GPxMUX register remains 0, and then write the desired value to the GPxMUX register to avoid pin faults.

To ensure proper operation of the QEP module, the input GPIO pins must be configured for synchronous input mode through the GPxQSEL register, and QEP input pins cannot use asynchronous mode. Internal pull-up can be configured in the GPxPUD register.

For details on the GPIO multiplexer, refer to the datasheet. For details on GPIO configuration, refer to the GPIO section.

33.5.3 Input modes

There are three input modes for QEP:

- Input strobe signal
- XCLK and XDIR pins
- Index or 0 marker

The QEP module requires the QEPA, QEPB, and QEPI inputs to be synchronized to SYSCLK before entering the module. For any GPIO pin enabling QEP, the application code can enable the synchronous GPIO input function. For more details on GPIO, refer to the GPIO section.

Input strobe signal

When the desired event occurs on the strobe pin, the general strobe signal (QEPS) can initialize or latch the position counter. To notify that the motor has reached a specific position, this signal is typically connected to a sensor or limit switch.

XCLK and XDIR pins

XCLK and XDIR pins can be used for direction count mode or quadrature clock mode.

(1) Direction count mode

In this mode, the clock signal and direction are provided directly from an external source, where the QEPA pin provides the clock input, and the QEPB pin provides the direction input. Some position encoders do not have quadrature outputs but have this type of output.

(2) Quadrature clock mode

The QEP encoder provides two square wave signals (QEPA and QEPB), and the QEPA and QEPB signals are electrically phase-shifted by 90 degrees. This

phase relationship is used to determine the number of QEP pulses from the index position and the rotational direction of the input shaft in order to derive relative position information. The quadrature decoder generates quadrature clock and direction signals through these two input signals. In the clockwise rotation or forward case, the QEPA signal leads the QEPB signal, and vice versa.

Index or 0 marker

The QEP encoder assigns the absolute starting position through the index input signal (QEPI), and from this absolute starting position, it uses quadrature pulses to increment the position information. By connecting this pin to the index output of the QEP encoder, the position counter can be selectively reset at each revolution. When the desired event occurs on the index pin, QEPI can be used to initialize or latch the position counter.

Polarity selection

The negation of each QEP input can be controlled through the QSP, QIP, QBP, and QAP bits. For example, setting the QIP bit will negate the QEPI input.

33.5.4 Timer

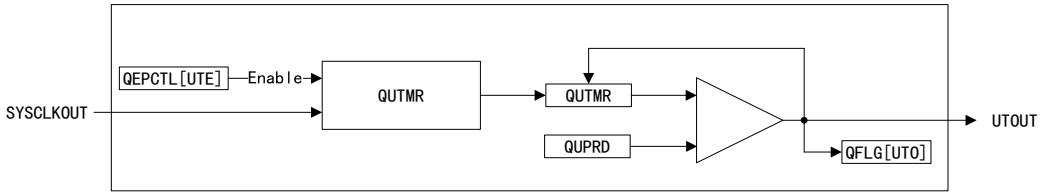
The QEP peripheral includes a 32-bit time base unit timer and a 16-bit watchdog timer.

32-bit timebase unit timer

As shown in the structure block diagram of 32-bit time base unit timer, the 32-bit timer clock source is SYSCLKOUT, and can generate periodic interrupts for speed calculation. If the unit time period value matches the timer value, the unit timer will be reset, and the unit timeout interrupt flag (UTO) will be generated. Whenever QUPRD equals QUTMR, the QUTMR will be reset.

To use the latched value for speed calculation, the QEP peripheral can latch the position counter, capture timer value, and capture period value on the unit timeout event.

Figure 152 Structure Block Diagram of 32-bit Timebase Unit Timer

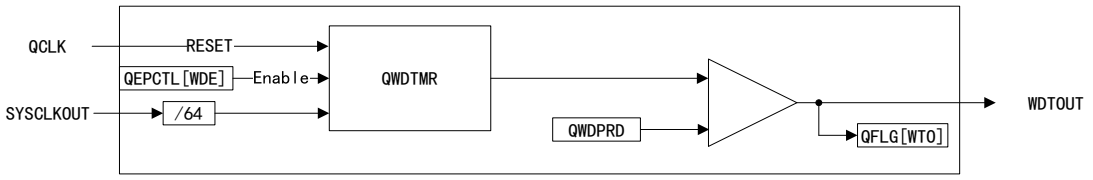


16-bit watchdog timer

The 16-bit watchdog timer is used to monitor the proper operation of the motion

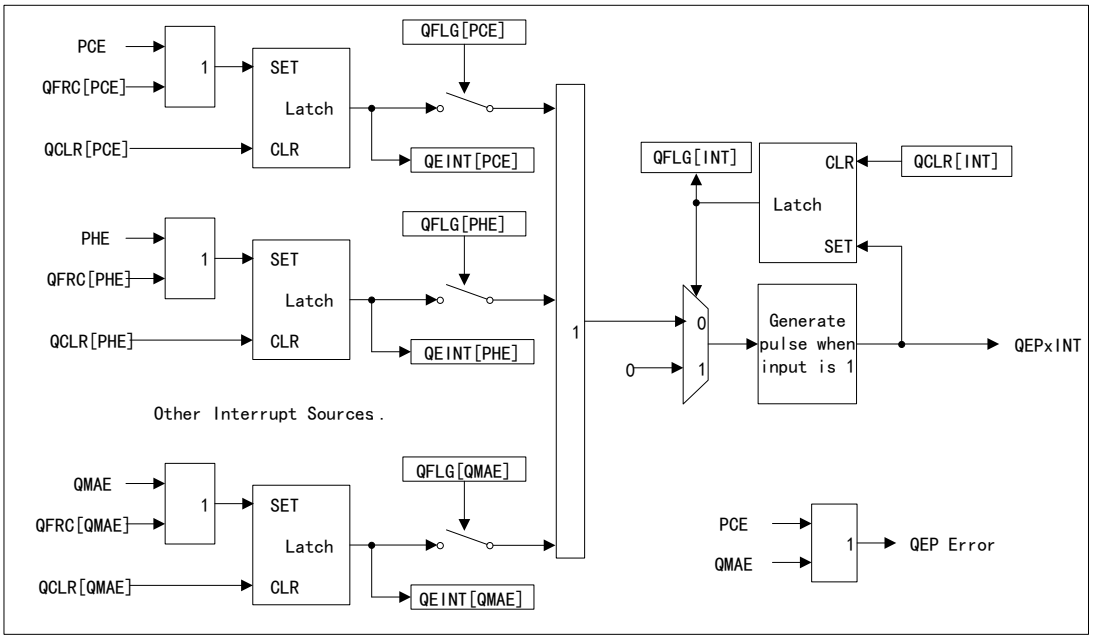
control system indicated by the quadrature clock. The clock source for the 16-bit watchdog timer is SYSCLKOUT/64, and the watchdog timer can be reset by the quadrature clock event (QWDTMR=QWDPRD). If no quadrature clock event is detected before the period match, the watchdog timer will expire and set the watchdog timeout interrupt flag (WTO). The timeout value can be programmed through the QWDPRD register.

Figure 153 16-bit Watchdog Timer Structure Block Diagram



33.5.5 QEP interrupt

Figure 154 QEP Interrupt Structure Block Diagram



NVIC interrupt pulse generation conditions

The NVIC will generate an interrupt pulse under the following conditions:

- The QEP event interrupt is enabled (through the QEINT register)
- The QEP event interrupt flag is set (through the QFLG register)
- The global interrupt flag (INT) for any previously generated interrupt event has been cleared. The interrupt service routine must clear the service event and INT bit through the QCLR register before generating any other interrupt pulses. As long as any flag in the QFLG register remains uncleared, the NVIC will not generate an interrupt

pulse for a new interrupt event. Interrupt events can be forcibly generated using the QFRC register.

33.6 Function description of quadrature mode adapter (QMA)

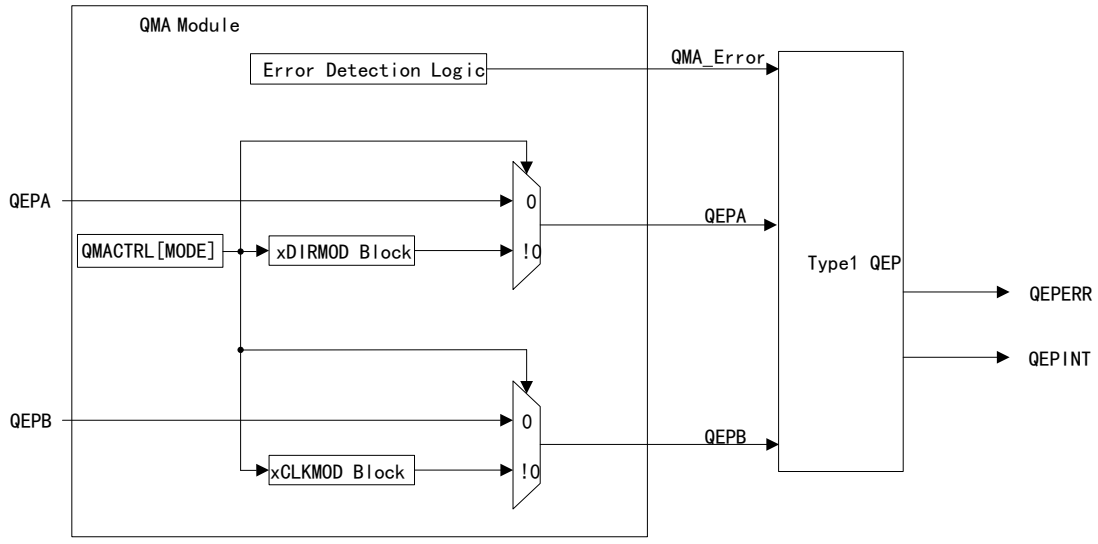
Functions of additional modules described for extending the QEP module to support

When the QMA is reset, it bypasses the QMA logic (by default), and the QEPA and QEPB inputs from the pins are directly fed into the QEP module. The QMA module can be enabled by configuring the MODE bit to handle the QEPA and QEPB inputs and transmit the modified QEPA and QEPB signals to the QEP module. For proper operation in the intended mode, the QMA module requires the QEP module to be configured in direction count mode and generate clock and direction signals on the QEPA and QEPB inputs as needed.

33.6.1 QMA structure block diagram

The transitions of the external QEPA and QEPB signals can be observed through the xCLKMOD and xDIRMOD blocks inside the QMA module, and the xCLKMOD block generates the clock signal on the QEPA input to the QEP module. The xDIRMOD block generates the direction signal on the QEPB input to the QEP module.

Figure 155 QMA Structure Block Diagram



33.6.2 Operation mode

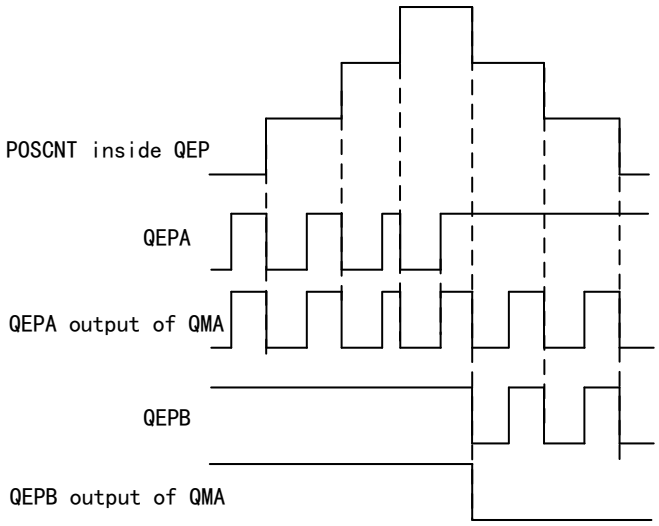
The operation mode of the QMA module can be configured through the QMACTRL register:

- MODE=001, select operation mode 1
- MODE=010, select operation mode 2

Operation mode 1

Operation mode 1 is used when the default state of the QEPA and QEPB input signals is high. In this case, the output of the QMA is as shown in the figure below, where the QEPA output is the logical "AND" of the QEPA and QEPB inputs from the pins, and the QEPB output is the direction signal generated by the QMA based on the QEPA and QEPB inputs.

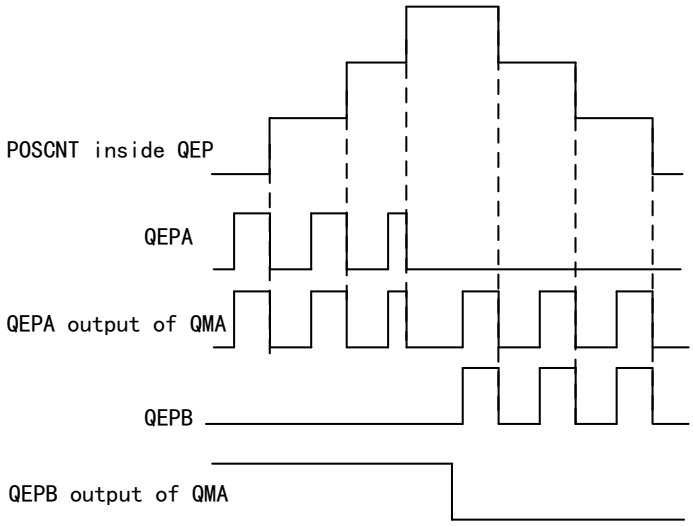
Figure 156 QMA Operation Mode 1 Output



Operation mode 2

Operation mode 2 is used when the default state of the QEPA and QEPB input signals is low. In this case, the output of the QMA is as shown in the figure below, where the QEPA output is the logical "OR" of the QEPA and QEPB inputs from the pins, and the QEPB output is the direction signal generated by the QMA based on the QEPA and QEPB inputs.

Figure 157 QMA Operation Mode 2 Output



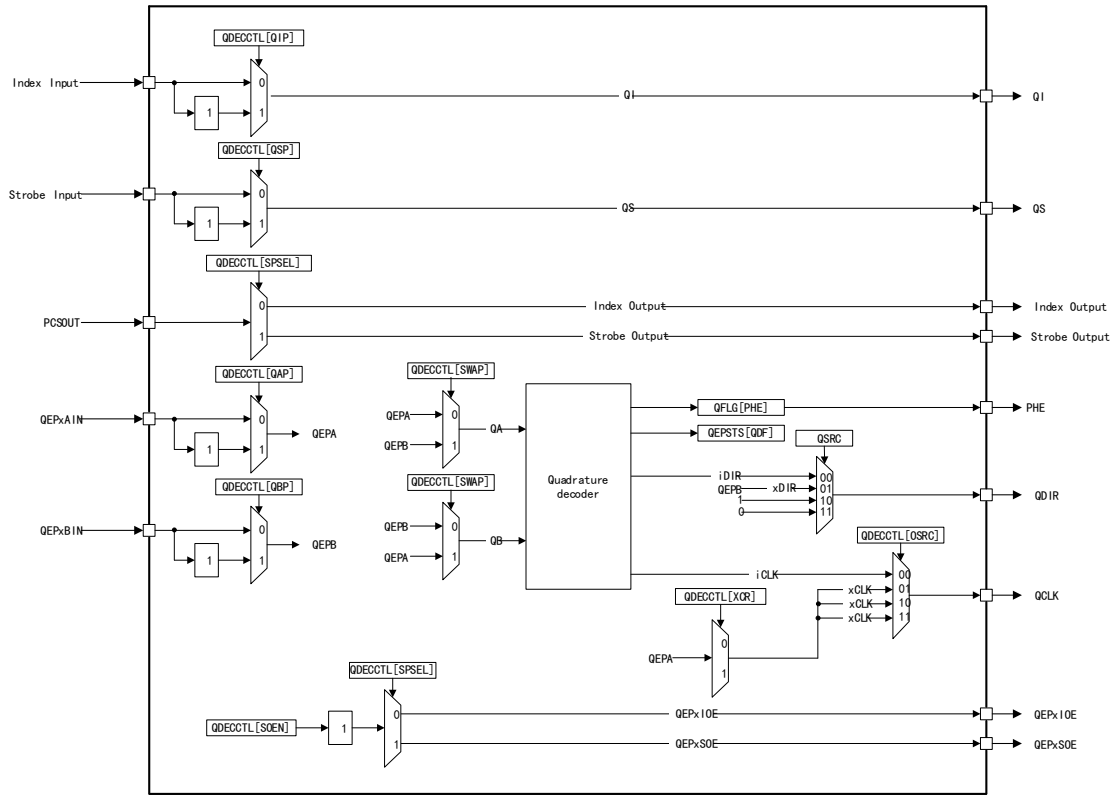
33.6.3 Error detection logic

The QMA module has error detection logic that can detect illegal conversions on the QEPA and QEPB signals and generate an error signal. This error signal can be used to generate error outputs and QEP interrupts, with the error and interrupt integration of the QMA module within the QEP module. For more details, refer to the QEP Interrupt section.

33.7 Function description of quadrature coder unit (QDU)

33.7.1 QDU structure block diagram

Figure 158 QDU Structure Block Diagram



33.7.2 Input mode

Depending on interface input requirements, the direction input and clock for the position counter can be selected through the QSRC bit:

- Count-up mode
- Count-down mode
- Quadrature counting mode
- Direction count mode

Count-up mode

The position counter is used to measure the frequency of the QEPA input, with

the counter direction signal hardwired to count upwards. Clearing the XCR bit generates a clock to the position counter on both ends of the QEPA input, thereby doubling the measurement resolution.

In the count-up mode, the application shall ensure that no signal edges are generated on the QEPB input or no QEPB is configured as the GPIO multiplex option.

Count-down mode

The position counter is used to measure the frequency of the QEPA input, with the counter direction signal used for downward counting. Clearing the XCR bit generates a clock to the position counter on both ends of the QEPA input, thereby doubling the measurement resolution.

In the count-down mode, the application shall ensure that no signal edges are generated on the QEPB input or no QEPB is configured as the GPIO multiplex option.

Quadrature counting mode

In quadrature counting mode, the quadrature decoder generates the direction and clock signals to the position counter.

(1) Direction decoding

The direction decoding logic of the QEP circuit determines whether the leading sequence is from QEPA or QEPB, and updates the corresponding direction information in the QDF bit. The quadrature decoding logic state mechanism diagram and truth value table are as follows.

Figure 159 Quadrature Decoding Logic State Mechanism Diagram

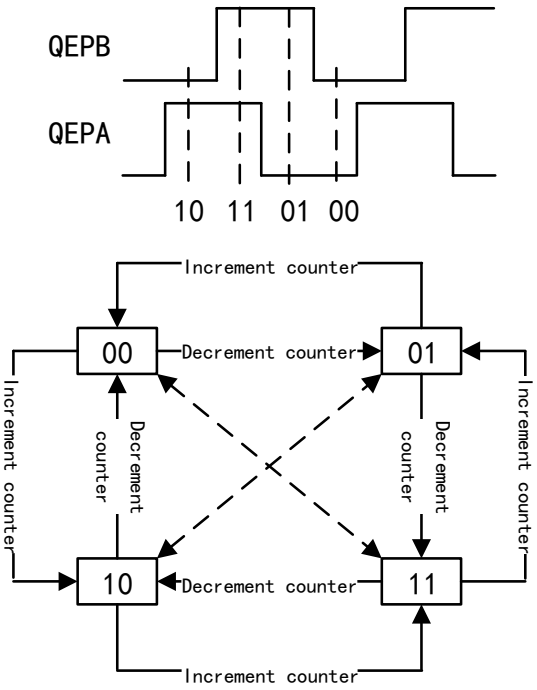


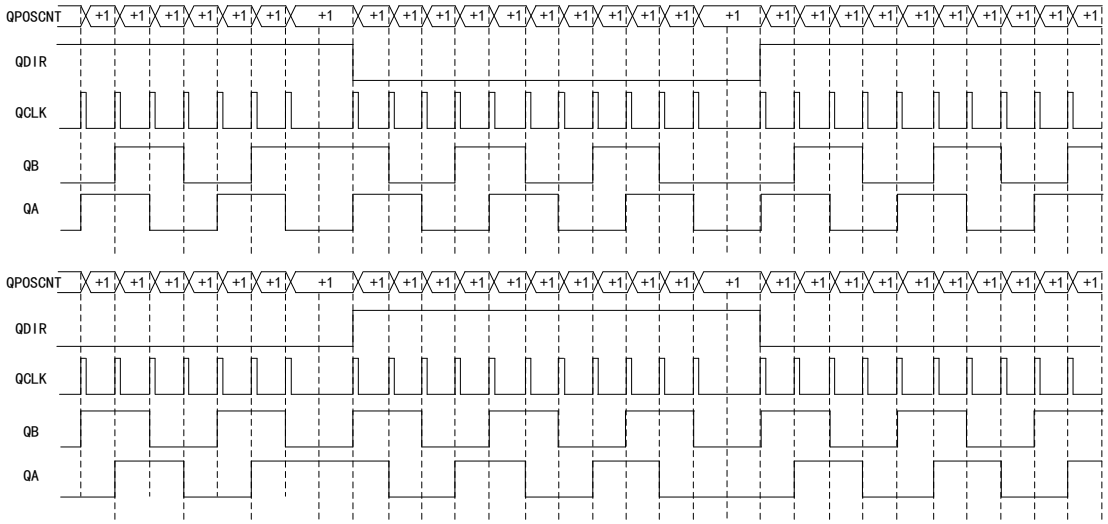
Table 161 Quadrature Decoding Logic Truth Value

Previous edge	Current edge	Counting direction	Count value (QPOSCNT)
QA is the rising edge	QA is the falling edge	Switch	Increment or decrement
	QB is the rising edge	Count up	Increment
	QB is the falling edge	Count down	Decrement
QA is the falling edge	QA is the rising edge	Switch	Increment or decrement
	QB is the rising edge	Count down	Decrement
	QB is the falling edge	Count up	Increment
QB is the rising edge	QA is the rising edge	Count down	Decrement
	QA is the falling edge	Count up	Increment
	QB is the falling edge	Switch	Increment or decrement
QB is the falling edge	QA is the rising edge	Count up	Increment
	QA is the falling edge	Count down	Decrement
	QB is the rising edge	Switch	Increment or decrement

When both edges of the QEPA and QEPB signals are detected, a count pulse for the position counter is generated. The direction decoding and quadrature clock generation for the QEP input signals are shown in the figure below. The clock frequency generated by the QEP logic is four times the frequency of each

input sequence.

Figure 160 Direction Decoding and Quadrature Clock for QEP Input Signals



(2) Count multiplication

The position counter generates QCLK on the rising/falling edges of the QEPA and QEPB input clocks, providing a resolution four times that of the input clock, as shown in the direction decoding and quadrature clock diagram for the QEP input signals.

(3) Reverse counting direction

In normal quadrature counting mode, the QEPA input is applied to the QA input of the quadrature decoder, while the QEPB input is applied to the QB input of the quadrature decoder. By setting the QDECCTL [SWAP] bit, quadrature clock inputs can be swapped to negate the counting direction.

(4) Quadrature phase error flag

The quadrature inputs QEPA and QEPB are phase-shifted by 90 degrees (under normal operating conditions). The quadrature phase error flag is set in the QFLG register, and the QPOSCNT value may be incorrect, and offset by a multiple of 1 or 3. This error can selectively generate an interrupt if edge conversions are detected simultaneously on both the QEPA and QEPB signals. The state conversions markered with dashed lines in the quadrature decoding logic state mechanism diagram represent invalid conversions that produce quadrature phase errors.

Direction count mode

When some position encoders provide direction and clock outputs instead of quadrature outputs, direction count mode can be used, where the QEPA input provides the clock for the position counter and the QEPB input provides direction information. If the direction input is high, the position counter

increments on each rising edge of the QEPA input; if the direction input is low, the position counter decrements.

33.8 Function description of position counter and control unit (PCCU)

The PCCU provides two configuration registers, QEPCTL and QPOSCTL, for setting the position counter initialization, operating mode, latch mode, and position comparison logic used for synchronization signal generation.

33.8.1 Initialization

The position counter can be initialized through software initialization events, strobe events, and index events.

Table 162 Initialize Position Counter

Event	Field	Description
Software initialization	SWI	Set the SWI bit to enable the software initialization of the position counter, but this bit can not be automatically cleared. When writing 1 to this bit again (while the bit remains set), the position counter will be re-initialized.
Strobe event initialization	SEI	When SEI=10, the position counter is initialized using the value from the QPOSINIT register on the rising edge of the strobe input.
		When SEI=11, the position counter is initialized using the value from the QPOSINIT register on the rising edge of the forward strobe input and the falling edge of the reverse strobe input.
Index event initialization	IEI	The QEPI signal can be used to trigger the initialization of the position counter on the rising/falling edge of the index input.
		When IEI=10, the position counter is initialized using the value from the QPOSINIT register on the rising edge of the QEPI signal.
		When IEI=11, the position counter is initialized using the value from the QPOSINIT register on the falling edge of the QEPI signal.

33.8.2 Configure the position counter operating mode

The position counter can be reset to one of the following modes according to the PCRM bits:

- PCRM=00, index event
- PCRM=01, maximum position
- PCRM=10, first index event

- PCRM=11, unit timeout event

In these four modes, the position counter is reset to 0 upon overflow, but the position counter is reset to QPOSMAX upon underflow. If the position counter continues to count upwards after reaching the maximum value, an overflow will occur and the PCO bit will be set. If the position counter is 0 but continues to count downwards, an underflow will occur and the PCU bit will be set.

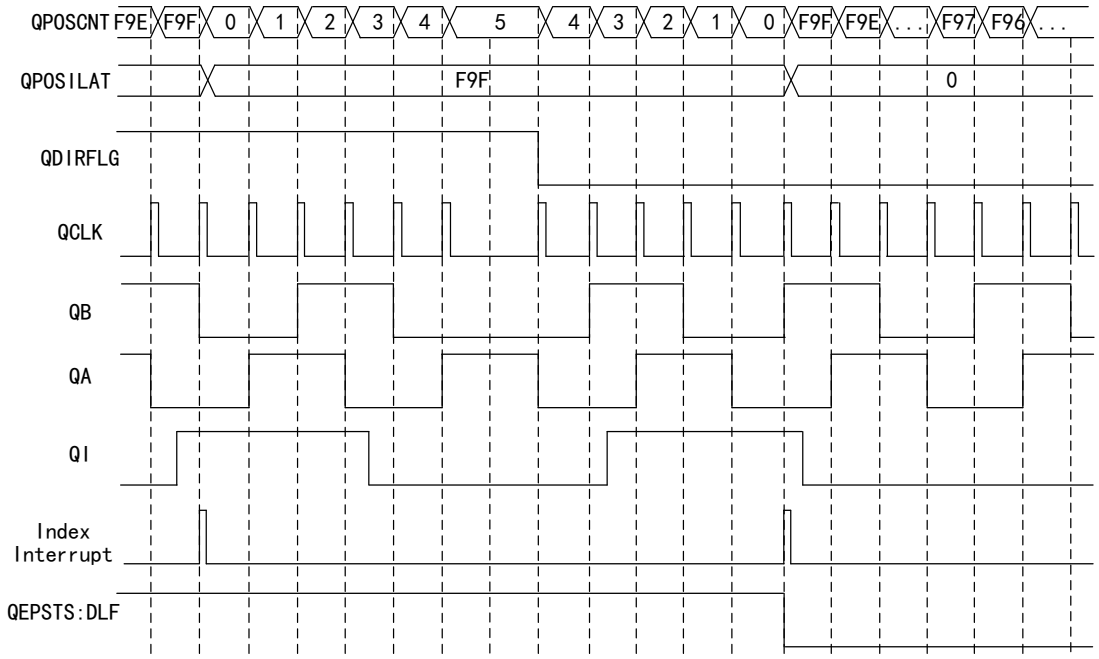
Reset the position counter on an index event

When an index event occurs during the following periods, the position counter will be reset on the next QEP clock:

- During the clockwise direction, the position counter is reset to 0
- During the counterclockwise direction, the position counter is reset to QPOSMAX

The quadrature edge after the first index edge is defined as the first index marker. The first index marker (FIMF) and the direction of the first index marker (FIDF) in the QEPSTS register are recorded by the QEP peripheral. Additionally, to allow the same relative quadrature conversions to reset the position counter on an index event, the QEP peripheral also marks the quadrature edge on the first index marker. When the first reset occurs on the falling edge of QEPB in the clockwise direction, subsequent resets must be aligned with the falling edge of QEPB in the clockwise direction, and in the counterclockwise direction, resets must be aligned with the rising edge of QEPB, as shown in the figure below.

Figure 161 Reset the Position Counter on an Index Event



The direction information is recorded in the QDLF bit for each index event marker, and the QPOSCNT value is latched into the QPOSILAT register. If the

value latched into the QPOSILAT register is not equal to 0 or QPOSMAX, the PCE (position counter error interrupt flag) and PCEF (position counter error flag) will be set. The PCE bit can only be cleared by software (if this bit is set wrongly), and the PCEF bit will be updated on each index event marker.

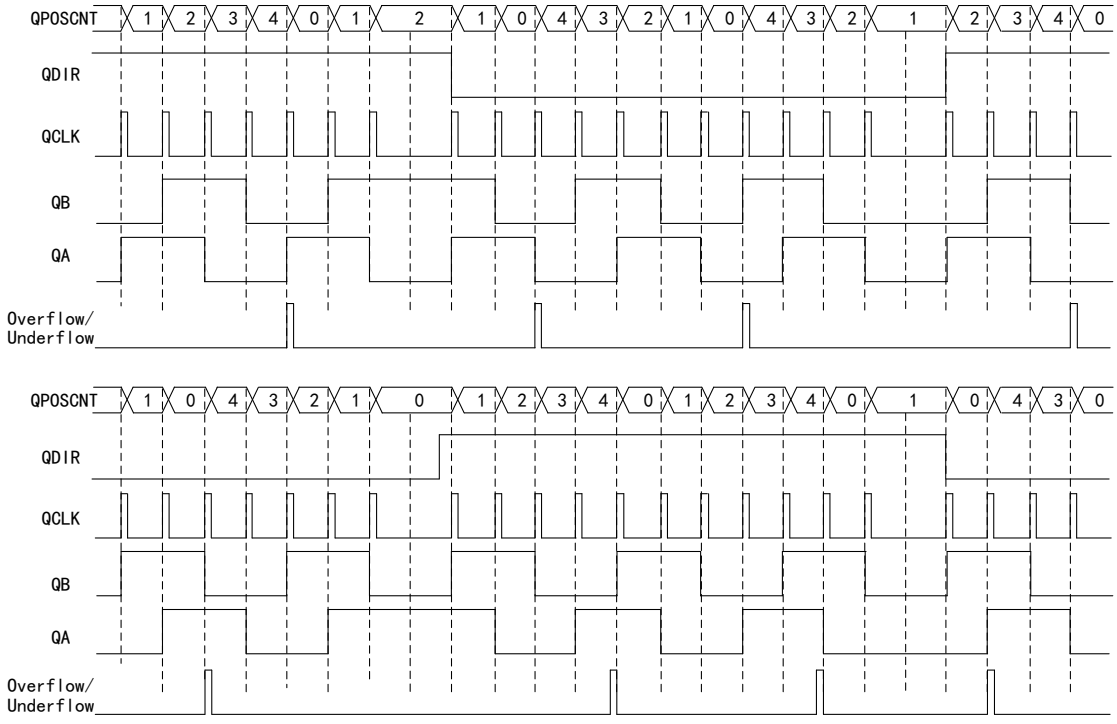
Latch the QPOSCNT into the QPOSILAT register on each index marker. If PCE bit/PCEF bit and PCRM=00 are generated only in this mode, the index event latch must configure IEL to 00 or 11.

Reset the position counter at the maximum position

The first index marker (FIMF) and FIDF do not apply to this mode.

As shown in the figure, the position counter is reset at the maximum position. If QPOSCNT=0, the position counter will be reset to QPOSMAX on the next counterclockwise QEP clock and set the PCU. If QPOSCNT=QPOSMAX, the position counter will be reset on the next clockwise QEP clock and set the PCO.

Figure 162 Reset the Position Counter when QPOSMAX=4



Reset the position counter on the first index event

When the first index event occurs during the following period, the position counter will be reset on the next QEP clock:

- During the clockwise direction, the position counter is reset to 0
- During the counterclockwise direction, the position counter is reset to QPOSMAX

The first index marker (FIMF) and FIDF do not apply to this mode.

Note: The position counter is reset only on the first index event, and subsequently, will be reset based on the maximum position. For details, refer to the Reset the Position Counter at the Maximum Position section.

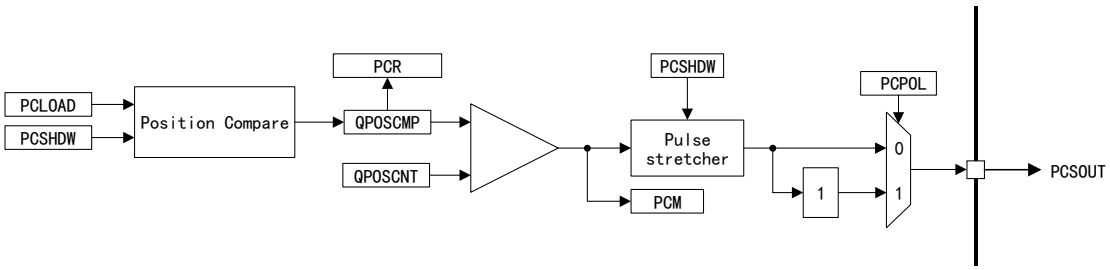
Reset the position counter on a unit timeout event

In this mode, the position counter mode selected by the QSRC bit on the unit time event determines whether QPOSCNT=0 or QPOSCNT=QPOSMAX. At this point, the mode can be used for frequency measurement.

33.8.3 Position compare unit

The position compare unit is used to generate a synchronized output and/or interrupt upon position compare match. The block diagram of the position compare unit is shown below. The QPOSCMP register is shadowed, and the shadow mode can be enabled/disabled via the QPOSCTL[PCSHDW] bit. The CPU writes directly to the active QPOSCMP register (when shadow mode is not enabled).

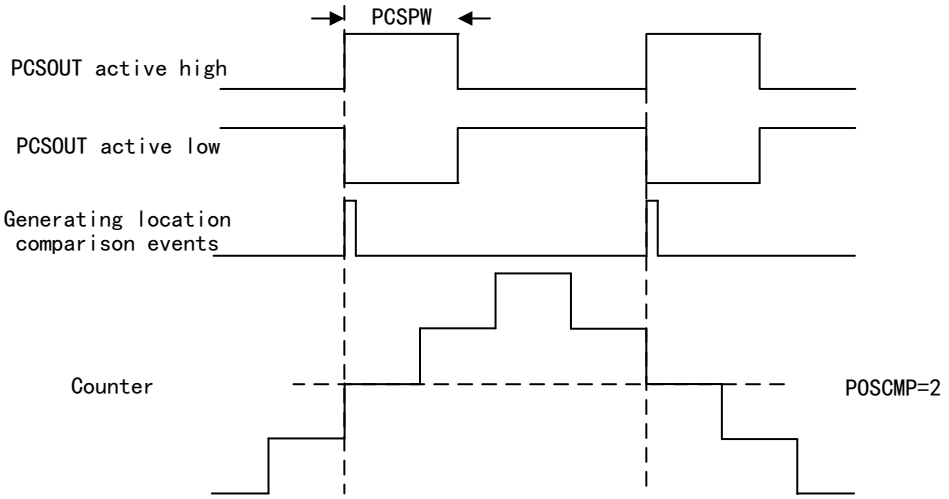
Figure 163 Structure Block Diagram of Position Compare Unit



To load the shadow register values into the active register for the compare match event and zero load event of position counter, and generate the PCR interrupt after loading, the position compare unit can be configured via the PCLOAD bit (in shadow mode).

When QPOSCNT matches QPOSCMP, PCM is set, and to trigger an external device, the compare synchronized output of the programmable pulse width position is generated upon compare match. As shown below, when QPOSCMP=2, the position compare unit generates a position compare event when the QEP position counter counts up from 1 to 2 and when the QEP position counter counts down from 3 to 2.

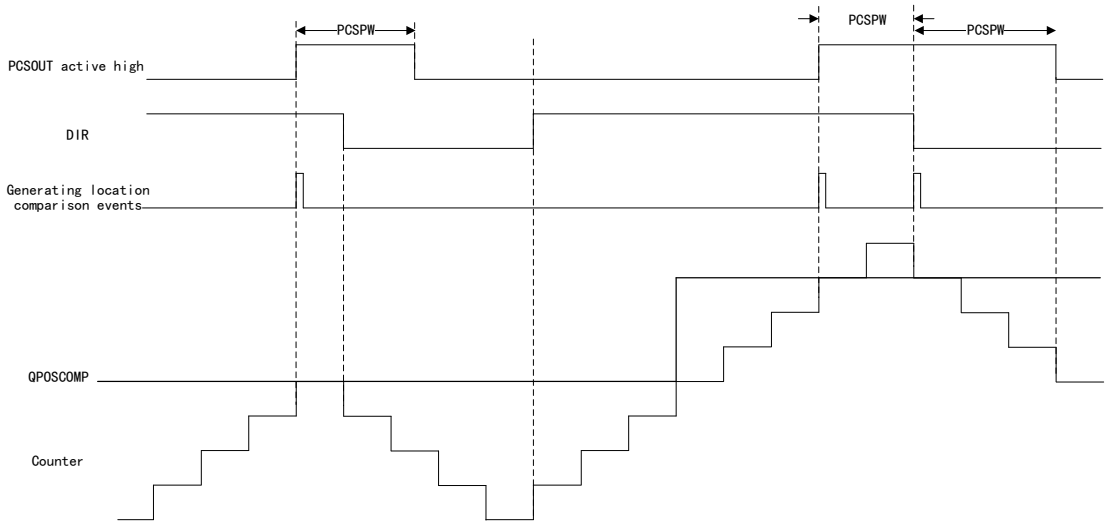
Figure 164 QEP Position Compare Event Generation Points



Pulse extender logic

When a position compare match occurs, the pulse extender logic will generate a programmable position compare synchronized pulse output. As shown below, if the previous position compare pulse is still valid, the pulse extender will generate a pulse of specified duration when a new position compare match occurs (generated in the new position compare event).

Figure 165 Position Compare Synchronization Output Pulse Extender



Synchronous output

When a compare match occurs between the QPOSCNT register and the QPOSCMP register, the position compare unit is used to generate the position compare synchronized signal. This synchronized signal can be used for the index/strobe pin output of the QEP peripheral. The position compare synchronization output can be enabled by setting SOEN, and the

synchronization output pin can be selected via the SPSEL bit.

33.8.4 Latch mode

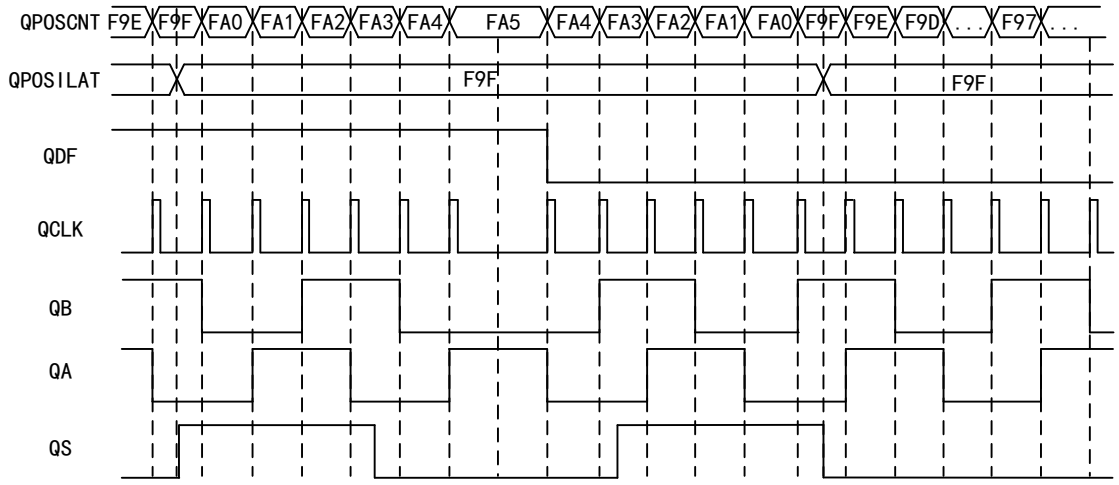
When a defined event occurs on the index or strobe pin, the QEP index and strobe inputs can be configured to latch the position counter into the QPOSILAT and QPOSSLAT registers, respectively.

Strobe event latch

As shown in the strobe event latch diagram, if the SEL bit is set, QPOSCNT is latched into the QPOSSLAT register on the rising edge of the strobe input when counting in the clockwise direction. When counting in the counterclockwise direction, QPOSCNT is latched on the falling edge of the strobe input. When the SEL bit is cleared, QPOSCNT is latched into the QPOSSLAT register on the rising edge of the strobe input.

If QPOSCNT is latched into the QPOSSLAT register, the SEL flag bit will be set.

Figure 166 Strobe Event Latch

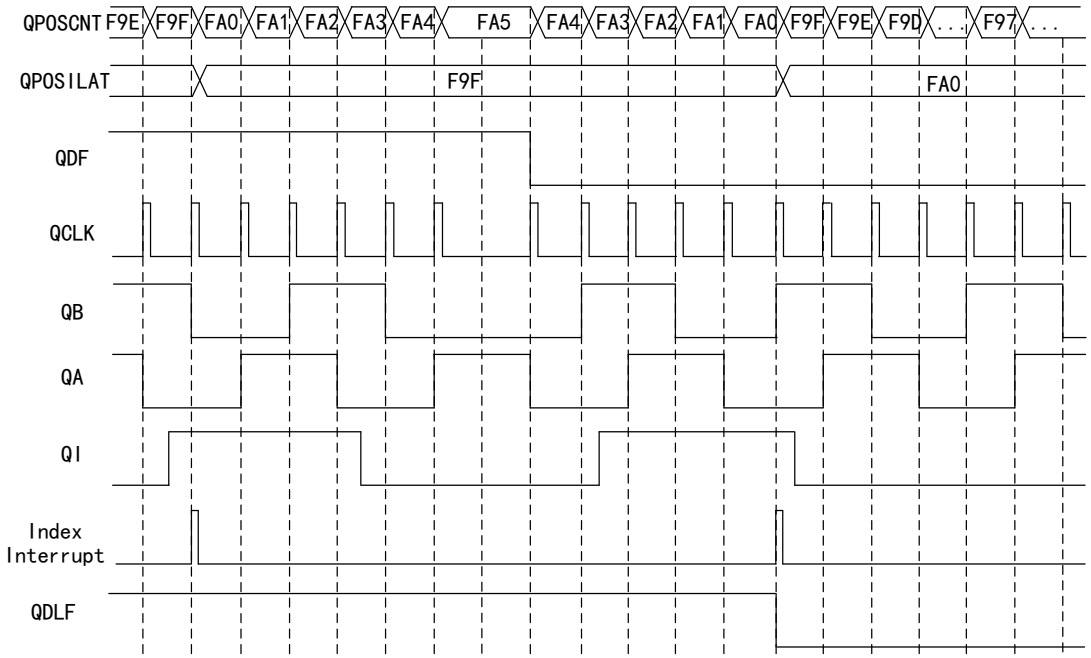


Index event latch

If the position counter is not to be reset on each index event, the position counter needs to be reset in full 32-bit mode, i.e., set PCRM=01 and PCRM=10. The position counter can be latched on the rising edge (IEL=01), falling edge (IEL=10), or by a software index marker (IEL=11) based on the IEL bit. Direction information can be recorded on each index event based on the QDLF bit, and the index event marker for PCRM is shown in the diagram.

Error detection mechanisms can check whether the position counter has accumulated the correct count (between index events) based on index event latch. If QPOSCNT is latched into the QPOSILAT register, the IEL bit is set. When PCRM=00, the IEL bit will be ignored.

Figure 167 1000-line Encoder Software Index Marker (QEPCTL[IEL] = 1)



Rising edge latch (IEL=01)

On each rising edge of the index input, QPOSCNT is latched into the QPOSILAT register.

Falling edge latch (IEL=10)

On each falling edge of the index input, QPOSCNT is latched into the QPOSILAT register.

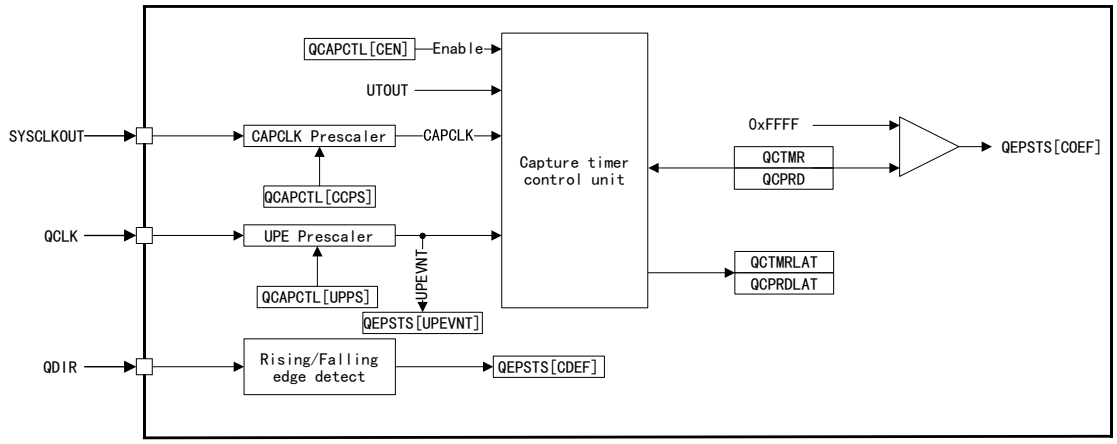
Software index marker latch (IEL=11)

The quadrature edge after the first index edge is defined as the first index marker. The first index marker (FIMF) and the direction of the first index marker (FIDF) in the QEPSTS register are recorded by the QEP peripheral. Additionally, to set IEL=11, which latches the position counter on the same relative quadrature transition, the QEP peripheral also marks the quadrature edge on the first index marker.

33.9 Function description of edge capture unit

The edge capture unit is used to measure the time elapsed between unit position events, with its structure block diagram shown below.

Figure 168 Structure Block Diagram of Edge Capture Unit



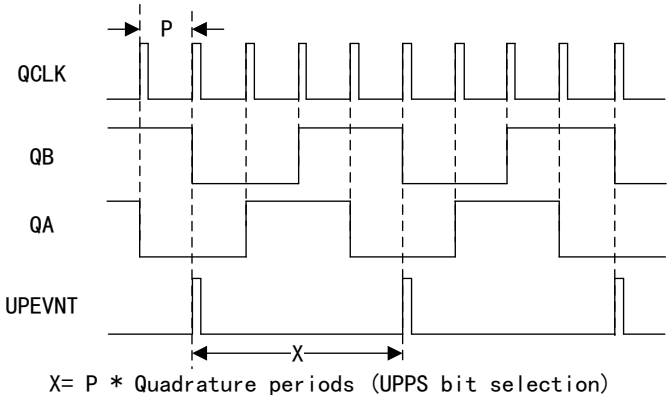
The function of the edge capture unit is typically used for low-speed measurements through the following formula:

$$v(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T}$$

Table 163 Supplementary Description

Parameter	Description
v (k)	Speed at time k
t (k)	Time at time k
t (k-1)	Time at time k-1
X	The unit position is determined by integer multiples of the quadrature edges, as shown in the timing diagram of unit position event
ΔT	Elapsed time of unit position event

Figure 169 Timing Diagram of Unit Position Event



The capture timer QCTMR runs from SYSCLKOUT, and the QEP capture timer clock prescaler is selected by the CCPS field. In each unit position event, QCTMR is latched into the QCTMRLAT register, and QCTMR is reset. The new value of QCTMR latched into the QCTMRLAT register is indicated by setting the

unit position event flag bit UPEVNT. Before reading the period register for low-speed measurement, the software can check the UPEVNT bit and clear this bit by writing 1.

If no direction change occurs between unit position events or the count between unit position events is less than 65,535, the time measurement between unit position events is accurate.

When a capture timer overflow occurs between unit position events, the QCPRDLAT register is set to FFFF, and the COEF bit is set. When a direction change occurs between unit position events, the QCPRDLAT register is set to FFFF, and the CDEF bit is set.

The QCTMR register and QCPRD register can be configured to latch unit timeout events and CPU reading of the QPOSCNT register.

When the QCLM bit is cleared and the CPU reads the QPOSCNT register, QCTMR is latched into the QCTMRLAT register, and QCPRD is latched into the QCPRDLAT register. When the QCLM bit is cleared and a unit timeout occurs, QPOSCNT, QCTMR, and QCPRD are latched into the QPOSLAT register, QCTMRLAT register, and QCPRDLAT register, respectively.

Note: To avoid errors in the prescaler, dynamic modification of the UPPS field is not allowed, such as switching UPE from QCLK/4 to QCLK/8. Dynamic modification of the CCPS field is only allowed after disabling the edge capture unit, such as switching CAPCLK from SYSCLK/4 to SYSCLK/8.

Figure 170 Timing Diagram of Edge Capture Unit

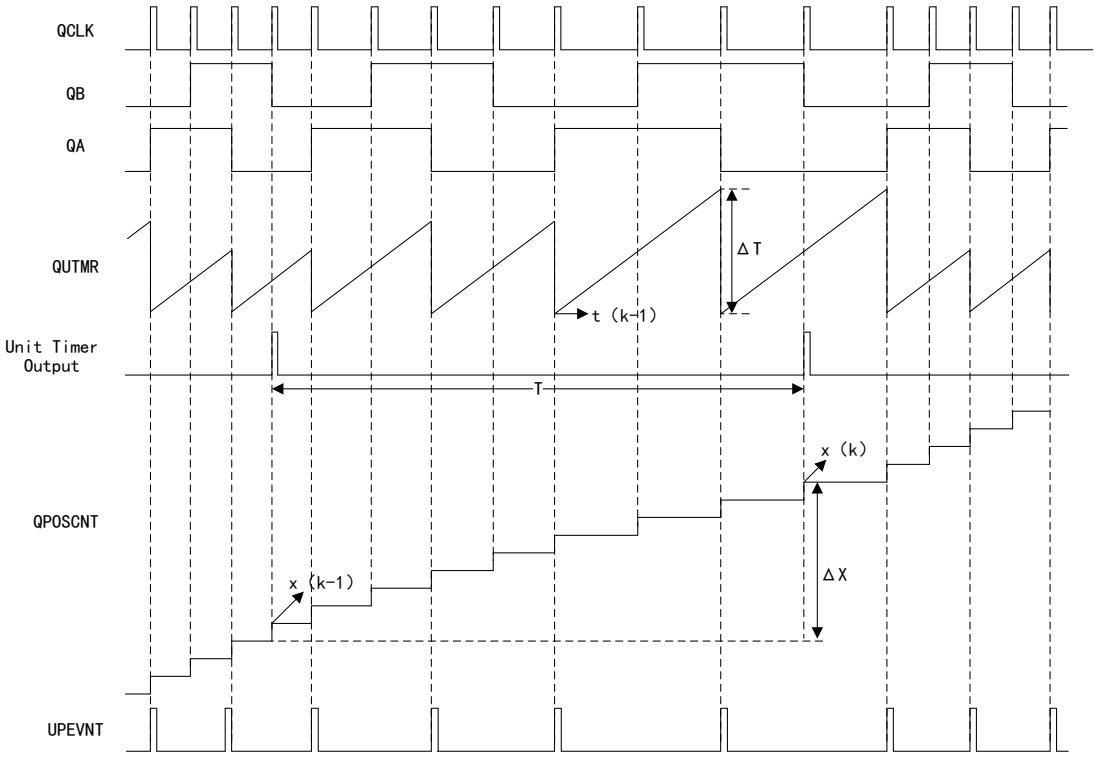


Table 164 Supplementary Description on Timing Diagram of Edge Capture Unit

Parameter	Description	Configuration information
v (k)	Speed at time k	-
x (k)	Position at time k	-
x (k-1)	Position at time k-1	-
X	Fixed unit position	Fixed unit position is defined by the UPPS field and sensor resolution
ΔX	Incremental position movement per unit time	Incremental position = QPOSLAT(k) - QPOSLAT(k-1)
t (k)	Time at time k	-
t (k-1)	Time at time k-1	-
T	Reciprocal of rate calculated with the fixed unit time or speed	QUPRD register
ΔT	Incremental time for unit position movement	QCPRDLAT register

Based on the timing diagram of the edge capture unit, its speed calculation formula is:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T}$$

According to the supplementary description on the edge capture unit timing diagram, the unit period and unit time are configured through the UPPS field and QUPRD register, while the increment position output and increment time output are provided by the QPOSLAT register and QCPRDLAT register.

33.10 Register bank address

Table 165 Register Bank Address

Device register	Register bank	Start address	End address
Qep1Regs	QEP_REGS	0x4001_3800	0x4001_3BFF

33.11 Register address mapping

Table 166 QEP_REGS Offset Address

Register name	**Register description**	**Offset address**	**WRPRT**
QPOSCNT	Position counter register	0x00	-
QPOSINIT	Initialize position counter register	0x04	-
QPOSMAX	Maximum position counter register	0x08	-
QPOSCMP	Position compare register	0x0C	-

Register name	Register description	Offset address	WRPRT
QPOSILAT	Index position latch register	0x10	-
QPOSSLAT	Strobe position latch register	0x14	-
QPOSLAT	Position latch register	0x18	-
QUTMR	QEP unit timer register	0x1C	-
QUPRD	QEP unit period register	0x20	-
QWDTMR	QEP watchdog timer register	0x24	-
QWDPRD	QEP watchdog period register	0x26	-
QDECCTL	Quadrature decoder control register	0x28	-
QEPCTL	QEP control register	0x2A	-
QCAPCTL	Quadrature capture control register	0x2C	-
QPOSCTL	Position compare control register	0x2E	-
QEINT	Enable QEP interrupt register	0x30	-
QFLG	QEP interrupt flag register	0x32	-
QCLR	Clear QEP interrupt register	0x34	-
QFRC	Force QEP interrupt register	0x36	-
QEPSTS	QEP flag register	0x38	-
QCTMR	QEP capture timer register	0x3A	-
QCPRD	QEP capture period register	0x3C	-
QCTMRLAT	QEP capture timer latch register	0x3E	-
QCPRDLAT	QEP capture period latch register	0x40	-
REV	QEP revision number register	0x60	-
QEPSTROBESEL	Strobe source select register	0x64	-
QMACTRL	QMA control register	0x68	-

33.12 Register functional description

33.12.1 Position counter register (QPOSCNT)

Offset address: 0x00

Reset type: SYSRSn

Based on the direction input, the 32-bit position counter register counts up or down for each QEP pulse. When the counter is configured for count up or count down mode, this register is read only. If the counter is used as a position integrator, the position of the selected reference point is proportional to the count value.

Field	Name	R/W	Description	Reset value
31:0	QPOSCNT	R/W	Position Counter Note: It is recommended to write to QPOSCNT during initialization (i.e., when QPEN=0). Writing to QPOSCNT when QPEN=1 will cause an error.	0h

33.12.2 Initialize position counter register (QPOSINIT)

Offset address: 0x04

Reset type: SYSRSn

Writing to this register shall always be 32-bit writes.

Field	Name	R/W	Description	Reset value
31:0	QPOSINIT	R/W	Position Counter Initialize The position counter can be initialized by software. This register contains the position value and can be used to initialize the position counter based on external strobe events or index events.	0h

33.12.3 Maximum position counter register (QPOSMAX)

Offset address: 0x08

Reset type: SYSRSn

Writing to this register shall always be 32-bit writes.

Field	Name	R/W	Description	Reset value
31:0	QPOSMAX	R/W	Maximum Position Counter This register contains the maximum position counter value.	0h

33.12.4 Position compare register (QPOSCMP)

Offset address: 0x0C

Reset type: SYSRSn

Writing to this register shall always be 32-bit writes.

Field	Name	R/W	Description	Reset value
31:0	QPOSCMP	R/W	Position Compare This register contains the position compare value. The position compare value is compared with QPOSCNT, and an interrupt and/or synchronized output can be generated upon compare match.	0h

33.12.5 Index position latch register (QPOSILAT)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	QPOSILAT	R	Index Position Latch The position counter is latched into this register during the index events defined by the IEL field.	0h

33.12.6 Strobe position latch register (QPOSSLAT)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	QPOSSLAT	R	Strobe Position Latch The position counter is latched into this register during the strobe events defined by the SEL bit.	0h

33.12.7 Position latch register (QPOSLAT)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	QPOSLAT	R	Position Latch The position counter is latched into this register during a unit timeout event.	0h

33.12.8 QEP unit timer register (QUTMR)

Offset address: 0x1C

Reset type: SYSRSn

Writing to this register shall always be 32-bit writes.

Field	Name	R/W	Description	Reset value
31:0	QUTMR	R/W	QEP Unit Timer This register can serve as the time base for unit time events. A unit time event is generated when the unit time period value matches the value of this timer.	0h

33.12.9 QEP unit period register (QUPRD)

Offset address: 0x20

Reset type: SYSRSn

Writing to this register shall always be 32-bit writes.

Field	Name	R/W	Description	Reset value
31:0	QUPRD	R/W	QEP Unit Period This register contains the unit counter period value that generates periodic unit time events. These events lock the QEP position based on the periodic interval and can optionally generate an interrupt.	0h

33.12.10 QEP watchdog timer register (QWDTMR)

Offset address: 0x24

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	QWDTMR	R/W	QEP Watchdog Timer This register can serve as the time base for watchdog checks for motor stalls. Reset this register when indicating a moving quadrature clock edge conversion. A watchdog timeout interrupt is generated when the watchdog period value matches the value of this timer.	0h

33.12.11 QEP watchdog period register (QWDPRD)

Offset address: 0x26

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	QWDPRD	R/W	QEP Watchdog Period This register contains the timeout value for the QEP watchdog timer. A watchdog timeout interrupt is generated when the watchdog period value matches the value of this timer.	0h

33.12.12 Quadrature decoder control register (QDECCTL)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	Reserved			0h
5	QSP	R/W	Negates QEPS Input 0: No effect 1: Negate QEPS input	0h
6	QIP	R/W	Negates QEPI Input 0: No effect 1: Negate QEPI input	0h
7	QBP	R/W	Negates QEPB input 0: No effect 1: Negate QEPB input	0h
8	QAP	R/W	Negates QEPA input 0: No effect 1: Negate QEPA input	0h
9	IGATE	R/W	Index Pulse Gating Select 0: Disable 1: Strobe index pin gating	0h
10	SWAP	R/W	Swapped Quadrature-clock Inputs Enable 0: Disable 1: Enable	0h
11	XCR	R/W	External Clock Rate Select 0: 2x resolution (count both rising and falling edges) 1: 1x resolution (count only rising edge)	0h
12	SPSEL	R/W	Sync Output Pin Select 0: Index pin 1: Strobe pin	0h
13	SOEN	R/W	Position-compare Ssync Output Enable 0: Disable 1: Enable	0h
15:14	QSRC	R/W	Position-counter Mode Select 00: Quadrature counting mode, QCLK = iCLK, QDIR = iDIR	0h

Field	Name	R/W	Description	Reset value
			01: Direction counting mode, QCLK = xCLK, QDIR = xDIR 10: Count-up mode, QCLK = xCLK, QDIR = 1 11: Count-down mode, QCLK = xCLK, QDIR = 0	

33.12.13 QEP control register (QEPCTL)

Offset address: 0x2A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	WDE	R/W	QEP Watchdog Enable 0: Disable 1: Enable	0h
1	UTE	R/W	QEP Unit Timer Enable 0: Disable 1: Enable	0h
2	QCLM	R/W	QEP Capture Latch Mode 0: CPU reads the latched position counter. If the CPU reads the QPOSCNT register, the capture timer value and capture period value are latched into the QCTMRLAT and QCPRDLAT registers. 1: Latch on unit timeout. If a unit timeout occurs, the capture timer, capture period value, and position counter are latched into the QPOSLAT, QCTMRLAT, and QCPRDLAT registers.	0h
3	QPEN	R/W	Quadrature Position Counter Enable 0: Disabled, some flags in the QFLG register will not be cleared or reset, and will display their actual status. Control/configuration registers are unaffected by software reset. Reset the internal operation flags/read-only registers of the QEP peripheral. 1: Enable	0h
5:4	IEL	R/W	Position Counter Index Event Latch Select 00: Reserved 01: Rising edge latch 10: Falling edge latch 11: Software index marker latch (for software index markers). Latch the quadrature direction flag on the index event marker (latch to the QDLF bit) and position counter (latch to the QPOSILAT register).	0h
6	SEL	R/W	Position Counter Strobe Event Latch Select 0: If QPOSSLAT = QPOSCNT, the position counter is latched on the rising edge of QEPS strobe. The position counter can be latched on the falling edge by negating the strobe input (using the QSP bit). 1: In clockwise direction, the position counter is latched on the rising edge of QEPS strobe. In counterclockwise direction, the position counter is latched on the falling edge of QEPS strobe.	0h

Field	Name	R/W	Description	Reset value
7	SWI	R/W	Software Initialize Position Counter Enable This bit will not be cleared automatically. 0: Disable 1: Enable (QPOSCNT = QPOSINIT)	0h
9:8	IEI	R/W	Index Event Init of Position Count Select 00: Disable 01: Disable 10: Initialize position counter on the rising edge of the QEPI signal when QPOSCNT = QPOSINIT 11: Initialize position counter on the falling edge of the QEPI signal when QPOSCNT = QPOSINIT	0h
11:10	SEI	R/W	Strobe Event Init of Position Count Select 00: Disable 01: Disable 10: Initialize position counter at the rising edge of the QEPS signal 11: In the clockwise direction, initialize the position counter at the rising edge of the QEPS strobe. In the counterclockwise direction, initialize the position counter at the falling edge of the QEPS strobe.	0h
13:12	PCRM	R/W	Position Counter Reset 00: Reset the position counter on the index event 01: Reset the position counter on the maximum position counter 10: Reset the position counter at the first index event 11: Reset the position counter on the unit time event	0h
15:14	FREE_SOFT	R/W	Emulation Mode Select QPOSCNT behavior 00: When simulation is suspended, the position counter stops immediately 01: The position counter continues counting until it flips. 10: The position counter is not affected by simulation suspension 11: The position counter is not affected by simulation suspension QWDTMR behavior 00: The watchdog timer stops immediately 01: The watchdog timer counts until the WDT period match flips 10: The watchdog timer is not affected by simulation suspension 11: The watchdog timer is not affected by simulation suspension QUTMR behavior 00: The unit timer stops immediately 01: The unit timer counts until the cycle flips 10: The unit timer is not affected by simulation suspension	0h

Field	Name	R/W	Description	Reset value
			11: The unit timer is not affected by simulation suspension QCTMR behavior 00: The capture timer stops immediately 01: The capture timer counts until the next unit cycle event 10: The capture timer is not affected by simulation suspension 11: The capture timer is not affected by simulation suspension	

33.12.14 Quadrature Capture Control Register (QCAPCTL)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	UPPS	R/W	Unit Position Event Prescale Select 0000: QCLK/1=UPE 0001: QCLK/2=UPE 0010: QCLK/4=UPE 0011: QCLK/8=UPE 0100: QCLK/16=UPE 0101: QCLK/32=UPE 0110: QCLK/64=UPE 0111: QCLK/128=UPE 1000: QCLK/256=UPE 1001: QCLK/512=UPE 1010: QCLK/1024=UPE 1011: QCLK/2048=UPE Others: Reserved	0h
6:4	CCPS	R/W	QEP Capture Timer Clock Prescale Select 000: SYSCLKOUT/1=CAPCLK 001: SYSCLKOUT/2=CAPCLK 010: SYSCLKOUT/4=CAPCLK 011: SYSCLKOUT/8=CAPCLK 100: SYSCLKOUT/16=CAPCLK 101: SYSCLKOUT/32=CAPCLK 110: SYSCLKOUT/64=CAPCLK 111: SYSCLKOUT/128=CAPCLK	0h
14:7	Reserved			0h
15	CEN	R/W	QEP Capture Enable 0: Disable 1: Enable	0h

33.12.15 Position Compare Control Register (QPOSCTL)

Offset address: 0x2E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
11:0	PCSPW	R/W	Position Compare Sync Output Pulse Width Select 0000 0000 0000: 1 * SYSCLKOUT cycle * 4 0000 0000 0001: 2 * SYSCLKOUT cycle * 4 ... 1111 1111 1111: 4096 * SYSCLKOUT cycle * 4	0h
12	PCE	R/W	Position Compare Enable 0: Disable 1: Enable	0h
13	PCPOL	R/W	Sync Output Polarity Select 0: High pulse 1: Low pulse	0h
14	PCLOAD	R/W	Position Compare of Shadow Load 0: Load when QPOSCNT = 0 1: Load when QPOSCNT = QPOSCMP	0h
15	PCSHDW	R/W	Position Compare of Shadow Enable 0: Disable, load immediately 1: Enable	0h

33.12.16 Enable QEP Interrupt Register (QEINT)

Offset address: 0x30

Reset type: SYSRSn

This register is used to enable/disable various interrupt event sources.

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	PCE	R/W	Position Count Error Interrupt Enable 0: Disable 1: Enable	0h
2	PHE	R/W	Quadrature Phase Error Interrupt Enable 0: Disable 1: Enable	0h
3	QDC	R/W	Quadrature Direction Change Interrupt Enable 0: Disable 1: Enable	0h
4	WTO	R/W	Watchdog Time out Interrupt Enable 0: Disable 1: Enable	0h
5	PCU	R/W	Position Counter Underflow Interrupt Enable 0: Disable 1: Enable	0h
6	PCO	R/W	Position Counter Overflow Interrupt Enable 0: Disable 1: Enable	0h
7	PCR	R/W	Position Compare ready Interrupt Enable 0: Disable	0h

Field	Name	R/W	Description	Reset value
			1: Enable	
8	PCM	R/W	Position Compare match Interrupt Enable 0: Disable 1: Enable	0h
9	SEL	R/W	Strobe Event Latch Interrupt Enable 0: Disable 1: Enable	0h
10	IEL	R/W	Index Event Latch Interrupt Enable 0: Disable 1: Enable	0h
11	UTO	R/W	Unit Time out Interrupt Enable 0: Disable 1: Enable	0h
12	QMAE	R/W	QMA Error Interrupt Enable 0: Disable 1: Enable	0h
15:13	Reserved			0h

33.12.17 QEP interrupt flag register (QFLG)

Offset address: 0x32

Reset type: SYSRSn

This register indicates whether any interrupt event is latched, including the global interrupt flag.

Field	Name	R/W	Description	Reset value
0	INT	R	Global Interrupt Flag 0: No interrupt occurs 1: Generate interrupt	0h
1	PCE	R	Position Counter Error Interrupt Flag 0: No interrupt occurs 1: Generate interrupt	0h
2	PHE	R	Quadrature Phase Error Interrupt Flag This bit is set when both QEPA and QEPB are converted simultaneously. 0: No interrupt occurs 1: Generate interrupt	0h
3	QDC	R	Quadrature Direction Change Interrupt Flag 0: No interrupt occurs 1: Generate interrupt	0h
4	WTO	R	Watchdog Time out Interrupt Flag This bit is set when a watchdog timeout occurs. 0: No interrupt occurs 1: Generate interrupt	0h
5	PCU	R	Position Counter Underflow Interrupt Flag This bit is set when the position counter underflows.	0h

Field	Name	R/W	Description	Reset value
			0: No interrupt occurs 1: Generate interrupt	
6	PCO	R	Position Counter Overflow Interrupt Flag This bit is set when the position counter overflows. 0: No interrupt occurs 1: Generate interrupt	0h
7	PCR	R	Position Compare Ready Interrupt Flag This bit is set after transferring the shadow register value to the active QPOSCMP register. 0: No interrupt occurs 1: Generate interrupt	0h
8	PCM	R	Position Compare Match Interrupt Flag This bit is set when a position compare match occurs. 0: No interrupt occurs 1: Generate interrupt	0h
9	SEL	R	Strobe Event Latch Interrupt Flag This bit is set after latching QPOSCNT into QPOSSLAT. 0: No interrupt occurs 1: Generate interrupt	0h
10	IEL	R	Index Event Latch Interrupt Flag This bit is set after latching QPOSCNT into QPOSILAT. 0: No interrupt occurs 1: Generate interrupt	0h
11	UTO	R	Unit Time Out Interrupt Flag This bit is set when a unit timer cycle match occurs. 0: No interrupt occurs 1: Generate interrupt	0h
12	QMAE	R	QMA Error Interrupt Flag 0: No interrupt occurs 1: Generate interrupt	0h
15:13	Reserved			0h

33.12.18 Clear QEP Interrupt Register (QCLR)

Offset address: 0x34

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INT	R/S	Global Interrupt Flag Clear 0: Invalid 1: Clear INT	0h
1	PCE	R/S	Position Counter Error Interrupt Flag Clear 0: Invalid 1: Clear PCE	0h
2	PHE	R/S	Quadrature Phase Error Interrupt Flag Clear	0h

Field	Name	R/W	Description	Reset value
			0: Invalid 1: Clear PHE	
3	QDC	R/S	Quadrature Direction Change Interrupt Flag Clear 0: Invalid 1: Clear QDC	0h
4	WTO	R/S	Watchdog Time Out Interrupt Flag Clear 0: Invalid 1: Clear WTO	0h
5	PCU	R/S	Position Counter Underflow Interrupt Flag Clear 0: Invalid 1: Clear PCU	0h
6	PCO	R/S	Position Counter Overflow Interrupt Flag Clear 0: Invalid 1: Clear PCO	0h
7	PCR	R/S	Position Compare Ready Interrupt Flag Clear 0: Invalid 1: Clear PCR	0h
8	PCM	R/S	Position Compare Match Interrupt Flag Clear 0: Invalid 1: Clear PCM	0h
9	SEL	R/S	Strobe Event Latch Interrupt Flag Clear 0: Invalid 1: Clear SEL	0h
10	IEL	R/S	Index Event Latch Interrupt Flag Clear 0: Invalid 1: Clear IEL	0h
11	UTO	R/S	Unit Time Out Interrupt Flag Clear 0: Invalid 1: Clear UTO	0h
12	QMAE	R/S	QMA Error Interrupt Flag Clear 0: Invalid 1: Clear QMAE	0h
15:13	Reserved			0h

33.12.19 Force QEP interrupt register (QFRC)

Offset address: 0x36

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	PCE	R/W	Force Position Counter Error Interrupt 0: Invalid 1: Force position counter error interrupt	0h
2	PHE	R/W	Force Quadrature Phase Error Interrupt 0: Invalid	0h

Field	Name	R/W	Description	Reset value
			1: Force quadrature phase error interrupt	
3	QDC	R/W	Force Quadrature Direction Change Interrupt 0: Invalid 1: Force quadrature direction change interrupt	0h
4	WTO	R/W	Force Watchdog Time Out Interrupt 0: Invalid 1: Force watchdog timeout interrupt	0h
5	PCU	R/W	Force Position Counter Underflow Interrupt 0: Invalid 1: Force position counter underflow interrupt	0h
6	PCO	R/W	Force Position Counter Overflow Interrupt 0: Invalid 1: Force position counter overflow interrupt	0h
7	PCR	R/W	Force Position Compare Ready Interrupt 0: Invalid 1: Force position comparison ready interrupt	0h
8	PCM	R/W	Force Position Compare Match Interrupt 0: Invalid 1: Force position comparison match interrupt	0h
9	SEL	R/W	Force Strobe Event Latch Interrupt 0: Invalid 1: Force strobe event latch interrupt	0h
10	IEL	R/W	Force Index Event Latch Interrupt 0: Invalid 1: Force index event latch interrupt	0h
11	UTO	R/W	Force Unit Time Out Interrupt 0: Invalid 1: Force unit timeout interrupt	0h
12	QMAE	R/W	Force QMA Error Interrupt 0: Invalid 1: Force QMA error interrupt	0h
15:13	Reserved			0h

33.12.20 QEP flag register (QEPSTS)

Offset address: 0x38

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PCEF	R	Position Counter Error Flag This bit will update for each index event. 0: No position counter error occurred during the last index conversion 1: Position counter error occurred	0h
1	FIMF	R/W	First Index Marker Flag Write 1 to clear this bit.	0h

Field	Name	R/W	Description	Reset value
			0: The first index pulse did not occur 1: Set when the first index pulse occurs	
2	CDEF	R/W	Capture Direction Error Flag Write 1 to clear this bit. 0: No capture direction error occurred 1: Direction change occurred between capture position events	0h
3	COEF	R/W	Capture Overflow Error Flag Write 1 to clear this bit. 0: No capture overflow occurred 1: QEP capture timer overflow occurred	0h
4	QDLF	R	QEP Direction Latch Flag 0: Counterclockwise rotation (on index event marker) 1: Clockwise rotation (on index event marker)	0h
5	QDF	R	Quadrature Direction Flag 0: Counterclockwise rotation 1: Clockwise rotation	0h
6	FIDF	R	Direction on the First Index Marker 0: Counterclockwise rotation 1: Clockwise rotation	0h
7	UPEVNT	R/W	Unit Position Event Flag Write 1 to clear this bit. 0: No unit position event occurred 1: Unit position event occurred	1h
15:8	Reserved			0h

33.12.21 QEP Capture Timer Register (QCTMR)

Offset address: 0x3A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	QCTMR	R/W	Capture Timer This register bit provides the time base for the edge capture unit.	0h

33.12.22 QEP Capture Period Register (QCPRD)

Offset address: 0x3C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	QCPRD	R/W	Capture Period The register keeps the period count value between the last continuous QEP position events.	0h

33.12.23 QEP Capture Timer Latch Register (QCTMRLAT)

Offset address: 0x3E

Reset type: SYSRSn

Field	Name	R	Description	Reset value
15:0	QCTMRLAT	R	Capture Timer Latch Between two unit timeout events, the QEP capture timer value can be latched into this register and the QEP position counter can be read.	0h

33.12.24 QEP Capture Period Latch Register (QCPRDLAT)

Offset address: 0x40

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	QCPRDLAT	R	Capture Period Latch Between two unit timeout events, the QEP capture period value can be latched into this register and the QEP position counter can be read.	0h

33.12.25 QEP Revision Number Register (REV)

Offset address: 0x60

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	MAJOR	R	Major Revision Number This field specifies the major revision number of the QEP IP.	1h
5:3	MINOR	R	Minor Revision Number This field specifies the minor revision number of the QEP IP.	0h
31:6	Reserved			0h

33.12.26 Strobe Source Select Register (QEPSTROBESEL)

Offset address: 0x64

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
1:0	STROBESEL	R/W	Strobe Source Select 0x: QEP strobe after the polarity multiplexer 10: QEP strobe after the logical "OR" operation of the polarity multiplexer using ADCSOCA 11: QEP strobe after the logical "OR" operation of the polarity multiplexer using ADCSOCB	0h
31:2	Reserved			0h

33.12.27 QMA control register (QMACTRL)

Offset address: 0x68

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	MODE	R/W	Select Mode for QMA Mode 000: QMA module is bypassed	0h

Field	Name	R/W	Description	Reset value
			001: Operating mode 1 010: Operating mode 2 Other: Reserved (QMA module is bypassed)	
31:3	Reserved			0h

34 Serial peripheral interface (SPI)

34.1 Full Name and Abbreviation Description of Terms

Table 167 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Most Significant Bit	MSB
Least Significant Bit	LSB
Slave In Master Out	SIMO
Slave Out Master In	SOMI
Master Out Master In	MOMI
Slave In Slave Out	SISO
Low-Speed Peripheral Clock	LPCLK

34.2 Introduction

SPI is a high-speed synchronous serial input/output port, and is used for communication between MCU and peripherals or other controllers, including extending peripherals or external I/O using ADC converters, shift registers, and display drivers. It can work in master or slave mode, supports multi-device communication, and has 125 kinds of programmable transmission baud rates, and the transmitted character length can be set to 1-16 bits. To reduce the CPU usage, it supports 16-level transmit/receive FIFO function.

34.3 Main characteristics

(1) The polarity and phase of the clock are programmable, and four clock schemes are supported
- No delayed rising edge
- Delayed rising edge
- No delayed falling edge
- Delayed falling edge

(2) 125 kinds of programmable transmission baud rates

(3) Support 1-16-bit transmission data

(4) Support master/slave mode

(5) Support FIFO function, with 16 programmable interrupt levels

(6) Support DMA function

(7) Support transmission delay control

(8) SPI high-speed mode

(9) SPI 3-line mode

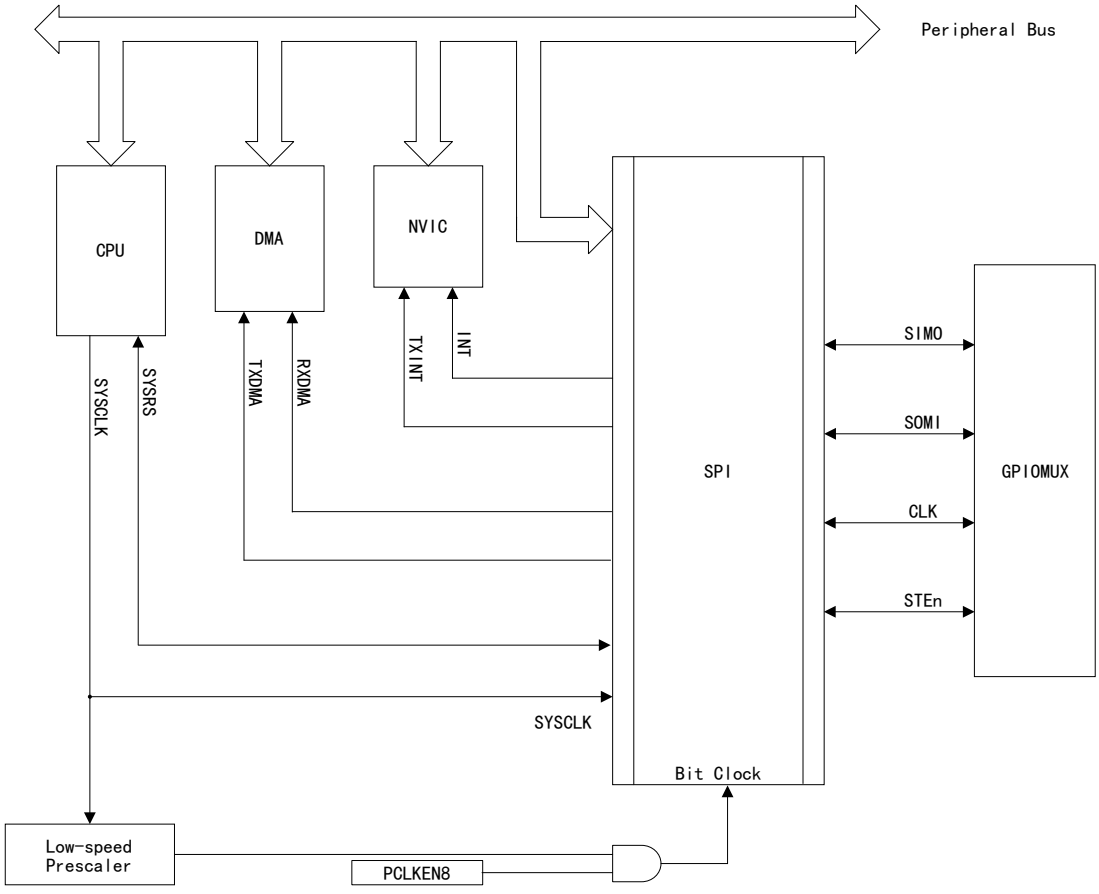
(10) Support digital audio transmission function

(11) Synchronous transmitting and receiving operations (the transmitting function can be disabled by software)

(12) Complete transmitting and receiving operations through interrupt drive or roll polling algorithm

34.4 Structure block diagram

Figure 171 CPU Interface Structure Block Diagram of SPI



34.5 Functional description

34.5.1 Description of SPI signal line

Table 168 SPI Signal Line Description

Signal category	Pin name	Description
External signal	SOMI	Slave: Output the pin and transmit data Master: Input the pin and receive data Data direction: From slave device to master device
	SIMO	Slave: Input the pin and receive data Master: Output the pin and transmit data Data direction: From master device to slave device
	CLK	Master: SPI clock output Slave: SPI clock input
	STEn	Enable slave transmitting
Interrupt signal	INT/RXINT	In FIFO mode, interrupt generated during data receiving In non-FIFO mode, interrupt generated when receiving/transmitting data (called an INT).
	TXINT	In FIFO mode, interrupt generated during data transmitting
DMA trigger signal	RXDMA	DMA receive request signal
	TXDMA	DMA transmit request signal
Control signal	LPCLK	SPI clock rate

STEn Signal Notes

When the SPI slave loses synchronization with the master, the internal bit counter and the various status flags in the SPI can be reset by flipping the SPIRDY value. After resetting the bit counter, the data transmitted by the next clock signal will be recognized by SPI as the first bit of new data. For details of SPIRDY reset, please refer to the Configuration Register section (SPI_CFG). When SPI is in slave mode, the STEn signal can gate the spurious clock and data pulses. When STEn is in high level, it prevents the slave from receiving data, to prevent the SPI slave from losing synchronization with the master. Therefore, frequent activation of STEn is not recommended.

Simulating STEn

Normally, an SPI master can connect more than one SPI slave through multiple STEn pins. However, in this device, even though there are no multiple STEn signals to connect multiple SPI slaves, the system can still simulate multiple STEn signals using software as follows:

Use GPIO pins to replace STEn pins. In this case, SPI must be configured in

master mode. In this configuration, each SPI slave is controlled through a dedicated GPIO pin. These GPIO pins need to be configured in output mode, without using the GPIO multiplexing to select STEn. Before transmitting data, set the GPIO pin corresponding to the target SPI slave as the active state. Once data transmission is complete, immediately set the GPIO pin as the non-active state. This operation will repeatedly occur on multiple slaves sharing the same master CLK, SIMO, and SOMI signal lines.

34.5.2 Pin configuration

By configuring the multiplexing registers in the GPIO section, select the corresponding pins to connect the SPI to the device. The GPIO multiplexing registers must be configured according to the correct steps outlined in the GPIO section, first configuring the GPxGMUX bit band (at which point the corresponding GPxMUX bit band is set as the default value of 0), and then writing the required value to the GPxMUX bit band.

The GPIO register is configured with some IO functions independent of SPI peripherals. The internal pull-up resistor status can be configured in the GPxPUD register. For input signals, the GPIO pin can be configured as asynchronous input by setting the corresponding GPxQSEL1/2[GPIOn]=11. For more information on GPIO multiplexing, refer to the GPIO section.

High-Speed Mode Pin Configuration

All GPIOs can be multiplexed for SPI high-speed mode. Enable SPI high-speed mode by setting SPI_CFG[HSEN] to 1.

Note: When using SPI high-speed mode, ensure that the capacitive load on the pins is within the values specified in the datasheet. If SPI operates in non-high-speed mode or the capacitive load on the pins exceeds the values specified in the datasheet, set SPI_CFG[HSEN] to 0.

34.5.3 SPI interrupt

There are two interrupt lines in SPI: INT/RXINT and TXINT.

(1) In FIFO mode, RXINT and TXINT interrupts can be generated;

(2) In non-FIFO mode, all available interrupts are routed together to generate a single INT interrupt.

Figure 172 SPI Interrupt Logic

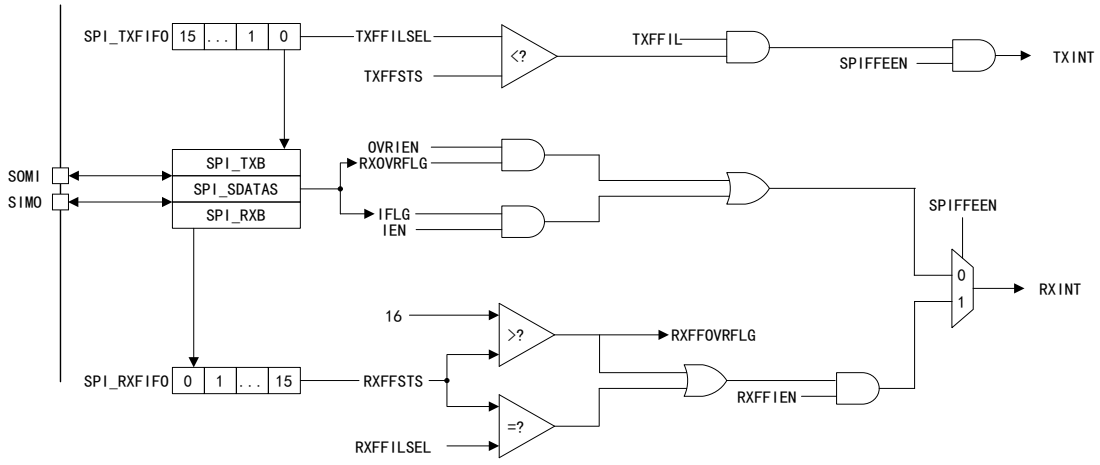


Table 169 SPI Interrupt Request

Mode	Interrupt lines	Interrupt event	Interrupt flag	Enable control bit
Non-FIFO mode (SPIFFEEN=0)	RXINT (INT)	The transmit buffer is empty	IFLG	IEN
		Receiver buffer overflow	RXOVRFLG	OVRIEN
		Data receiving	IFLG	IEN
FIFO mode (SPIFFEEN=1)	RXINT	The transmit buffer is empty	TXFFILSEL	TXFFIEN
	TXINT	FIFO receiving	RXFFILSEL	RXFFIEN

Note:

(1) In non-FIFO mode, the interrupts generated by RXINT and INT are collectively referred to as INT.

34.5.3.1 INT/RXINT Interrupt

FIFO mode

In FIFO mode, SPI generates a CPU interrupt by determining the relationship between the receive FIFO status (RXFFSTS) and the selected receive FIFO interrupt level (RXFFILSEL). When RXFFSTS ≥ RXFFILSEL, the receive FIFO interrupt flag (RXFFIFLG) is set. At this point, if the receive FIFO interrupt is enabled (RXFFIEN = 1), an RXINT interrupt will be triggered in the NVIC module.

Non-FIFO mode

In non-FIFO mode, the interrupts generated are collectively referred to as INT; when FIFO interrupt enhancement is enabled, the generated interrupt is

referred to as RXINT. In the interrupt controller (NVIC), these interrupts share the same interrupt vector.

Both of the following conditions can trigger an interrupt and share the INT vector:

(1) Transmission completion: Transmission completion indicates that the last bit of data has been transmitted/received during the SPI transmission, and the system is ready for the next operation. At this point, the interrupt flag bit is set (IFLG = 1), and the received character is placed into the receive buffer (SPI_RXB). If the receive FIFO interrupt is enabled (RXFFIEN = 1), IFLG will generate an interrupt on the INT vector.

(2) Receive overflow: Receive overflow occurs when a transmission/reception operation is completed before the previous character is read from the buffer. At this point, the receive overflow flag bit (RXOVRFLG) is set. If the overflow interrupt is enabled (OVRIEN = 1) and the previous RXOVRFLG bit is cleared, RXOVRFLG will trigger an interrupt on the INT vector.

34.5.3.2 TXINT interrupt

FIFO mode

In FIFO mode, the method for generating a TXINT interrupt is similar to that of generating an RXINT interrupt. SPI generates a TXINT interrupt by determining the relationship between the transmission FIFO status (TXFFSTS) and the selected transmission FIFO interrupt level (TXFFILSEL). When TXFFSTS ≤ TXFFILSEL, the transmission FIFO interrupt flag (TXFFIFLG) is set. At this point, if the transmission FIFO interrupt is enabled (TXFFIEN= 1), an TXINT interrupt will be triggered in the NVIC module.

Non-FIFO mode

TXINT interrupts are not available in non-FIFO mode.

34.5.4 DMA function

34.5.4.1 DMA events

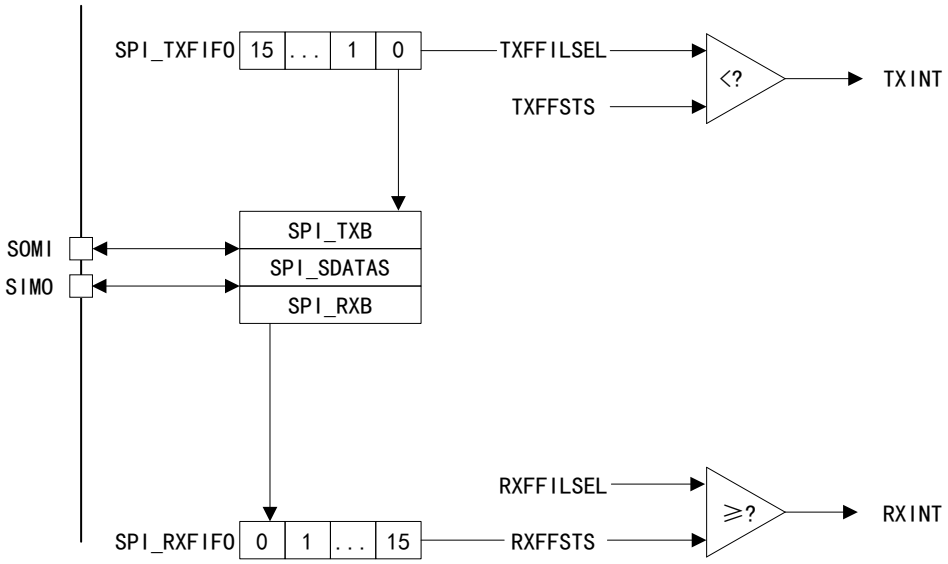
The SPI data register can be accessed by both the CPU and DMA via the internal peripheral bus, and can only be read/written through a 16-bit register. DMA can be triggered only when FIFO enhancement is enabled. Each SPI module can generate two interrupt events, i.e. SPITXDMA and SPIRXDMA, to activate the corresponding DMA events according to the following configuration:

(1) SPITXDMA Event: In the SPI_TXFIFO register, when TXFFSTS < TXFFILSEL, the SPITXDMA event is activated.

(2) SPIRXDMA Event: In the SPI_RXFIFO register, when RXFFSTS< RXFFILSEL, the SPITXDMA event is activated.

The SPI triggers a DMA event as shown in the figure below:

Figure 173SPI-triggered DMA Event



34.5.4.2 DMA transmission

Transmit data

When configuring the TX FIFO with DMA function for SPI communication, to prevent DMA from continuing to write data to a full FIFO and thus causing data loss, ensure that the DMA burst size (BURST_SIZE[BURSTSIZE]) does not exceed (16 - SPI_TXFIFO[TXFFILSEL]).

Data transmission steps are as follows:

- Calculate the total number of words to be transmitted, with WORDNUM representing the total word count of the data to be transmitted;
- Select the FIFO interrupt level in SPI_TXFIFO[TXFFILSEL];
- Calculate the size of the DMA transmission in TRANSFER_SIZE[TRANSFERSIZE];
- Calculate the DMA burst size in BURST_SIZE[BURSTSIZE];
- Configure DMA based on the calculated transfer size and burst size to meet the data transmission requirements;
- Set SPI and FIFO: Adjust SPI-related parameters according to the calculated values;

Note: Avoid configuring SPI_TXFIFO[TXFFILSEL] as 0x00 or 0x10 to ensure the proper functioning of the DMA configuration.

For example, to use DMA to transmit 128-word data, the configuration of each

parameter is as shown in the table below:

Table 170 128-word DMA Data Transmission Example

Parameter	Calculation process	Calculation result
WORDNUM	-	128
TXFFILSEL	-	8
TXSIZE	=(WORDNUM/TXFFILSEL)-1=(128/8)–1=15	15 (16 transmissions)
BURSTSIZE	(16- TXFFILSEL)-1=16-8–1=7	7 (8 words per burst);

Note: Avoid configuring SPI_TXFIFO[TXFFILSEL] as 0x00 or 0x10 to ensure the proper functioning of the DMA configuration.

Receive data

When configuring the RX FIFO with DMA function for SPI communication, to prevent DMA from reading data from an empty FIFO, ensure that the DMA burst size (BURST_SIZE[BURSTSIZE]) does not exceed SPI_RXFIFO[RXFFILSEL]. BURST_SIZE[BURSTSIZE] shall be configured as SPI_RXFIFO[RXFFILSEL] or a divisor of the total SPI transmission count to ensure that all data from the RX FIFO can be correctly received by DMA.

Data reception process is as follows:

(1) Calculate the number of words to be received, with WORDNUM representing the total word count of the data to be received;

- Select the receive FIFO interrupt level in SPI_RXFIFO[RXFFILSEL];
- Calculate the size of the DMA transmission in TRANSFER_SIZE[TRANSFERSIZE];
- Calculate the DMA burst size in BURST_SIZE[BURSTSIZE];
- Configure DMA based on the calculated transfer size and burst size to meet the data transmission requirements;
- Set SPI and FIFO: Adjust SPI-related parameters according to the calculated values;

Note: To ensure reasonable DMA configuration, avoid configuring SPI_TXFIFO[TXFFILSEL] as 0x00.

Table 171 200-word DMA Data Reception Example

Parameter	Calculation process	Calculation result
WORDNUM	-	200
RXFFILSEL	-	4
TXSIZE	=(WORDNUM/RXFFILSEL)-1=(200/4)–1=49	49 (50 transmissions)
BURSTSIZE	RXFFILSEL-1=4–1=3	3 (4 words per burst);

34.5.5 SPI mode

34.5.5.1 SPI master/slave connection

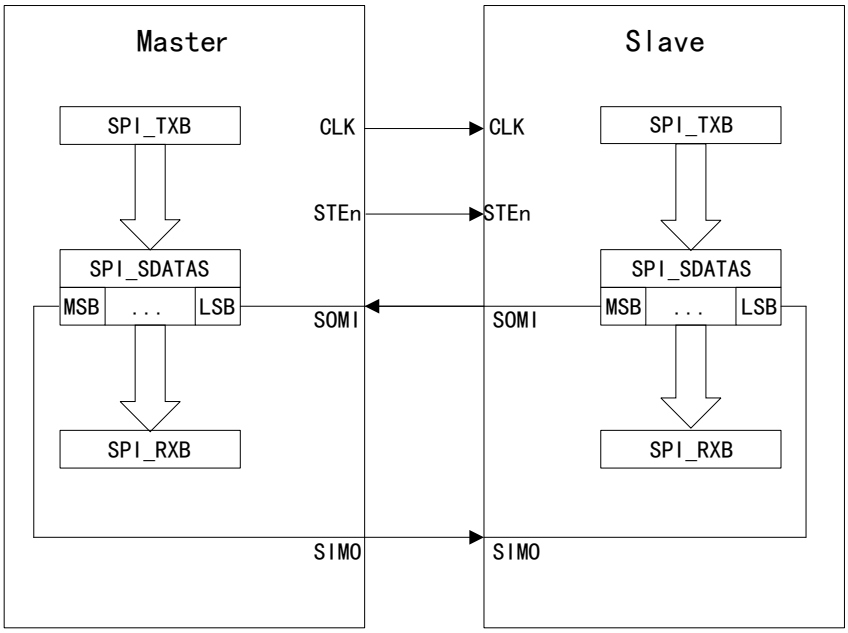
The master transmits data through the CLK signal. For both SPI master and slave, when SPI_CTRL[PHASEL]=0, data is shifted out of the shift register on one edge (rising or falling) of the CLK clock and latched into the shift register on the opposite edge (falling or rising). When SPI_CTRL[PHASEL]=1, the data transmission and reception occur during the first half cycle of the CLK signal conversion. Therefore, SPI allows both the master and slave to transmit and receive data simultaneously, and software needs to determine whether the transmitted data is valid. There are three possible data transmission methods in total:

Table 172 Data Transmission Methods

Mode	Master	Slave
1	Transmit valid data	Transmit invalid data
2	Transmit valid data	Transmit valid data
3	Transmit invalid data	Transmit valid data

Since the SPI master controls the CLK signal, data transmission can be initiated at any time, but the method by which the master detects whether the slave is ready to transmit data is determined by software. SPI can select the operating mode through SPI_CTRL[MSCFG], that is, selecting the working mode and the source of the CLK signal through this bit. The SPI master/slave connection is shown in the figure below:

Figure 174 SPI Master/Slave Connection



34.5.5.2 SPI master mode

Configure SPI_CTRL[MSCFG] as1 to select the SPI operating mode as master mode. In master mode, data is output through the SIMO pin, and the SOMI pin receives and latches the data. SPI provides a serial clock for the serial communication network via the CLK pin.

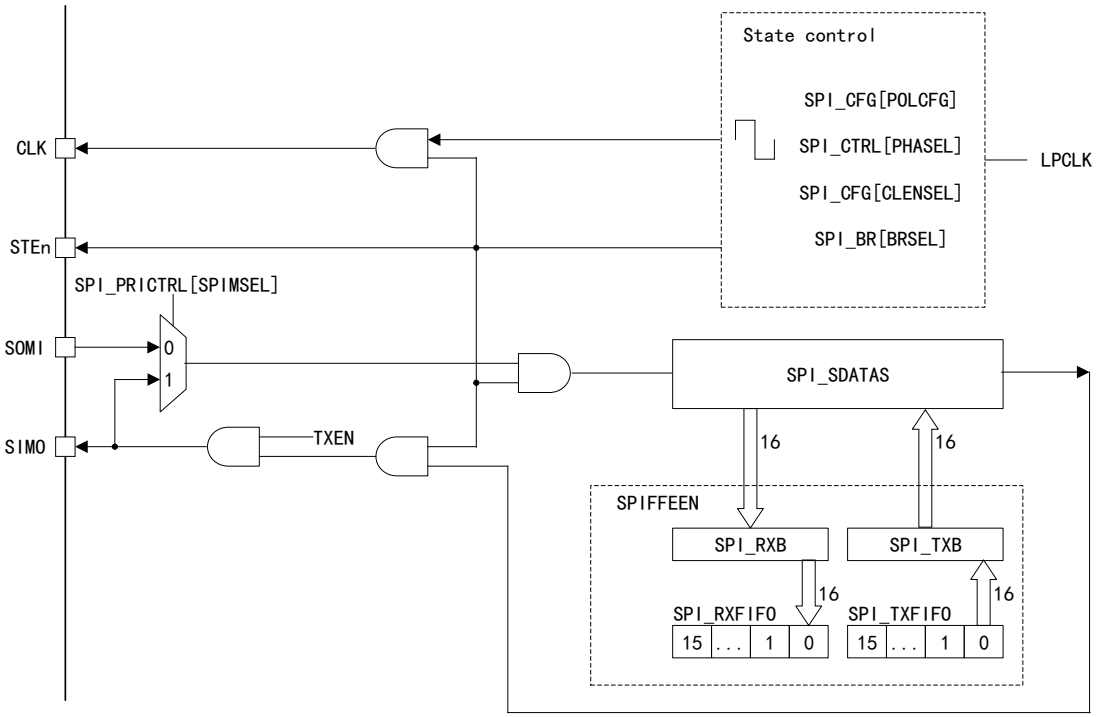
The bit transmission rate is configured in the baud rate register (SPI_BR), with a total of 125 data transfer rates available for selection. After writing data into the serial data shift register (SPI_SDATAS) and transmit buffer register (SPI_TXB), data is transmitted sequentially from the SIMO pin starting from the most significant bit (MSB). While data is being transmitted, the data received on the SOMI pin is sequentially filled into the least significant bit (LSB) of the serial data shift register (SPI_SDATAS). After the data transmission of the predetermined sequence is completed, the received data is transferred to the receive buffer register (SPI_RXB). In SPI_RXB, the data is stored in right-aligned format for the CPU to perform read operations.

When a set number of data is moved into SPI_SDATAS, the data in SPI_SDATAS is transferred to SPI_RXB, and at the same time the interrupt flag (IFLG = 1) is set. At this point, if there is valid data in SPI_TXB (i.e., the transmit buffer full flag TXBFFLG = 1), that data is moved to SPI_SDATAS and transmitted. Otherwise, when all data bits are shifted out of SPI_SDATAS, the CLK stops. When SPI interrupt is enabled (i.e., IEN = 1), an SPI interrupt is generated.

STEn is used as the SPI slave chip select enable pin in typical applications. It is driven low by the master before the data is transmitted to the master, and driven high after the transmission is complete.

The structure block diagram of the SPI in master mode is shown below:

Figure 175 SPI Master Mode



34.5.5.3 SPI slave mode

Configure SPI_CTRL[MSCFG] = 0 to select the SPI operating mode as slave mode. In slave mode, data is output through the SOMI pin and input through the SIMO pin. The external master provides the serial shift clock and inputs it through the CLK pin. In addition, the serial shift clock defines the transmission rate, and the input frequency of CLK shall not exceed fLPCLK/4.

In slave mode, when the appropriate edge jump of the CLK clock from the master is detected, the data written into SPI_SDATAS and SPI_TXB is transmitted, according to the character shift status in the SPI_SDATAS register:

- When all data bits of the current character in SPI_SDATAS have been shifted out (i.e., the data has been fully transmitted), the character written into the SPI_TXB register is immediately copied to the SPI_SDATAS register;
- When SPI_SDATAS is empty, i.e., there is no character copied into SPI_SDATAS in advance, the character written into the SPI_TXB register is immediately copied to the SPI_SDATAS register;
- When SPI_SDATAS is not empty, i.e., there is a character copied into SPI_SDATAS in advance, any data written to SPI_TXB will not be copied into SPI_SDATAS until the current character in SPI_SDATAS is shifted out

When receiving data, the SPI module waits for the CLK clock signal from the master and then shifts the data received on the SIMO pin into SPI_SDATAS.

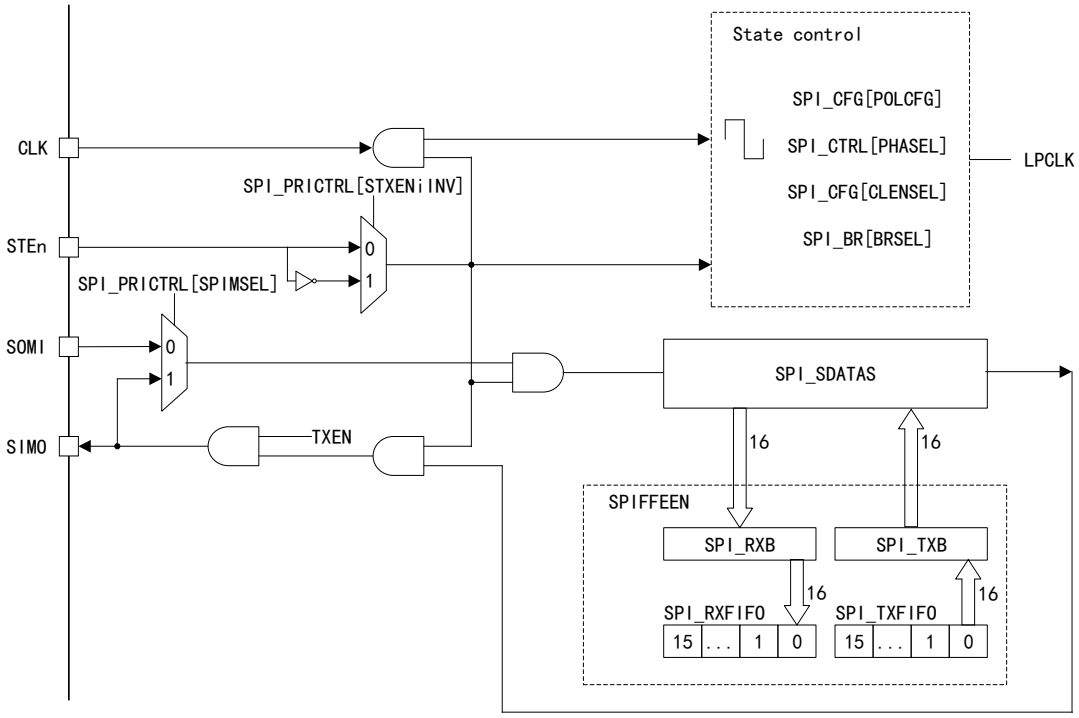
When data needs to be synchronously transmitted by the slave, if the data has not been loaded into SPI_SDATAS in advance, the character to be transmitted must be written into SPI_TXB before the CLK signal starts.

The TXEN bit is used to control whether data can be transmitted. When the TXEN bit is cleared, the data transmission function is disabled by setting the SOMI pin in a high-impedance state. The TXEN bit is cleared during data transmission, but the data being transmitted will still be fully transmitted. To ensure the SPI module can correctly receive input data, the SPI module will not force the SOMI pin into a high-impedance state before the data transmission is completed. It will only force the SOMI pin into a high-impedance state after the current data transmission is completed. The TXEN bit can be used to control which device may transmit data. Multiple slave devices can be connected in the SPI network, but only one device is allowed to drive the SOMI pin at any given time.

The STEn pin is the slave select pin. When STEn is in low level, the slave SPI enables data transmission to the serial data line; when STEn is in high level, the slave SPI serial shift register (SPI_SDATAS) stops, and the output pin enters a high-impedance state. Although only one slave device can communicate at any given time, this setup allows multiple devices to be connected on the same bus.

The structure block diagram of the SPI in slave mode is shown below:

Figure 176 SPI Slave Mode



34.5.6 Data format

The CLENSEL field in the configuration register (SPI_CFG) selects the

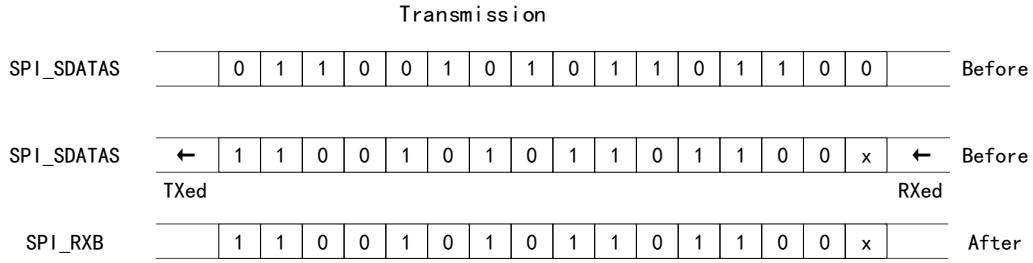
character length (1 to 16 bits). The CLENSEL field allows the state control logic unit to calculate the number of bits transmitted to determine the time when a complete character is processed.

For characters smaller than 16 bits, data must be written into SPI_SDATAS and SPI_TXB in left-aligned format and read from SPI_RXB in right-aligned format. In addition, SPI_RXB contains the most recently received character (right-aligned), as well as any remaining characters shifted to the left from previous transmissions.

The bit transmission in SPI_RXB is shown in the figure below.

Assume the length of the transmission character is 1 bit, and the current value in SPI_SDATAS is 0x656C.

Figure 177 SPI_RXB Bit Transmission Example



Assume in master mode, when the SOMI signal is in high level, x = 1; when the SOMI signal is in low level, x = 0.

34.5.7 Transmission baud rate

SPI offers four clock schemes to choose from and supports 125 different baud rates.

(1) In master mode, SPI generates the SPI clock and outputs it through the CLK pin. The clock frequency cannot exceed LPCLK/4;

(2) In slave mode, the SPI clock is generated by an external source and input through the CLK pin. The clock frequency cannot exceed LPCLK/4;

Note: When configuring the baud rate, it must not exceed the maximum switching frequency of the GPIO. For details on the maximum rated switching frequency of the GPIO, refer to the datasheet.

Select transmission baud rate

- When SPI_BR = 0/1/2:

$$\text{SPI baud rate} = \frac{\text{LPCLK}}{4}$$

- When SPI_BR = 3-127:

$$\text{SPI baud rate} = \frac{LPCLK}{(SPI_BR + 1)}$$

Where:

LPCLK: Low-speed peripheral clock frequency

SPI_BR: Value of baud rate register in the master SPI

The system clock frequency (LPCLK) and the operating baud rate must be known to determine the value to be written into the SPI_BR register.

Calculate transmission baud rate

For example, with LPCLK = 60 MHz, SPI_BR = 5, and SPI operating in non-high-speed mode (HSEN = 0):

$$\text{SPI baud rate} = \frac{LSPCLK}{(SPI_BR + 1)}$$

$$= \frac{60 \times 10^6}{5 + 1}$$

$$= 10.0\text{Mbps}$$

34.5.8 Phase and polarity of clock signal

The clock polarity is controlled by the SPI_CFG[POLCFG] bit, and the clock phase is controlled by the SPI_CTRL[PHASEL] bit.

Clock polarity refers to the effective edge of the CLK signal line.

- When POLCFG = 0, SPI transmits data on the clock's rising edge and receives data on the falling edge. The CLK signal line is at low level during SPI idle state.
- When POLCFG = 1, SPI transmits data on the clock's falling edge and receives data on the rising edge. The CLK signal line is at high level during SPI idle state

Clock phase refers to whether the CLK signal is delayed by half a cycle.

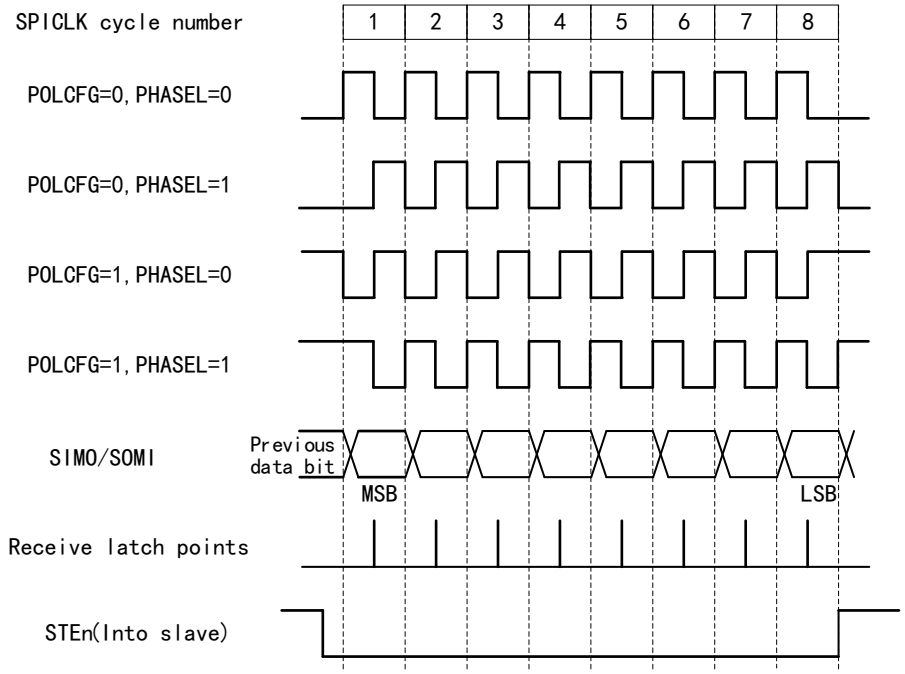
- When PHASEL = 0, there is no delay.
- When PHASEL = 1, the CLK signal is delayed by half a cycle.

According to the different states of clock phase POLCFG and clock polarity PHASEL, there are four clock schemes.

Table 173 Four Clock Schemes of SPI

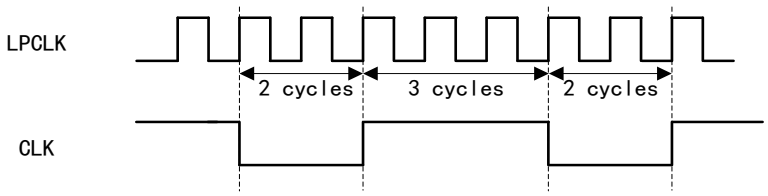
POLCFG	PHASEL	Idle CLK clock	Delay	Clock scheme
0	0	Low level	None	No delayed rising edge. That is, SPI transmits data on the rising edge of the clock and receives data on the falling edge.
0	1	Low level	Yes	Delayed rising edge. That is, SPI transmits data during the first half cycle of the rising edge and receives data on the rising edge.
1	0	High level	None	No delayed falling edge. That is, SPI transmits data on the falling edge of the clock and receives data on the rising edge.
1	1	High level	Yes	Delayed falling edge. That is, SPI transmits data during the first half cycle of the falling edge and receives data on the falling edge.

Figure 178 SPI Signal Options



Select the appropriate waveform based on the required operation to determine the configuration of clock phase and polarity.

Figure 179 CLK-LPCLK Characteristics



As shown in the figure, the symmetry of CLK is affected by SPI_BR and POLCFG:

(1) When SPI_BR + 1 is an even value, CLK retains symmetry

(2) When SPI_BR + 1 > 3 and is an odd value, CLK is asymmetric. In this case:

- If POLCFG = 0, the low-level pulse of SPI is one LPCLK cycle longer than the high-level pulse
- If POLCFG = 1, the high-level pulse of SPI is one LPCLK cycle longer than the low-level pulse

34.5.9 FIFO characteristics

(1) Enable FIFO Mode: Set SPI_TXFIFO[SPIFFEEN] to 1 to enable FIFO mode. At any time during SPI operation, FIFO mode can be reset by setting SPI_TXFIFO[SPIEN] to 1. After enabling FIFO mode, all SPI registers and SPI FIFO registers are activated.

(2) SPI Reset: Upon reset, SPI starts in standard mode, and the FIFO function is disabled. Therefore, FIFO registers (SPI_TXFIFO, SPI_RXFIFO, and SPI_FIFOCTRL) are also disabled;

(3) FIFO Status: The SPI_TXFIFO[TXFFSTS] and SPI_RXFIFO[RXFFSTS] bits indicate the number of valid words in the transmit and receive buffers, respectively. When SPI_TXFIFO[TXFFEN] and SPI_RXFIFO[RXFFEN] are set to 1, the FIFO pointers are reset; When SPI_TXFIFO[TXFFEN] and SPI_RXFIFO[RXFFEN] are cleared, the FIFO begins the reset operation.

(4) Buffer: In addition to the original transmit and receive buffers, a 16-word FIFO is added for each. In standard SPI, the transmit FIFO and shift register use a 1-word transmit buffer register (SPI_TXB) as the transmission buffer. The transmit FIFO will load data into SPI_TXB only after the last bit of the character in the shift register is shifted out.

(5) Interrupts:

- In non-FIFO mode, INT/RXINT is used as the interrupt source.
- In FIFO mode, there are two interrupt sources: TXINT and RXINT, which are used for the transmit and receive FIFOs, respectively. SPI FIFO receive, receive errors, and receive FIFO overflow share the

RXINT interrupt. A single INT interrupt is used for both transmit and receive in non-FIFO mode, and disabled in FIFO mode. RXINT serves as the SPI FIFO receive interrupt. For detailed information about interrupts, refer to the SPI Interrupt section.

- Interrupt levels are programmable. Both the transmit FIFO and the receive FIFO can generate DMA trigger interrupts and CPU interrupts. When the TXFFILSEL bit and TXFFSTS bit match (i.e., TXFFSTS ≤ TXFFILSEL), a transmit FIFO interrupt request (TXINT) is generated. When the RXFFILSEL bit and RXFFSTS bit match (i.e., RXFFSTS ≥ RXFFILSEL), a receive FIFO interrupt request (RXINT) is generated. These operations provide programmable interrupt levels for SPI transmit and receive. RXFFILSEL is configured by default as 0x11111, and TXFFILSEL is configured by default as 0x00000.

(6) Transmission Delay: The rate at which data is transmitted from the FIFO to the shift register can be adjusted. The SPI_FFCTRL[FFTXDLYSEL] bit defines the delay between the transmit FIFO buffer and the transmit shift register. This delay is defined by the number of SPI clock cycles. When there is no delay, SPI transmits data in continuous mode, i.e., words are shifted out of the FIFO one by one. When there is a delay of 255 clock cycles, SPI transmits data in maximum delay mode. The delay of 255 CLK cycles of each word in the FIFO is shifted out of the FIFO. Transmission delay facilitates non-glue interface applications with various slow SPI peripherals, such as EEPROMs, ADCs, and DACs.

34.5.10 SPI special mode

34.5.10.1 High-speed mode

Communication speed

In SPI's high-speed mode, full-duplex data transmission can be performed at a maximum rate of LPCLK/4.

In high-speed mode, SPI must be configured for a single-master-to-single slave connection, and the pin load must be within the values specified in the datasheet to reach the maximum full-duplex speed. For information on the maximum rated speed, refer to the datasheet.

The process to configure the maximum rated speed is as follows, assuming operation at 100 MHz:

(1) Set LPCLK = SYSCLK;

(2) Select appropriate GPIO multiplexed pins;

(3) Configure SPI as follows:
- HSEN = 1, enabling high-speed mode;
- BRSEL = 3, i.e., CLK = LPCLK / (BRSEL + 1) = 25;

Other configurations in high-speed mode are the same as those in normal SPI mode; data transmission and reception, DMA, and interrupt methods are also consistent with those of the normal SPI mode. For detailed information on the SPI configuration process, refer to the SPI Configuration section.

34.5.10.2 3-wire mode

Differences between 3-wire and 4-wire modes

In normal mode, SPI communicates via four pins. The 3-wire mode of SPI allows communication via three pins.

Enable 3-wire mode by setting SPIMSEL as 1. In master mode, the SOMIx pin is no longer used, and the SIMOx pin serves as a bidirectional SPIMOMI pin. In slave mode, the SIMOx pin is no longer used, and the SOMIx pin serves as a bidirectional SPISISO pin.

The pin function differences between 3-wire and 4-wire modes are shown in the table below:

Table 174 Pin Function Differences between 3-wire and 4-wire Modes

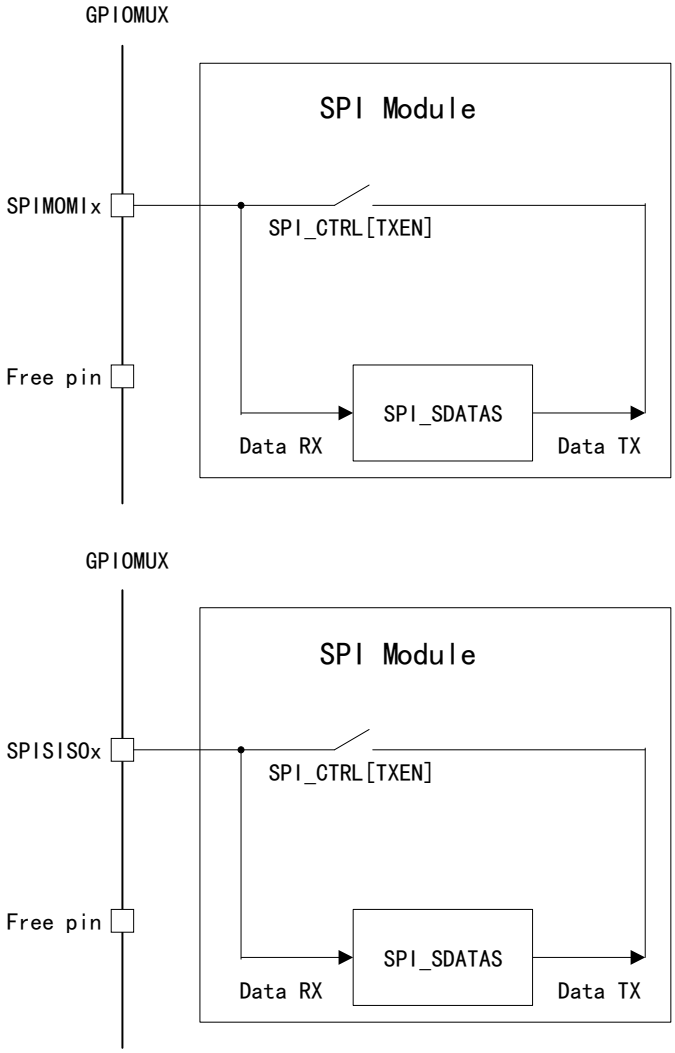
4-wire mode	3-wire mode (master)	3-wire mode (slave)
SOMIx	Idle	SISOx
SIMOx	MOMIx	Idle
CLKx	CLKx	CLKx
STEx	STEx	STEx

Data transmission method

In 3-wire mode, since the receive and transmit paths in SPI are connected together, any data transmitted by the SPI module will also be received by the SPI module itself. To clear any additional received data in the SPI data register, the software layer must perform a virtual read.

3-wire mode data transmission is shown in the diagram below:

Figure 180 SPI 3-wire mode



Set SPI_CTRL[TXEN] to 1 to transmit data, and clear this bit before reading data to receive it. In master mode, to initiate read operations, before reading from the data register, the software layer must write virtual data to the SPI data register (SPI_SDATAS or SPI_RXB) while SPI_CTRL[TXEN] is cleared (with no data being transmitted to SPIMOMI).

When SPI_CTRL[TXEN] is set or cleared, the pin configurations for each SPI mode are shown in the table below:

Table 175 3-wire SPI Pin Configuration

Master/slave mode	Mode	SPI_PRICTRL[SPIMSEL]	SPI_CTRL[TXEN]	SIMO	SOMI
Master mode	4-wire mode	0	X	TX	RX
	3-wire mode	1	0	RX	Not used
			1	TX/RX	

Master/slave mode	Mode	SPI_PRICTRL[SPIMSEL]	SPI_CTRL[TXEN]	SIMO	SOMI
Slave mode	4-wire mode	0	X	RX	TX
	3-wire mode	1	0	Not used	RX
			1		TX/RX

3-wire mode transmit/receive example

In 3-wire mode, in addition to the general SPI initialization steps (refer to the Reset and Initialization section for details), it is also required to set SPI_PRICTRL[SPIMSEL] to configure SPI for 3-wire mode. When transmitting and receiving data in 3-wire mode, the considerations for master and slave modes are as follows:

(1) Master mode transmit: Configure the CLKx, STEx, and SIMOx pins as SPI pins, and the SOMIx pin as a non-SPI pin. Since SIMOx and SOMIx are internally connected in 3-wire mode, when the master transmits data, it will receive the data it has transmitted. Based on the above characteristics, it is required to clear the junk data received in SPI_RXB each time data is transmitted.

(2) Master mode receive: Configure the CLKx, STEx, and SIMOx pins as SPI pins, and the SOMIx pin as a non-SPI pin. Additionally, to initiate the read operation, the master must clear the TXEN bit to disable the transmission path, then transmit virtual data to the SPI data register. Different from transmission mode, since the TXEN bit is cleared, the virtual data transmitted by the master will not be transmitted to the SIMOx pin, nor will it be received by the master. Instead, the master will receive data from the slave.

(3) Slave mode transmit : Configure the CLKx, STEx, and SOMIx pins as SPI pins, and the SIMOx pin as a non-SPI pin. Since SOMIx and SIMOx are internally connected in 3-wire mode, when the slave transmits data, it will receive the data it has transmitted. Based on the above characteristics, it is required to clear the junk data received in SPI_RXB each time data is transmitted.

(4) Slave mode receive: Configure the CLKx, STEx, and SOMIx pins as SPI pins, and the SIMOx pin as a non-SPI pin. Additionally, to initiate the read operation, the slave must clear the TXEN bit to disable the transmission path, then transmit virtual data to the SPI data register. Otherwise, the slave will receive data as usual.

34.5.11 Reset and Initialization

When the system is reset, the SPI is forcibly initialized to the default

configuration. The default configuration is as follows:

(1) SPI is configured in slave mode (MSCFG=0);

(2) Disable the SPI interrupt function;

(3) Disable the transmission function (TXEN=0);

(4) Character length is 1 bit (CLENSEL=0);

(5) The value in the SPI_SDATAS register is the reset value 0x0000;

(6) Latch data on the input end of the CLK signal's falling edge.

34.5.12 Configuration process

Before making initialization changes, SPIRDY must be cleared to avoid unpredictable or unnecessary errors during or after initialization. Set SPIRDY after initialization is completed. When SPI remains in the reset state (i.e., SPIRDY=0), there is no specific order for the configuration. Since SPI registers can be updated with a single 16-bit write operation, apart from the SPIRDY configuration, there is no need for special attention to the order of other register configurations. The following list shows the SPI configuration process in logical order.

Table 176 SPI Configuration Process

<table>
<tr><th>Logical order</th><th colspan="3">Configuration process</th><th>Field configuration</th></tr>
<tr><td>1</td><td colspan="3">Force SPI into reset state</td><td>SPIRDY=0</td></tr>
<tr><td rowspan="12">2</td><td rowspan="12">Configure SPI as needed</td><td colspan="2">Configure master/slave mode</td><td>MSCFG</td></tr>
<tr><td colspan="2">Configure CLK polarity</td><td>POLCFG</td></tr>
<tr><td colspan="2">Select CLK phase</td><td>PHASEL</td></tr>
<tr><td colspan="2">Select baud rate</td><td>BRSEL</td></tr>
<tr><td colspan="2">(Optional) Enable high-speed mode</td><td>HSEN</td></tr>
<tr><td colspan="2">Select character length</td><td>CLENSEL</td></tr>
<tr><td colspan="2">Clear SPI flags</td><td>IFLG, RXOVRFLG</td></tr>
<tr><td colspan="2">(Optional) Enable inverted STEn</td><td>STEINV</td></tr>
<tr><td colspan="2">(Optional) Enable 3-wire mode</td><td>SPIMSEL</td></tr>
<tr><td rowspan="3">(Optional) Use FIFO enhancement</td><td>Enable SPI FIFO</td><td>SPIFFEEN</td></tr>
<tr><td>Clear FIFO flag</td><td>TXFFICLR, RXFFOVRCLR, RXFFICLR</td></tr>
<tr><td>Release transmit and receive FIFO reset</td><td>TXFFEN, RXFFEN</td></tr>
</table>

Logical order	Configuration process			Field configuration
			Release SPI FIFO channel from reset	SPIEN
3	(Optional) Use interrupts	FIFO mode	Set transmit and receive interrupt levels, and enable interrupts	TXFFILSEL, RXFFILSEL, TXFFIEN, RXFFIEN
		Non-FIFO mode	Enable receive overflow and/or SPI interrupts	OVRIEN, IEN
4	Release SPI from reset state			SPIRDY=1

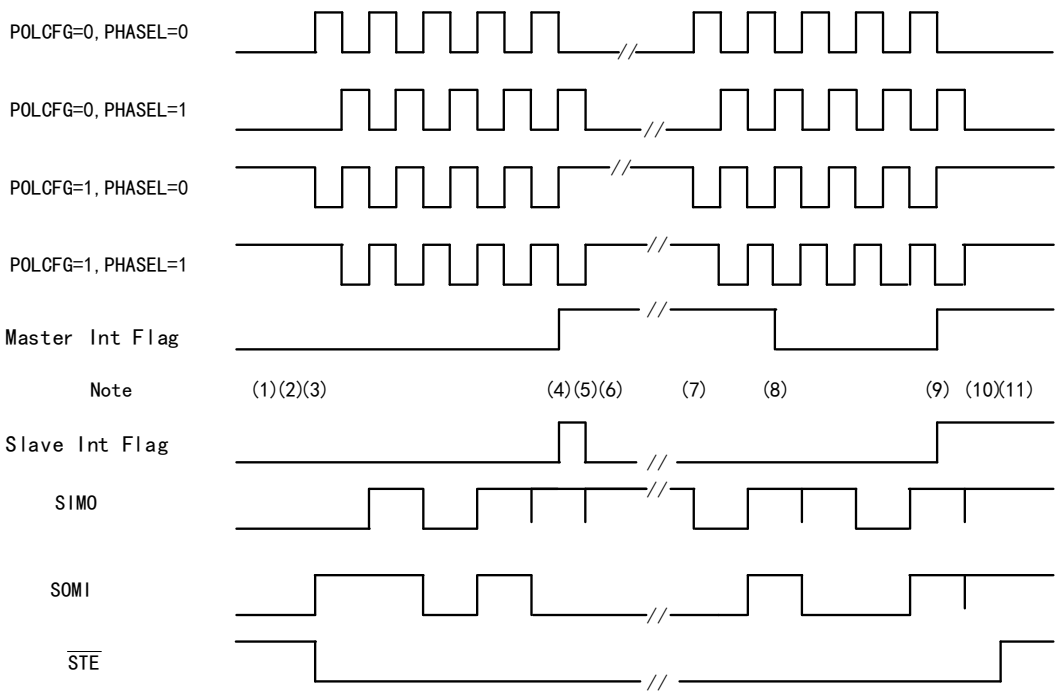
Note: Changing the SPI configuration is prohibited during SPI communication.

34.5.13 SPI transmission timing

For example, with a character length of 5 bits and a symmetric CLK signal, the data transmission between the SPI master and slave is shown below:

Note: The following diagram applies only to SPI with 8-bit data transmission and does not apply to APM52 series devices that can process 16-bit data. This diagram is for reference only.

Figure 181 SPI Data Transmission



Where:

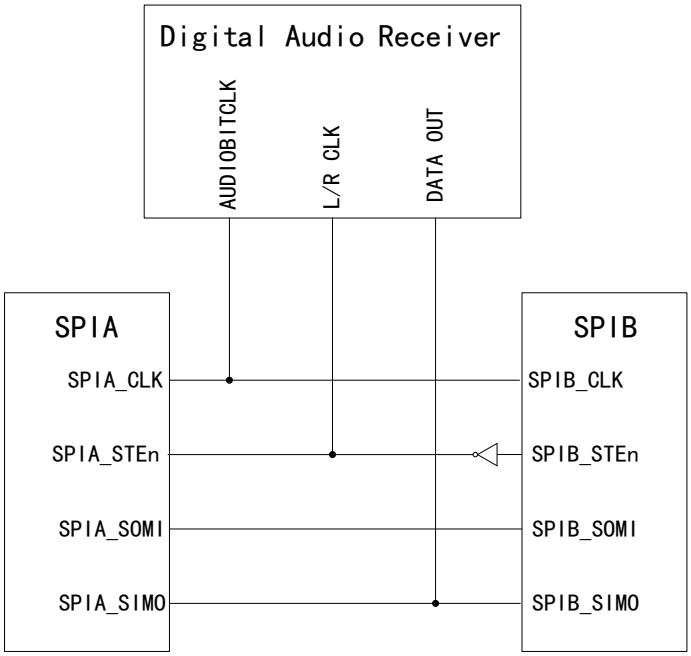
Comment	Description
(1)	The slave writes 0x0D0 to SPI_SDATAS and waits for the master to shift the data out

Comment	Description
(2)	The master sets the STEn of the slave to low active (activated)
(3)	The master writes 0x058 to SPI_SDATAS, and SPI_SDATAS starts the transmission
(4)	After completing the transmission of the first byte, set the interrupt flag
(5)	The slave reads data 0x0B from SPI_RXB in right-aligned form
(6)	The slave writes 0x04C to SPI_SDATAS and waits for the master to shift the data out
(7)	The master writes 0x06C to SPI_SDATAS, and SPI_SDATAS starts the transmission
(8)	The master reads data 0x01A from SPI_RXB in right-aligned form
(9)	After completing the transmission of the second byte, set the interrupt flag
(10)	The slave reads data 0x8D, while the master reads data 0x89. After unused bits are masked by software, the master receives 0x09, and the slave receives 0x0D
(11)	The master clears the STEn signal of the slave, and the STEn signal is high active (inactivated)

34.5.14 Digital audio transmission function

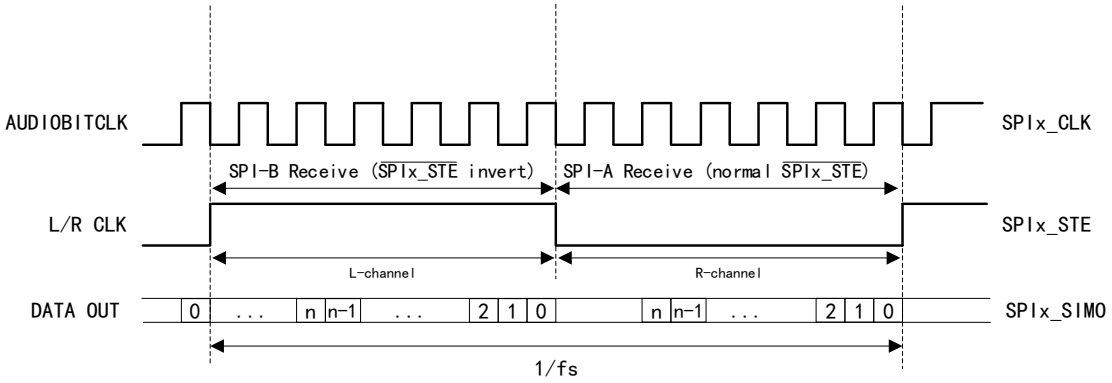
In devices with two SPI peripherals, to receive stereo digital audio data from the left and right channels in slave mode, enable the STEINV bit for one of the SPI peripherals. Right-channel data is stored when the low STEn signal is received, while left-channel data from the master is stored when the opposite high-level STEn signal is received. The SPI peripherals are connected as shown in the figure below to receive audio data from the digital audio interface:

Figure 182 Digital Audio Reception



Note: The STEINV bit is only used in SPI slave mode. When SPI is in master mode, writing to the STEINV bit has no effect on the STEn pin.

Figure 183 Digital Audio Interface Data - Standard Right-aligned Format



When configuring SPI clock polarity SPI_CFG[POLCFG]=0 and phase SPI_CTRL[PHASEL]=1 (i.e., latching data on the rising edge of the clock), the digital audio interface data in standard right-aligned format is as shown in the above figure.

When using dual SPI and the STEINV bit to receive digital audio, the number of supported digital audio interface formats is limited by the SPI timing specifications. For details on SPI timing specifications, please refer to the datasheet.

34.6 Register bank address

Table 177 SPI Register Bank Address

Device register	Register bank	Start address	End address
SpiRegs	SPI_REGS	0x4000_1400	0x4000_17FF

34.7 Register address mapping

Table 178 SPI_REGS Register Address Mapping

Register name	Register description	Offset address	WRPRT
SPI_CFG	Configuration register	0x00	-
SPI_CTRL	Control register	0x02	-
SPI_STS	Status register	0x04	-
SPI_BR	Baud rate register	0x08	-
SPI_EMUB	Simulate buffer register	0x0C	-
SPI_RXB	Receive buffer register	0x0E	-
SPI_TXB	Transmit buffer register	0x10	-
SPI_SDATAS	Serial data shift register	0x12	-
SPI_TXFIFO	Transmit FIFO register F	0x14	-

Register name	Register description	Offset address	WRPRT
SPI_RXFIFO	Receive FIFO register	0x16	-
SPI_FFCTRL	FIFO Control Register	0x18	-
SPI_PRICTRL	Priority control register	0x1E	-

34.8 Register functional description

34.8.1 Configuration register (SPI_CFG)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	CLENSEL	R/W	Character Length Select Select the number of bits in a shift sequence that a single character is shifted in or out of CLENSEL[0]. CLENSEL= Word length - 1 0000: 1-bit word 0001: 2-bit word … 1111: 16-bit word	0h
4	LBEN	R/W	Loopback Mode Enable Loopback mode is only valid in SPI master mode. After loopback mode is enabled, module verification can be performed during device testing. 0: Disable 1: Enable, where SIMO and SOMI are internal connections for module self-test	0h
5	HSEN	R/W	High Speed Mode Enable When enabling high-speed mode, correct GPIO shall be selected through the GPIO multiplexing register. 0: Disable 1: Enable	0h
6	POLCFG	R/W	Clock Polarity Configure This bit configures the valid edge of the CLK signal. This bit, along with PHASEL, configures the four clock schemes on the CLK pin. 0: SPI transmits data on the rising edge of CLK and receives data on the falling edge. The CLK signal line is at low level during SPI idle state. The PHASEL bit determines the data transmit/receive edge. PHASEL=0: Data is transmitted on the rising edge of CLK, and received data is latched on the falling edge of CLK. PHASEL=1: Data is transmitted on the first half cycle of the rising edge of CLK and the subsequent falling edge, while received data is latched on the rising edge of CLK. 1: SPI transmits data on the falling edge of CLK and receives data on the rising edge. The CLK signal line is	0h

Field	Name	R/W	Description	Reset value
			at high level when SPI is idle. The PHASEL bit determines the data transmit/receive edge. PHASEL=0: Data is transmitted on the falling edge of CLK, and received data is latched on the rising edge of CLK. PHASEL=1: Data is transmitted during the first half cycle of the falling edge of CLK and the subsequent rising edge, while received data is latched on the falling edge of CLK.	
7	SPIRDY	R/W	SPI Ready When SPIRDY=0, setting this bit will prevent the character written to the transmitter from shifting out, so a new character must be written to the serial data shift register. Once set, CLK will return to a non-active state for one cycle. 0: Initialize SPI flags as reset conditions, i.e., clear TXBFFLG, IFLG, and RXOVRFLG flags, immediately drive CLK in low level, regardless of clock polarity, and drive STE to a non-active state. When SPIRDY=0, the SPI configuration remains unchanged. 1: SPI is ready, and SPI has been ready to transmit/receive the next character Note: This bit shall be cleared before changing the SPI configuration, and set again before resuming SPI operation.	0h
15:8	Reserved			0h

34.8.2 Control register (SPI_CTRL)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	IEN	R/W	SPI Interrupt Enable Enable SPI transmit/receive interrupts. The interrupt flag (IFLG) bit is not affected by this bit. 0: Disable 1: Enable	0h
1	TXEN	R/W	Transmit Enable Disable data transmission (in master/slave mode) by placing the serial output data in a high-impedance state. When TXEN is disabled during transmission, the transmit shift register continues to operate until the previous character is shifted out. When this bit is disabled, SPI can still receive characters and update the corresponding flag bits. This bit is cleared by a system reset. 0: Disable transmission (in master mode, the SIMO pin will be in a high-impedance state if GPIO pins are not configured; in slave mode, the SOMI pin will be in a high-impedance state if GPIO pins are not configured.)	0h

Field	Name	R/W	Description	Reset value
			1: Enable transmission (for 4-wire mode, ensure the STEn input pin of the receiver is enabled).	
2	MSCFG	R/W	Master/Slave Mode Configure 0: Slave mode 1: Master mode	0h
3	PHASEL	R/W	Clock Phase Select This bit selects whether the CLK signal is delayed by half a cycle. This bit, along with POLCFG, configures the four clock schemes on the CLK pin. 0: No delay. At this point, data input/output depends on POLCFG 1: The CLK signal is delayed by half a cycle. At this point, the data input/output polarity depends on POLCFG. In any SPI mode, SPI (master/slave) provides the first bit of data after writing to the SPI_SDATAS register and before the first edge of the CLK signal.	0h
4	OVRIEN	R/W	Overrun Interrupt Enable This bit controls whether an overflow interrupt can be generated. When set, an interrupt is generated when the hardware sets the receive overflow flag (RXOVRFLG) bit. The receive overflow flag (RXOVRFLG) and interrupt flag (IFLG) share the same interrupt vector. 0: Disable 1: Enable	0h
15:5	Reserved			0h

34.8.3 Status register (SPI_STS)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	Reserved			0h
5	TXBFFLG	R	Transmit Buffer Full Flag This bit is set when a character is written to the SPI transmit buffer. It is automatically cleared after the previous character shifts out of the SPI transmit buffer and is automatically loaded into the SPI_SDATAS register. 0: The transmit data buffer has not been full 1: The transmit buffer has been full	0h
6	IFLG	RC_R	Interrupt Flag This bit is set when SPI has completed transmitting/receiving the last bit and is ready for the next operation. At the same time, the received character is placed into the receive buffer. If IEN=1, setting this bit will generate an interrupt request. This bit can be cleared by one of three methods: system	0h

Field	Name	R/W	Description	Reset value
			reset, writing 0 to the SPIRDY bit, or reading the receive buffer register. 0: A complete word has not been received or transmitted 1: SPI has completed transmitting/receiving the last bit and is ready for the next operation Note: When FIFO mode is enabled, this bit is disabled. The process of copying received words from the SPI receive buffer to the receive FIFO will clear this bit. Similar function is served in FIFO mode through FIFO status or FIFO interrupt bits.	
7	RXOVRFLG	RC_W1	Receive Overrun Flag This bit is set when a receive/transmit operation is completed before the previous character is read from the buffer. This bit can be cleared by one of three methods: system reset, writing 1 to this bit, or writing 0 to the SPIRDY bit. If OVRIEN=1, SPI will generate only one interrupt request when this bit is set for the first time. When RXOVRFLG has already been set, subsequent overflows will not generate additional interrupt requests. Therefore, after each overflow, this bit must be cleared by writing 1 to it to allow new overflow interrupt requests. Otherwise, if this bit is not cleared in the interrupt service routine, a new overflow interrupt request will not be immediately triggered upon exiting the routine. Since this bit shares the same interrupt vector with the IFLG bit, it must be cleared during the interrupt service routine to eliminate the possibility of interrupt source when the next byte is received. 0: No receive overflow occurs 1: The last received character is overwritten and lost (the SPI receive buffer is rewritten by the SPI module before the user application reads the previous character).	0h
15:8	Reserved			0h

34.8.4 Baud rate register (SPI_BR)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
6:0	BRSEL	R/W	Baud Rate Select When SPI is operating as a slave, it receives the clock from the master via the CLK pin. At this point, the CLK signal is not affected by the BRSEL bit. Note: The clock frequency received from the master must not exceed SPI slave $f_{LPCLK}/4$. When SPI is operating as a master, this bit field selects the bit transmission rate for data transfer. There are a total of 125 data transfer rates available, each as a	0h

Field	Name	R/W	Description	Reset value
			function of the CPU clock LPCLK. The CLK clock is output on the baud rate clock of the CLK pin, shifting one data bit per CLK cycle. In master mode, SPI generates the clock and outputs it via the CLK pin. For details on the calculation formula for transmission baud rate, refer to the Transmission Baud Rate section. 000000: SPI baud rate = LPCLK/4 000001: SPI baud rate = LPCLK/4 000010: SPI baud rate = LPCLK/4 Others: SPI baud rate = LPCLK/(SPI_BR + 1)	
15:7	Reserved			0h

34.8.5 Simulation Buffer Register (SPI_EMUB)

Offset address: 0x0C

Reset type: SYSRSn

It is recommended to view this register in normal emulator operation mode.

Field	Name	R/W	Description	Reset value
15:0	EMUB	R	Emulation Buffer When the SPI_SDATAS register receives a complete character, it will be shifted into the SPI_RXB and SPI_EMUB registers; therefore, the received character can be read from these registers, and an interrupt flag (IFLG) is set during the shift. The SPI_EMUB register functions almost identically to the SPI_RXB register, except that reading the contents of the SPI_EMUB register does not clear the interrupt flag (IFLG). In essence, SPI_EMUB is a virtual address that allows the emulator to read the contents of the SPI_RXB register without clearing the interrupt flag (IFLG). This register is designed to support emulation. Reading the SPI_RXB register clears the interrupt flag (IFLG). During normal emulation, the control registers are read to continuously update the contents on the display. By reading this register, the simulator can correctly update the content on the display. In other words, reading this register allows the simulator to more accurately simulate the real environment of the SPI.	0h

34.8.6 Receive Buffer Register (SPI_RXB)

Offset address: 0x0E

Reset type: SYSRSn

When the SPI_SDATAS register receives a complete character, it will be shifted into this register, and an interrupt flag (IFLG) will be set during the shift. Reading the SPI_RXB register will also clear the interrupt flag (IFLG).

When FIFO mode is enabled, reading this register also decrements SPI_RXFIFO[RXFFSTS].

Field	Name	R/W	Description	Reset value
15:0	RXB	R	Received Buffer The received data is first shifted into the most significant bit of the SPI and stored in this register in a right-aligned format.	0h

34.8.7 Transmit Buffer Register (SPI_TXB)

Offset address: 0x10

Reset type: SYSRSn

The next character to be transmitted is written to this register, and the TXBFFLG flag is set simultaneously. If the transmission of the current character is completed, the data in this register will be automatically loaded into the SPI_SDATAS register, and the TXBFFLG flag will be cleared. If no transmission is currently in active state, the data written to SPI_TXB will be transferred to the SPI_SDATAS register without setting the TXBFFLG flag.

In master mode, if no transmission is currently in active state, writing to this register initiates a transmission in the same way as writing to the SPI_SDATAS register.

Field	Name	R/W	Description	Reset value
15:0	TXB	R/W	Transmit Buffer Data is written to this register in a left-aligned format.	0h

34.8.8 Serial Data Shift Register (SPI_SDATAS)

Offset address: 0x12

Reset type: SYSRSn

After data is written to this register, it is shifted out from the most significant bit (MSB) during subsequent CLK cycles. For each bit shifted out (MSB), one bit is shifted into the register from the least significant bit (LSB).

Field	Name	R/W	Description	Reset value
15:0	SDATAS	R/W	Serial Data Shift When TXEN=1, the data output on the serial output pin will be provided by this bit; When the SPI is in master mode, data transmission will be initiated. For information on the phase and polarity of the transmission clock, refer to the POLCFG and PHASEL bits. In master mode, writing virtual data to this register will initiate the receiver sequence. Since data is not hardware-aligned, when the character length of the transmission is less than 16 bits, the transmitted data must be written in a left-aligned format and read in a right-aligned format.	0h

34.8.9 Transmit FIFO Register (SPI_TXFIFO)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	TXFFILSEL	R/W	Transmit FIFO Interrupt Level Select When this bit matches the TXFFSTS bit (i.e., TXFFSTS ≤ TXFFILSEL), a transmit FIFO interrupt request will be generated. 00000: No word remains in the transmit buffer 00001: No word or only 1 word remains in the transmit buffer 00010: No word or fewer than 2 words remain in the transmit buffer … 10000: No word or fewer than 16 words remain in the transmit buffer Others: No word or fewer than 16 words remain in the transmit buffer	0h
5	TXFFIEN	R/W	Transmit FIFO Interrupt Enable Enable transmit FIFO interrupts based on TXFFILSEL matching 0: Disable 1: Enable	0h
6	TXFFICLR	W	Transmit FIFO Interrupt Flag Clear Clear the TXFFIFLG flag. 0: No effect, reads as 0 1: Clear the TXFFIFLG flag	0h
7	TXFFIFLG	R	Transmit FIFO Interrupt Flag 0: No transmit FIFO interrupt occurs 1: A transmit FIFO interrupt occurs	0h
12:8	TXFFSTS	R	Transmit FIFO Status 00000: Transmit buffer is empty 00001: Transmit buffer contains 1 word 00010: Transmit buffer contains 2 words … 10000: Transmit buffer contains 16 words (maximum value) Others: Transmit buffer contains 16 words (maximum value)	0h
13	TXFFEN	R/W	Transmit FIFO Enable 0: Reset the FIFO pointer to 0 and keep it in the reset state 1: Release the transmit FIFO from the reset state	1h
14	SPIFFEEN	R/W	SPI FIFO Enhancements Enable 0: Disable 1: Enable	0h
15	SPIEN	R/W	SPI Enable 0: Reset the SPI transmit and receive channels and keep the SPI FIFO register configuration unchanged 1: Restore the transmit and receive function of SPI FIFO without affecting the SPI registers.	1h

34.8.10 Receive FIFO Register (SPI_RXFIFO)

Offset address: 0x16

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	RXFFILSEL	R/W	Receive FIFO Interrupt Level Select When this bit matches the RXFFSTS bit (i.e., RXFFSTS≥RXFFILSEL), a receive FIFO interrupt request is generated. 00000: The receive buffer contains 0 or more words 00001: The receive buffer contains 1 or more words 00010: The receive buffer contains 2 or more words … 10000: The receive buffer contains 16 words Others: The receive buffer contains 16 words	1Fh
5	RXFFIEN	R/W	Receive FIFO Interrupt Enable Enable receive FIFO interrupts based on RXFFILSEL matching 0: Disable 1: Enable	0h
6	RXFFICLR	W	Receive FIFO Interrupt Flag Clear Clear the RXFFIFLG flag. 0: No effect, reads as 0 1: Clear the RXFFIFLG flag	0h
7	RXFFIFLG	R	Receive FIFO Interrupt Flag 0: No receive FIFO interrupt occurs 1: A receive FIFO interrupt occurs	0h
12:8	RXFFSTS	R	Receive FIFO Status 00000: Receive buffer is empty 00001: Receive buffer contains 1 word 00010: Receive buffer contains 2 words … 10000: Receive buffer contains 16 words (maximum value) Others: Receive buffer contains 16 words (maximum value)	0h
13	RXFFEN	R/W	Receive FIFO Enable 0: Reset the FIFO pointer to 0 and keep it in the reset state 1: Release the receive FIFO from the reset state and re-enable the receive FIFO	1h
14	RXFFOVRCLR	W	Receive FIFO Overflow Flag Clear Clear the RXFFOVRFLG flag. 0: No effect, reads as 0 1: Clear the RXFFOVRFLG flag	0h
15	RXFFOVRFLG	R	Receive FIFO Overflow Flag 0: No receive FIFO overflow occurs	0h

Field	Name	R/W	Description	Reset value
			1: A receive FIFO overflow occurs, i.e., the number of words received in the FIFO exceeded 16, and the first received word was lost	

34.8.11 FIFO control register (SPI_FFCTRL)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	FFTXDLYSEL	R/W	FIFO Transmit Delay Select This field defines the delay between transfers from the FIFO transmit buffer to the transmit shift register. The delay is defined using SPI serial clock cycles, with the range of this 8-bit register being 0 to 255 serial clock cycles. In FIFO mode, to propagate the delay between transmissions into the data stream, the transmit buffer (SPI_TXB) between the field shift register and the FIFO shall be filled only after the last bit of the shift register has completed shifting out. In FIFO mode, SPI_TXB cannot be used as an additional buffer level. 0000 0000: After the transmission of the previous word is completed, the next word from the FIFO transmit buffer is immediately transmitted to SPI_TXB 0000 0001: After the transmission of the previous word is completed, the next word from the FIFO transmit buffer is transmitted to SPI_TXB after 1 serial clock cycle 0000 0010: After the transmission of the previous word is completed, the next word from the FIFO transmit buffer is transmitted to SPI_TXB after 2 serial clock cycles ….. 1111 1111: After the transmission of the previous word is completed, the next word from the FIFO transmit buffer is transmitted to SPI_TXB after 255 serial clock cycles	0h
15:8	Reserved			0h

34.8.12 Priority control (SPI_PRICTRL)

Offset address: 0x1E

Reset value: 0x0000

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	SPIMSEL	R/W	SPI Mode Select In 3-wire mode, unused pins will function as GPIO pins. In master mode, the SIMO pin serves as SPIMOMI (master transmit and master receive) while	0h

Field	Name	R/W	Description	Reset value
			SOMI is available for use by other modules. In slave mode, the SOMI pin serves as SPISOSI (slave transmit and master receive) while SIMO is available for use by other modules. 0: 4-wire mode (normal mode) 1: 3-wire mode	
1	STEINV	R/W	STEn Inversion This bit is only used in SPI slave mode, and is not valid when written to SPI master mode. In devices with two SPI peripherals, to receive stereo digital audio data from the left and right channels, the STE signal of one module needs to be inverted. 0: STEn signal is active low (normal) 1: STE signal is active high (inverted)	0h
3:2	Reserved			0h
4	EMUFREEEN	R/W	Emulation Free Run Enable This bit configures what happens when simulation is suspended (e.g., when the debugger hits a breakpoint). In free-running mode, peripherals can continue executing the current operation; In stop mode, the operation stops immediately upon suspension or after completing the current operation (current transmit/receive sequence). 0: Simulation mode is selected by EMUSOFT 1: Free-running mode, where peripherals continue executing the current operation	0h
5	EMUSOFT	R/W	Emulation Soft Run This bit is valid only when EMUFREEEN=0. The start time of transmission is determined by the baud rate in use. In standard SPI mode, 0: If TSUSPEND is asserted, the transmission stops midway through the bit stream. Once TSUSPEND is deasserted and a system reset is not performed, the remaining bits in the data buffer to be transmitted will be shifted out. For example: If 3 out of 8 bits in SPI_SDATAS have already been shifted, communication will pause at the third bit. If TSUSPEND is later deasserted and no SPI reset is performed, SPI will resume transmission from where it stopped (i.e., the fourth bit) and continue transmitting the remaining 8 bits from that point. 1: If simulation suspension occurs before transmission starts (i.e., before the first CLK pulse), no transmission occurs. If simulation suspension occurs after transmission starts (i.e., after the first CLK pulse), the data will be completely shifted out of SPI_SDATAS. The start time of transmission is determined by the baud rate in use. Transmission stops after shifting between the shift register and buffer is completed. In standard SPI mode, transmission stops when both SPI_TXB and SPI_SDATAS are empty. In FIFO mode,	0h

Field	Name	R/W	Description	Reset value
			transmission stops when both transmit FIFO and SPI_SDATAS are empty.	
15:6	Reserved			0h

35 Universal asynchronous receiver/transmitter (UART)

35.1 Full Name and Abbreviation Description of Terms

Table 179 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
first-in first-out	FIFO

35.2 Introduction

USART is a serial communication device that can flexibly exchange full-duplex and half-duplex data with external devices, and meets the requirements of external devices for industry standard NRZ asynchronous serial data format. UART also provides a wide range of baud rate and supports multiprocessor communication.

35.3 Main characteristics

(1) Full-duplex asynchronous communication

(2) Single-line half-duplex communication

(3) NRZ standard format

(4) Characteristics of programmable serial port:
- Data bits: 1 to 8 bits
- Parity bit: Even parity check, odd parity check
- Support 1 and 2 stop bits

(5) Can distinguish data and address bits

(6) Double-buffering receiving and transmitting

(7) Independent transmitter and receiver enable bit

(8) Independent interrupt enable bit

(9) The transmitter and receiver operations can be completed through interrupt or roll polling algorithm-based statuses

(10) 16-level transmit/receive FIFO

(11) Programmable baud rate generator

(12) Automatic baud rate detection

(13) Multiprocessor communication:

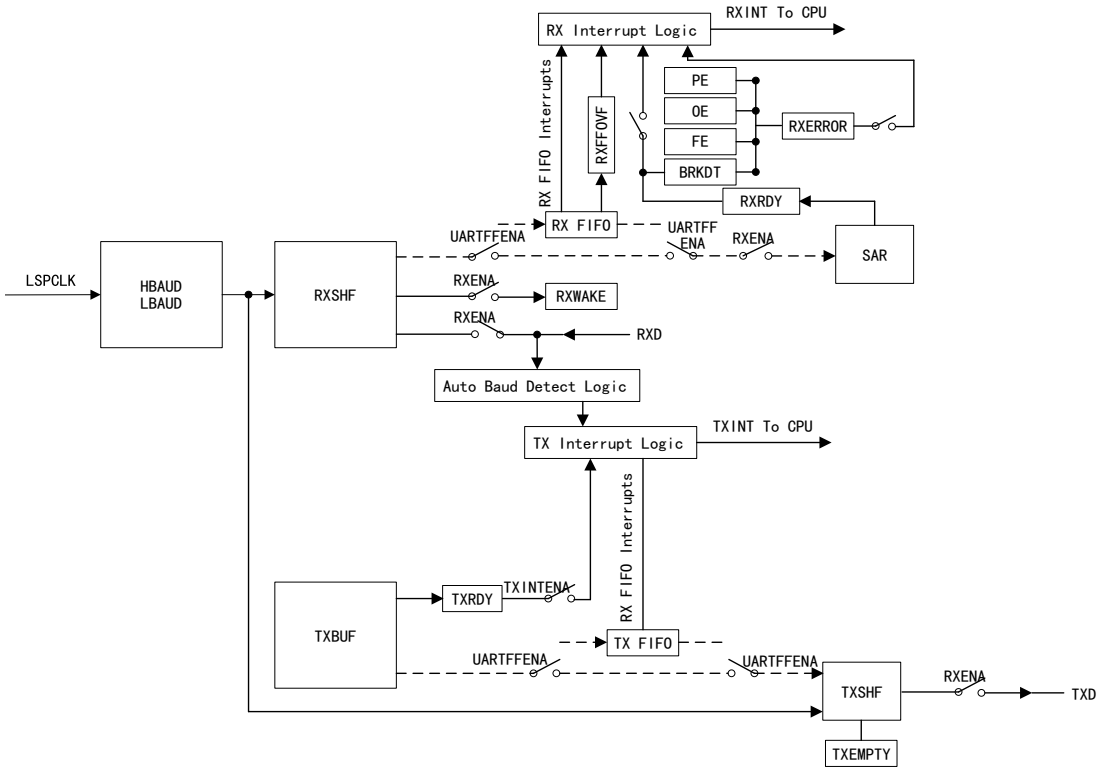
- Idle line mode
- Address bit mode

(14) Status flag bit:

- Transmission detection flag: The transmit register is empty, while the receive register is not empty
- Error detection flag: Overrun error, interrupt detection, parity error, frame error

35.4 Structure block diagram

Figure 184 Structure Block Diagram



35.5 Functional description

Table 180 UART Signal Description

Signal	Type	Description
Baud rate clock	Control signal	LSPCLK prescaler clock
TXD	External signal	Asynchronous serial port transmit data
RXD		Asynchronous serial port receive data
TXINT	Interrupt signal	Transmit interrupt
RXINT		Receive interrupt

35.5.1 Configure pins

Table 181 UART Pin-related Functions

Operation	Function
Configure GPIO multiplexing registers	Connect the peripheral to the device pins
Set the appropriate GPIO input type bits to 11	Set the GPIO input qualification to asynchronous mode
Configure GPIO high multiplex n bits first, then configure GPIO low multiplex n bits	Avoid burrs on the pins

Internal pull-up resistors can be configured in the GPIO pull-up/down n bits. Some IO functions are defined by GPIO register settings independent of the peripheral.

35.5.2 Frame format

The frame format of data frame is controlled by UARTCCR register

- The UARTCHAR bit controls the character length, which can be set from 1 to 8 bits
- The PARITYENA bit controls whether the check bit is enabled
- The PARITY bit controls the check bit to make it odd or even
- The STOPBITS bit controls the stop bit length, which can be set to 1 or 2 bits

Check bit

The PARITY bit in UARTCCR determines the parity bit. When PARITY=0, it is odd parity; otherwise, it is even parity.

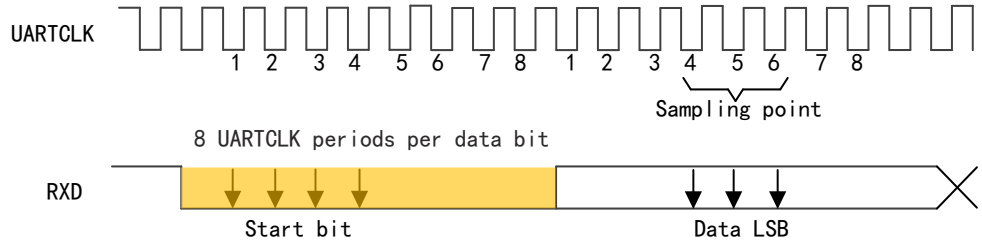
- Even check: When the number of frame data and check bit '1' is even, the even check bit is 0; otherwise it is 1.
- Odd check: When the number of frame data and check bit '1' is even, the odd check bit is 1; otherwise it is 0.

35.5.3 Communication format

Using single-wire or dual-wire communication, each data bit has 8 UARTCLK cycles.

A valid start bit is identified after four consecutive low levels on the internal UARTCLK cycles, at which point the receiver begins operation. If no zero bit is detected, the processor resumes searching for a start bit. The processor determines the bits following the start bit by sampling during the fourth, fifth, and sixth UARTCLK cycles of the bit, with the value based on the majority occurrence. Since the receiver automatically synchronizes with the frame, external transmitters and receivers do not need to use a synchronized serial clock, allowing the clock to be locally generated.

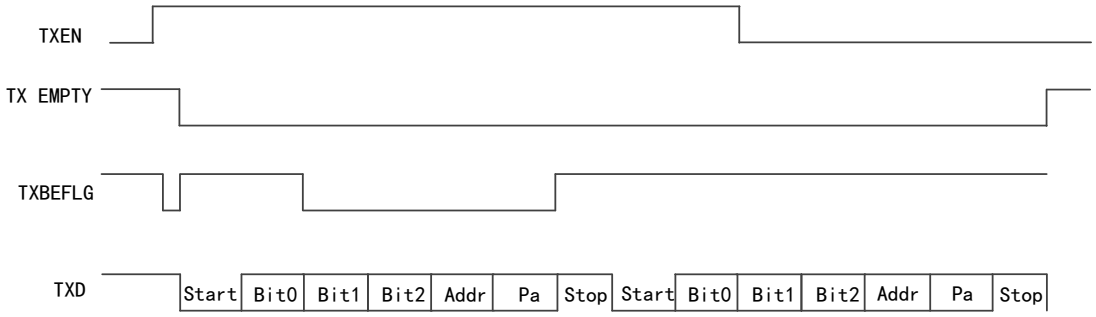
Figure 185 Asynchronous Communication Format



Transmitter signal

In address bit wake-up mode, setting TXENA high enables the transmitter, while setting it low disables the transmitter, allowing the current character to be transmitted. After the first character transmission is completed, the transmission of the second character to the TXSHF shift register begins. UART transfers data to TXSHF. When TXRDY becomes high, it indicates that the transmitter is ready to transmit the second character and requests an interrupt. After TXRDY becomes high, the program writes the second character to UARTTXBUF. If each character is in three bits, the transmitter signal timing diagram is as follows:

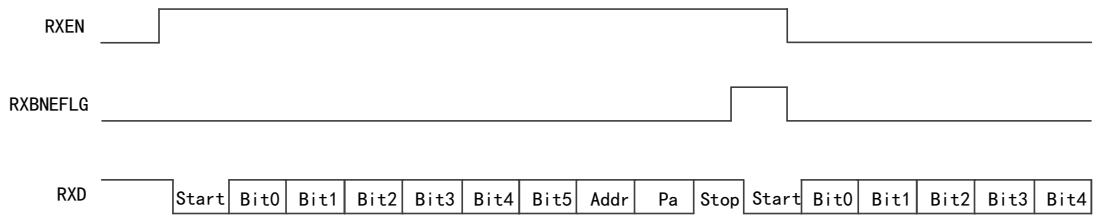
Figure 186 Transmitter Signal Timing



Receiver signal

In address bit wake-up mode, setting RXENA high enables the receiver, while setting it low disables the receiver. Data continues to assemble in RXSHF but will not be transmitted to the receive buffer register. A start bit can be detected when data arrives at the RXD pin. Data moves from RXSHF to UARTRXBUF, requiring an interrupt request. The RXRDY flag bit becomes high, indicating that a new character has been received and is automatically cleared when UARTRXBUF is read. If each character is in six bits, the receiver signal timing diagram is as follows:

Figure 187 Receiver Signal Timing



35.5.4 Baud rate

The low-speed peripheral clock, and the UARTHBAUD and UARTLBAUD registers determine the internally generated serial clock. For a given LSPCLK, UART can select one of 64K different serial clock rates using the 16-bit value in the UARTHBAUD and UARTLBAUD registers. Refer to the formula for calculating asynchronous baud rate. LSPCLK/16 is the maximum baud rate. If LSPCLK is 100 MHz, the maximum baud rate is 6.25 Mbps.

Table 182 Asynchronous Baud Rate Register Values

LSPCLK clock frequency division	100 MHz actual baud rate	Error rate
324	38400	0.16
650	19200	0.01
1301	9601	0.01
2603	4800	0
5207	2400	0

35.5.5 Automatic baud rate

General UARTs typically do not have built-in automatic baud rate detection. These UART modules integrate an embedded controller clock frequency that depends on the PLL reset value. The clock frequency of general embedded controllers often varies.

35.5.6 Automatic baud rate detection sequence

Setting the UARTRST bit enables automatic baud rate detection. The CDC and ABD bits in UARTFFCT control the automatic baud rate.

If CDC=1, ABD is set, which indicates automatic baud rate alignment; at this point, UART will trigger a FIFO interrupt. After the interrupt service, the CDC bit must be cleared via software. If CDC remains set to 1 after the interrupt service, no further interrupts will occur.

At high baud rates, both the transmitter and connector may affect the voltage conversion rate of incoming data bits. If the baud rate exceeds 100k, the reliability of automatic baud rate detection may be impacted and the automatic baud rate lock function may be caused to fail. A lower baud rate can be used

between the master and the UART bootloader to achieve baud rate locking, and the master can then set the baud rate register as a higher desired baud rate to avoid the above issues.

35.5.7 Multi-processor protocol

The multi-processor protocol allows efficient data transmission between multiple processors. It is divided into idle line mode and address bit mode.

35.5.8 Multiprocessor communication

Multi-processor communication enables one processor to efficiently transmit data blocks to other processors on the same serial line. At any given time, only one data block can be transmitted on the serial line.

Sleep mode

If a processor is to be interrupted only when an address byte is detected, all processors on the serial line will set SLEEP. When a processor reads an address that matches the CPU device address set in the software, the program must clear the SLEEP bit to enable the UART to generate an interrupt for each received data byte. Even if SLEEP=1, the receiver can still operate. At this point, the receiver will only set RXRDY, RXINT, or the receiver error status bit to 1 when it detects that an address byte and the address bit in the received frame is 1. SLEEP is not set by the UART itself and must be configured by software.

Address byte

All monitors will read the address byte, which is the first byte in the transmitted information block. Monitors with the correct address will be interrupted by subsequent data bytes. Monitors with an incorrect address will not be interrupted until the next address byte.

Identify address byte

- Idle line mode is used in typical non-multi-processor UART communication. This mode includes a silent period before the address byte. Without additional address/data bits, it can process data blocks with more than ten bytes more quickly.
- Address bit mode has an extra address bit added to each byte to distinguish between address and data. Since it does not require waiting between data blocks, this mode is more efficient for handling small data blocks. However, at high transmission speeds, the program may not be fast enough to avoid the 10-bit idle time in the transmission stream.

Transmitting and receiving

The ADDRIDLE_MODE bit configures multi-processor mode. Both modes use

the TXWAKE, RXWAKE, and SLEEP bits to control the transmitter and receiver functions in these modes.

Receive sequence

When an address block is received, the UART port wakes up and requests an interrupt. It reads the first frame in the data block containing the destination address. The software program is triggered by the interrupt and checks the incoming address. The address byte is compared with the device address byte stored in memory. If the data block is intended for the device CPU, the CPU will clear the SLEEP bit and read the rest of the data block. Otherwise, the software routine exits, the SLEEP bit remains set, and no receive interrupt occurs until the next data block starts.

35.5.9 Idle line mode

In this mode, the idle time between two data blocks is longer than the idle time between two frames within a data block. An idle time of ten or more high-level bits after a frame indicates the start of a new data block. The time to transmit one bit can be calculated based on the baud rate.

Step

(1) Upon receiving the "block start" signal, the UART wakes up and requests an interrupt.

(2) The processor recognizes the next UART interrupt.

(3) The interrupt service routine will compare the received address with its own address.

(4) If the CPU is addressed, the service routine will clear the SLEEP bit and receive the rest of the data block. Otherwise, the SLEEP bit remains unchanged. At this point, the CPU can continue executing its main program and will not be interrupted by the UART port until the start of the next data block is detected.

In idle line mode, if the UART needs longer than 10 bits to read all RXDATA from the FIFO, it may miss the immediate block to be detected. RXWAKE is asserted upon recognizing a 10-bit IDLE period. If UARTRXBUF is read, RXWAKE will not be asserted again, even if the line remains idle afterward.

When the CPU requires UART clock cycles with more than 10 bits to read data from UARTRXBUF/FIFO, use the following methods:

- If RXWAKE is set before reading the UARTRXBUF register, do not set the SLEEP bit at the end of the ISR.
- Set UARTCTL1. Perform SWRESET after reading all RX data at the end of the ISR.

Block start signal

The methods for transmitting a block start signal are as follows:

- Delay the idle time of ten or more bits between the transmission of the last frame of data in the previous block and the transmission of the address frame of the new block.
- Set the TXWAKE bit before writing to the UARTTXBUF register. 11 bits of idle time can be transmitted. At this time, the idle period on the serial communication line will not exceed the necessary duration.

Temporary wake flag

The temporary wake flag is an internal flag, double-buffered with the TXWAKE bit. When loading TXSHF from UARTTXBUF, load WUT from TXWAKE and clear the TXWAKE bit.

To transmit a block start signal, first set the TXWAKE bit, then write it to the UARTTXBUF register. When TXSHF becomes idle again, shift the data from UARTTXBUF into TXSHF and the value of TXWAKE into WUT, and then clear TXWAKE. Setting TXWAKE will replace the start bit, data bits, and parity bit with an 11-bit idle time, and transmit after the stop bit of the previous frame. Write a new address value to UARTTXBUF. A data word must be written to the UARTTXBUF register for the value of TXWAKE to be shifted into WUT. Since TXSHF and WUT are double-buffered, UARTTXBUF and TXWAKE can be written again after the data word is shifted into the TXSHF register.

Receiver operation

The receiver is not affected by the SLEEP bit. Until an address frame is detected, the receiver neither sets RXRDY and error status bits, nor does it request a receive interrupt.

35.5.10 Address bit mode

Set the ADDRIDLE_MODE bit to select address bit mode, where an additional bit called the address bit follows the last data bit of the frame. The address bit is set to 1 in the first frame of a block and set to 0 in all other frames. The idle period duration will not be affected. The address bit format is typically used for data frames with 11 or fewer bytes, able to add one bit to all transmitted data bytes. The idle format is typically used for data frames with 12 bytes or more.

Transmit address

The value of TXWAKE is included in the address bit. During transmission, when the UARTTXBUF register and TXWAKE are loaded into the TXSHF register and WUT, respectively, TXWAKE is reset, and WUT becomes the value of the address bit for the current frame.

To transmit an address, first set the TXWAKE bit, and write the address value to the UARTTXBUF register. When this address value is transferred to the TXSHF register and shifted out, its address bit is transmitted as 1. This indicates other processors on the serial line to read the address. After loading TXSHF and WUT, UARTTXBUF and TXWAKE can be written to immediately. Subsequently, set the TXWAKE bit to transmit non-address frames within the block.

35.5.11 Interrupt

The receiver and transmitter can be controlled through interrupts. When the interrupt enable bit is not set, interrupts will not be asserted but can reflect the transmit and receive status. UARTCTL2 [TXRDY] indicates the interrupt condition status. The BRKDT, RXRDY, and RXERROR bits in UARTRXST are all interrupt flag bits. The receiver and transmitter have independent peripheral interrupt vectors. Peripheral interrupt requests are represented by priority bits output from the peripheral to the NVIC controller, which can be either high or low priority. When interrupt requests with the same priority occur to TX and RX simultaneously, the receiver is given higher priority than the transmitter to reduce the risk of receiver overflow.

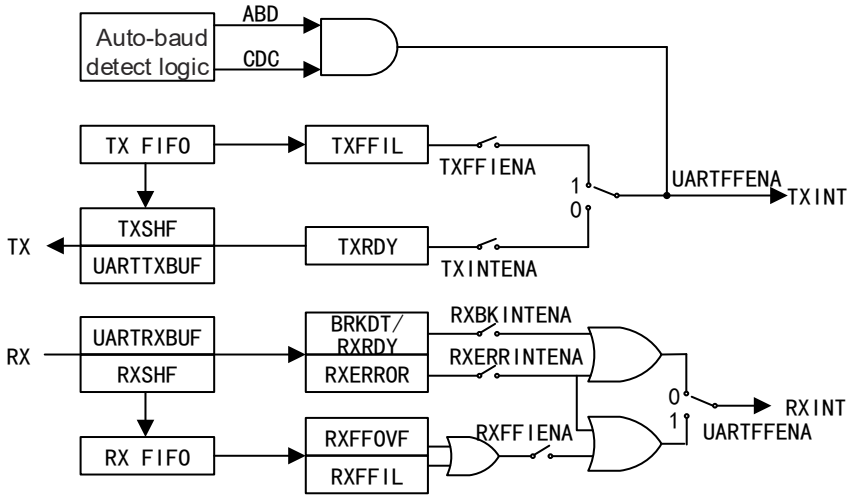
If the TXINTENA bit is set, a transmitter peripheral interrupt request is asserted when data in UARTTXBUF is transferred to the TXSHF register. At this point, the CPU can write it to UARTTXBUF, which will set the TXRDY bit and generate an interrupt.

If the RXBKINTENA bit is set, the RXRDY bit is set and an interrupt is generated when a complete frame is received, and data is transferred from the RXSHF register to the UARTRXBUF register. When an interrupt detection condition occurs, the BRKDT bit is set and an interrupt is generated.

Table 183 UART Interrupt Request

Interrupt event	UARTFFENA bit	Event flag bit	Enable bit
Transmit empty	0	TXRDY	TXINTENA
Receive error		RXERROR	RXERRINTENA
Receive break		BRKDT	RXBKINTENA
Receive data		RXRDY	
Transmit empty	1	TXFFIL	TXFFIENA
Receive error or receive break		RXERROR	RXERRINTENA
Receive FIFO		RXFFIL	RXFFIENA
Automatic baud rate detection	-	ABD	-

Figure 188 Interrupt Mapping



35.5.12 FIFO

Table 184 Functions and Descriptions

Function	Description
Reset	During reset, start in standard mode with FIFO functionality disabled.
Standard mode	In standard mode, TXINT/RXINT interrupts are generally used as the module's interrupt sources.
Enable FIFO	Set UARTFFTX [UARTFFENA] to enable FIFO mode. The UARTRST bit resets the FIFO mode.
Activate	Activate all registers.
Interrupt	In FIFO mode, interrupts are used separately for transmit FIFO and receive FIFO. In standard mode, TXINT is disabled as the transmit FIFO interrupt. RXINT is a universal interrupt for receive errors, receive FIFO, and receive FIFO overflow conditions.
Programmable interrupt levels	Both transmit and receive FIFOs can generate CPU interrupts. An interrupt is triggered when the TXFFST field matches the TXFFIL field. The default trigger level for the receive FIFO is 0x11111, and for the transmit FIFO, it is 0x00000. These levels are programmable.
Buffer	The transmit and receive buffers have two 16-level FIFOs. The widths of the transmit FIFO register and receive FIFO register are 8 bits and 10 bits, respectively. In standard UART, the UARTTXBUF serves as a transition buffer before the transmit FIFO and the shift register. When FIFO is enabled: Since UARTTXBUF is not an additional buffer, data is loaded into the FIFO only after the last bit of the shift register is shifted out. TXSHF loads directly from the FIFO after an optional delay. Characters written to UARTTXBUF are queued into TXFIFO, and characters received from RXFIFO can be read via UARTRXBUF.

Function	Description
Transmission delay	The word transmission rate from the FIFO is programmable. FFTXDLY can define a delay of 0-256 baud rate clock cycles between word transmissions. With zero delay, data can be transmitted in continuous mode, with words in the FIFO output consecutively. With a delay of 256 clock cycles, data is transmitted in maximum delay mode, with words in the FIFO separated by 256 baud rate clock cycles. Programmable delay can communicate with slow UART, reducing CPU intervention.
FIFO status bits	The transmit and receive FIFOs each have their own status bits, which indicate the number of words available in the FIFO at any time. If the TXFIFORESET and RXFIFORESET bits are set, the FIFO will restart from the beginning. When cleared, the FIFO pointers will be reset to zero.

35.5.13 DMA Scheme

As the R502's DMA lacks handshaking signals such as 'request' and 'ack', it relies solely on a unidirectional trigger signal sent from the peripheral to the DMA. The UART's DMA trigger signal can be derived from the logic used to generate UART interrupts.

FIFO Mode

In FIFO mode, the TX DMA request signal is determined by TXFFIL (Transmit FIFO interrupt level) and TXFFST (TX FIFO status); when the data stored in the TX FIFO (TXFFST) is less than TXFFIL, a TX DMA request signal is output.

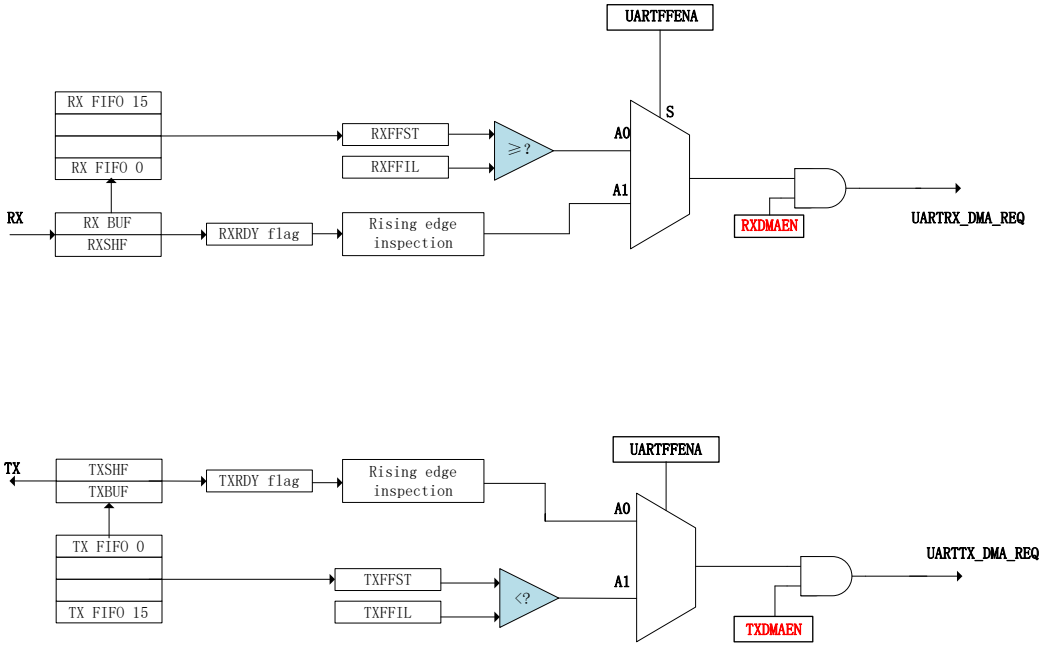
In FIFO mode, the RX DMA request signal is determined by RXFFIL (Transmit FIFO interrupt level) and RXFFST (RX FIFO status). When the data stored in the RX FIFO (TXFFST) is greater than or equal to RXFFIL, an RX DMA request pulse is output.

Non-FIFO mode

In non-FIFO mode, the TX DMA request signal is determined by TXRDY (Transmitter buffer register ready flag); when the data in TX BUF has been transferred to TX SHF, a TX DMA request pulse is output.

In non-FIFO mode, the RX DMA request signal is determined by RXRDY (UART receiver-ready flag); an RX DMA request pulse is output when the data in the RX SHF has been transferred to the RX BUF.

Figure189 Schematic diagram of the UART DMA request signal



35.6 Register bank address

Table 185 Register Bank Address

Device register	Register bank	Start address	End address
UartaRegs	UART_REGS	0x4000_0800	0x4000_0BFF
UartbRegs	UART_REGS	0x4000_0C00	0x4000_0FFF
UartcRegs	UART_REGS	0x4000_1000	0x4000_13FF

35.7 Register address mapping

Table 186 Register Address Mapping

Register name	Register description	Offset address	WRPRT
UARTCCR	Communication control register	0x00	-
UARTCTL1	Control register	0x02	-
UARTHBAUD	Baud rate high register	0x04	-
UARTLBAUD	Baud rate low register	0x06	-
UARTCTL2	Control register 2	0x08	-
UARTRXST	Receive status register	0x0A	-
UARTRXEMU	Simulator/Emulator register	0x0C	-
UARTRXBUF	Receive buffer register	0x0E	-

Register name	Register description	Offset address	WRPRT
UARTTXBUF	Transmit buffer register	0x12	-
UARTFFTX	Transmit FIFO register	0x14	-
UARTFFRX	Receive FIFO register	0x16	-
UARTFFCT	FIFO/Automatic baud rate control register	0x18	-
UARTPRI	Select mode register	0x1E	-

35.8 Register functional description

35.8.1 Communication Control Register (UARTCCR)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	UARTCHAR	R/W	Character Length Configure Configures character lengths of 1 to 8 bits. Characters with fewer than 8 bits are right-aligned in UARTRXBUF and UARTRXEMU. UARTRXBUF is padded with leading zeros, while UARTRXEMU is not. 000: 1 bit 001: 2 bits … 111: 8 bits	0h
3	ADDRIDLE_MODE	R/W	Multiprocessor Mode Select Multiprocessor communication differs from other communication modes as it requires use of SLEEP and TXWAKE. Address bit mode has an extra bit added to the frame, while idle line mode does not have this bit added and is compatible with RS-232 type communication. Idle line mode is typically used for normal communication. 0: Idle line mode protocol 1: Address bit mode protocol	0h
4	LOOPBKENA	R/W	Loop Back Test Mode Enable Enable loopback test mode, where the Tx pin is internally connected to the Rx pin. 0: Disable 1: Enable	0h
5	PARITYENA	R/W	Parity Enable If ADDRIDLE_MODE=1, the address bit is included in parity calculations. For characters with fewer than 8 bits, the remaining unused bits shall be masked during parity calculations. 0: Disable 1: Enable	0h

Field	Name	R/W	Description	Reset value
6	PARITY	R/W	Odd/Even Parity Select When PARITYENA=1, this bit indicates odd or even parity. 0: Odd parity check 1: Even parity check	0h
7	STOPBITS	R/W	STOP Bit Configure Specify the number of stop bits to be transmitted. The receiver only checks one stop bit. 0: 1 1: 2	0h
15:8	Reserved			0h

35.8.2 Control Register (UARTCTL1)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	RXENA	R/W	Receive Enable On the RXD pin, data is received and then transmitted to the receiver buffer through the receiver shift register. 0: Prevent received characters from being transferred to UARTRXBUF, and no receiver interrupt will be generated for UARTRXEMU. However, the receiver shift register can continue assembling characters. 1: Transmit received characters to UARTRXBUF and UARTRXEMU.	0h
1	TXENA	R/W	Transmit Enable When this bit is disabled, data written to UARTTXBUF will not be transmitted, even if later enabled. When this bit is set, data is transmitted via the TXD pin. If the data previously written to UARTTXBUF has already been transmitted, transmission will stop once reset. 0: Disable 1: Enable	0h
2	SLEEP	R/W	Sleep Mode Enabled Control the receiver sleep function in a multiprocessor configuration. Clearing this bit will exit the sleep mode. When SLEEP=1, the receiver can still operate but does not update UARTRXST [6:2] unless any address byte is detected. SLEEP is not cleared when an address byte is detected. 0: Disable 1: Enable	0h
3	TXWAKE	R/W	Data-Transmit Feature Select Determined by the setting of the ADDRIDLE_MODE bit. 0: Data-transmit feature not selected.	0h

Field	Name	R/W	Description	Reset value
			In idle line mode: Write 1 to this bit, and then write data to the UARTTXBUF register, generating an idle period of 11 data bits. In address bit mode: Write 1 to this bit, and then write data to UARTTXBUF, setting the address bit of the frame to 1. 1: Data-transmit feature selection depends on the mode. This bit is cleared by system reset or when transferred to the WUT flag.	
4	Reserved			0h
5	SWRESET	R/W	Software Reset This bit does not affect any configuration bits. All affected logic remains in the specified reset state until this bit is set. Once this bit is asserted, flags will be frozen until deasserted. 0: Writing 0 to this bit initializes the flag bits of UARTCTL2 and UARTRXST. 1: After a system reset, writing 1 to this bit is required to re-enable the UART.	0h
6	RXERRINTENA	R/W	Receive Error Interrupt Enable If RXERROR is set due to an error, setting this bit enables the receive error interrupt. 0: Disable 1: Enable	0h
13:7	Reserved			0h
14	RXDMAENA	R/W	RX DMA request 0: Disable 1: Enable	0h
15	TXDMAENA	R/W	TX DMA request 0: Disable 1: Enable	0h

35.8.3 High Baud Rate Register (UARTHBAUD)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	BAUD	R/W	High Baud Set The internally generated serial clock is determined by the low-speed peripheral clock signal and the baud rate registers. The 16-bit value of these registers are used to select one of 64K serial clock rates for communication mode. If 0<BRR<65536: BRR = (UARTHBAUD<<8) + (UARTLBAUD) The formula for calculating the baud rate is as follows: Asynchronous baud rate = LSPCLK / ((BRR+1)*8) BRR = LSPCLK / (UART asynchronous baud rate*8) - 1	0h

Field	Name	R/W	Description	Reset value
			If BRR = 0: Asynchronous baud rate = LSPCLK / 16 BRR = 16-bit value in the baud rate register	
15:8	Reserved			0h

35.8.4 Low Baud Rate Register (UARTLBAUD)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	BAUD	R/W	Low Baud Set Refer to the BAUD bits in the above table.	0h
15:8	Reserved			0h

35.8.5 Control register 2 (UARTCTL2)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	TXINTENA	R/W	Transmit Data Buffer Empty Interrupt Enable In non-FIFO mode, to generate the first transmit interrupt, a data word must be written to UARTTXBUF. If this bit is enabled after writing data to UARTTXBUF, no interrupt will be generated. 0: Disable 1: Enable	0h
1	RXBKINTENA	R/W	Receiver Buffer/Break Interrupt Enable 0: Disable 1: Enable	0h
5:2	Reserved			0h
6	TXEMPTY	R	Transmitter Empty Flag The SWRESET bit is set to 1 by resetting or enabling the system. This will not cause an interrupt request. 0: The transmit buffer or shift register is not empty 1: The transmit buffer and shift register are empty	0h
7	TXRDY	R	Transmit Data Buffer Empty Flag Writing data to UARTTXBUF will automatically clear this bit. 0: UARTTXBUF is full 1: UARTTXBUF is ready to receive the next character. If TXINTENA=0, this flag asserts a transmitter interrupt request. The SWRESET bit is set to 1 by resetting or enabling the system.	0h
15:8	Reserved			0h

35.8.6 Receive Status Register (UARTRXST)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	RXWAKE	R	Receiver Wake up Occur Flag In address-bit multiprocessor mode, this bit reflects the address bit value of the character contained in UARTRXBUF. In idle line multiprocessor mode, this bit is set if the RXD data line is detected to be idle. This bit can be cleared by reading UARTRXBUF, enabling SWRESET, UARTRST, system reset, or transmitting the first byte after the address byte to UARTRXBUF. 0: No receiver wake up 1: Receiver wake up detected	0h
2	PE	R	Parity Error Occur Flag Set this bit when the received character does not match its parity bit. Disable this bit if parity generation and detection are not enabled. This bit can be cleared by enabling SWRESET, UARTRST, or system reset. 0: No parity error or parity disabled 1: Parity error is detected	0h
3	OE	R	Overrun Error Occur Flag This bit is set when a new character is transferred into UARTRXEMU and UARTRXBUF registers while they already contain a character, causing the previous character to be overwritten and lost. This bit can be cleared by enabling SWRESET, UARTRST, or system reset. 0: No overrun error 1: Overrun error is detected	0h
4	FE	R	Frame Error Occur Flag Only the first stop bit is checked. A missing stop bit indicates loss of synchronization with the start bit and results in a framing error. This bit can be cleared by enabling SWRESET, UARTRST, or system reset. Note: In sleep mode, when RXWAKE is not present and the RXD line remains low for more than 10 bits, only BRKDT is marked. Otherwise, this bit will be marked before BRKDT. 0: No frame error 1: A frame error is detected	0h
5	BRKDT	R	Break Occur Flag When RXD remains low continuously for more than 9.625 bits, starting from the first missing stop bit, a break occurs. If the RX line becomes high during the 9.625-bit period, no interrupt detection will be marked.	0h

Field	Name	R/W	Description	Reset value
			BRKDT interrupt occurs even if SLEEP=1. If RXBKINTENA=1, the interrupt will cause a receiver interrupt but will not load the receiver buffer. After detecting the interrupt, this bit can be cleared by enabling SWRESET, UARTRST, or system reset. This bit and the RXRDY bit will not be set simultaneously; if not reset after an interrupt, no further interrupts will occur. 0: No break condition 1: Break condition occurs	
6	RXRDY	R	Receive Data Buffer Not Empty Flag If RXBKINTENA=1, a receiver interrupt is generated. This bit can be cleared by reading the UARTRXBUF register, enabling SWRESET, UARTRST, or system reset. 0: UARTRXBUF is empty 1: UARTRXBUF is not empty	0h
7	RXERROR	R	Receive Occur Error Flag Indicate that an error flag in UARTRXST has been set. This bit can be used for quick error status checking during an interrupt service routine. It cannot be cleared directly; but can be cleared by enabling SWRESET, UARTRST, or system reset. 0: No error flag 1: An error flag is detected	0h
15:8	Reserved			0h

35.8.7 Simulator/Emulator Register (UARTRXEMU)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	ERXDT	R	Receive Emulation Buffer General data reception operations involve reading received data from the UARTRXBUF register. This register is primarily used by the simulator because it can continuously read received data for updates. It is cleared by system reset. This register must be used in the simulator monitor window to view the contents of the UARTRXBUF register. UARTRXEMU is simply a different address location to access the UARTRXBUF register without clearing the RXRDY flag.	0h
15:8	Reserved			0h

35.8.8 Receive Buffer Register (UARTRXBUF)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	SAR	R	Receive Character Bit	0h
13:8	Reserved			0h
14	UARTFFPE	R	FIFO Parity Error Flag This flag is set if a parity error occurs when a 7-0 bit character is received. This bit is associated with the character at the top of the FIFO. 0: No parity check error 1: A parity check error occurred	0h
15	UARTFFFE	R	FIFO Framing Error Flag This flag is set if a frame error occurs when a 7-0 bit character is received. This bit is associated with the character at the top of the FIFO. 0: No frame error 1: A frame error occurred	0h

35.8.9 Transmit Buffer Register (UARTTXBUF)

Offset address: 0x12

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXDT	R/W	Transmit Buffer If the character length is less than 8 bits, the leftmost bits will be ignored. The data bits to be transmitted must be right-aligned when written to TXDT, and the TXRDY flag is set when data is transferred from this register to the TXSHF transmitter shift register. If TXINTENA=1, this data transmission will also trigger an interrupt.	0h
15:8	Reserved			0h

35.8.10 Transmit FIFO Register (UARTFFTX)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	TXFFIL	R/W	Transmit FIFO Interrupt Level The transmit FIFO generates an interrupt whenever TXFFST is less than or equal to TXFFIL. The maximum value that can be assigned to these bits to generate an interrupt cannot exceed the depth of the TX FIFO. After reset, the default value of these bits is 00000.	0h
5	TXFFIENA	R/W	Transmit FIFO Interrupt Enable 0: Disable 1: Enable. The interrupt is triggered when the TXFFST bit matches the TXFFIL bit.	0h
6	TXFFINTCLR	R/W	Transmit FIFO Interrupt Clear The TXFFINT flag can be cleared by setting this bit.	0h
7	TXFFINT	R	Transmit FIFO Interrupt	0h

Field	Name	R/W	Description	Reset value
			0: No interrupt occurs 1: Interrupt occurs	
12:8	TXFFST	R	Transmit FIFO Word Number 00000: Transmit FIFO is empty 00001: Transmit FIFO contains 1 word 00010: Transmit FIFO contains 2 words … 10000: Transmit FIFO contains 16 words	0h
13	TXFIFORESET	R/W	Re-Transmit FIFO 0: Reset the FIFO pointer to zero and hold it 1: Re-transmit FIFO operation	1h
14	UARTFFENA	R/W	FIFO Enable 0: Disable 1: Enable	0h
15	UARTRST	R/W	UART Reset 0: SWRESET, TXFFINT and RXFFINT are reset, basically clearing the TX/RX FIFO contents. The UART remains in a reset state until 1 is written. It also resets the UARTRXST register and sets the TXEMPTY and TXRDY bits as 1. 1: The FIFO can resume transmit or receive operations. This bit remains 1 even when auto-baud logic is active.	1h

35.8.11 Receive FIFO Register (UARTFFRX)

Offset address: 0x16

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	RXFFIL	R/W	Receive FIFO Interrupt Level The transmit FIFO generates an interrupt whenever RXFFST is less than or equal to RXFFIL. The maximum value that can be assigned to these bits to generate an interrupt cannot exceed the depth of the RX FIFO. After reset, the default value of these bits is 11111.	1Fh
5	RXFFIENA	R/W	Receive FIFO Interrupt Enable 0: Disable 1: Enable. The interrupt is triggered when the RXFFST bit matches the RXFFIL bit.	0h
6	RXFFINTCLR	W	Receive FIFO Interrupt Clear The RXFFINT flag can be cleared by setting this bit . RXFFINTCLR and RXFFOVRCLR need to be cleared at the same time; otherwise, further interrupts will be blocked.	0h
7	RXFFINT	R	Receive FIFO Interrupt 0: No interrupt occurs 1: Interrupt occurs	0h

Field	Name	R/W	Description	Reset value
12:8	RXFFST	R	Receive FIFO Word Count 00000: Receive FIFO is empty 00001: Receive FIFO contains 1 word 00010: Receive FIFO contains 2 words … 10000: Receive FIFO contains 16 words	0h
13	RXFIFORESET	R/W	Re-Receive FIFO 0: Reset the FIFO pointer to zero and hold it 1: Re-receive FIFO operation	1h
14	RXFFOVRCLR	R/W	Receive FIFO Overrun Flag Clear The RXFFOVF flag can be cleared by setting this bit. RXFFINTCLR and RXFFOVRCLR need to be cleared at the same time; otherwise, further interrupts will be blocked.	0h
15	RXFFOVF	R	Receive FIFO Overrun Flag Set when the receive interrupt is active. This bit is cleared by RXFFOVRCLR, UARTRST, or system reset. 0: No overrun 1: Overrun. More than 16 words are received into the FIFO, and the first word is lost	0h

35.8.12 FIFO/Auto-baud Control Register (UARTFFCT)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	FFTXDLY	R/W	FIFO Transfer Delay The delay between each transmission from the FIFO transmit buffer to the transmit shift register ranges from 0 to 256 baud clock cycles. In FIFO mode, the buffer between the shift register and FIFO needs to be filled after the shift register has shifted the last bit. In FIFO mode, TXBUF cannot be considered as an additional buffer level. When there is only one UART stop bit, the delay inserted by FFTXDLY is the baud clock cycles of FFTXDLY. When there are two stop bits, the delay inserted by FFTXDLY requires subtracting 1.	0h
12:8	Reserved			0h
13	CDC	R/W	Auto-baud Alignment Enable 0: Disable 1: Enable	0h
14	ABDCLR	W	Auto-baud Detect Flag Clear Setting this bit can clear the ABD flag.	0h
15	ABD	R	Auto-baud Detect Flag 0: Auto baud detection not completed 1: Auto baud detection completed	0h

35.8.13 Select Mode Register (UARTPRI)

Offset address: 0x1E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	Reserved			0h
4:3	FREESOFT	R/W	Select the Mode When an Emulation Event Suspends 00: Suspend and stop immediately 01: Complete the current receive/transmit sequence before stopping 10: Free running 11: Free running	0h
15:5	Reserved			0h

36 Internal integrated circuit interface (I2C)

36.1 Full Name and Abbreviation Description of Terms

Table 187 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Serial Data	SDA
Serial Clock	SCL
Clock	CLK
Arbitration	ARB
No Acknowledgement	NACK

36.2 Introduction

Provide an interface between the device and device compliant with I2C Bus Specification version 2.1, and connect via the I2C bus. External components connected to this 2-wire serial bus can transmit/receive 1 to 8 bits of data to/from the device through the I2C module. Support I2C-compatible master or slave.

36.3 Main characteristics

(1) Master mode or slave mode can be selected
- Multi-master transmitting and slave receiving
- Multi-slave transmitting and master receiving
- Master transmitting/receiving and receiving/transmitting modes

(2) Free data format mode

(3) 8-bit format transmission

(4) 7-bit and 10-bit addressing mode

(5) Broadcast communication

(6) Start byte mode

(7) Data transmission rate: 10 kbps~400 kbps

(8) Receive FIFO and transmit FIFO

(9) I2C interrupt
- Ready to transmit data
- Ready to receive data
- Register access ready

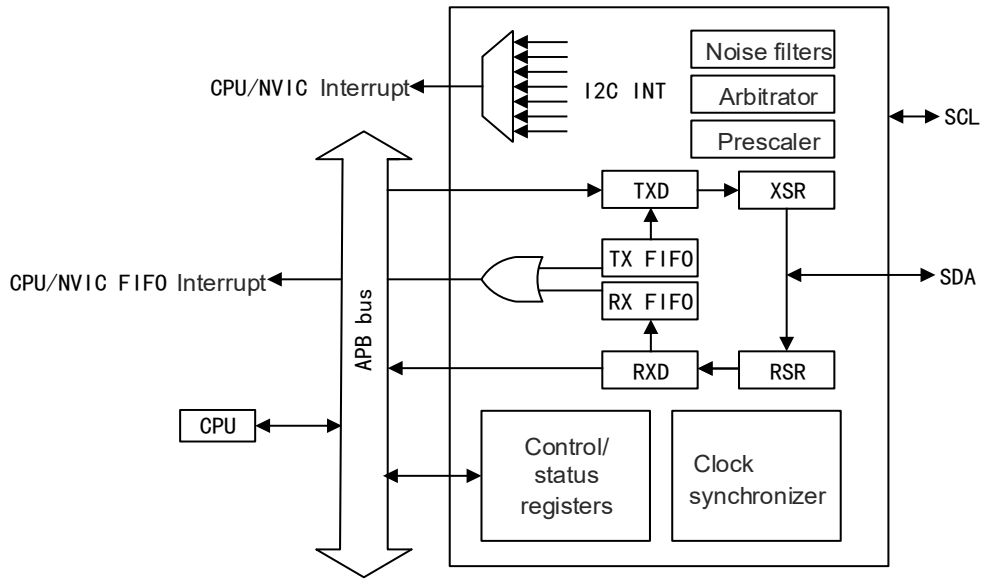
- Addressed as a slave
- No answer received
- Bus arbitration fails
- Stop condition is detected

(10) I2CFIFO interrupt

- Transmit FIFO interrupt
- Receive FIFO interrupt

36.4 Structure block diagram

Figure 190 Structure Block Diagram



36.5 Functional description

36.5.1 Reset or Disable

When I2CEN = 0, all status bits in the I2CSTS register are forced to their default values, and the I2C module remains disabled until the I2CEN bit is set. The SDA and SCL pins are in a high-impedance state. Setting the XRS pin low will reset the entire device. When releasing the XRS pin, all I2C registers are resest to their default values. The I2CEN bit is forced to 0, indicating that the I2C is reset, which can save power and clear error conditions. The I2C can be configured or reconfigured only when I2CEN = 0.

36.5.2 Configure pins

Table 188 I2C Pin-related Functions

Operation	Function
Configure GPIO multiplexing registers	Connect the peripheral to the device pins

Operation	Function
Set the appropriate GPIO input type register n bits to 11	Set the GPIO input qualification to asynchronous mode
First configure the GPIO pin multiplex high register n bits, then configure the GPIO pin multiplex low register n bits	Avoid burrs on the pins

The internal pull-up resistor can be configured in the GPIO pull-up/down configuration register n bits. Some IO functions are defined by GPIO register settings independent of the peripheral.

36.5.3 Operation mode

Input and output level

The master generates one clock pulse for each bit of data transmitted. Due to the various technical devices that can be connected to the I2C bus, the high/low levels depend on the associated VDD level.

Data validity

The high/low state of SDA can only change when the clock signal SCL is in low level. During the high level of the clock, data on SDA must remain stable.

Operation mode

There are four basic operational modes to support data transmission as a master and a slave. If the I2C is the master, it operates as a master transmitter, typically transmitting an address to a specific slave. To receive data from the slave, the I2C must be switched to master receiver mode. If the I2C is the slave, it begins as a slave receiver and transmits an acknowledgment signal when the master recognizes its slave address. If the master requests data from the I2C, the module must be switched to slave transmitter mode.

(1) Master Transmit Mode

The I2C, as the master, transmits control information and data to the slave. All masters start in this mode, and data in 7-bit/10-bit address format is shifted out on SDA. The transfer of data bits is synchronized with clock pulses generated on SCL. After transmitting one byte, if device intervention is required, clock pulses are inhibited, and SCL is held low.

(2) Master Receive Mode

The I2C, as the master, receives data from the slave. In master transmit mode, the I2C transmits a command to the slave before entering this mode. When using 7-bit/10-bit address format, the I2C enters master receive mode after transmitting the slave address byte and R/W = 1.

Serial data bits are shifted into the I2C according to clock pulses on SCL. After

receiving one byte, if device intervention is required, clock pulses are inhibited, and SCL is held low.

(3) Slave Transmit Mode

The I2C, as the slave, transmits data. In slave receive mode, the I2C receives a command from the master before entering this mode. When using 7-bit/10-bit address format, if the slave address byte matches its own address and the master has transmitted R/W = 1, the I2C enters slave transmit mode.

As a slave transmitter, the I2C shifts serial data bits out to SDA according to clock pulses generated by the master. Write data to the TXD register to release SCL. As a slave, the I2C does not generate clock signals; after transmitting one byte, if device intervention is required, SCL can be held low.

(4) Slave Receive Mode

The I2C, as the slave, receives data. All slaves start in this mode. In this mode, clock pulses generated by the master shift received serial data bits into the I2C. Read data from the RXD register to release SCL. As a slave, the I2C does not generate clock signals; after receiving one byte, if device intervention is required, SCL can be held low.

36.5.4 Start and Stop

The I2C can generate start and stop conditions when configured as a master. A transition from high to low on SDA when SCL is in high level is called a start condition. The master transmits this condition to indicate the beginning of data transmission. A transition from low to high on SDA while SCL is in high level is called a stop condition. The master transmits this condition to indicate the end of data transmission.

Between the start condition and the next stop condition, the I2C bus is in a busy state, and at this point, BBSYFLG = 1. Between the stop condition and the next start condition, the I2C bus is in an idle state, and at this point, BBSYFLG = 0. When the I2C is in reset, start or stop conditions cannot be detected, and the BBSYFLG bit always remains 0. When the I2C is out of reset, the BBSYFLG bit cannot correctly reflect the I2C bus status until a start or stop condition is detected. When the MSCFG bit and STAGEN bit are set, the I2C transmits a start condition at the beginning of data transmission. When the STOGEN bit is set, the I2C transmits the stop condition at the end of data transmission. When both the BBSYFLG bit and the STAGEN bit are set simultaneously, a repeated start condition is generated.

Before the first data transmission using the I2C, the following steps must be performed:

(1) Set the I2CEN bit to enable the I2C, then wait for a period longer than the maximum data transmission time. During this period, ensure that at

least one start or stop condition has been captured by the BBSYFLG bit. Afterward, the BBSYFLG bit will correctly reflect the I2C bus status.

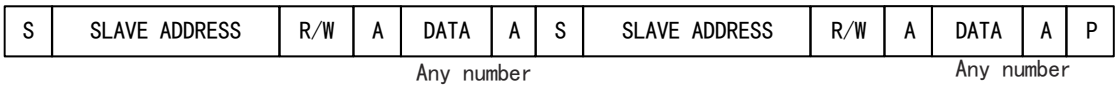
(2) Check if the BBSYFLG bit is 0 before starting data transmission.

Not resetting the I2C during transmission ensures that the BBSYFLG bit reflects the actual I2C bus status. If resetting the I2C during transmission is necessary, the above steps must be repeated each time the I2C is brought out of reset.

36.5.5 Repeated start condition

The I2C master can communicate with multiple slave addresses by transmitting another start condition at the end of each data transmission. Repeated start conditions can be used with 7-bit addressing, 10-bit addressing, and free data format. Repeated start conditions are shown as shown below:

Figure 191 Repeated Start Conditions



Remarks:

(1) S: Start signal

(2) SLAVE ADDRESS: Slave address

(3) R/W: Selection bit of transmission direction

(4) P: Stop signal

36.5.6 Repeated and Non-repeated

Repeated Mode (REEN = 1)

(1) The software controls the number of bytes to be transmitted or received.

(2) The RAIFLG bit is set at the end of each byte transmit or receive.

(3) Once I2C transmit starts, switching to another mode is not allowed until the I2C transmission completes.

Non-repeated Mode (REEN = 0)

(1) The I2CCNT register determines the number of bytes to be transmitted or received.

(2) If STOGEN = 0, the RAIFLG bit is set when the internal data counter counts down to 0.

(3) If STOGEN = 1, the I2C generates a stop condition when the internal data counter counts down to 0, and the RAIREN bit is not set.

(4) Once I2C transmit starts, switching to another mode is not allowed until the I2C transmission completes.

(5) If I2CCNT is set to 0, the I2C state machine expects to transmit or receive 65536 bytes rather than 0 bytes.

36.5.7 Addressing Format

7-bit Addressing

After reset, the default is 7-bit addressing format. This format is enabled when extended addressing and free data format are disabled. In this case, the first byte after the start condition consists of a 7-bit slave address and an R/W bit. R/W determines the direction of data:

- R/W = 0: I2C master writes data to the addressed slave (RXTXCFG = 1).
- R/W = 1: I2C master reads data from the slave (RXTXCFG = 0).

A dedicated ACK clock cycle is inserted after each byte. If the slave inserts an ACK bit after the master's first byte, the transmitter transmits the data set by the DBSEL bit. After the data bit transfer is complete, the receiver inserts an ACK bit.

10-bit Addressing

Setting the extended address and disabling the free data format enables the 10-bit addressing format. It is similar to the 7-bit addressing format, but the master transmits the slave address in two separate bytes:

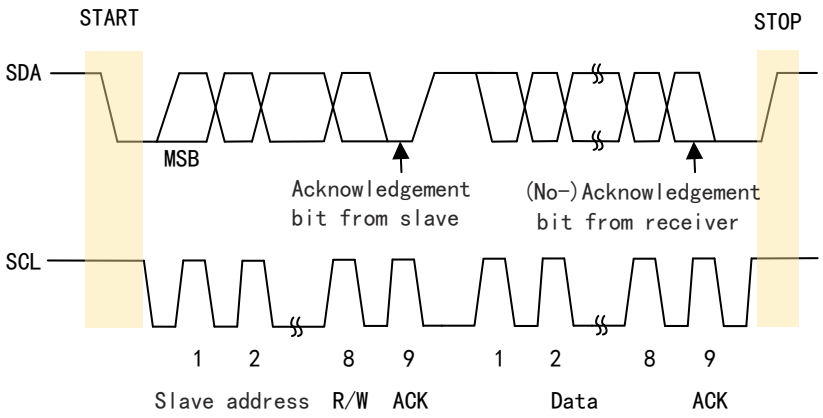
- The first byte consists of 11110, the two most significant bits of the 10-bit slave address, and the R/W bit.
- The second byte is the remaining 8 bits of the 10-bit slave address.

The slave needs to transmit an acknowledgment after each byte is transmitted. Once the master has written the second byte to the slave, the master can write data or use a repeated start condition to change the data direction.

36.5.8 Serial Data Format

Data values from 1 to 8 bits are supported. Each bit on the SDA line corresponds to one pulse on the SCL line; values of most significant bit are transmitted first. The number of data values that can be transmitted or received is unlimited. The following is an example of the 7-bit addressing format:

Figure 192 Serial Data Format



36.5.9 Free data format

FDFEN = 1 enables the free data format. In this case, the first byte after the start condition is the data byte. The number of bits in this byte is determined by the DBSEL bit. An ACK bit is inserted after each data byte. No address or data direction bit is transmitted. Therefore, both the transmitter and receiver support the free data format, and the direction of the data remains constant throughout the transmission. Digital loopback mode is not supported in this format.

Table 189 Configuration of FDFEN and MSCFG Bits

FDFEN	MSCFG	I2C status	RXTXCFG Status
0	0	Non-free data format, slave mode	Unaffected. The I2C responds as either a transmitter or receiver, depending on the master's command.
1	0	Free data format, slave mode	The free data format requires the I2C to remain as either a transmitter or receiver throughout the transmission. RXTXCFG = 0: I2C is the receiver RXTXCFG = 1: I2C is the transmitter
0	1	Non-free data format, master mode	RXTXCFG = 0: I2C is the receiver RXTXCFG = 1: I2C is the transmitter
1	1	Free data format, master mode	

36.5.10 Clock prescaler register

A programmable prescaler divides SYSCLK to generate the I2C module clock (7~12 MHz), which determines the operating frequency of the I2C module. Further division produces the I2C master clock on the SCL pin. The PSC bit field must be initialised before the prescaler value can be specified. The correspondence is as follows:

$$\text{I2C Clock} = \text{SYSCLK} / (\text{PSC} + 1)$$

The prescaler can only be initialised whilst the I2C is in the reset state. The scaled frequency takes effect once the I2C is enabled. When the I2C is configured as the master on the I2C bus, the SCL pin acts as the master clock transmit pin . This clock controls the communication timing between the I2C and the slave device.

When the I2C acts as a master, the I2C clock can be re-divided as the master clock on the SCL pin. The jumping of master clock is determined by CLKH and CLKL.

The relationship between the master clock period and the module clock period is as follows:

Master Clock Period = [(CLKH+d)+(CLKL+d)] / I2C Module Clock

Where d is the delay, the PSC field determines the value of d, when PSC is 0, d is 7. When PSC is 1, d is 6. In all other cases, d is 5.

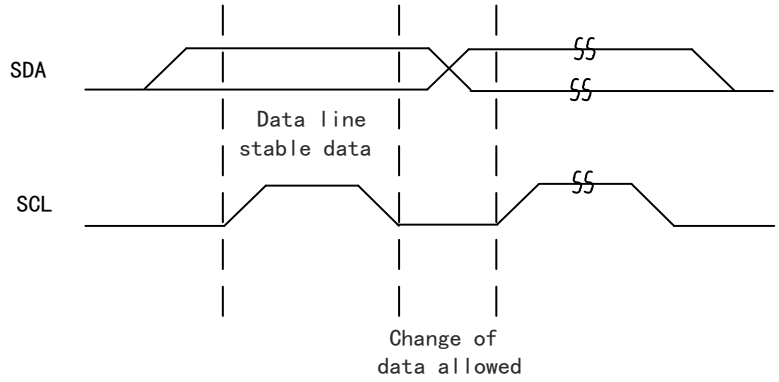
36.5.11 Input/Output Levels

The master generates one clock pulse for each bit of data transmitted. Due to the various technical devices that can be connected to the I2C bus, the levels of logic 0 and logic 1 depend on the associated VDD level. For details, refer to the datasheet for.

36.5.12 Data validity

During the high level of the clock, data on SDA must remain stable. The high or low level state of the data line SDA can only be changed when the clock signal SCL is in low level.

Figure 193 Data Validity



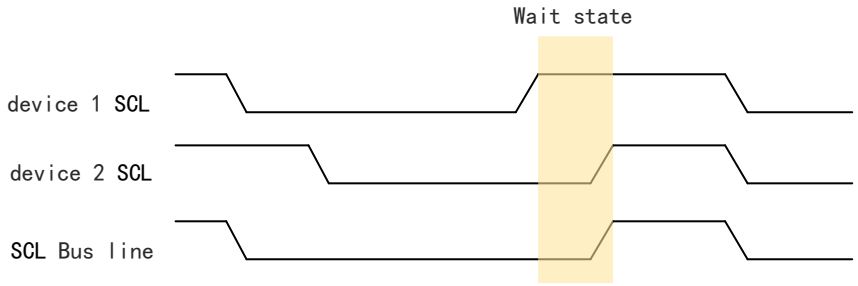
36.5.13 Clock synchronization

Generally, only one master generates the clock signal SCL. However, during arbitration, there may be two or more masters, so the clock needs to be synchronized to compare data output.

The first device to generate a low level on SCL overrides the others. During the

transition from high to low level, it forces the clock generators of other devices to begin their own low-level periods. The device with the longest low-level period pulls SCL low. Other devices that complete their low-level periods must wait for SCL to be released before beginning their high-level periods. If one device pulls the clock line low for a longer time, all clock generators enter a wait state. Thus, slaves can slow down high-speed masters, and slow devices can create sufficient time to store bytes to be transmitted or that have been received. In the synchronized signal obtained on SCL, the slowest device determines the length of the low-level period, and the fastest device determines the length of the high-level period.

Figure Synchronization of Two I2C Clock Generators194



36.5.14 Arbitration

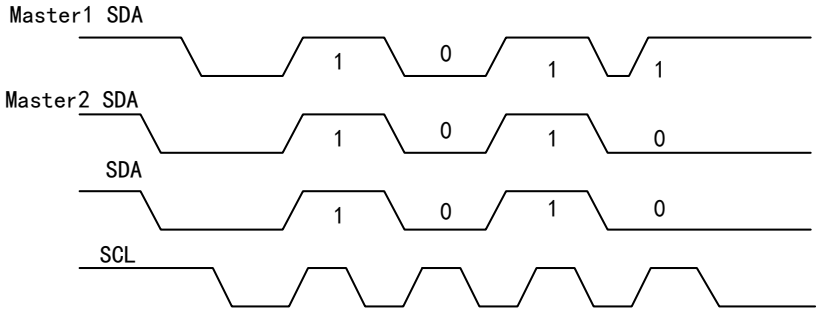
Arbitration occurs when two or more master-transmitters attempt to initiate a transmission on the same bus at the same time. When multiple transmitters compete for the use of SDA, the arbitration mechanism determines the priority of the transmission. Arbitration prioritizes the data stream with the lowest binary value. The master-transmitter that pulls SDA low overrides the first master-transmitter that releases SDA high. If two or more devices transmit the same first byte, arbitration continues on subsequent bytes.

If the I2C is the master that loses arbitration, it switches to slave-receiver mode, sets the ARBLIFLG bit, and generates an arbitration lost interrupt request. If the arbitration process is still ongoing during the transmission, but a repeated start condition or stop condition is also transmitted to SDA, the relevant master-transmitter must transmit the repeated start condition or stop condition at the same position in the format frame.

Arbitration cannot occur in the following situations:

- Between a repeated start condition and a data bit
- Between a repeated start condition and a stop condition
- Between a stop condition and a data bit

Figure 195 Arbitration Procedure



Note: Master1 Arbitration failure

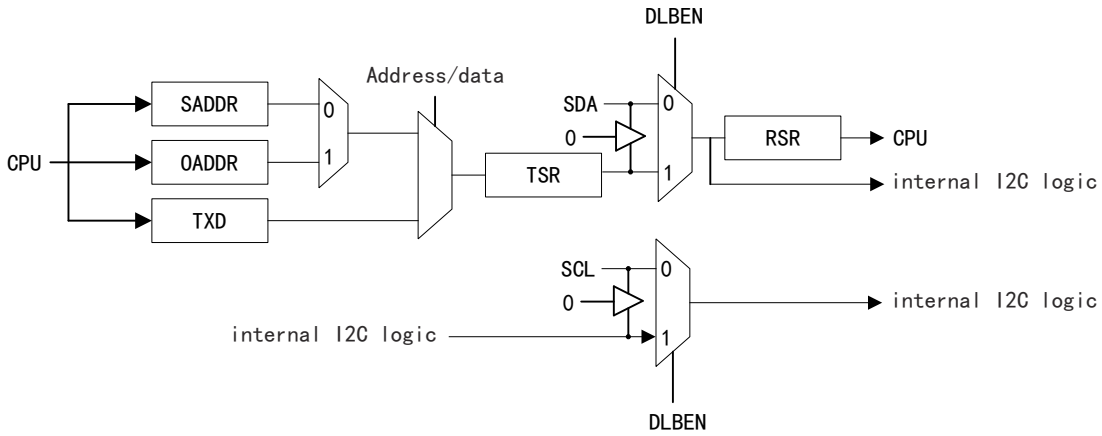
36.5.15 Digital Loopback

Setting the DLBEN bit enables the digital loopback mode. In this mode, data transmitted from the TXD register is transmitted along an internal path and takes n cycles to reach RXD, where n is determined by:

n = 8 * (SYSCLK) / (I2C Module Clock)

The transmit and receive clocks are the same. The address seen on the external SDA pin is the address in the OADDR register. The free data format is not supported in this mode.

Figure 196 Digital Loopback Mode



36.5.16 Generate a NACK Bit

When the I2C acts as a receiver, the NACK bit can acknowledge or ignore the bit transmitted by the transmitter. The I2C ignores new bits by transmitting a not-acknowledge bit during the acknowledge cycle on the bus.

The methods for generating a NACK bit vary depending on the mode, as follows:

(1) Master receiver mode and repeated mode

- Reset the module

- Generate a stop condition
- Set the NACKCFG bit before the rising edge of the last data bit planned to be received

(2) Master receiver mode and non-repeated mode

- Reset the module
- If STOGEN = 0, set STOGEN = 1 to generate a stop condition
- If STOGEN = 1, allow the internal data counter to count down to 0 to force the generation of a stop condition
- Set the NACKCFG bit before the rising edge of the last data bit planned to be received

(3) Slave receiver mode

- Reset the module
- Allow an overflow condition
- Set the NACKCFG bit before the rising edge of the last data bit planned to be received

36.5.17 Interrupt request

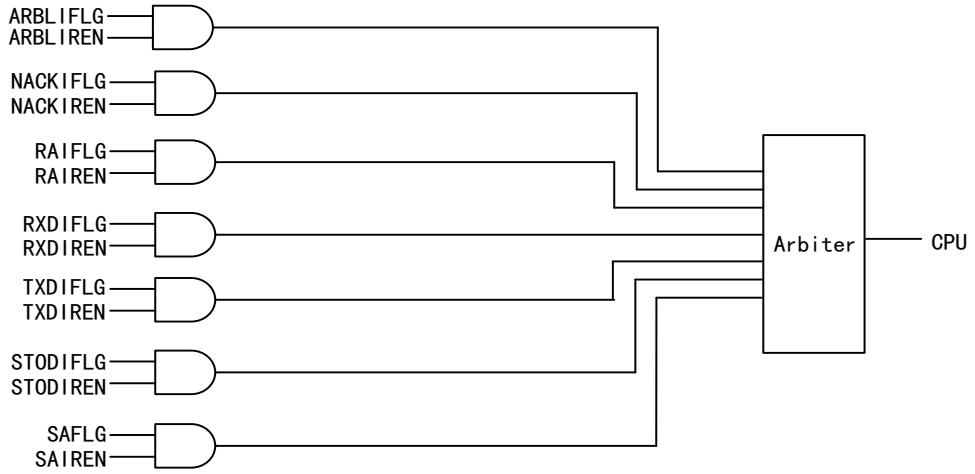
Basic Interrupt Requests

All requests are multiplexed through an arbiter to a single I2C interrupt request transmitted to the CPU. Each interrupt request has an enable bit and a flag bit. The flag bit is set when the specified event occurs. If the corresponding enabled bit is 0 at this time, the interrupt request is blocked. If it is 1, the request is forwarded to the CPU as an I2C interrupt. The basic interrupt requests are prioritized from highest to lowest as follows:

Table 190 I2C Interrupt Request

Interrupt event	Event flag bit	Interrupt control bit
Arbitration loss	ARBLIFLG	ARBLIREN
No response	NACKIFLG	NACKIREN
Register access ready	RAIFLG	RAIREN
Ready to receive data	RXDIFLG	RXDIREN
Ready to transmit data	TXDIFLG	TXDIREN
Stop Condition Detection	STODIFLG	STODIREN
Slave address	SAFLG	SAIREN

Figure 197 I2C Interrupt Mapping

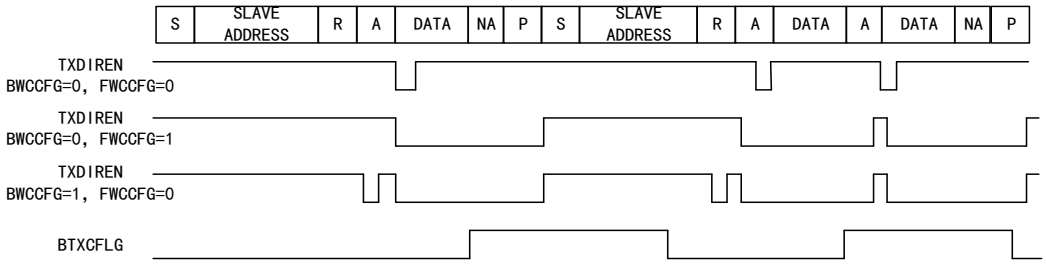


Setting the FWCCFG bit avoids a situation where the normal transmit interrupt timing might cause old data to remain in the transmit buffer if an interrupt during a transmit byte transaction is interrupted. If this bit is set, a transmit data ready interrupt is generated only when required by the bus transaction.

- In master mode, the interrupt is first generated when the ACK for the address byte is received.
- In slave mode, the interrupt is first generated when the address matches. Further interrupts are generated when data is ACKed. TXDIREN is asserted simultaneously with the transmit ready interrupt.

The BWCCFG and FWCCFG bits in the EXT register affect the I2C and interrupts. When configured as a slave-transmitter, the effects of these two bits on the I2C module registers and interrupts are shown in the following figure:

Figure 198 Backward Compatible Bit and Forward Compatible Bit



Remarks:

(1) S: Start signal

(2) SLAVE ADDRESS: Slave address

(3) R/W: Selection bit of transmission direction

(4) A: Acknowledge bit

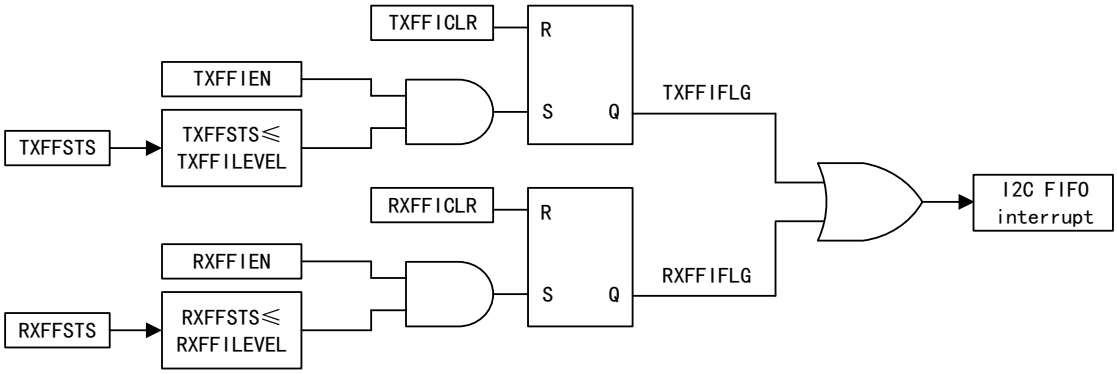
(5) NA: Not-acknowledge bit

(6) P: Stop signal

FIFO interrupt

In addition to the basic I2C interrupts, both the transmit and receive FIFOs can generate interrupts. These can be configured to generate an interrupt after transmitting/receiving a specified number of bytes. These two interrupt sources are OR-ed together to form a single maskable CPU interrupt.

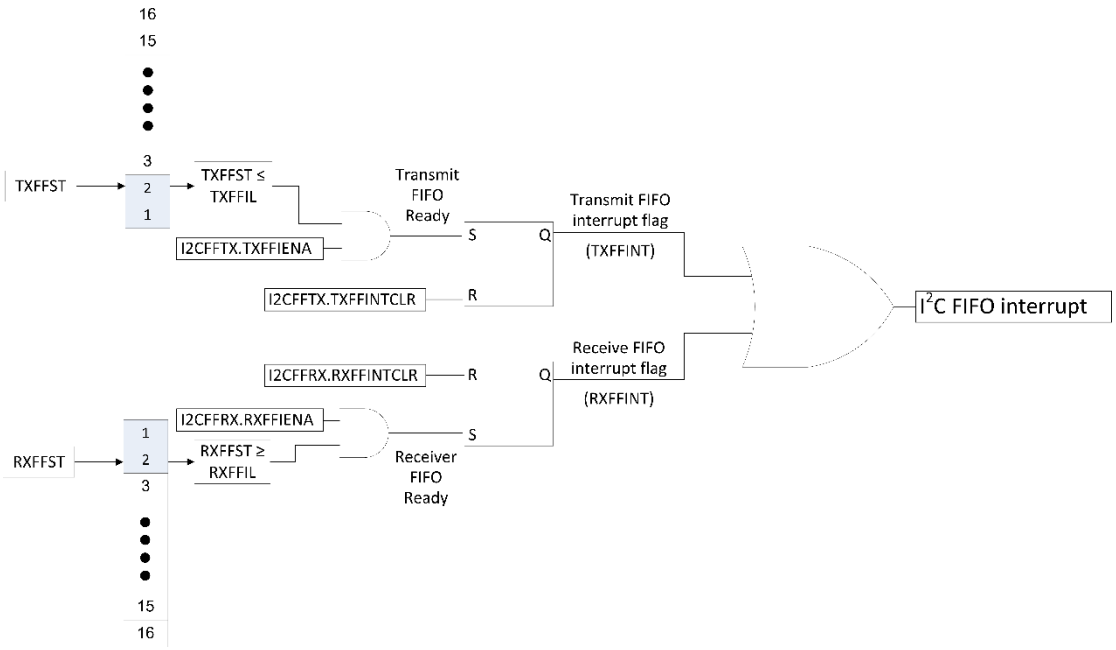
Figure 199 FIFO Interrupt



36.5.18 DMA Support Scheme

As the R502's DMA does not feature handshake signals such as 'request' and 'ack', it relies solely on unidirectional trigger signals sent from the peripheral to the DMA. The logic for generating the I2C DMA trigger signal can be based on that used for I2C interrupts.

Figure200 Schematic of the I2C DMA Request Signal



FIFO Mode

In FIFO mode, once the interrupt is enabled, the generation of the TX interrupt is determined by TXFFIL (Transmit FIFO interrupt level) and TXFFST (FIFO status). When the number of data items in the TX FIFO (TXFFST) is less than or equal to TXFFIL, the TXFFINT interrupt flag is set high and a TX interrupt is generated.

After enabling I²C FIFO mode, the TXFFINT flag is set to zero; once the TX FIFO interrupt and the transmit FIFO are enabled, the TXFFINT flag is set high and a TX interrupt is generated.

In FIFO mode, triggering TX DMA requires a comparison between TXFFST and TXFFIL; a TX DMA pulse is output when TXFFST is less than or equal to TXFFIL.

In FIFO mode, once the interrupt is enabled, the generation of the RX interrupt is determined by RXFFIL (Receive FIFO interrupt level bits) and RXFFST (Receive FIFO status). When the data received by the RX FIFO (RXFFST) is greater than or equal to RXFFIL, the RXFFINT interrupt flag is set high and an RX interrupt is generated.

In FIFO mode, triggering the RX DMA requires a comparison between RXFFST and RXFFIL; when RXFFST is greater than or equal to RXFFIL, an RX DMA pulse is output.

Non-FIFO mode

In non-FIFO mode, the logic for generating the I²C TX interrupt consists primarily of the XRDY flag bit. XRDY is the transmit buffer register ready flag; after a data item has been written to the data transmission register I²CDXR, when the data is copied to the transmission shift register I²CXSR, I²CDXR becomes empty, the XRDY flag bit is set high, and the TX interrupt signal is set high.The TX interrupt signal is in pulse form.

After I²C is enabled, the XRDY flag is set to zero; once the XRDYINT interrupt is enabled, the XRDY flag is set high, triggering the XRDYINT interrupt.

In non-FIFO mode, the XRDY flag can be used as a trigger pulse for DMA.

In non-FIFO mode, the logic for generating the I2C RX interrupt is primarily based on the RRDY flag. RRDY is the receive buffer register ready flag. The RX interrupt signal is in pulse form.

When received data is copied from the receive shift register (I2CRSR) to the data receive register (I2CDRR), the RRDY flag is set to 1 and an RRDYINT interrupt is generated.

In non-FIFO mode, the RRDY flag can be used as a trigger pulse for DMA.

36.6 Register bank address

Table 191 Register Bank Address

Device register	Register bank	Start address	End address
I2CARegs	I2C_REGS	0x4000_1800	0x4000_1BFF
I2CBRegs	I2C_REGS	0x4000_1C00	0x4000_1FFF

36.7 Register address mapping

Table 192 Register Address Mapping

Register name	Register description	Offset address	WRPRT
OADDR	I2C address register	0x00	-
IREN	Enable interrupt request register	0x02	-
STS	Status register	0x04	-
CLKL	Clock Low Frequency Divider	0x06	-
CLKH	Clock High Frequency Divider	0x08	-
CNT	Counter	0x0A	-
RXD	Receive data register	0x0C	-
SADDR	Slave address register	0x0E	-
TXD	Transmit data register	0x10	-
CTRL	Control register	0x12	-
ISRC	Interrupt source register	0x14	-
EXT	Extension Register	0x16	-
PSC	Prescale register	0x18	-
TXFIFO	Transmit FIFO register	0x40	-
RXFIFO	Receive FIFO register	0x42	-

36.8 Register functional description

36.8.1 I2C Address Register (I2COADDR)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	OADDR	R/W	I2C Own Address The I2C module defines its own slave address through this register, which is used to distinguish it from other slave devices on the I2C bus. 7-bit Addressing Mode: Field [6:0] provides the 7-bit slave address of the I2C. 10-bit Addressing Mode: Field [9:0] provides the 10-bit slave address of the I2C.	0h
15:10	Reserved			0h

36.8.2 Interrupt Request Enable Register (I2CIREN)

Offset address: 0x02

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ARBLIREN	R/W	Arbitration Lost Interrupt Request Enable 0: Disable 1: Enable	0h
1	NACKIREN	R/W	No Acknowledgment Interrupt Request Enable 0: Disable 1: Enable	0h
2	RAIREN	R/W	Register Access Ready Interrupt Request Enable 0: Disable 1: Enable	0h
3	RXDIREN	R/W	Receive Data Ready Interrupt Request Enable This bit cannot be set when using FIFO mode. 0: Disable 1: Enable	0h
4	TXDIREN	R/W	Transmit Data Ready Interrupt Request Enable This bit cannot be set when using FIFO mode. 0: Disable 1: Enable	0h
5	STODIREN	R/W	Transmit Data Ready Interrupt Request Enable 0: Disable 1: Enable	0h
6	SAIREN	R/W	Save Addressed Interrupt Request Enable 0: Disable 1: Enable	0h
15:7	Reserved			0h

36.8.3 Status register (I2CSTS)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ARBLIFLG	R/W	Arbitration Lost Interrupt Flag	0h

Field	Name	R/W	Description	Reset value
			Applicable only when I2C is the master transmitter. The CPU can poll ARBLIFLG or use the ARBLIFLG interrupt request. This bit is set when the I2C senses that it has lost arbitration with two or more transmitters that began transmission simultaneously, or when the I2C attempts to begin transmission while BBSYFLG = 1. 0: Arbitration not lost 1: Arbitration is lost Writing a 1 to this bit, resetting the I2C, an ACK having been transmitted from the receiver, or the CPU reading the I2CISRC register will clear this bit.	
1	NACKIFLG	R/W	No Acknowledgment Interrupt Flag Apply NACK when the I2C is the master transmitter. The CPU can poll NACKIFLG or use the NACKIFLG interrupt request. NACKIFLG = 1 when the I2C module performs a general call transfer. 0: NACK is not received 1: NACK is received Writing a 1 to this bit, resetting the I2C, an ACK having been transmitted from the receiver, or the CPU reading the I2CISRC register will clear this bit.	0h
2	RAIFLG	R/W	Register Access Ready Interrupt Flag Applicable only when the I2C module is in master mode. The CPU can poll RAIFLG or use the RAIFLG interrupt request. In non-repeated mode, if STOGEN = 0, this bit is set when the internal data counter counts down to 0. In repeated mode, this bit is set at the end of each byte transmitted by TXD. 0: Not ready 1. Ready Writing a 1 to this bit, resetting the I2C, or when the I2C starts using the contents of the current register, will clear this bit.	0h
3	RXDIFLG	R/W	Receive Data Ready Interrupt Flag When not in FIFO mode, RXD is ready to be read. When in FIFO mode, use RXFFIFLG instead. The CPU can poll RXDIFLG or use the RXDIFLG interrupt request. 0: Not ready 1. Ready Writing a 1 to this bit, resetting the I2C, or when RXD is read by the CPU will clear this bit.	0h
4	TXDIFLG	R	Transmit Data Ready Interrupt Flag When not in FIFO mode, this bit indicates that TXD is ready to accept new data. When in FIFO mode, use TXFFIFLG instead. The CPU can poll TXDIFLG or use the TXDIFLG interrupt request. 0: Not ready 1. Ready This bit is forced to 1 when the I2C is reset. This bit will be cleared when data is written to I2CTXD.	1h

Field	Name	R/W	Description	Reset value
5	STODIFLG	R/W	Stop Condition Detected Interrupt Flag This bit is set when the I2C transmits or receives a stop condition. When this bit is 0, the I2C delays the clearing of the STOGEN bit. 0: No stop condition is detected 1: Stop condition is detected Writing a 1 to this bit, resetting the I2C, or when the CPU reads a value of 110 from I2CISR will clear this bit.	0h
6	BTXCFLG	R/W	Byte Transmit Complete Flag This bit is set when the master/slave successfully transmits a byte on SCL/SDA. 0: Byte transmission not complete 1: Byte transmission completed Writing a 1 to this bit or resetting the I2C will clear this bit.	0h
7	Reserved			0h
8	ZEROAFLG	R	All Zeros Address Flag 0: All-zero address not detected 1: All-zero address detected	0h
9	SAFLG	R	Slave Address Flag 0: In 7-bit addressing mode, this bit is cleared when a NACK, STOP condition, or repeated START condition is received. In 10-bit addressing mode, this bit is cleared when a NACK, STOP condition, or a slave address different from the I2C's own slave address is received. 1: The I2C has recognized its own slave address or the all-zero address	0h
10	TXSNEFLG	R	Transmit Shift Register not Empty Flag An underflow occurs when the transmit shift register is empty, but the TXD register is not loaded. The next transmission does not occur until new data is in TXD. If the new data is not transmitted in time, the previous data may be retransmitted on the SDA pin. 0: Transmit shift register empty 1: The transmit shift register is empty Writing to TXD or resetting the I2C will set this bit.	1h
11	RXSFFLG	R	Receive Shift Register Full Flag An overflow occurs if new data is received before the old data in the shift register has been read out. When new bits arrive from the SDA pin, they overwrite the bits in the receive shift register. The new data is not copied into RXD until the old data has been read. 0: No overrun is detected 1: Overrun is detected This bit is cleared when RXD is read by the CPU or the I2C is reset.	0h
12	BBSYFLG	R	Bus Busy Flag	0h

Field	Name	R/W	Description	Reset value
			0: The bus is idle 1: The bus is busy This bit is cleared when the I2C transmits/receives a STOP condition or the I2C is reset.	
13	TXNACKFLG	R/W	Transmit No Acknowledgement Flag This bit is used when the I2C is in receive mode. 0: Not transmit NACK 1: Transmit NACK Writing a 1 to this bit or resetting the I2C will clear this bit.	0h
14	SFLG	R/W	Slave Flag 0: I2C is not addressed as a slave 1: I2C is addressed as a slave Writing a 1 to this bit, enabling digital loopback mode, or a START or STOP condition occurring on the I2C bus will clear this bit.	0h
15	Reserved			0h

36.8.4 Clock Low Frequency Divider (I2CCLKL)

Offset address: 0x06

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CLKL	R/W	Clock Low-time Divide Value To generate a lower master clock duration, the module clock period is multiplied by (CLKL + d), where d is an adjustment factor based on the prescaler. Note: It must be a non-zero value to generate the I2C clock correctly.	0h

36.8.5 Clock High Frequency Divider (I2CCLKH)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CLKH	R/W	Clock High-time Divide Value To generate a higher master clock duration, the module clock period is multiplied by (CLKH + d), where d is an adjustment factor based on the prescaler. Note: It must be a non-zero value to generate the I2C clock correctly.	0h

36.8.6 Counter (I2CCNT)

Offset address: 0x0A

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	CNT	R/W	Counter Value	0h

Field	Name	R/W	Description	Reset value
			The number of data bytes to be transmitted or received. If a STOP condition is specified, CNT decrements after each byte is transmitted until it reaches zero, which in turn generates a STOP condition. 0000000000000000: Count value is 65536 0000000000000001: Count value is 1 0000000000000010: Count value is 2 … 1111111111111111: Count value is 65535	

36.8.7 Receive data register (I2CRXD)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	RXD	R	Receive Data The I2C can receive data bytes from 1 to 8 bits. The number of bits is determined by the DBSEL bit. When a complete data byte is received, the I2C copies the data byte from the receive shift register to RXD. The CPU cannot directly access the receive shift register. If a data byte of fewer than 8 bits is present in RXD, the data value is right-aligned, and the other bits of RXD are undefined. For example, if DBSEL = 011, the received data is RXD[2:0], and the contents of RXD[7:3] are undefined. When in receive FIFO mode, the RXD register acts as the receive FIFO buffer.	0h
15:8	Reserved			0h

36.8.8 Slave address register (I2CSADDR)

Offset address: 0x0E

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	SADDR	R/W	Slave Address Set In 7-bit addressing mode: SADDR[6:0] provides the 7-bit slave address that the I2C transmits in master transmit mode. In 10-bit addressing mode: SADDR[9:0] provides the 10-bit slave address that the I2C transmits in master transmit mode.	3FFh
15:10	Reserved			0h

36.8.9 Transmit data register (I2CTXD)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXD	R/W	Transmit data	0h

Field	Name	R/W	Description	Reset value
			The I2C can accept data bytes from 1 to 8 bits. The number of bits is determined by the DBSEL bit. When a data byte of fewer than 8 bits is written, the data value is right-aligned, and the other bits of TXD are undefined. After the data byte is written to TXD, the I2C copies the data byte to the transmit shift register. The CPU cannot directly access the transmit shift register. When in transmit FIFO mode, the TXD register acts as the transmit FIFO buffer.	
15:8	Reserved			0h

36.8.10 Control register (I2CCTRL)

Offset address: 0x12

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	DBSEL	R/W	Data Type Select The number of bits in the next data byte that the I2C will receive or transmit. It must match the data size of the other device. The address byte is always 8 bits, regardless of this bit. When DBSEL = 000, a data byte has 8 bits. Transmit data written to TXD must be right-aligned. If the number of bits is less than 8, the received data is right-aligned in RXD[7:0], and the other RXD[7:0] bits are undefined. The following indicates how many bits are in each data byte: 000: 8 bits 001: 1 bit 010: 2 bits … 111: 7 bits	0h
3	FDFEN	R/W	Free Data Format Mode Enable 0: Disable. Transmit 7/10-bit Addressing Format 1: Enable. Transmit Free Data Format	0h
4	STABEN	R/W	Free Data Format Mode Enable Applicable only when I2C is the master. The START byte can help slaves that require extra time to detect the START condition. When the I2C is a slave, the START byte from the master is ignored. 0: Disable 1: Enable. When STAGEN = 1, the I2C can generate a start condition, a START byte, a repeated start condition, and a dummy acknowledge clock pulse. Then, as usual, the I2C transmits the slave address in SADDR.	0h
5	I2CEN	R/W	I2C Enable 0: Disable. When this bit is cleared, all status bits are set to their default values 1: Enable. It has the effect of releasing the I2C bus	0h
6	DLBEN	R/W	Digital Loopback Mode Enable 0: Disable	0h

Field	Name	R/W	Description	Reset value
			1: Enable (MSCFG=1) The transmit clock and receive clock are the same. The address transmitted on the SDA pin is the address in OADDR. In this mode, data transmitted from TXD is received by RXD via an internal path after n device cycles, where n = ((I2C Input Clock Frequency / Module Clock Frequency) × 8)	
7	REEN	R/W	Repeat Mode Enable Applicable only when I2C is the master transmitter or master receiver. The REEN, STOGEN, and STAGEN bits determine when the I2C module starts and stops data transmission. 0: Non-repeated mode. The value in I2CCNT determines the number of bytes the I2C receives/transmits 1: Repeated mode. One data byte is transmitted each time TXD is written until the STOGEN bit is manually set. The value of CNT is ignored. The RAIFLG bit/interrupt can determine when TXD is ready or when all data has been transmitted and the CPU is allowed to write the STOGEN bit.	0h
8	ADDRLEN	R/W	Address Length Configure 0: 7-bit addressing mode. I2C transmits the 7-bit slave address 1: 10-bit addressing mode. I2C transmits the 10-bit slave address	0h
9	RXTXCFG	R/W	Receive Transmit Mode Configure 0: Receive mode. I2C is a receiver, receiving data on the SDA pin 1: Transmit mode. I2C is a transmitter, transmitting data via the SDA pin	0h
10	MSCFG	R/W	Master/Slave Mode Configure This bit changes from 1 to 0 when the I2C master generates a stop condition. 0: Slave mode. I2C is the slave, receiving the serial clock from the master. 1: Master mode. I2C is the master, generating the serial clock on the SCL pin.	0h
11	STOGEN	R/W	Stop Condition Generating Applicable only when I2C is the master. This bit is not writable when I2CEN = 0. The REEN, STOGEN, and STAGEN bits determine when the I2C module starts and stops data transmission, and the STOGEN and STAGEN bits can terminate repeated mode. When in non-repeated mode, at least one byte must be transmitted before a stop condition is generated. The I2C delays clearing this bit until the STODIFLG bit is set. Clearing this bit and then initiating a new message avoids interrupting the I2C state machine. 0: Automatically cleared after a stop condition is generated 1: A stop condition is generated when the I2C internal counter counts down to 0	0h

Field	Name	R/W	Description	Reset value
12	Reserved			0h
13	STAGEN	R/W	Start Condition Generating Applicable only when I2C is the master. This bit is not writable when I2CEN = 0. The REEN, STOGEN, and STAGEN bits determine when the I2C module starts and stops data transmission, and the STOGEN and STAGEN bits can terminate repeated mode. 0: Automatically cleared after a start condition is generated 1: A start condition is generated on the I2C bus	0h
14	FREEEN	R/W	I2C Free Run Enable 0: When I2C is master: Upon breakpoint, if SCL is low, the I2C immediately halts and keeps driving SCL low. If SCL is high, the I2C module waits until SCL goes low, then stops. When I2C is slave: Upon transmit/receive completion, the breakpoint forces the I2C to stop. 1: I2C free run	0h
15	NACKCFG	R/W	NACK Mode Configure Applicable only when I2C is the receiver. 0: In master-receiver mode: The I2C transmits an ACK bit during each acknowledge cycle until the counter counts down to 0. At this time, the I2C transmits a NACK bit to the transmitter. To transmit the NACK bit earlier, the NACKCFG bit must be set. In slave-receiver mode: The I2C transmits an ACK to the transmitter during each acknowledge cycle on the bus. 1: In slave-receiver or master-receiver mode: The I2C transmits a NACK to the transmitter on the next acknowledge cycle on the bus. Once the NACK is transmitted, this bit is cleared. To transmit a NACK bit in the next acknowledge cycle, this bit must be set before the rising edge of the last data bit.	0h

36.8.11 Interrupt Source Register (I2CISRC)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	IFLG	R	Interrupt Flag If another lower-priority interrupt is pending and enabled, the value corresponding to that interrupt is loaded. Otherwise, the value remains cleared. The following interrupt events are listed in descending order of priority. A CPU read operation can clear this bit. A CPU read operation will also clear the relevant bits in the I2CSTS register in the event of arbitration loss, a no-acknowledge condition being detected, or a stop condition. 000: No selection 001: Arbitration is lost 010: No-acknowledge state detected 011: Register ready to be accessed	0h

Field	Name	R/W	Description	Reset value
			100: Receive data ready 101: Transmit data ready 110: Stop state detected 111: Addressed as a slave	
7:3	Reserved			0h
11:8	INTEST	R/W	Internal Testing Reserved bit locations shall always be written as zero.	0h
15:12	Reserved			0h

36.8.12 Extension Register (I2CEXT)

Offset address: 0x16

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	BWCCFG	R/W	Backwards Compatibility Mode Configure It will affect the timing of the TXDIFLG and TXSNEFLG bits when in slave transmitter mode.	1h
1	FWCCFG	R/W	Forward Compatibility Mode Configure When programmed, this bit only functions to request transmit data when it is needed. To avoid affecting the TXDIFLG bit, it needs to be set after I2CEN = 0. 0: Legacy function requesting transmit data when the buffer is copied to shift register or when a start condition is activated. Stale data is reused in cases of illegal starts, arbitration loss, NACK, etc. 1: New function of requesting data is active only on ACK.	0h
15:2	Reserved			0h

36.8.13 Prescaler Register (I2CPSC)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	PSC	R/W	Prescale Value It determines the I2C module clock and must be initialized during I2C reset. Module Clock Frequency = I2C Input Clock Frequency / (PSC + 1)	0h
15:8	Reserved			0h

36.8.14 Transmit FIFO Register (I2CTXFIFO)

Offset address: 0x40

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	TXFFILEVEL	R/W	Transmit FIFO Interrupt Level Set	0h

Field	Name	R/W	Description	Reset value
			Since the I2C on this device has a 16-level transmit FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. The TXFFIFLG bit is set when TXFFSTS ≤ TXFFILEVEL. If TXFFIEN = 1, an interrupt is generated.	
5	TXFFIEN	R/W	Transmit FIFO Interrupt Enable 0: Disable 1: Enable	0h
6	TXFFICLR	R/W	Transmit FIFO Interrupt Flag Clear 0: Invalid 1: Clear the TXFFIFLG bit	0h
7	TXFFIFLG	R	Transmit FIFO Interrupt Flag If TXFFIEN = 1, this bit will generate an interrupt when set. The CPU can clear this bit by writing 1 to the TXFFICLR bit. 0: Not occurred 1: Occurred	0h
12:8	TXFFSTS	R	Transmit FIFO Status xxxxx Transmit FIFO contains xxxxx bytes, and 00000 Receive FIFO is empty. Since these bits are reset to zero, when the transmit FIFO is enabled and the I2C is released from reset, the TXFFIFLG and TXFFICLR bits are set. If enabled, a transmit FIFO interrupt is generated.	0h
13	TXFFEN	R/W	Transmit FIFO Enable 0: Reset the transmit FIFO pointer to 0000 and hold the transmit FIFO in reset 1: Enable	0h
14	I2CFFEN	R/W	I2C FIFO Enable This bit must be enabled for the transmit or receive FIFO to function properly. 0: Disable 1: Enable	0h
15	Reserved			0h

36.8.15 Receive FIFO Register (I2CRXFIFO)

Offset address: 0x42

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	RXFFILEVEL	R/W	Receive FIFO Interrupt Level Set Since the I2C on this device has a 16-level transmit FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. The RXFFIFLG bit is set when RXFFSTS ≤ RXFFILEVEL. If RXFFIEN = 1, an interrupt is generated. To avoid a situation where a receive FIFO interrupt is never generated because these bits are reset to zero, the receive FIFO is enabled, and the I2C is taken out of reset, these bits	0h

Field	Name	R/W	Description	Reset value
			need to be modified before the RXFFEN bit is set, or on the same instruction.	
5	RXFFIEN	R/W	Receive FIFO Interrupt Enable 0: Disable 1: Enable	0h
6	RXFFICLR	R/W	Receive FIFO Interrupt Flag Clear 0: Invalid 1: Clear the RXFFIFLG bit	0h
7	RXFFIFLG	R	Receive FIFO Interrupt Flag If RXFFIEN = 1, this bit will generate an interrupt when set. The CPU can clear this bit by writing 1 to the RXFFICLR bit. 0: Not occurred 1: Occurred	0h
12:8	RXFFSTS	R	Receive FIFO Status xxxxx Receive FIFO contains xxxxx bytes, and 00000 Receive FIFO is empty.	0h
13	RXFFEN	R/W	Receive FIFO Enable 0: Reset the receive FIFO pointer to 0000 and hold the receive FIFO in reset 1: Enable	0h
15:14	Reserved			0h

37 Power management bus (PMBUS)

37.1 Full Name and Abbreviation Description of Terms

Table 193 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Power Management Bus	PMBUS
System Management Bus	SMBus
Packet Error Checking	PEC
End of Message	EOM

37.2 Introduction

PMBUS is a communication protocol based on SMBus, used as an interface for communication between microprocessors and devices. It is similar to I2C at the physical layer, and this module complies with the SMI Forum's PMBUS Specification Part I v1.0 and Part II v1.1.

37.3 Main characteristics

(1) Have PEC function, supporting master-slave mode and I2C mode

(2) There are four signal lines: ALERT, SCL, SDA, and CONTROL

(3) Control the signal level and timing, and resolve addresses and buffer data

(4) Support clock high-level and low-level timeout detection

(5) Have transmit and receive buffers, with a length of up to four bytes

(6) Support two different communication speeds, including standard mode (up to 100kHz) and fast mode (up to 400kHz)

(7) Can handle multiple types of messages in master-slave mode

(8) Can be configured to manually or automatically confirm the device address and command bytes

(9) Transmit EOM interrupt at the end of message transmission

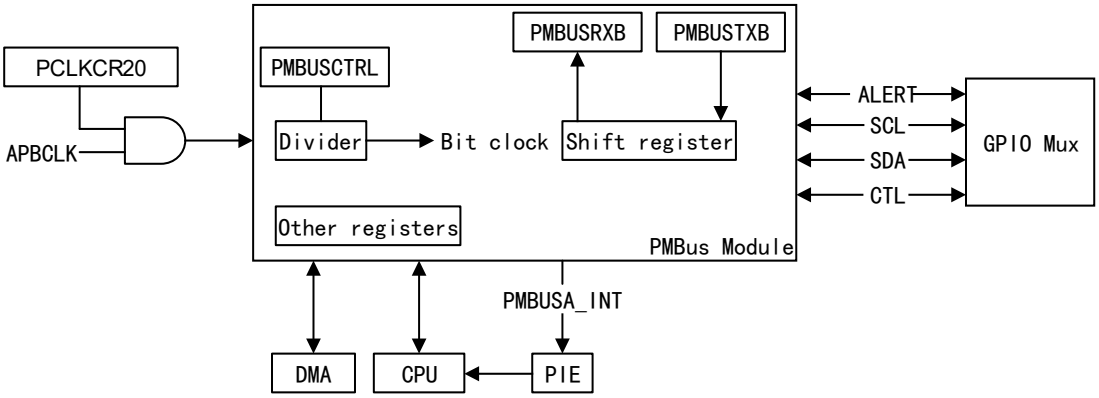
(10) Conditions for triggering a maskable interrupt:

- Low/high timeout of clock
- Bus idle
- The transmit buffer is empty
- Alarm input assertion

- Received data ready/slave address
- Transmit EOM interrupt

37.4 Structure block diagram

Figure 201 PMBUS Module Structure Block Diagram



This module is responsible for implementing the lower layers of the PMBUS communication protocol. The PMBUS module contains the following four signal lines:

Table 194 PMBUS Module Signal Lines

Signal Line	Function
ALERT	Slave output signal and master input signal, allowing the slave to issue an attention request to the master
SCL	The bus clock line, typically controlled by the master, can be temporarily pulled low by the slave when it requires more time to process data, thereby pausing the transaction.
SDA	Line used for data transmission, allowing bidirectional data flow between master and slave
CONTROL	Slave input signal that can shut down the slave or trigger an interrupt

37.5 Functional description

37.5.1 Pin and Mode Configuration

Pin configuration

If this peripheral is to be connected to the pins of the device, the GPIO pins need to be multiplexed. Some IO functions are defined by the GPIO register settings, which are independent of the current peripheral. To avoid generating burrs on the pins, the following steps shall be followed for configuration:

(1) Set the GPyGMUX register; the corresponding GPyMUX field remains at the default value of zero;

(2) Set the GPyMUX register to the target value.

To prevent the input signal from being affected by clock synchronization, configure the GPIO input qualification to asynchronous mode (set the corresponding bits in the GPxQSELn register to 0x11); to configure the internal pull-up resistor, configure the GPyPUD register. For details on GPIO multiplexing and settings, see refer to the GPIO section.

Master Mode Configuration

(1) Write the appropriate clock divider value to the PMBUSCTRL[CLKDIV] bits to generate a bit clock frequency below 10 MHz;

(2) Set the PMBUSCTRL[MASTEREN] bit and clear the PMBUSCTRL[SLAVEEN] bit to activate the master mode.

Before each transmission, the PMBUS module needs to be configured and the PMBUSMCTRL register operated as follows (during transmission, received data does not need to be manually acknowledged):

Table 195 Master Mode Configuration

Configuration	Specific Steps
Set the slave address for the current message	Set the SADDR bits to the correct slave address for the next transmission
Enable the PEC byte	Set the PECBYTEEN bit when the bus is to use PEC
Configure the bytes used for the command code	Set the BYTECFG bit to configure two bytes for the command code
Enable the use of command codes on master-initiated messages	Set the COMCEN bit to transmit a command byte at the beginning of the message transmission
Number of data bytes transmitted in the current message	Set the BYTENUM bits to determine the number of data bytes to be transmitted, excluding the block length byte (automatically generated when needed)
Enable transmitting group command messages and enable transmitting call messages	Set the GCOMEN bit to enable transmitting group command messages during a group command process, and set the TXPROMEN bit to enable transmitting call messages during a process call

Write the configured option values above to the PMBUSMCTRL register to start the transmission.

Configuration of slave mode

(1) Write the appropriate clock divider value to the PMBUSCTRL[CLKDIV] bits to generate a bit clock frequency below 10 MHz;

(2) Set the PMBUSCTRL[SLAVEEN] bit to activate slave mode.

(3) Configure the PMBUSSCTRL register to define parameters related to message reception:

Table 196 PMBUSSCTRL Register Configuration

Configuration	Specific Steps
Set the slave address and mask	Set the SLAVEEN and SMASKEN bits to configure the address and mask used when the slave receives messages
Manual slave address acknowledgment	Enable the SADDRACKEN bit: Allow software to decide whether to acknowledge an address; Disable the SADDRACKEN bit: Automatically make acknowledgment decisions based on the preset slave address and mask
PEC enable	Set the PECBYTEEN bit if using PEC
Manual command byte acknowledgment	Set the COMC bit to allow software to decide whether to acknowledge a command byte
Number of automatically acknowledged bytes	Configure the number of receive buffer bytes to be automatically acknowledged by setting the AUTOACK bits, typically set to the maximum value (unless faster detection of or non-acknowledgment of error messages is required)

(4) If the complete message length (excluding the address) exceeds 4 bytes, the received message needs to be manually acknowledged:

- For every 4-byte data packet received, manually acknowledge by writing 1 to the PMBUSACK register (necessary even in auto-acknowledge mode);
- The PMBUS module temporarily pulls the clock line low while waiting for the acknowledge signal;
- After the acknowledge signal is issued, the module further pulls the data line low and subsequently releases the clock line to transmit the acknowledge signal to the bus master;

(5) If the complete message length or the last part of the message does not exceed 4 bytes or is less than the limit set in the PMBUSSCTRL[AUTOACK] bits, the module automatically acknowledges, and there is no need to write to the PMBUSACK register:

- If a not-acknowledge signal needs to be transmitted, it can be done by writing 0 to PMBUSACK (this can only be done when the module is waiting for an acknowledgment; writing zero at any other time will unexpectedly transmit a not-acknowledge signal in the next message, causing interference).

37.5.2 Message Handling in Master Mode

37.5.2.1 Transmit Byte

The device address, a single data byte, and an optional PEC byte constitute a Transmit Byte, with the steps for a Transmit Byte message as follows:

(1) Initialize the Transmit Byte message: Load the data byte into the PMBUSTXB[7:0] bits. This data byte is transmitted to the slave;

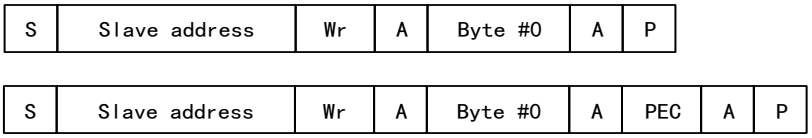
(2) Set the device address, i.e., configure the PMBUSMCTRL register. When programming the address, if the PMBUSMCTRL[PECBYTEEN] bit is asserted high, the PEC byte can be transmitted in the message;

(3) After programming the PMBUSMCTRL register, the PMBUS module transmits the Transmit Byte message;

(4) After the message is transmitted, the module issues an EOM interrupt, at which point the firmware can verify the accuracy of the message transmission;

(5) After receiving the end interrupt signal, it is necessary to verify whether the slave correctly acknowledged the transmitted data (by reading the PMBUSSTS register).

Figure 202 Transmit Byte Message with and without PEC



37.5.2.2 Receive Byte

The device address, a single data byte, and an optional PEC byte constitute a Receive Byte message, with the steps for a Receive Byte message as follows:

(1) Read the data from the slave to initialize the Receive Byte message;

(2) Program the PMBUSMCTRL register. The firmware programs the RW bit, the device address, and the optional PECBYTEEN bit. At this point, to indicate the message type, the RW bit can be set high, indicating data transmitted from the slave to the master;

(3) After programming the PMBUSMCTRL register, the PMBUS module transmits the Receive Byte message;

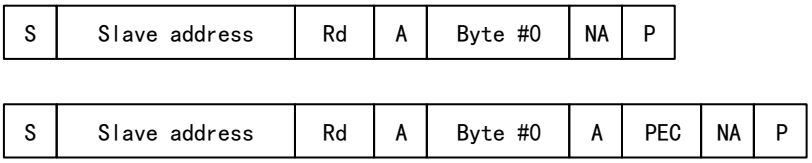
(4) After the message is transmitted, the module issues an EOM interrupt, at which point the firmware can verify the accuracy of the message;

(5) After receiving the end-of-message interrupt, it is necessary to determine whether there is data to be read from the PMBUSRXB

register and to verify that the slave address was correctly acknowledged.

The data received from the slave shall contain the correct PEC byte. If the PMBUSMCTRL[PECBYTEEN] bit is asserted during this process, the PMBUSSTS[PECVALID] bit also needs to be checked.

Figure 203 Receive Byte Message with and without PEC



37.5.2.3 Write Byte and Write Word

The device address, a command byte, the data byte(s) being transmitted, and an optional PEC byte constitute a Write Byte and Write Word, and the transmission process is similar to the Transmit Byte protocol, with the option of transmitting 1 or 2 bytes (a Write Byte message transmits only 1 byte, while a Write Word message can transmit 2 bytes). The steps for writing the Write Byte and Write Word message are as follows:

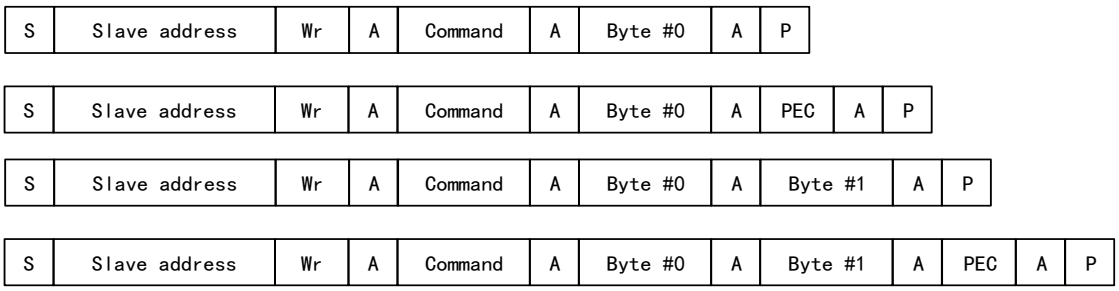
(1) Configure the PMBUSMCTRL register. To enable command byte transmission, the optional PECBYTEEN and COMCEN bits need to be set;

(2) After enabling command byte transmission, the firmware writes to the PMBUSTXB register, where bits [7:0] are written with the command byte transmitted to the slave, and bits [15:8] and [23:16] are written with the data bytes (the write format is different from the Transmit Byte protocol);

(3) After programming the PMBUSMCTRL register, the PMBUS module transmits the Write Byte/Write Word message;

(4) After the message is transmitted, the module issues an EOM interrupt, at which point the firmware can verify the accuracy of the message, and the PMBUSSTS register indicates whether the slave correctly acknowledged the message.

Figure 204 Write Byte and Write Word Message with and without PEC



37.5.2.4 Read Byte and Read Word

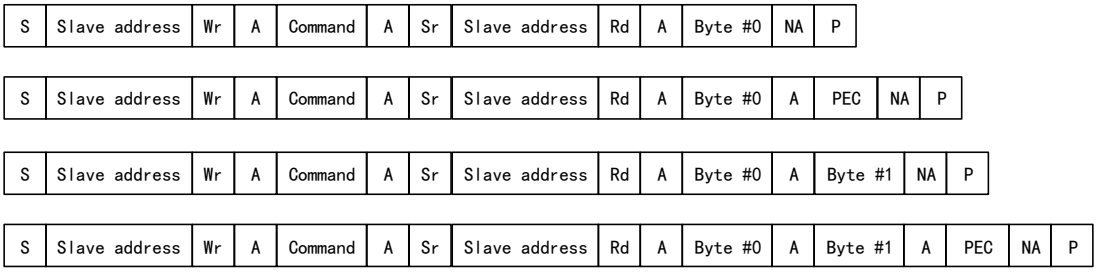
The device address, a command byte, the data byte(s) received from the slave, and an optional PEC byte constitute the Read Byte and Read Word message. The received message is returned by the slave. The reception process is similar to the Receive Byte protocol, with the option of receiving 1 or 2 bytes (a Read Byte message receives 1 byte, while a Read Word message supports receiving 2 bytes). The steps for reading the Read Byte and Read Word message are as follows:

(1) Configure the PMBUSMCTRL register, configure the PECBYTEEN bit to read the PEC byte appended to the message (can be set to receive or not receive the PEC byte), and set the COMCEN bit;

(2) After receiving the message, the slave will process it. If the slave fails to acknowledge the message correctly, or after the slave receives the predetermined number of bytes, the PMBUS module will automatically terminate the message;

In addition, the firmware needs to load the command byte into the PMBUSRXB[7:0] bits and access this register to obtain the data received from the slave.

Figure 205 Read Byte and Read Word Message with and without PEC



37.5.2.5 Quick Command

In master mode, configuring the PMBUSMCTRL register can trigger a quick command, and the target slave address needs to be written to this register. The steps to initiate a quick command are as follows:

(1) Configure the byte count to 0 bytes (write all 0s to the PMBUSMCTRL[15:8] bits);

(2) Transmit the device address. During transmission, the PMBUS module monitors whether the address is acknowledged by the slave:

- If the slave does not acknowledge the address, the PMBUS module terminates the message by enabling the PMBUSSTS[RXDFLG] bit and then triggering a stop condition on the bus;
- If the slave acknowledges the address, a data request is issued to the processor;

(3) Upon receiving the data request, force the PMBUS module to write a stop condition to the bus to terminate the message (firmware writes zero to the PMBUSACK bit).

Figure 206 Quick Command Message



37.5.2.6 Extended Command

Extended commands are to add the additional 256 command codes to the original commands. The PMBUS module supports extended commands, which can be added to the Write Byte and Write Word, Read Byte, and Read Word protocols. The steps for operating a message with an extended command are as follows:

(1) Set the PMBUSMCTRL[BYTECFG] bit to active;

(2) After the extended command is enabled, two command bytes are used when transmitting a message. The second command byte is not transmitted separately but is merged with subsequent data bytes and loaded into the PMBUSTXB[15:8] bits when programming the first part of the write data or read message;

(3) In addition to adding the extended command, the Write Byte and Write Word need to include the slave address and a repeated start condition. Other operations are the same as Transmit Byte;

The PMBUS module adjusts the protocol format based on the setting of the BYTECFG bit and automatically handles the extra bytes in Write Byte and Write Word (no firmware support is required). .

Figure 207 Extended Command Read Byte and Read Word Message with and without PEC

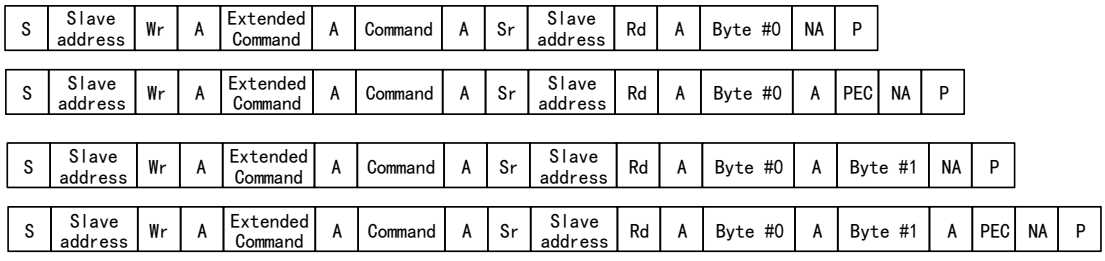
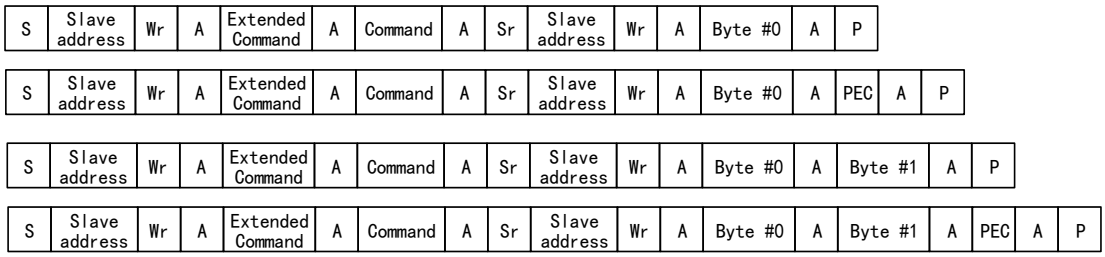


Figure 208 Extended Command Write Byte and Write Word Message with and without PEC



37.5.2.7 **Group Command**

In the group command protocol, commands for multiple devices can be transmitted in the same message. The steps for initiating a group command are as follows:

(1) Write the message to PMBUSMCTRL. When the slave address of the first device is written in the message, group command mode can be enabled (set the PMBUSMCTRL[GCOMEN] bit);

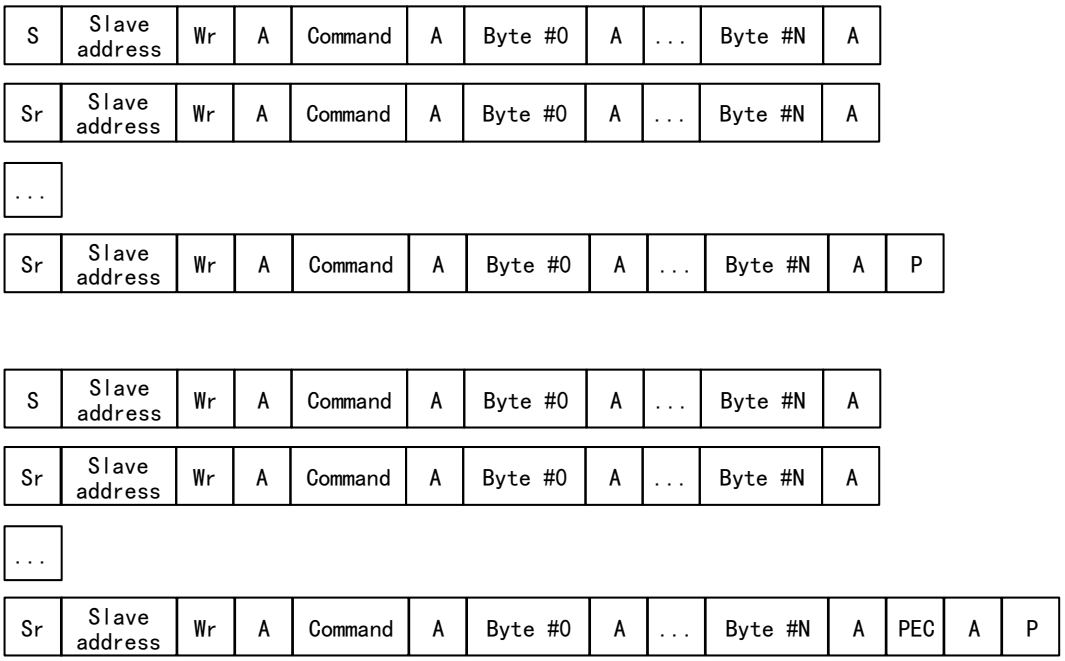
(2) The remaining part of the message is written sequentially, following the Write Byte/Write Word message format;

(3) In the group command protocol, the message is divided into multiple parts, each corresponding to a command for a device. When the command for the first device ends, the firmware simply writes the next device address, triggering the PMBUS module to transmit a repeated start condition on the bus and begin the next part of the message.

(4) When the firmware writes the last device address of the message in the GCOMEN register, it indicates the end of the group command. At this point, disable the GCOMEN bit before writing to the PMBUSMCTRL register.

Note: When devices on the bus receive a group command message, they check whether the end of the message is a stop condition. If a stop condition is detected at the end of the message, the device executes the received command without waiting for other devices' commands to complete.

Figure 209 Group Command Message with and without PEC



37.5.2.8 **Process Call**

The Write Word message and the Read Word message constitute the process call protocol, which allows writing data and then immediately reading data in a single communication. Process call has an optional function: When the slave transmits read data to the master, a PEC byte can be appended after the data to verify the accuracy of the received data. The steps for executing the process call protocol are as follows:

(1) Load the command byte into the PMBUSTXB[7:0] bits and the data byte into bits 23:8 to complete the Write Word message;

(2) Write to the PMBUSMCTRL register and set the TXPROMEN bit to enable. The process call message begins transmission. At this time, the PMBUS module automatically generates a repeated start condition and initiates the Read Word message part;

(3) To determine if the message is valid, the firmware waits for the EOM interrupt after receiving the acknowledgment and data from the slave;

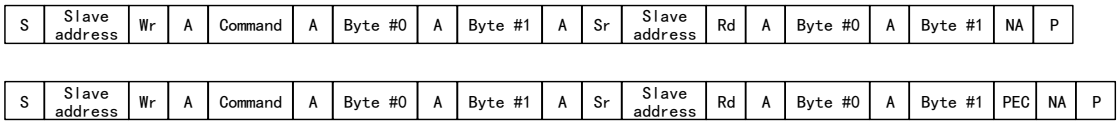
(4) Upon receiving the EOM, the PMBUSSTS register can be checked to see the slave's acknowledgment status for the transmitted data. It can also indicate whether the two bytes of the Read Word portion were received. If it is necessary to determine whether the PEC byte received from the slave in the Read Word portion is correct, PEC processing needs to be enabled first, and the PMBUSSTS[PECVALID] bit shall be checked;

(5) If the next operation to be performed is not a process call message, the PMBUSMCTRL[TXPROMEN] bit needs to be disabled;

(6) Repeat the above process to execute the next message or other operations.

Note: Since writing any value to the PMBUSMCTRL register triggers a message, do not reconfigure the master before the firmware transmits a new message.

Figure 210 Process Call Message with and without PEC



37.5.2.9 **Block Write**

The Block Write protocol is used to transmit messages exceeding two data bytes and can append a PEC byte at the end of the slave data. This protocol is similar in structure to the Write Word protocol but includes an additional feature: the first data byte after the command byte specifies the length of the subsequent data block. The block length indicates how many data bytes the slave will receive after the command, excluding the command byte itself and the first data byte of the block length. The steps for Block Write are as follows:

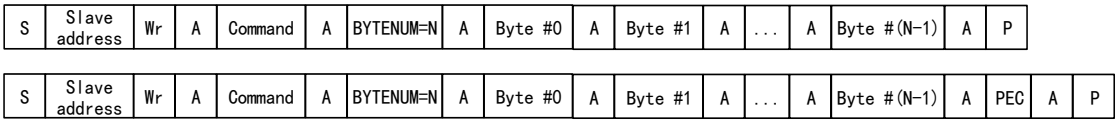
(1) Initiate a Block Write message. Write the block length to the PMBUSMCTRL[BYTENUM] bits. When the number of data bytes the firmware wants to transmit exceeds two, the PMBUS module automatically inserts the block length into the message;

(2) Load the initial write data into the PMBUSTXB register, where bits [7:0] are used to store the command byte, followed by the block length, and the remaining three bytes are used to store the first three data bytes after the block length;

(3) Write to the PMBUSMCTRL register to transmit the Block Write message: If the block length exceeds 3 bytes, the PMBUS module provides a data request interrupt, requesting the firmware to provide additional data bytes. The firmware shall follow these rules:
- If the message is less than 4 bytes, write only the required data bytes to the PMBUSTXB register;
- If the message exceeds 4 bytes, write using all 4 bytes of the PMBUSTXB register;

(4) After the message is complete, the PMBUS module issues an EOM interrupt and verifies whether the slave accepted the written data block (check the PMBUSSTS register).

Figure 211 Block Write Message with and without PEC



37.5.2.10 Block Read

The Block Read protocol can read more than two bytes of data from a slave device. If PEC processing is enabled, the slave can append a PEC byte at the end of the message. The structure of the Block Read protocol is similar to the Read Word protocol. The first data byte transmitted by the slave indicates the length of the data being written by the slave. The steps for Block Read are as follows:

(1) Write the command byte to be transmitted to the slave into the PMBUSTXB[7:0] bits;

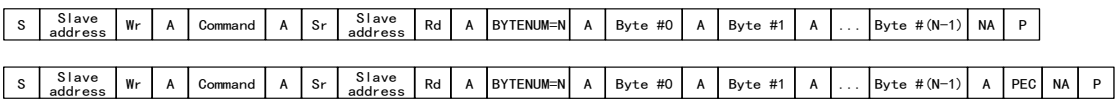
(2) Write the block length to the PMBUSMCTRL[BYTENUM] bits to initiate the Block Read message. The block length count does not include the command byte, any slave address, or the block length byte in the message;

(3) Transmit the Block Read message. The module is configured to trigger an interrupt after receiving 4 data bytes:

- If the block length is 3, the EOM interrupt and the data ready interrupt are received simultaneously;
- If the block length exceeds 3, only the data ready interrupt is asserted because the maximum amount of data the firmware can read at a time is 4 bytes;

(4) After the EOM interrupt is detected, the firmware can verify the received PEC. The PMBUSRXB register may store fewer than 4 bytes. By reading the value of the PMBUSSTS[RXBYTE] bits, the firmware can determine the actual number of bytes available in the last data transfer.

Figure 212 Block Read Message with and without PEC



37.5.2.11 Block Write - Block Read Process Call

The Block Write and Block Read protocols constitute the Block Write - Block Read process call protocol, which allows writing data and then immediately reading data in a single communication. Block Write - The steps for a Block Write-Block Read process call are as follows:

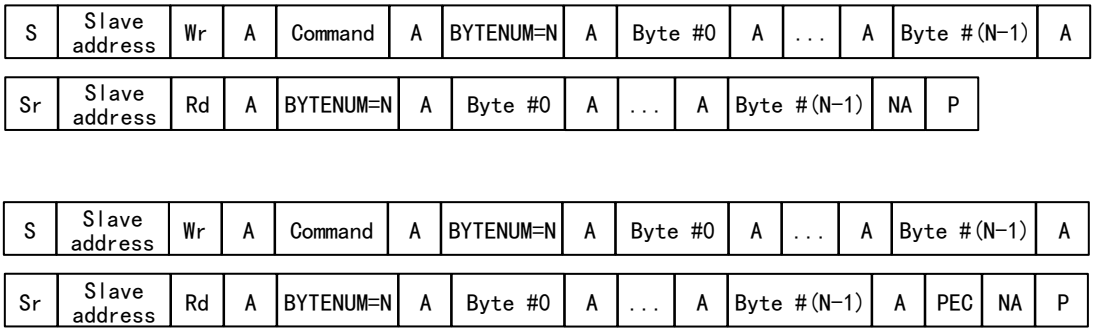
(1) The master performs a Block Write operation, setting the PMBUSMCTRL[BYTENUM] bits to specify the length of the data block to be written;

(2) Enable the PMBUSMCTRL[TXPROMEN] bit to transmit the data;

(3) After completing the Block Write portion of the message, the PMBUS module automatically issues a repeated start condition on the bus;

(4) After the repeated start condition, the PMBUS module begins transmitting the Block Read message, operating identically to the Block Read message, allowing the master to read data from the slave device.

Figure 213 Block Write with or without PEC - Block Read Process Call Message



37.5.2.12 Alert Response

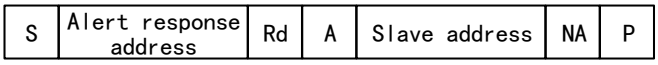
In master mode, the alert response message is used when the master detects an alert condition from a slave. This message is essentially a Receive Byte message, with the difference that it has a reserved address specifically for alert responses, setting the slave device address to 0xC and disabling the PEC function. The steps for alert response are as follows:

(1) When the PMBUS module detects an alert condition at its input, it notifies the firmware via an interrupt signal, indicating that a slave has asserted an alert condition, meaning that the slave needs to communicate with the master;

(2) To initiate the alert response message, the firmware needs to program the PMBUSMCTRL register, setting the alert response address and specifying the slave address requesting service;

(3) After receiving the EOM interrupt, the firmware can read the address of the slave that transmitted the alert from the PMBUSRXB register.

Figure 214 Alert Response Message



37.5.3 Message Handling in Slave Mode

In slave mode, if the most efficient operating mode with automatic address and command acknowledgment is enabled, and the PEC function is enabled, the type of message received can be determined based on the flag bits:

- If automatic command acknowledgment is enabled, the DISDRDY bit is set to 1;
- If automatic address acknowledgment is disabled, all messages require manually setting the DISSADDRRDY bit to 1 to start message transmission. In this mode, there is one dedicated command each for reading and writing.

If the message has no PEC, the length of its effective data is reduced by one byte. For example, a quick command with PEC has one byte, while a quick command without PEC has no bytes.

Note:

(1) The corresponding bits in the PMBUSSTS register are set only under specific conditions (i.e., receiving a stop signal, the receive buffer being full, or a fault occurring);

(2) When a quick command is received, writing 1 to the PMBUSACK register acknowledges the message;

(3) When receiving data, the number of bytes received is not updated in real-time.

37.5.3.1 Transmit Byte

The device address, a single data byte, and an optional PEC byte constitute the Transmit Byte message. To acknowledge the PEC byte, PEC processing must be enabled in the PMBUSSCTRL register, where the PMBUSSTS[PECVALID] bit indicates whether the received PEC byte is valid. There are two acknowledgment modes:

- Automatic Address Acknowledgment Mode: The module automatically acknowledges data bytes and optional PEC bytes without firmware intervention. After receiving the Transmit Byte message, the module generates an EOM interrupt, reads, and sets PMBUSSTS[READDATA].
- Manual Mode: The firmware acknowledges the address, and the module acknowledges the remaining data bytes and PEC byte.

The following are the steps for the PMBUS module to handle the Transmit Byte message:

(1) Before acknowledging, read the data byte from the PMBUSRXB register (the first received data byte is stored in the PMBUSRXB[7:0] bits);

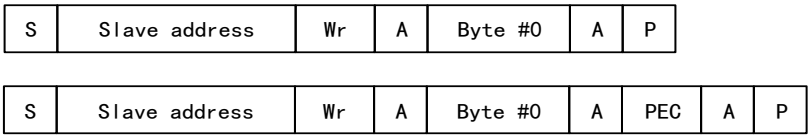
(2) Write 1 to the PMBUSACK register to receive the Transmit Byte message;

(3) When the Transmit Byte message is received, set the PMBUSSTS[READDATA] bit and EOM. If PEC is received and valid, the PMBUSSTS[PECVALID] bit is also set;

(4) Read the PMBUSSTS[RXBYTE] bits:

- If PEC processing is not enabled, PMBUSSTS[RXBYTE] will indicate that one byte was received;
- If PEC processing is enabled, PMBUSSTS[RXBYTE] indicates that two bytes were received (data byte and PEC byte) when the value is 2. The PEC byte is stored in the PMBUSRXB[15:8] bits.

Figure 215 Transmit Byte Message with and without PEC



37.5.3.2 Receive Byte

The device address, a single data byte, and an optional PEC byte constitute the Receive Byte message. Under the automatic address acknowledgment module, the steps for handling a Receive Byte message are as follows:

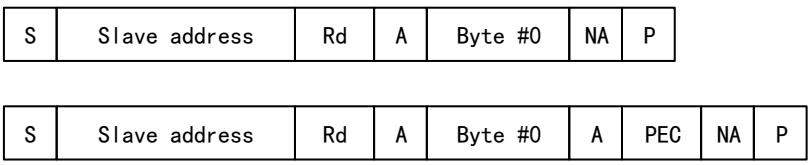
(1) Detect the slave address and automatically acknowledge the address to the master;

(2) The firmware receives a data request interrupt, and the data byte to be transmitted to the master is stored in the PMBUSTXB[7:0] bits;

(3) Set the PMBUSSCTRL[VALBYTESEL] bit to 1 to transmit one byte. If PEC processing is enabled, the firmware also sets the PMBUSSCTRL[TXPEC] and PMBUSSCTRL[PECPROEN] bits;

(4) If PEC is enabled, the slave automatically calculates and adds the PEC byte after transmitting the data byte.

Figure 216 Receive Byte Message with and without PEC



37.5.3.3 Write Byte and Write Word

The slave address, command word, data byte(s), and optional PEC byte constitute the Write Byte and Write Word messages, with the former containing one data byte, and the latter containing two data bytes.

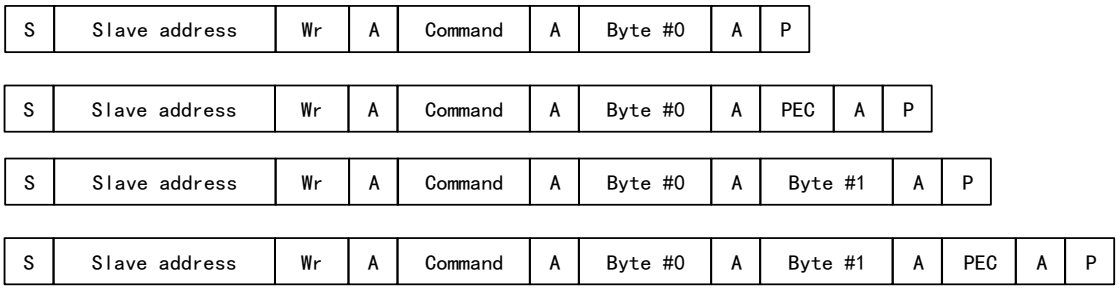
In automatic address acknowledgment mode, the module automatically

acknowledges data bytes and optional PEC bytes (without firmware intervention). The PMBUSSCTRL register needs to be configured to acknowledge the command word. The steps for handling Write Byte and Write Word messages are as follows:

(1) For Write Byte messages and Write Word messages without PEC:
- When the value of the PMBUSSTS[RXBYTE] bit is 3, it indicates that 3 bytes have been received (command byte, data byte, and PEC byte);
- The transmission ends, and the firmware receives an EOM interrupt. At this time, read and set the PMBUSSTS[READDATA] bit;
- Read the data from the PMBUSRXB register. The firmware does not need to transmit an acknowledgment to the master.

(2) For Write Word messages with a PEC byte:
- When the value of the PMBUSSTS[RXBYTE] bit is 4, it indicates that 4 bytes have been received (command byte, 2 data bytes, and PEC byte);
- The transmission ends, and the PMBUSSTS[READDATA] interrupt is enabled;
- The firmware reads the data from the PMBUSRXB register and writes to the PMBUSACK register to acknowledge the master;
- The PMBUS module keeps SCL low until the firmware responds to the received data.

Figure 217 Write Byte and Write Word Message with and without PEC



37.5.3.4 **Read Byte and Read Word**

The slave address, command word, data byte(s) received from the slave, and optional PEC byte constitute the Read Byte and Read Word messages. The steps for responding to Read Byte and Read Word messages are as follows:

(1) Configure the PMBUSSCTRL register to acknowledge the slave address and command word;

(2) After receiving the command byte, store it in the PMBUSRXB[7:0] bits;

(3) In automatic mode, after receiving the repeated start condition and slave address, the PMBUS module provides data ready and data request interrupts.

- Set the PMBUSSTS[RXBYTE] bit to 1 to trigger the data ready interrupt;

However, at this point, the module cannot distinguish the incoming group command Transmit Byte message from the Read Byte/Read Word message:

- If a read command is transmitted using the same device address, asserting the data request bit indicates that the slave shall transmit data to the master;
- If data has already been written to the PMBUSTXB register before the device address is received, the data request bit is not asserted. If the firmware wants to be able to respond to Transmit Byte, it shall also set the PMBUSSTS[RXBYTE] bit to 1 after receiving the address to prepare for reading data;

(4) After transmitting data to the master, wait and see whether the next event is EOM or a data request:

- If the event is EOM, process it as a Transmit Byte message;
- If the event is a data request, process it as a read command, which could be a Read Byte, Word, or Block;

(5) After identifying it as a read command, the firmware needs to respond: write the data to the PMBUSTXB register and correctly set the PMBUSSCTRL[VALBYTESEL] and PEC bits in the PMBUSSCTRL register (if PEC processing is enabled);

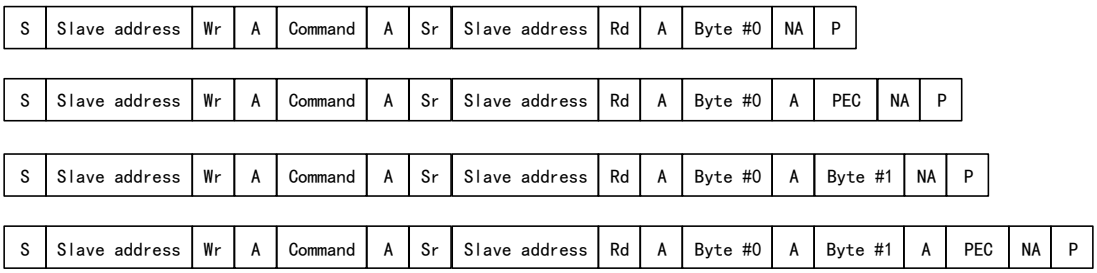
(6) For Read Byte, set the transmit byte count to 1. If a PEC byte is to be transmitted, set the PMBUSSCTRL[TXPEC] bit;

(7) Write all bytes to PMBUSTXB simultaneously to start transmission;

(8) After the master receives the correct number of bytes in the message, the Read Byte message is complete. At this point, the master will not acknowledge the last byte but will set the PMBUSSTS[ENDFLG] bit to notify the firmware;

Note: The data ready and data request bits may change within the polling interval. The firmware design shall handle this situation appropriately.

Figure 218 Read Byte and Read Word Message with and without PEC

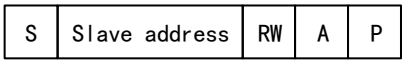


37.5.3.5 Quick Command

In slave mode, when the PMBUS module receives a quick command, it acknowledges the received device address based on different modes:

- In automatic address acknowledgment mode, it automatically acknowledges the device address that transmitted the command without firmware intervention. After receiving the EOM, the firmware can optionally read the PMBUSHSADDR register, which contains the received address information;
- In manual address acknowledgment mode, the firmware needs to write to the PMBUSACK register to acknowledge the address.

Figure 219 Quick Command Message



37.5.3.6 Extended Command

Extended commands are to add the additional 256 command codes to the original commands. The data byte and two command bytes in extended commands are stored in the PMBUSRXB[15:0] bits. The PMBUS module uses the PMBUSSTS[RXRESFLG] and [RXBYTE] bits to process extended command messages.

(1) The steps for handling extended command Write Byte and Write Word messages are as follows:

- Store the two command bytes in the PMBUSRXB[15:0] bits;
- The first command byte (the first byte in the command code) must contain a specific command extension code, which indicates that the extended command protocol is being used;
- Once the device address is re-transmitted, the PMBUSSTS[RXRESFLG] bit is set, indicating the start of an extended command;
- The PMBUSSTS[RXBYTE] bits are used to distinguish the type of write operation:
  - If the value is 3, it represents an extended command Write Byte message;
  - If the value is 4, it represents an extended command Write Word message.

(2) The steps for handling extended command Read Byte and Read Word messages are as follows:

- When the module receives the device address again, it triggers data ready and data request interrupts, preparing to transmit data to the master;
- The two command bytes can be found in the PMBUSRXB[15:0] bits;

- The first command byte will be matched against the command extension code to ensure it is an extended command;
- To complete the read operation and return data to the master, the firmware must load the data to be transmitted.

Figure 220 Extended Command Write Byte and Write Word Message with and without PEC

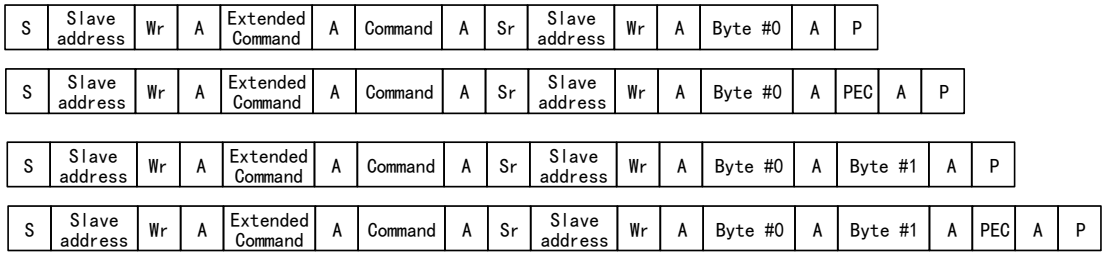
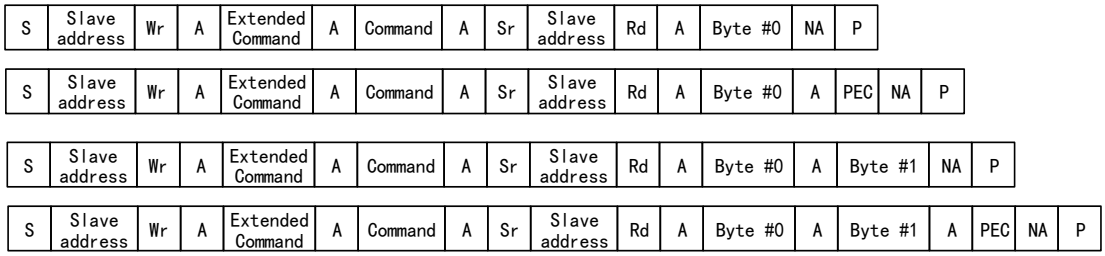


Figure 221 Extended Command Read Byte and Read Word Message with and without PEC



37.5.3.7 Group Command

The group command protocol of the PMBUS module allows the user to transmit one message and issue instructions to multiple devices simultaneously. The steps for initiating a group command are as follows:

(1) When devices detect a stop condition at the end of a group command message, they will execute the received instructions collaboratively;

(2) After the address and command are acknowledged, if the module detects 4 data bytes or a repeated start signal on the bus, it generates a data ready interrupt;

(3) After receiving the repeated start condition, set the data ready bit. The data is read into memory;

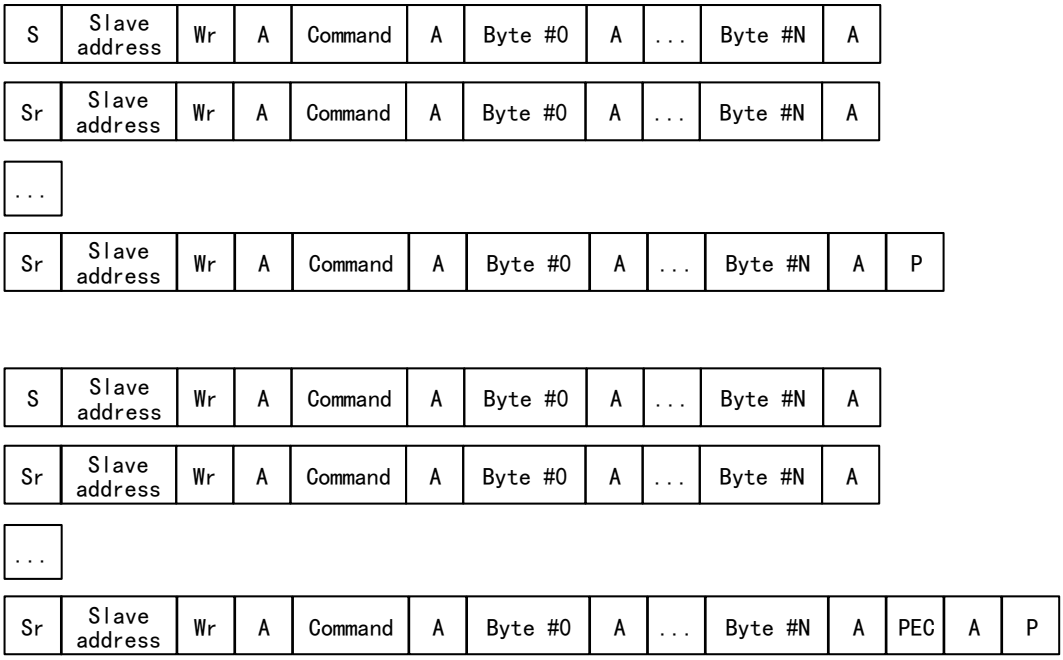
(4) Wait for the EOM interrupt, as all messages need to be fully received. This is a crucial step in executing group commands;

(5) After the EOM bit is set, the firmware processes the received data.

Waiting for the EOM interrupt is the only difference between group commands and other write messages. After receiving a group command, the remaining

processing flow of the slave firmware is the same as handling any other write message.

Figure 222 Group Command Message with and without PEC



37.5.3.8 **Process Call**

The Write Word message and the Read Word message constitute the process call protocol, which allows writing data and then immediately reading data in a single communication. The steps for executing the process call protocol are as follows:

(1) Configure the PMBUSSCTRL register to acknowledge the address and command;

(2) In automatic mode, after receiving the repeated start condition and slave address, the PMBUS module provides data ready and data request interrupts;

(3) PMBUSSTS[RXRESFLG] is set, indicating that the Write Word message of the process call has been received;

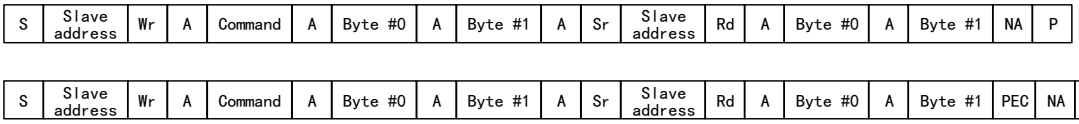
(4) The received command byte is stored in the PMBUSRXB[7:0] bits, and the two data bytes received from the master are stored in the PMBUSRXB[23:8] bits;

(5) The master initiates a read request by transmitting another repeated start signal followed by the slave address;

(6) Upon receiving the read request, the firmware writes 2 data bytes to PMBUSTXB, which are ready to be transmitted back to the master. If PEC processing is enabled, the PMBUSSCTRL[TXPEC] bit is set;

(7) The EOM interrupt is triggered, completing the Read Word portion.

Figure 223 Process Call Message with and without PEC



37.5.3.9 **Block Write**

The Block Write protocol can transmit more than 2 data bytes and is similar in structure to the Write Word protocol. The steps for executing the Block Write protocol are as follows:

(1) After receiving the command byte, block length, and 2 data bytes, the PMBUS module triggers a data ready interrupt, prompting the firmware to start processing the received data;

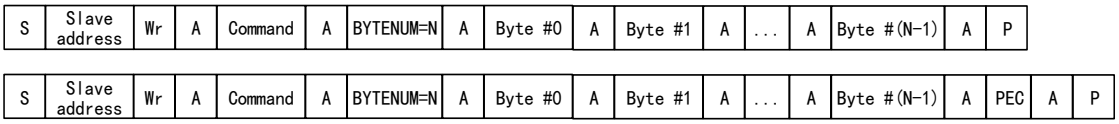
(2) While waiting for the firmware to read and process the received data, the module pauses the bus by pulling the clock line low and programs the corresponding acknowledge register. The bus resumes operation only when the firmware has finished processing and is ready to receive new data;

(3) During message transmission, the number of bytes received is stored in the PMBUSSTS register. When a cumulative total of 4 data bytes is received, the module triggers the data ready interrupt again;

(4) When the message transmission is complete, the last group of data bytes, which may be less than four, is stored in the PMBUSRXB register;

(5) The firmware checks the PMBUSSTS[PECVALID] bit to determine whether the received PEC value is correct.

Figure 224 Block Write Message with and without PEC



37.5.3.10 **Block Read**

The Block Read protocol is used to transmit messages exceeding two data bytes and structurally similar to Read Word. The steps for executing the Block Read protocol are as follows:

(1) After receiving the repeated start condition and slave address, the PMBUS module generates data ready and data request interrupts. At this time, the command byte transmitted by the master is stored in the PMBUSRXB[7:0] bits;

(2) The PMBUS module keeps the SCL line pulled low to pause communication until the firmware is ready to transmit data;

(3) The firmware writes data bytes to the PMBUSTXB register (the first write needs to set the PMBUSTXB[7:0] bits to specify the data block length);

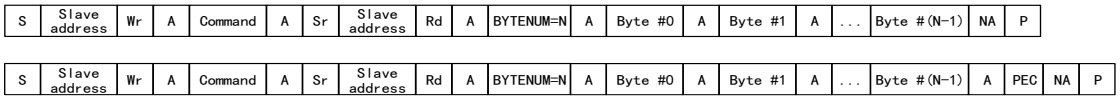
(4) During the first data transmission, the PMBUSSCTRL[VALBYTESEL] bit is set to 4, indicating that the slave will transmit four bytes of data, but the [TXPEC] bit is not set, indicating that the PEC byte is not included in this transmission;

(5) After transmitting 4 bytes, the firmware waits for a data request, rather than EOM, indicating that the master is ready to receive more data. At this time, the module generates a data request interrupt again and pulls SCL low again until the firmware has written more data into the PMBUSTXB register;

(6) When transmitting the last four or fewer data bytes, the firmware writes the exact number of bytes into the PMBUSSCTRL[VALBYTESEL] bits and writes these bytes into the PMBUSTXB register. At this time, the firmware sets the [TXPEC] bit, preparing to transmit the PEC byte;

(7) The PMBUS module outputs these data bytes to the bus, followed by the PEC byte. If the master does not acknowledge after receiving the PEC, the slave sets the EOM bit.

Figure 225 Block Read Message with and without PEC



37.5.3.11 Block Write - Block Read Process Call

The Block-Write and Block Read protocols combine to constitute the Block Write-Block Read process call protocol, allowing writing a block of data and then immediately reading a block of data in a single communication. This protocol's message handling mode is similar to the working mode of the process call message. The steps for a Block Write-Block Read process call are as follows:

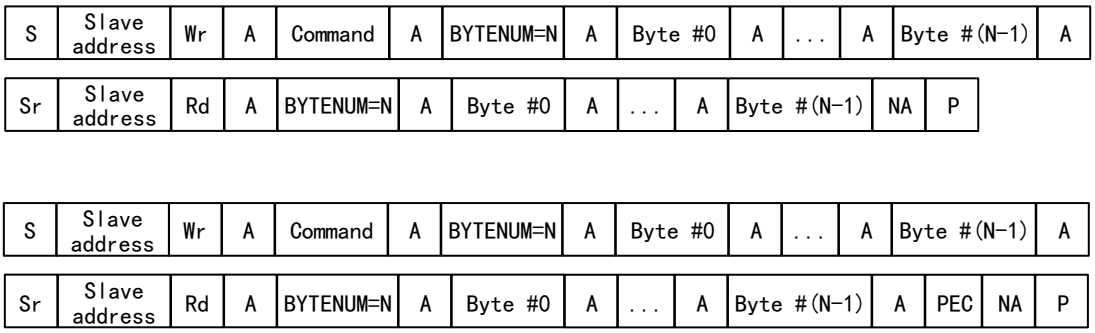
(1) Configure the PMBUSSCTRL register to acknowledge the address and command;

(2) The PMBUS module generates a data ready interrupt when it detects 4 data bytes or a repeated start condition;

(3) After receiving the repeated start condition, the firmware needs to prepare the data and load it into the transmit buffer to be transmitted to the master;

(4) When preparing to transmit data, the firmware must set the length of the data block to be transmitted in the PMBUSTXB[7:0] bits.

Figure 226 Block Write with or without PEC - Block Read Process Call Message



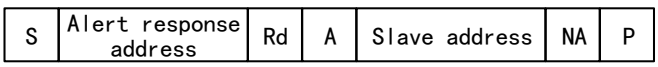
37.5.3.12 Alert Response

When the master on the PMBUS detects an alert status from the slave, it triggers an alert response message. The method of handling the message depends on whether the PMBUS module is in automatic or manual address acknowledgment mode:

(1) Automatic Address Acknowledgment Mode:
- After detecting the alert response address, the PMBUS module transmits an acknowledgment signal to the master;
- It will transmit the slave address programmed within the PMBUSSCTRL register;
- The module responds to the alert response message provided that the PMBUSCTRL[ALERTLOW] bit has been set;
- After processing the alert response message, the module clears the alert status and disables the relevant enable bit within the PMBUSCTRL register.

(2) Manual Address Acknowledgment Mode:
- The firmware reads the address already received from the PMBUSRXB register;
- It will transmit the target slave address to the master;
- To respond to the alert response message initiated by the master, the firmware must also reconfigure the PMBUSCTRL register to clear the ALERTLOW bit.

Figure 227 Alert Response Message



37.5.4 DMA Support Scheme

As the R502's DMA does not feature handshaking signals such as 'request' and 'ack', it relies solely on unidirectional trigger signals sent from the peripheral to

the DMA. The PMBUS DMA trigger signal can be understood by referring to the logic behind the generation of the PMBUS interrupt.

The logic for generating the PMBUS TX interrupt is primarily determined by the DATA_REQUEST flag. DATA_REQUEST is the data request flag; when an address has been transmitted and the PMBTXBUF is empty, the DATA_REQUEST flag is set to high, causing the TX interrupt signal to go high. The TX interrupt signal is in pulse form. The DATA_REQUEST flag can be used as the DMA trigger pulse.

The generation logic for the PMBUS RX interrupt is primarily determined by the DATA_READY flag. DATA_READY is the receive buffer register full flag, indicating that the PMBUS interface's read buffer is full and that the firmware must read the data before further bus activity can take place; when the DATA_READY flag is set to high, the TX interrupt signal is raised.The RX interrupt signal is in pulse form. The DATA_READY flag can be used as a trigger pulse for DMA.

37.6 Register bank address

Table 197 Register Bank Address

Device register	Register bank	Start address	End address
PmbusaRegs	PMBUS_REGS	0x4000_2000	0x4000_23FF

37.7 Register address mapping

Table 198 PMBUS_REGS Offset Addresses

Register name	Register description	Offset address	WRPRT
PMBUSMCTRL	Master mode control register	0x00	√
PMBUSTXB	Transmit buffer register	0x04	-
PMBUSRXB	Receive buffer register	0x08	-
PMBUSACK	Acknowledge Register	0x0C	-
PMBUSSTS	Status register	0x10	-
PMBUSIMASK	Interrupt mask register	0x14	√
PMBUSSCTRL	Slave mode control register	0x18	√
PMBUSHSADDR	Keep slave address register	0x1C	-
PMBUSCTRL	Control register	0x20	√
PMBUSTIMCTRL	Timing control register	0x24	√
PMBUSCLKTIM	Clock timing register	0x28	√
PMBUSSTATIM	Start Setup Time Register	0x2C	√

Register name	Register description	Offset address	WRPRT
PMBUSBUSIDTIM	Bus Idle Time Register	0x30	√
PMBUSCLKLTOTIIM	Clock Low Level Timeout Value Register	0x34	√
PMBUSCLKHTOTIIM	Clock High Level Timeout Value Register	0x38	√

37.8 Register functional description

37.8.1 Master mode control register (PMBUSMCTRL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	RWCFG	R/W	Message Read and Write Configure 0: Write message, the master transmits data to the slave 1: Read message, the slave transmits data to the master	0h
7:1	SADDR	R/W	Current Message Slave Address	0h
15:8	BYTENUM	R/W	Number of Bytes of Data Transferred in the Current Message The device address, command byte, or block length in a block message are not included in the data byte count. When performing a byte write operation through the PMBus interface, a maximum of 255 bytes of data can be transmitted in a single message. In a block message, based on the number of data bytes transmitted and set, the PMBus interface automatically inserts the corresponding block length into the message. The firmware only needs to load the transmission address, command byte, and data.	0h
16	COMCEN	R/W	Use of Command Code on Master Initiated Messages Enable 0: Disable 1: Enable	0h
17	BYTECFG	R/W	Use bytes for Command Code Configure 0: 1 byte 1: 2 bytes	0h
18	PECBYTEEN	R/W	PEC Byte Enables 0: Disable PEC processing 1: Enable PEC transmission/reception	0h
19	GCOMEN	R/W	Transmit of Group Command Message Enable 0: Default message state (except for group command messages) 1: Enable transmitting group command messages	0h
20	TXPROMEN	R/W	Transmit Process Call Message Enable 0: Default message state (except for process call messages) 1: Enable transmitting process call messages	0h

Field	Name	R/W	Description	Reset value
31:21	Reserved			0h

37.8.2 Transmit Buffer Register (PMBUSTXB)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	TXB	R/W	Transmit Buffer [7:0]: Byte 0 (first data byte transmitted from the transmit buffer) [15:8]: Byte 1 (second data byte transmitted from the transmit buffer) [23:16]: Byte 2 (third data byte transmitted from the transmit buffer) [31:24]: Byte 3 (last data byte transmitted from the transmit buffer)	0h

37.8.3 Receive Buffer Register (PMBUSRXB)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
31:0	RXB	R	Receive Buffer [7:0]: Byte 0 (first data byte received in the receive buffer) [15:8]: Byte 1 (second data byte received in the receive buffer) [23:16]: Byte 2 (third data byte received in the receive buffer) [31:24]: Byte 3 (last data byte received in the receive buffer)	0h

37.8.4 Acknowledge Register (PMBUSACK)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ACK	R/W	Acknowledge Received Data 0: Received data not replied 1: Acknowledge data reception (this bit is cleared when PMBus issues an acknowledge signal)	0h
31:1	Reserved			0h

37.8.5 Status register (PMBUSSTS)

Offset address: 0x10

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	RXBYTE	RC	Receive Byte Select 0: No data received	0h

Field	Name	R/W	Description	Reset value
			1: 1 byte of data received (data is in LIN_RXB register bits [7:0]) 2: 2 bytes of data received (data is in LIN_RXB register bits [15:0]) 3: 3 bytes of data received (data is in LIN_RXB register bits [23:0]) 4: 4 bytes of data received (data is in LIN_RXB register bits [31:0])	
3	READDATA	RC	Read Data Prior to Bus Activity 0: No data available for the processor to read 1: Read data because the PMBus interface read buffer is full. To wait for the firmware to read the data, the PMBus interface enables the clock stretching function to pause other activities on the bus	0h
4	ADITDATA	RC	Request Additional Data 0: No additional data required 1: Additional data requested. The PMBus interface enables the clock stretching function to pause other activities on the bus	0h
5	ENDFLG	RC	Current Message End Flag 0: PMBus is idle or transmitting a message 1: Message transmission complete	0h
6	RXDFLG	RC	Receive Data Flag 0: Received 1: Not received	0h
7	PECVALID	RC	Received PEC is Valid 0: Invalid (provided that EOM is asserted) 1: Valid Note: This bit is only valid after EOM. Ignore the status of this bit during message transmission.	0h
8	CLKLTOFLG	RC	Clock Low Timeout Flag 0: Clock low timeout flag not detected 1: Clock low timeout flag detected, and the clock low timeout hold time is greater than 35ms	0h
9	CLKHFLG	RC	Clock High Flag 0: Clock high flag not detected 1: During message transmission, the clock high exceeded 50μs	0h
10	RDYRSADDR	RC	Ready Read Slave Address 0: No slave address to read 1: Ready to read the slave address from LIN_RXB register bits 6:0	0h
11	RXRESFLG	RC	Received Repeated Star Flag 0: Not received 1: Received	0h
12	BUSYFLG	RC	Busy Flag	0h

Field	Name	R/W	Description	Reset value
			0: PMBus interface is idle, ready to transmit/receive messages 1: PMBus interface is busy, processing the current message	
13	AVAFLG	RC	Available Flag 0: Not available, PMBus is processing the current message 1: Available to process a new message	0h
14	LOSTCTRL	RC	Master Lost Control of PMBus 0: Master can control the PMBus 1: Master arbitration failed, cannot control the PMBus	0h
15	MASTERFLG	RC	Master Mode Flag 0: In slave mode or idle mode 1: In master mode	0h
16	ALERTTRAN	RC	Alert Pin Transitioned 0: No conversion 1: Converted, asserted by another device on the PMBus	0h
17	CTRLTRAN	RC	Control Pin Transitioned 0: No conversion 1: Converted, asserted by another device on the PMBus	0h
18	ALERTOLL	R	Alert Pin at Logic Level High 0: This pin is observed at a low level 1: This pin is observed at a high level	1h
19	CTRLOLL	R	Control Pin at Logic Level High 0: This pin is observed at a low logic level 1: This pin is observed at a high logic level	0h
20	SDAOLL	R	Data Pin at Logic Level High 0: This pin is observed at a low logic level 1: This pin is observed at a high logic level	1h
21	SCLOLL	R	Clock Pin at Logic Level High 0: This pin is observed at a low logic level 1: This pin is observed at a high logic level	1h
31:22	Reserved			0h

37.8.6 Interrupt mask register (PMBUSIMASK)

Offset address: 0x14

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	DISBUSIDLE	R/W	Disables Bus Idle Generation Interrupt 0: Generate an interrupt when the bus idle is asserted	1h

Field	Name	R/W	Description	Reset value
			1: Disable interrupt generation when the bus idle is asserted	
1	DISCLKLTO	R/W	Disables Clock Low Timeout Generation Interrupt 0: Generate an interrupt when the clock low timeout flag is asserted 1: Disable interrupt generation when the clock low timeout flag is asserted	1h
2	DISDRDY	R/W	Disables Data Ready Generation Interrupt 0: Generate an interrupt when data ready is asserted 1: Disable interrupt generation when data ready is asserted	1h
3	DISDREQ	R/W	Disables Data Request Generation Interrupt 0: Generate an interrupt when a data request is asserted 1: Disable interrupt generation when data request is asserted	1h
4	DISSADDRRDY	R/W	Disables Slave Address Ready Generation Interrupt 0: Generate an interrupt when the slave address ready is asserted 1: Disable interrupt generation when the slave address ready is asserted	1h
5	DISEND	R/W	Disables Message End Generation Interrupt 0: Generate an interrupt when the message end is asserted 1: Disable interrupt generation when the message end is asserted	1h
6	DISALERT	R/W	Disables Alert Generation Interrupt 0: Generate an interrupt when an alert is asserted 1: Disable interrupt generation when an alert is asserted	1h
7	DISCTRL	R/W	Disables Control Generation Interrupt 0: Generate an interrupt when control is asserted 1: Disable interrupt generation when control is asserted	1h
8	DISLOSTARB	R/W	Disables Lost Arbitration Generation Interrupt 0: Generate an interrupt when loss of arbitration is asserted 1: Disable interrupt generation when loss of arbitration is asserted	1h
9	DISCLKH	R/W	Disables Clock High Generation Interrupt 0: Generate an interrupt when clock high is asserted 1: Disable interrupt generation when clock high is asserted	1h
31:10	Reserved			0h

37.8.7 Slave mode control register (PMBUSSCTRL)

Offset address: 0x18

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
6:0	SDEVADDR	R/W	Slave Current Device Address Set The slave is set to the default mode, i.e., automatic slave address acknowledgment mode. After receiving the device address, the PMBus interface compares the received address with the preset slave address after mask filtering. If they match, the slave automatically acknowledges.	7Ch
7	SADDRACKEN	R/W	Manual Slave Address Acknowledgement Enable 0: The slave automatically acknowledges the device address specified by the SDEVADDR bits 1: Manually set the slave acknowledgment address, especially when the PMBUSCTRL[MODECFG] bit is 1. In this mode, the firmware needs to check the received address and acknowledge the message.	0h
14:8	SMASKEN	R/W	Slave Mode Enable When detecting an address, the slave can use a mask to acknowledge multiple device addresses. (1) After power-up, the slave mask defaults to 7Fh. At this time, the slave only responds when the received address completely matches bits [6:0] of the slave address; (2) When all bits of the mask are 0, any device address can be acknowledged; (3) When any bit in the mask is 0, the slave can acknowledge when the slave address is 0/1 on that bit.	7Fh
15	PECPROEN	R/W	PEC Processing Enable 0: Disable 1: Enable	0h
18:16	VALBYTESEL	R/W	Valid Bytes Select 000: No valid bytes 001: One valid byte, LIN_TXB register bits [7:0]: Byte #0 010: Two valid bytes, LIN_TXB register bits [15:0]: Byte #0 and Byte #1 011: Three valid bytes, LIN_TXB register bits [23:0]: Byte #0, Byte #1, and Byte #2 100: Four valid bytes, LIN_TXB register bits [31:0]: Byte #0, Byte #1, Byte #2, and Byte #3	0h
19	TXPEC	R/W	Transmit a PEC Byte at the End of the Message When this bit is set, a PEC byte will be transmitted at the end of the message. The PMBus interface transmits according to the number of data bytes specified by the VALBYTESEL bits and then transmits the PEC byte.	0h

Field	Name	R/W	Description	Reset value
20	COMC	R/W	Data Request Flag Generated after Receive of Command Code 0: The slave automatically acknowledges after receiving the command code 1: After receiving the command code, the slave generates a data request flag and waits for the firmware to issue an acknowledgment signal before continuing message transmission	0h
22:21	AUTOACK	R/W	Slave mode automatic acknowledge data bytes Configure 00: The slave receives 1 byte, the hardware does not automatically acknowledge the received byte, and the firmware manually acknowledges each received byte 01: The slave receives 2 bytes, the hardware automatically acknowledges the first received byte, and the firmware manually acknowledges the second byte 10: The slave receives 3 bytes, the hardware automatically acknowledges the first two received bytes, and the firmware manually acknowledges the third byte 11: The slave receives 4 bytes, the hardware automatically acknowledges the first three received bytes, and the firmware manually acknowledges the fourth byte	3h
31:23	Reserved			0h

37.8.8 Hold Slave Address Register (PMBUSHSADDR)

Offset address: 0x1C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ADDRRW	R	Address Stored R/W Bit 0: Write access 1: Read access	0h
7:1	DEVADDR	R	Stored Device Address	0h
31:8	Reserved			0h

37.8.9 Control register (PMBUSCTRL)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	STSMRSTEN	R/W	State Machines Reset Enable 0: Default state, the state machine is not reset	0h

Field	Name	R/W	Description	Reset value
			1: Resets the state machine to its initial state Note: A software reset does not clear the PMBUSSTS register, and the register needs to be manually read to clear it.	
1	ALERTLOW	R/W	PMBus Alert Driven Low By Slave 0: The PMBus alert is held high by a pull-up resistor and is not controlled by the slave 1: The PMBus alert is pulled low by the slave	0h
2	CLKLTOCFG	R/W	Clock Low Timeout Generated Interrupt Configure 0: An interrupt is generated on the rising edge of the clock low timeout 1: An interrupt is generated on the falling edge of the clock low timeout	0h
3	FASTEN	R/W	Fast Mode Enable 0: Enable standard 100KHz mode 1: Enable fast 400KHz mode	0h
4	Reserved			0h
5	CTRLCFG	R/W	Control Generated Interrupt Configure 0: An interrupt is generated on the rising edge of control 1: An interrupt is generated on the falling edge of control	0h
6	ALERTCFG	R/W	Alert Pin Configure 0: Configure as function mode 1: Configured as a GPIO pin	0h
7	ALERTDRVCFG	R/W	GPIO Mode Alert Pin Driven Configure 0: Low level 1: High level	0h
8	ALERTDIRCFG	R/W	Alert Pin Direction Configure 0: Output 1: Input	0h
9	PINCFG	R/W	Control Pin Configure 0: Default function mode 1: GPIO pin	0h
10	CTRLDRVCFG	R/W	GPIO Mode Control Pin Driven Configure 0: Low level 1: High level	0h
11	CTRLDIRCFG	R/W	Control Pin Direction Configure 0: Output 1: Input	0h
12	SDACFG	R/W	Data Pin Configure 0: Low level 1: High level	0h

Field	Name	R/W	Description	Reset value
13	SDADRVCFG	R/W	GPIO Mode Data Pin Driven Configure 0: Low level 1: High level	0h
14	SDADIRCFG	R/W	Data Pin Direction Configure 0: Output 1: Input	0h
15	SCLCFG	R/W	Clock Pin Configure 0: Function mode 1: GPIO pin	0h
16	SCLDRVCFG	R/W	GPIO Mode Clock Pin Driven Configure 0: Low level 1: High level	0h
17	SCLDIRCFG	R/W	Clock Pin Direction Configure 0: Output 1: Input	0h
19:18	Reserved			0h
20	DISCLKLTO	R/W	Clock Low Timeout Disabled 0: Enable 1: Disable	0h
21	SLAVEEN	R/W	PMBus Slave Enables 0: Disable 1: Enable	1h
22	MASTEREN	R/W	PMBus Master Enables 0: Disable 1: Enable	0h
27:23	CLKDIV	R/W	SYSCLK Clock Divider The PMBus transmit/receive FSM clock (FSMCLK) is obtained through the SYSLCK frequency division, where FMSCLK ≤ 10MHz. The formula is as follows: FMSCLK frequency = SYSCLK frequency / (CLKDIV + 1)	0h
30:28	Reserved			0h
31	MODECFG	R/W	Mode Configure 0: PMBus mode 1:I2C mode	0h

37.8.10 Timing Control Register (PMBUSTIMCTRL)

Offset address: 0x24

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	PARCFG	R/W	FSM Parameters Configure When this bit is set to 1, the FSM uses the settings of the PMBUSCLKTIM, PMBUSSTATIM, PMBUSBUSIDTIM, PMBUSCLKLTOTIIM, and PMBUSCLKHTOTIIM registers; by default, the FSM uses preset timing parameters.	0h
31:1	Reserved			0h

37.8.11 Clock Timing Register (PMBUSCLKTIM)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	CLKHIGHP	R/W	Number of FSM Input Clock in the PMBus Master Clock High Pulse Set Number of FSM clocks in one clock high pulse = (CLKHIGHP + 3)	2Fh
15:8	Reserved			0h
23:16	CLKTIM	R/W	Number of FSM Input Clock in the PMBus Master Clock Period Number of FSM clocks in one clock period = (CLKTIM + 4)	60h
31:24	Reserved			0h

37.8.12 Start Setup Time Register (PMBUSSTATIM)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	STATIM	R/W	The Time between the Last Rising Edge of the PMBus Master Clock and the Next Starting Edge This bit defines the clock cycle of the PMBus FSM.	2Fh
31:8	Reserved			0h

37.8.13 Bus Idle Time Register (PMBUSBUSIDTIM)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	BUSIDTIM	R/W	Bus Idle Time Set The value of this bit represents the duration of consecutive logic 1, in units of PMBus clock cycles. When the PMBus clock and data lines remain in the	1F3h

Field	Name	R/W	Description	Reset value
			logic 1 state for a period of time, the bus can be considered idle.	
31:10	Reserved			0h

37.8.14 Clock Low Timeout Value Register (PMBUSCLKLTOTIIM)

Offset address: 0x34

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
19:0	CLKLTOTIIM	R/W	Clock Low Timeout Time This bit represents the duration of consecutive low levels, in units of PMBus FSM clock cycles. A clock low timeout occurs when the PMBus clock line remains low for a duration exceeding the CLKLTOTIIM value.	0005 572Fh
31:20	Reserved			0h

37.8.15 Clock High Timeout Value Register (PMBUSCLKHTOTIIM)

Offset address: 0x38

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
9:0	CLKHTOTIIM	R/W	Clock High Timeout Time This bit represents the duration of consecutive high levels, in units of PMBus FSM clock cycles. A clock high timeout occurs when the PMBus clock line remains low for a duration exceeding the CLKHTOTIIM value.	1F3h
31:10	Reserved			0h

38 Controller area network (CAN)

38.1 Full Name and Abbreviation Description of Terms

Table 199 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Data Length Code	DLC
Bus-off State	Bus-off
Auto-Bus-on	ABO
Bit Stream Processor	BSP
Mod-2 sum	-
Information Processing Time	IPT

38.2 Introduction

CAN is an abbreviation for Controller Area Network, a serial communication protocol that complies with the ISO11898-1 (CAN 2.0B) protocol specification. The CAN module supports bit rates up to 1 Mbit/s and distributed real-time control with high reliability.

38.3 Main characteristics

(1) The maximum bit rate of communication is 1Mbps

(2) Comply with ISO11898-1 Protocol Specifications (CAN protocol 2.0A and 2.0B)

(3) Two interrupt lines: CANINT0 and CANINT1

(4) Support DMA function

(5) Support multiple clock sources

(6) Support software module reset

(7) Support message RAM parity check mechanism

(8) Support suspension mode for debugging operation

(9) Support programmable loopback mode for self-check operation

(10) After entering the Bus-off state, the 32-bit programmable timer will automatically return to bus-on

(11) 32 emails, with each having the following characteristics:

- Support transmitting or receiving function

- Support two types of frames: data frame and remote frame
- Support 11-bit standard identifiers or 29-bit extended identifiers, and the identifier receiving masks are configurable
- Able to accommodate data of 0-8 bytes
- Support data RAM and parity configuration
- Each email has an independent identifier mask
- Support programmable FIFO mode

Note:

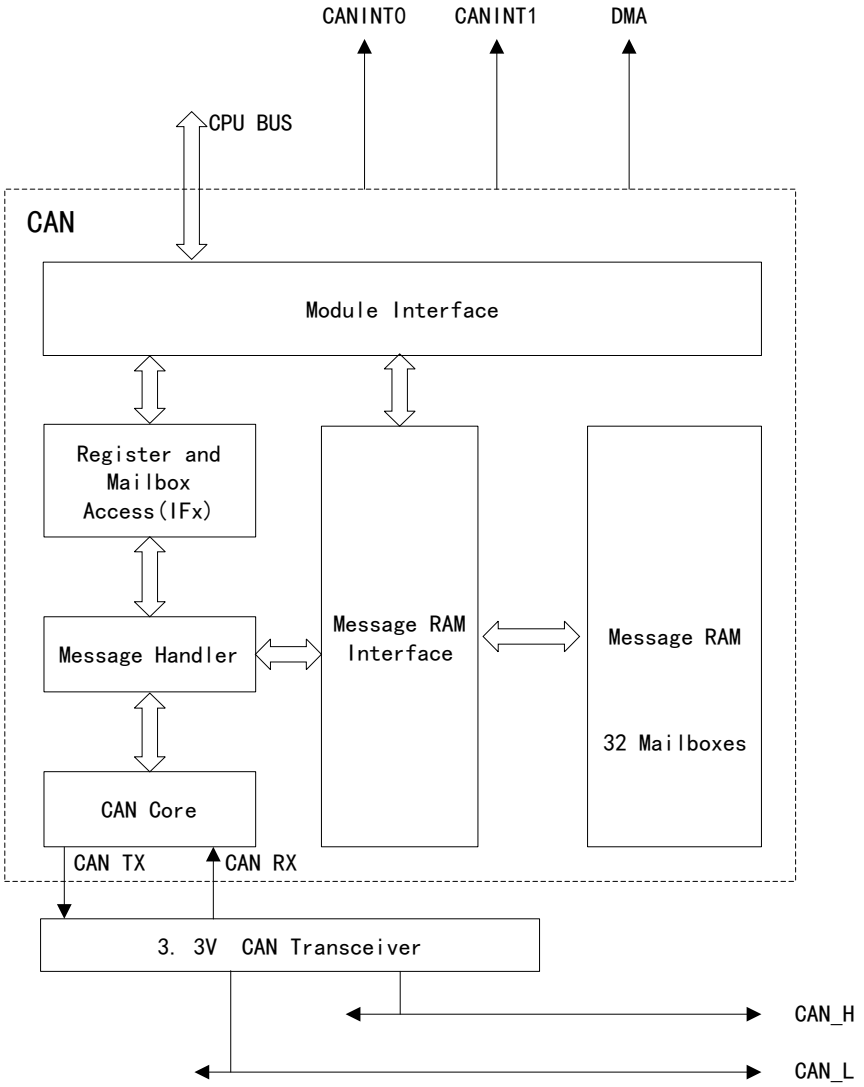
(1) According to the timing settings used, the accuracy of the on-chip built-in crystal oscillator might not meet the requirements of the CAN protocol. In such case, an external clock source must be used. For details about the accuracy of the on-chip built-in crystal oscillators, please refer to the Datasheet.

(2) For a 100 MHz CAN bit clock, the minimum available bit rate is 3.90625 kbps.

(3) The mailbox base address is offset by 0x400 from the CAN base address.

38.4 Structure block diagram

Figure 228 CAN Structure Block Diagram



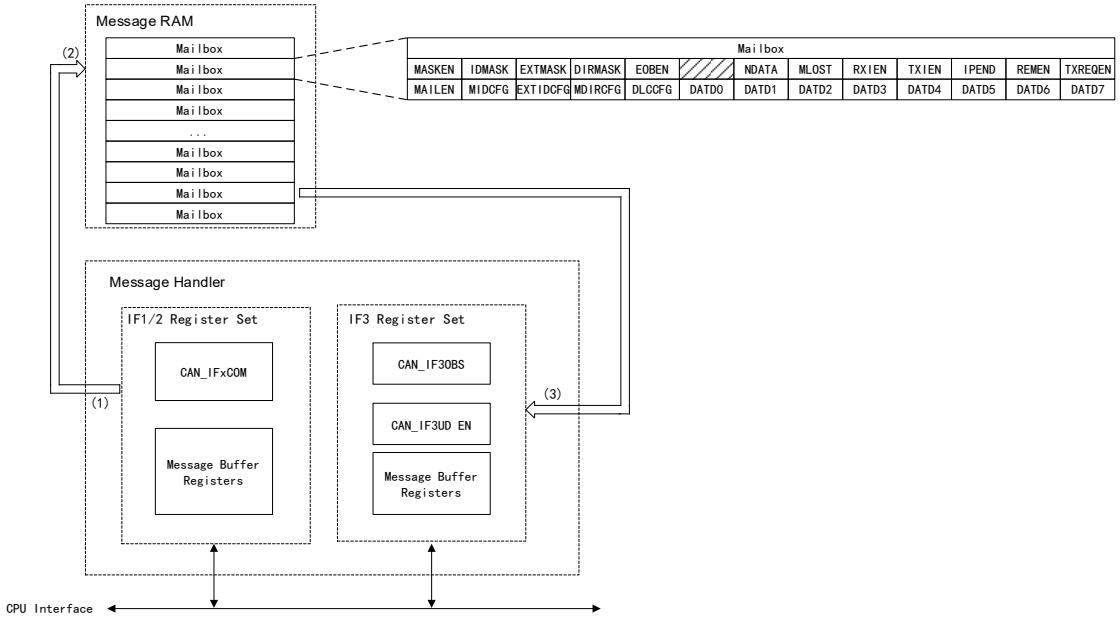
38.4.1 Internal Structure

Table 200 CAN Internal Structure

Internal Structure	Function
CAN Core	Consist of Rx/Tx shift registers and a CAN protocol controller, responsible for handling all functions related to the ISO 11898-1 protocol
Message RAM	Store 32 messages. In dedicated test mode, the message RAM is mapped to memory, allowing direct access.
Message Processor	(1) A state machine that controls data transfer between the single-port message RAM and the Rx/Tx shift registers of the CAN core (2) Process acceptance filtering and generate corresponding interrupt request signals based on the configuration in the control registers

Internal Structure	Function
Register and Mailbox Access (IFx)	During normal operation, all CPU and DMA accesses to the message RAM are indirect via IFx to ensure data consistency during read and write operations. The IFx registers can be seen as a window for indirect mailbox access. CPU read and write operations to the message RAM are controlled by three sets of interface registers. The IF1 and IF2 register banks can perform read and write operations, while the IF3 register bank is used only for read operations. The word length of IFx is the same as that of the message RAM.

Figure 229 Indirect Mailbox Access via IFx Registers



The access steps are as follows:

(1) Select the transmission direction and the register to be transmitted in the CAN_IFxCOM register of the IF1/2 register bank;

(2) Select the mailbox to be transmitted and perform the transmission in the CAN_IFxCOM register of the IF1/2 register bank

(3) When automatic update operation is enabled, perform automatic updates through the CAN_IF3OBS and CAN_IF3UDEN registers of the IF3 register bank

38.5 Functional description

As shown in the block diagram, the CAN module is connected to the CAN bus (physical layer) through a CAN transceiver chip.

Independent mailboxes can be configured for communication on the CAN network. These mailboxes and their identifier masks are stored in the message

RAM.

The message processor is responsible for executing all functions related to message handling, including:

- Processing transmission/DMA requests and generating interrupts
- Transferring messages between the CAN core and message RAM
- Receiving filtering

The CAN core and message processor are controlled and configured, and message RAM is accessed through the CAN register bank, directly accessible by the CPU via the module interface.

38.5.1 Pin configuration

By configuring the multiplexing registers in the GPIO section, select the corresponding pins to connect the CAN to the device. The GPIO multiplexing registers must be configured according to the correct steps outlined in the GPIO section, first configuring the GPxGMUX bit band (at which point the corresponding GPxMUX bit band is set as the default value of 0), and then writing the required value to the GPxMUX bit band to avoid burrs on the pins. The GPIO register is configured with some IO functions independent of CAN peripherals. The internal pull-up resistor status can be configured in the GPxPUD register. For input signals, the GPIO pin can be configured as asynchronous input by setting the corresponding GPxQSEL1/2[GPIOn]=11. For more information on GPIO multiplexing, refer to the GPIO section.

38.5.2 Addressing Scheme

The CAN module uses a special addressing scheme to support byte access. It is recommended to use the HWREG_BP() instruction only with the built-in __byte_peripheral_32() function for 32-bit access to CAN registers. For 16-bit access, write the lower 16 bits to the register address and the upper 16 bits to the register address + 2.

Due to the presence of bus bridging, when viewing the CAN module register space through the memory window of IDEs such as Keil, the observed addresses may not match the actual hardware addressing.

- In 32-bit or 16-bit view modes, ignore odd addresses and duplicate even addresses;
- In 8-bit view mode, even addresses from the CAN module are copied to odd addresses in the CCS memory window, and odd addresses from the module are not displayed.

Table 201 Software Access to CAN Registers

Software Access			**CAN Register Space**		
Format	Access	Data	Name	Address	Data
8bit	__byte((int *)0x00,0)	0x0000	CAN_CTRL	0x00	0x33221100

Software Access			CAN Register Space		
	__byte((int *)0x01,0)	0x0011	CAN_EFLG	0x04	0x33221100
	__byte((int *)0x02,0)	0x0022	CAN_ECNT	0x08	0x33221100
	__byte((int *)0x03,0)	0x0033	CAN_BTIM	0x0C	0x33221100
	__byte((int *)0x04,0)	0x0044			
	__byte((int *)0x05,0)	0x0055			
	__byte((int *)0x06,0)	0x0066			
	__byte((int *)0x07,0)	0x0077			
	__byte((int *)0x08,0)	0x0088			
	__byte((int *)0x09,0)	0x0099			
	__byte((int *)0x0A,0)	0x00AA			
	__byte((int *)0x0B,0)	0x00BB			
	__byte((int *)0x0C,0)	0x00CC			
	__byte((int *)0x0D,0)	0x00DD			
	__byte((int *)0x0E,0)	0x00EE			
	__byte((int *)0x0F,0)	0x00FF			
16 bits	(*((short *)(0x00)))	0x1100	CAN_CTRL	0x00	0x33221100
	(*((short *)(0x01)))	0x1111	CAN_EFLG	0x04	0x33221100
	(*((short *)(0x02)))	0x3322	CAN_ECNT	0x08	0x33221100
	(*((short *)(0x03)))	0x3322	CAN_BTIM	0x0C	0x33221100
	(*((short *)(0x04)))	0x5544			
	(*((short *)(0x05)))	0x5544			
	(*((short *)(0x06)))	0x7766			
	(*((short *)(0x07)))	0x7766			
	(*((short *)(0x08)))	0x9988			
	(*((short *)(0x09)))	0x9988			
	(*((short *)(0x0A)))	0xBBAA			
	(*((short *)(0x0B)))	0xBBAA			
	(*((short *)(0x0C)))	0xDDCC			
	(*((short *)(0x0D)))	0xDDCC			
	(*((short *)(0x0E)))	0xFFEE			
	(*((short *)(0x0F)))	0xFFEE			
32 bits	(*((long *)(0x00)))	0x33221100	CAN_CTRL	0x00	0x33221100
	(*((long *)(0x01)))	0x33221100	CAN_EFLG	0x04	0x33221100

Software Access			CAN Register Space		
	(*((long *)(0x02)))	0x33221100	CAN_ECNT	0x08	0x33221100
	(*((long *)(0x03)))	0x33221100	CAN_BTIM	0x0C	0x33221100
	(*((long *)(0x04)))	0x77665544			
	(*((long *)(0x05)))	0x77665544			
	(*((long *)(0x06)))	0x77665544			
	(*((long *)(0x07)))	0x77665544			
	(*((long *)(0x08)))	0xBBAA9988			
	(*((long *)(0x09)))	0xBBAA9988			
	(*((long *)(0x0A)))	0xBBAA9988			
	(*((long *)(0x0B)))	0xBBAA9988			
	(*((long *)(0x0C)))	0xFFEEDDCC			
	(*((long *)(0x0D)))	0xFFEEDDCC			
	(*((long *)(0x0E)))	0xFFEEDDCC			
	(*((long *)(0x0F)))	0xFFEEDDCC			

Table 202 IDE (Keil, etc.) Access to CAN Registers

Format	Access	Displayed data
8bit	0x00	0x00
	0x01	0x00
	0x02	0x22
	0x03	0x22
	0x04	0x44
	0x05	0x44
	0x06	0x66
	0x07	0x66
	0x08	0x88
	0x09	0x88
	0x0A	0xAA
	0x0B	0xAA
	0x0C	0xCC
	0x0D	0xCC
	0x0E	0xEE
	0x0F	0xEE
16 bits	0x00	0x1100

Format	Access	Displayed data
	0x01	0x1100
	0x02	0x3322
	0x03	0x3322
	0x04	0x5544
	0x05	0x5544
	0x06	0x7766
	0x07	0x7766
	0x08	0x9988
	0x09	0x9988
	0x0A	0xBBAA
	0x0B	0xBBAA
	0x0C	0xDDCC
	0x0D	0xDDCC
	0x0E	0xFFEE
	0x0F	0xFFEE
32 bits	0x00	0x11001100
	0x02	0x33223322
	0x04	0x554455544
	0x06	0x77667766
	0x08	0x99889988
	0x0A	0xBBAABBAA
	0x0C	0xDDCCDDCC
	0x0E	0xFFEEFFEE

38.5.3 Operating mode

38.5.3.1 Initialization mode

Initialization mode can be entered in one of three ways:

- Hardware reset
- Software Reset: Set CAN_CTRL[INIT]
- Entering bus-off state

When CAN_CTRL[INIT]=1, stop message transmission on the CAN bus. Also, CAN error counters are not updated, and CAN_TX outputs a recessive level (high). Setting CAN_CTRL[INIT] will not affect other configuration registers.

When initializing CAN, the CPU must configure the CAN bit timing, and

mailboxes must be used for CAN communication. Disable unused mailboxes in the CAN communication by clearing the corresponding MAILEN bits.

When CAN_CTRL[INIT] and CAN_CTRL[CFGWACCEN] are set, access the bit timing registers to configure timing.

Complete software initialization by clearing CAN_CTRL[INIT]. Subsequently, to synchronize with data transmission on the CAN bus, the BSP needs to wait for the occurrence of 11 consecutive high levels (indicating a bus idle state) before participating in bus activities and starting message transmission. For detailed information on bit timing, please refer to the Bit Timing section.

Mailbox initialization is not affected by CAN_CTRL[INIT]. However, all mailboxes must be configured with specific identifiers or set to "invalid" before starting message transmission.

During normal CPU operation, mailbox configuration can be changed. After configuring and transferring a mailbox from the interface registers to the message RAM, if the modified mailbox number is the same as or smaller than a previously found mailbox, acceptance filtering is performed on this mailbox. This ensures data consistency even when changing mailboxes, for example, when there is a pending CAN frame reception.

38.5.3.2 Normal mode

After CAN initialization and resetting CAN_CTRL[INIT], the CAN core synchronizes with the CAN bus and is ready for message transmission.

When a received message passes acceptance filtering, the entire message, including MSGID, DLC, and up to 8 data bytes, is stored in the corresponding mailbox. Thus, if an identifier mask is used, some bits can change without affecting message reception.

Because the message processor ensures data consistency during concurrent access, the CPU can read or write each message through the interface registers at any time.

The CPU can update pending transmit messages. For messages with permanent mailboxes (configured with MSGID and control bits, set for multiple CAN transmissions), only data bytes can be updated. When multiple pending transmit messages are assigned to the same mailbox, the entire mailbox must be configured before issuing the transmit request. Transmit requests for multiple mailboxes can be served simultaneously. Mailboxes are then transmitted sequentially based on their internal priority. Even if the transmission request is pending, the message can still be updated or set to "invalid" at any time. However, if the message is updated before a pending request starts transmission, the message's data bytes are discarded. Depending on the mailbox configuration, transmission can be automatically requested upon receiving a remote frame with a matching identifier.

38.5.4 Automatic Bus On

After entering the Bus-off state, the CPU initiates the Bus-off recovery sequence by resetting CAN_CTRL[INIT]; otherwise, the CAN module remains in Bus-off state. Enable automatic bus on by setting CAN_CTRL[ABOEN], at this point, the CAN automatically starts the Bus-off recovery sequence. The user can define the number of delay clock cycles for initiating the Bus-off recovery sequence to control the startup time of the sequence.

Note: When the CAN module enters the Bus-Off state due to multiple bus errors, it stops all bus activities and automatically sets CAN_CTRL[INIT]. After the application clears CAN_CTRL[INIT] or enables automatic bus on, the device resumes normal operation after waiting for 129 bus idle bits (129 * 11 consecutive recessive bits). Setting or clearing CAN_CTRL[INIT] does not shorten the Bus-Off recovery sequence. Error counters are reset at the end of the Bus-Off recovery sequence.

To allow the CPU to detect whether the CAN bus is continuously disturbed or stuck at a dominant bit, and to monitor the progress of the Bus-off recovery sequence, after CAN_CTRL[INIT] is reset, whenever 11 consecutive recessive bits (i.e., logic "1") are detected, the error code Bit 0 is written to the Error Flag Register (CAN_EFLG).

38.5.5 Disable automatic retransmission

Automatic retransmission is enabled by default and can be disabled by setting CAN_CTRL[DISARETX].

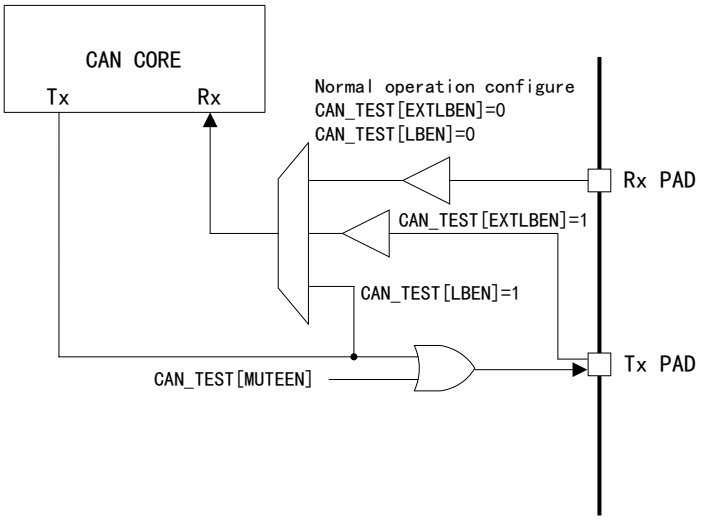
Based on the CAN specifications, automatic retransmission retransmits frames with arbitration loss or error disturbances. CAN does not provide the user with confirmation of whether the frame transmission w s successful until the transmission is successfully completed. For more information on this mode, see the Event-driven Message Transmit section.

38.5.6 Test Mode

The test mode is primarily used for the self-inspection of the CAN module. For all test modes, the CAN_CTRL[TESTEN] bit must be set first to enable write operations to the test register (CAN_TEST).

The operations of each test mode are shown in the figure below. It is important to note that the gate-level precision functionality of the module is not reflected in this figure. This diagram only illustrates the operations of the module and does not include GPIO multiplexing or I/O buffer modules.

Figure 18 CAN_MUX



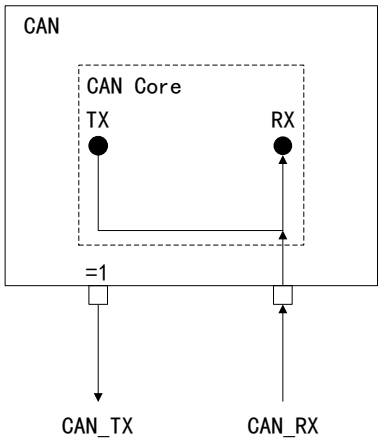
38.5.6.1 Mute mode

Set CAN_TEST[SLTEN] to enable silent mode.

In this mode, only recessive bits (logic "1") can be transmitted to the CAN bus, while dominant bits (logic "0") cannot be transmitted. Valid data frames and remote frames can be received from the bus and routed to the CAN core for processing. Since dominant bits (e.g., acknowledgment bits, overrun flags, active error flags) cannot be transmitted to influence the CAN bus, silent mode allows for analyzing the information flow on the CAN bus without affecting it.

In silent mode, the connection between CAN_TX, CAN_RX, and the CAN core is as shown in the figure below:

Figure 230 CAN Works in Mute Mode



In ISO 11898-1, the silent mode is referred to as bus monitoring mode.

38.5.6.2 Loopback mode

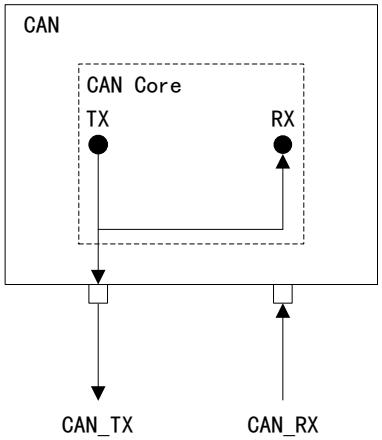
Set CAN_TEST[LBEN] to enable loopback mode.

In this mode, within the CAN module, the Tx output of the CAN core is connected to the Rx input. The transmitted message is directly looped back to the input and treated as a received message. When a message passes through the receive filters, it can be stored in a mailbox. The CAN core ignores the actual value of the external CAN_RX input, and the transmitted message can be monitored on the CAN_TX pin

The loopback mode is mainly used for hardware self-inspection function. In the loopback mode, acknowledgment errors (i.e., recessive bits sampled during the acknowledgment slot of a data frame/remote frame) are ignored by the CAN core.

The connection among CAN_TX, CAN_RX, and the CAN cores in the loopback mode is shown in the figure below:

Figure 231 CAN Works in Loopback Mode



Note: In the loopback mode, the signal path between the CAN core and the Tx pin, as well as the Tx pin itself, are ignored. For testing the above content, refer to the Loopback Silent Mode section for details.

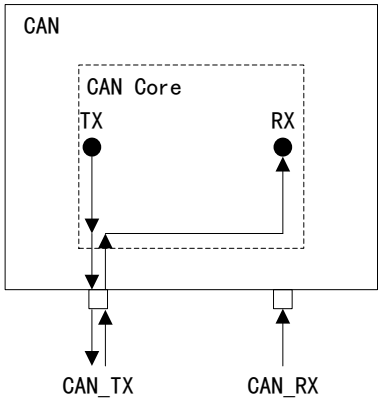
38.5.6.3 Loopback mute mode

Setting both CAN_TEST[LBEN] and CAN_TEST[SLTEN] enables the loopback silent mode.

In this mode, the transmitted data is directly looped back to the input for reception. The CAN_RX pin is not connected to the CAN core, and data cannot be received from the CAN bus. The CAN_TX pin can only transmit recessive bits (logic "1") to the bus, not dominant bits (logic "0"). Loopback silent mode allows testing of CAN hardware without interfering with the normal operation of the CAN network.

The connection among CAN_TX, CAN_RX, and CAN cores in the loopback silent mode is shown in the figure below:

Figure 232 CAN Operating in Loopback Silent Mode



38.5.6.4 External Loopback Mode

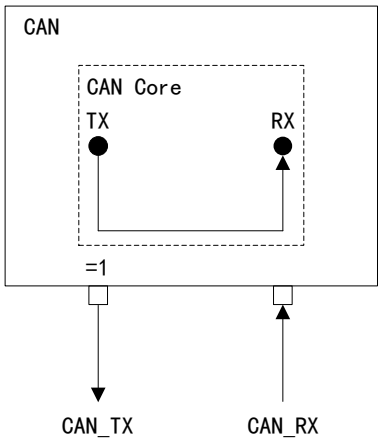
Set CAN_TEST[EXTLBEN] to enable the external loopback mode. When loopback mode is activated (i.e., CAN_TEST[LBEN] is set), the setting of CAN_TEST[EXTLBEN] will be ignored by the system.

In this mode, the CAN core is connected to the input buffer of the CAN_TX pin, forming a complete signal path that includes data transmission from the CAN core to the CAN_TX pin and its return to the CAN core via the TX pin.

Similar to the loopback mode, the external loopback mode is used to test whether the I/O circuit of the Tx pin is functioning properly.

The connection among CAN_TX, CAN_RX, and the CAN cores in the external loopback mode is shown in the figure below:

Figure 233 CAN Works in External Loopback Mode



38.5.7 Clock source

Typically, the system clock (SYSCLK) provides the CAN bit timing clock. Additionally, an external clock source (XTAL) can also provide the CAN bit clock, if required. For information on configuring the CAN module clock source, refer to the System Control Interrupts section and the datasheet.

When configuring the CAN core, ensure that each bit time is at least 8 clock cycles. To achieve a 1Mbps transmission rate, an oscillator with a frequency of at least 8MHz or higher is required.

38.5.8 Interrupt

38.5.8.1 Interrupt lines

CAN mode has two interrupt lines: CANINT0 and CANINT1.

Set CAN_CTRL[IEN0] to enable the CANINT0 interrupt line; set CAN_CTRL[IEN1] to enable the CANINT1 interrupt line.

The CAN module is equipped with module-level interrupt enable and acknowledge functions. Set CAN_GLBINTEN[GLBINT0EN] to enable the CANINT0 global interrupt; set CAN_GLBINTEN[GLBINT1EN] to enable the CANINT1 global interrupt. When handling an interrupt, prioritize clearing individual message or status change flags before acknowledging the interrupt using CAN_GLBINTCLR and NVIC_ACKn.

38.5.8.2 Interrupt source type

CAN interrupts can be categorized into the following three interrupt sources:

- Error interrupt
- Mailbox Interrupt
- Status change interrupt

The interrupt sources are identified by INT0 and INT1 in the Interrupt Register (CAN_INT). When INT0 and INT1 are set to 0, it indicates that there are no pending interrupts. When interrupt line CANINT0 or CANINT1 is triggered, the interrupt line remains active until either of the following conditions is met:

(1) The corresponding bit INT0 or INT1 in the Interrupt Register (CAN_INT) is reset to 0 (indicating that the event that triggered the interrupt has been handled);

(2) The corresponding bit CAN_CTRL[IEN0] or CAN_CTRL[IEN1] is reset.

When CAN_INT[INT0] = 0x8000, it indicates that a pending interrupt is generated due to the CAN core updating the error flag register, and this interrupt has the highest priority. Note that this update operation does not necessarily mean that the value in the error flag register has changed.

By reading the error flag register, the CPU can update (or reset) the RXCFLG, TXCFLG, and LEC flag bits. When the CPU performs a write operation on the error flag register, it will neither generate nor reset an interrupt. When the value of CAN_INT[INT0] or CAN_INT[INT1] is between 0x0001 and 0x0020, this value corresponds to the mailbox number, indicating that an interrupt is generated by the corresponding mailbox. CAN_INT[INT0] and CAN_INT[INT1] point to the pending interrupt with the highest priority. The smaller the mailbox number, the

higher the priority; conversely, the larger the number, the lower the priority.

The interrupt service routine can read the interrupt source message while also reading the message and resetting the IPEND bit of the mailbox by clearing the IFx command register (CAN_IFxCOM) (x=1/2). When the IPEND bit is cleared, the interrupt register points to the next mailbox with a pending interrupt.

Error interrupt

Enable the error interrupt group by setting CAN_CTRL[ERRIEN]. Error interrupts are only routed to the CANINT0 interrupt line, and the CANINT0 interrupt line is enabled by setting CAN_CTRL[IEN0].

Error interrupts encompass the following three events:

- Error Warning (EWFLG)
- Bus-off (BUSOFLG)
- Parity Error (PEFLG)

Mailbox Interrupt

Mailbox interrupts can be routed to both CANINT0 and CANINT1 interrupt lines.

Setting CAN_IFxMCTRL[IPEND] can force an interrupt.

Mailbox interrupts are generated by mailbox events and controlled by the IPEND, TXIEN, and RXIEN flags within the mailbox structure.

Status change interrupt

Enable the status change interrupt group by setting CAN_CTRL[SCHIEN]. Once set, a status change interrupt is generated for each CAN frame, regardless of active CAN communication, bus errors, and message RAM configuration.

Status change interrupts are only routed to the CANINT0 interrupt line, and the CANINT0 interrupt line is enabled by setting CAN_CTRL[IEN0].

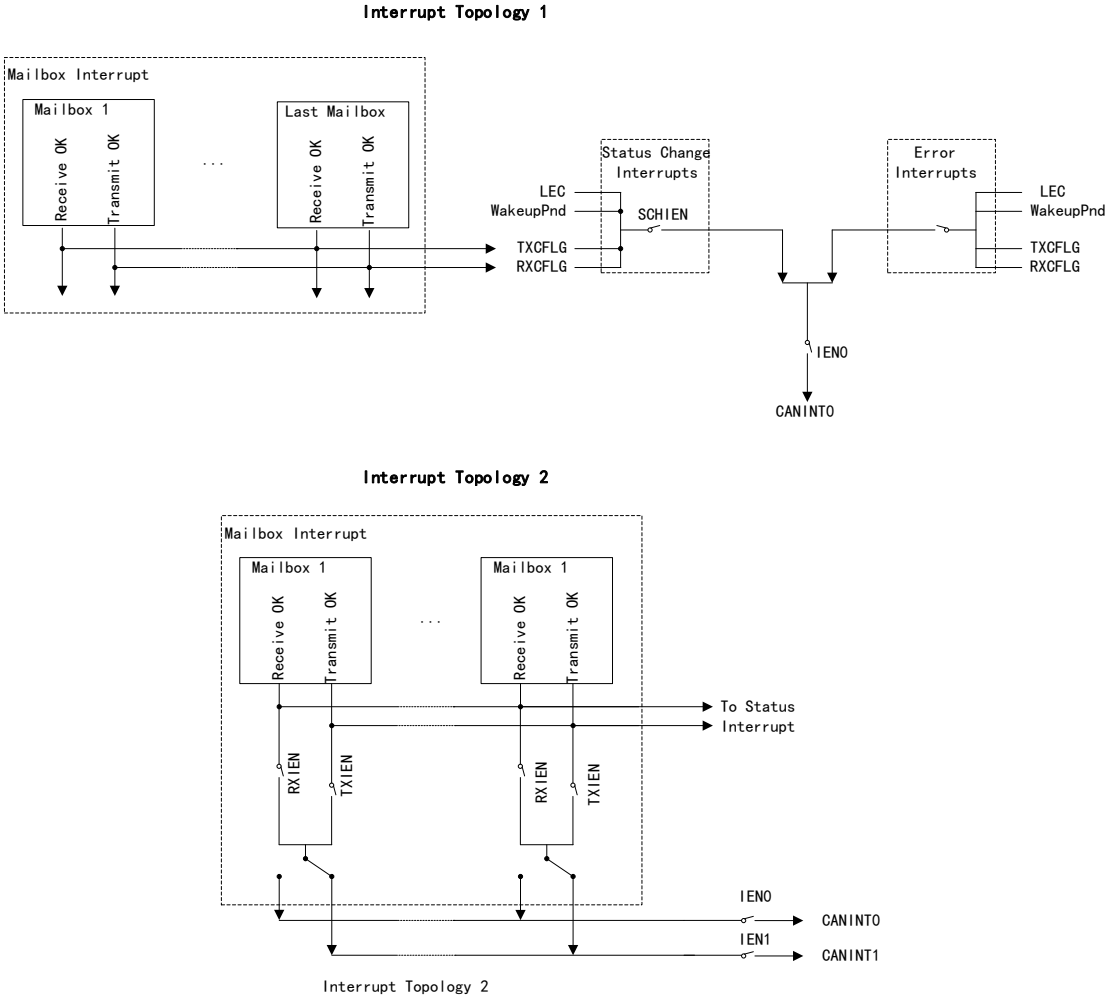
Status change interrupts encompass the following three events:

- Last Error Code (LEC)
- Transmit completed (TXCFLG)
- Receive completed (RXCFLG)

38.5.8.3 Interrupt Topology

The CAN interrupt topology is shown in the figure below:

Figure 234 Interrupt Topology



As shown in the figure above, error interrupts and status change interrupts can only be routed to the CANINT0 interrupt line, while mailbox interrupts for transmit and receive operations can be routed to both CANINT0 and CANINT1 interrupt lines.

38.5.9 DMA function

The DMA function of the CAN module is implemented through the interface registers IFx. The three interface registers IF1/2/3 correspond to the three DMA trigger outputs of the CAN module. Enable the DMA request for the corresponding interface register by setting CAN_CTRL[IFDENx] (x=1..3).

Upon completion of the internal IFx update, a DMA request is triggered and held. The first read or write operation to the IFx register bank clears the DMA request.

For IF1 and IF2 registers, writing to the IF1 and IF2 command registers initiates the update of the IF1 and IF2 registers. After initiation, REQDMAIFx in the IF1/2 command register is set, triggering a DMA request when the corresponding

interface becomes available. For the IF3 register, when a mailbox is configured to automatically update IF3 and a CAN message is received, the contents of the IF3 register are automatically updated. If the DMA request for IF3 is enabled, a DMA request is triggered for each IF3 update.

38.5.10 Parity Check Mechanism

To ensure the integrity of the message RAM data, the CAN module provides a parity check mechanism. A parity bit is calculated for every 32 bits of data (one word).

Configure CAN_CTRL[PEN]=1 to enable the parity check mechanism. In this case, when data is written to the message RAM, the parity bit is also written. When reading data, the parity bit stored in the message RAM is compared to check data integrity. When the parity mechanism is enabled, the CAN module automatically generates and checks parity bits. Parity bit generation follows the modulo-2 sum rule. If the modulo-2 sum of the data bits is 1, the parity bit is set. In other words, when the parity bit is set, the number of 1s in the data is odd.

Configure CAN_CTRL[PEN]=0 to disable the parity mechanism. In this case, when writing data to the message RAM, the parity bit in the message RAM remains unchanged, and the parity bit is not compared when reading data.

Parity Error Handling Mechanism

Each read operation of the message RAM performs a parity check on the mailbox, for example, at the start of CAN frame transmission. When a parity error is detected, CAN_EFLG[PEFLG] is set to indicate the occurrence of a parity error. At this time, the MVAL bit of the mailbox is cleared to avoid transmitting invalid data on the CAN bus. Furthermore, if error interrupts are enabled, an interrupt is also generated when a parity error is detected.

Since the CPU can still read data from the mailbox even if there is a parity error, the application must ensure that the CPU reads valid data. For example, this can be achieved by immediately checking the Parity Error Register (CAN_PE) when a parity error interrupt occurs.

38.5.11 Debug mode

When an external debugger is connected to the CAN module and pauses the CAN core, the debug mode is automatically entered. The debug mode of the CAN module provides functions such as pausing CAN activity and allowing the debugger to access the contents of the message RAM to support the use of external debugging units.

The system waits for the current transmission operation to start or the reception operation to complete, or for the bus to enter an idle state, before transitioning to debug mode. When CAN_CTRL[IDBGSEN] is set, the debugger can immediately interrupt the currently ongoing transmit or receive. After entering

the debug mode, CAN_CTRL[DBGFLG] is set to indicate that the CAN module is currently in the debug mode.

All CAN registers can be accessed in the debug mode. Reading reserved bits returns 0, and writing to reserved bits has no effect. The message RAM is memory-mapped, meaning that the external debugging unit can directly read the contents of the message RAM. For specific information on the memory structure, see the Message RAM Representation in Debug Mode section.

In the debug mode, the automatic clearing function of the error flag register and the IF1/IF2 command registers is disabled.

Note:

(1) In the debug mode, writing to control registers may affect the CAN state machine and further impact message handling.

(2) In debug mode, message RAM is directly memory-mapped, allowing the debugger to access it directly. During this time, the message RAM cannot be accessed through the interface register set.

38.5.12 CAN initialization

Initial State of CAN after Hardware Reset

After a hardware reset, all CAN protocol functions are disabled, and CAN_CTRL[INIT] is set. The bit timing and mailboxes must be configured before enabling CAN protocol functions.

Configure CAN Bit Timing

When configuring bit timing, both CAN_CTRL[INIT] and CAN_CTRL[CFGWACCEN] need to be set.

When CAN_CTRL[INIT] is cleared, the CAN core's protocol controller state machine and message handler state machine begin to control the internal data flow of the CAN module, indicating that the CAN module is transitioned from initialization mode to normal operation mode. Messages to be transmitted are stored in the shift register of the CAN core and transmitted over the CAN bus according to the CAN protocol. Messages that pass through the receive filters are stored in the message RAM.

For information on bit timing configuration, refer to the Bit Timing Configuration section for details.

Configure CAN Mailboxes

When configuring mailboxes, it is not necessary to set CAN_CTRL[CFGWACCEN].

A hardware reset resets the MVAL, NDATA, IPEND, and TXREQ bits of the mailbox to 0. Configure the mailboxes by setting the Mask Register, Arbitration Register, Message Control Register, and Data Registers of the IF1/IF2 interface register banks. Write the mailbox number to the VALMNUM bit of the corresponding IF1/IF2 Command Register to load the settings in the IF1/IF2 registers into the corresponding mailbox in the message RAM.

For information on mailbox configuration, refer to the Mailbox Configuration section for details.

Interrupt enable

After the CPU clears CAN_CTRL[INIT] and CAN_CTRL[CFGWACCEN], both CANINT0 and CANINT1 interrupt lines can be enabled by setting CAN_CTRL[IEN0] and CAN_CTRL[IEN1] simultaneously. In addition, both error interrupts and status change interrupts can be enabled by setting CAN_CTRL[ERRIEN] and CAN_CTRL[SCHIEN] simultaneously.

Interrupt-driven/Polling Mode Initialization

CAN communication supports both interrupt-driven and polling modes for data transmission control.

(1) In interrupt-driven mode, when the status of a mailbox changes (e.g., receiving a new message or completing message transmission), the corresponding mailbox's interrupt pending (IPEND) bit is set to 1. At this time, the interrupt register points to the mailboxes with their IPEND bit set to 1. Even if the interrupt lines CANINT0 and CANINT1 are disabled (i.e., CAN_CTRL[IEN0] and CAN_CTRL[IEN1] = 0), the interrupt register is still updated.

(2) In polling mode, the CPU polls the NDATA bits of the new data registers (CAN_NDATA) and the TXREQ bits of the transmission request registers (CAN_TXREQ) for all mailboxes. To simplify polling operations, it is recommended to allocate all transmit mailboxes to the lower number group and all receive mailboxes to the higher number group.

38.5.13 Message RAM

The message RAM is used to store mailboxes and their parity bits, with the following characteristics:

(1) Number of mailboxes: A total of 32 mailboxes are stored in the message RAM;

(2) Indirect access: During normal operation, the CPU cannot directly access the message RAM; it can only access the message RAM indirectly through the interface register banks;

(3) Debug mode access: The message RAM can only be accessed in the debug mode.

(4) The base address of the message RAM = CAN peripheral base address + 0x400

(5) Interface Register Bank:

- IF1 and IF2: These two sets of registers support the CPU in performing indirect read/write operations on the message RAM and buffer the control information and user data to be transmitted to or sent from the mailbox.
- IF3: The IF3 register bank automatically receives control information and user data from the message RAM after a CAN message is received and the mailbox is updated. The CPU does not need to initiate the transmission from the message RAM to the IF3 register bank.

(6) Conflict avoidance: The message handler resolves potential conflicts between concurrent access to the message RAM and CAN frame reception/transmission.

38.5.13.1 Mailbox Addressing

In the message RAM, starting address of a mailbox = mailbox base address + mailbox number * 0x20

For example:

Mailbox 1 starting address = mailbox base address + 1 * 0x20, i.e., the offset address of mailbox 1 is 0x20

Mailbox 2 starting address = mailbox base address + 2 * 0x20, i.e., the offset address of mailbox 2 is 0x40

Note:

(1) Mailbox numbers are 1-32; 0 is an invalid mailbox number

(2) The smaller the mailbox number, the higher the priority; conversely, the larger the number, the lower the priority. Therefore, mailbox 32 (the mailbox with the largest number) at address 0x00 has the lowest priority, and mailbox 1 has the highest priority.

(3) Writing data to a message RAM address that has not been allocated or defined as a valid mailbox by the system may overwrite the data of implemented mailboxes.

Table 203 Message RAM Addressing in Debug Mode

Debug mode	Mailbox Number	Offset address	Word Number
Parity	1	0x20	1
MXtd, MDir, Mask		0x24	2

Debug mode	Mailbox Number	Offset address	Word Number
Xtd, Dir, ID		0x28	3
Ctrl		0x2C	4
Data Bytes 3-0		0x30	5
Data Bytes 7-4		0x34	6
Parity	2	0x40	1
MXtd, MDir, Mask		0x44	2
Xtd, Dir, ID		0x48	3
Ctrl		0x4C	4
Data Bytes 3-0		0x50	5
Data Bytes 7-4		0x54	6
…	…	…	…
Parity	31	0x3E0	1
MXtd, MDir, Mask		0x3E4	2
Xtd, Dir, ID		0x3E8	3
Ctrl		0x3EC	4
Data Bytes 3-0		0x3F0	5
Data Bytes 7-4		0x3F4	6
Parity	32	0x00	1
MXtd, MDir, Mask		0x04	2
Xtd, Dir, ID		0x08	3
Ctrl		0x0C	4
Data Bytes 3-0		0x10	5
Data Bytes 7-4		0x014	6

Note: Mailbox 32 is the highest implemented mailbox number.

38.5.13.2 Message RAM Representation in Debug Mode

In the debug mode, because the message RAM is memory-mapped, the external debugging unit can directly access it. At this time, the message RAM cannot be accessed through the IFx register banks.

Figure 235 Message RAM Structure in Debug Mode

Message RAM

31	30	...	18	17	16
15	14	...	2	1	0

Message Address + 0x00

Reserved	
Reserved	Parity[4:0]

Message Address + 0x04

EXTMASK	DIRMASK	Reserved	IDMASK[28:16]
IDMASK[15:0]			

Message Address + 0x08

Reserved	EXTIDCFG	MDIRCFG	MIDCFG[28:16]
MIDCFG[15:0]			

Message Address + 0x0C

Reserved										
Reserved	MLOST	Reserved	MASKEN	TXIEN	RXIEN	REMEN	Reserved	EOBEN	Reserved	DLCCFG[3:0]

Message Address + 0x10

DATD3	DATD2
DATD1	DATD0

Message Address + 0x14

DATD7	DATD6
DATD5	DATD4

38.5.14 Mailbox Structure

The mailbox structure is shown in the figure below, where NDATA, IPEND, TXREQEN, and MAILEN are represented by dedicated registers.

Figure 236 Mailbox Structure

Mailbox												
MASKEN	IDMASK	EXTMASK	DIRMASK	EOBEN		NDATA	MLOST	RXIEN	TXIEN	IPEND	REMEN	TXREQEN
MAILEN	MIDCFG	EXTIDCFG	MDIRCFG	DLCCFG	DATD0	DATD1	DATD2	DATD3	DATD4	DATD5	DATD6	DATD7

Unused

MASKEN - Mask Enable

This bit indicates whether the Mask bit fields (IDMASK, EXTMASK, DIRMASK) are used for receive filtering. When this bit is set to 1, the mailbox's MASK bit fields must be configured during mailbox initialization before setting

CAN_MVAL[MVAL].

0: Ignored and not used for receive filtering.

1: Used for receive filtering.

IDMASK - Identifier Mask

This bit indicates whether the corresponding bit in the mailbox identifier is used for receive filtering.

0: Not used for receive filtering.

1: Used for receive filtering.

EXTMASK - Extended Identifier Mask

This bit indicates whether the extended identifier (CAN_IFxARB[EXTIDCFG]) is used for receive filtering.

When the mailbox uses an 11-bit standard format identifier, the 11-bit identifier of the received data frame is placed in MIDCFG[28:18]. During receive filtering, only MIDCFG[28:18] and the corresponding IDMASK[28:18] bits are considered.

0: Not used for receive filtering.

1: Used for receive filtering.

DIRMASK - Message Direction Mask

This bit indicates whether the message direction bit (CAN_IFxARB[MDIRCFG]) is used for receive filtering.

0: Not used for receive filtering.

1: Used for receive filtering.

EOBEN - End of Block Enable

This bit is used to cascade multiple mailboxes to construct a FIFO buffer. When used as a standalone mailbox, this bit must always be set to 1.

0: The current mailbox is part of a FIFO buffer block but not the last mailbox in the block.

1: The current mailbox is a standalone mailbox, not belonging to any FIFO buffer block, or the current mailbox is the last mailbox in a FIFO buffer block

NDATA - New Data

This bit indicates whether the data bytes of the mailbox have been updated.

0: Data not updated (no new data has been written to the data section of this mailbox by the message processor since this flag was last cleared).

1: New data (written by the CPU or message processor) is present in the data section of this mailbox.

MLOST – Message Lost

This bit applies only to mailboxes configured for the receive direction.

0: No message loss occurred since the CPU last cleared this bit.

1: A message loss occurred, i.e., the message processor received a new message while CAN_NDATA[NDATA] was set and stored it in this mailbox, causing the previous message to be overwritten by the new one.

RXIEN - Receive Interrupt Enable

This bit indicates whether IPEND is triggered after a frame is successfully received.

0: IPEND not triggered.

1: IPEND triggered.

TXIEN - Transmit Interrupt Enable

This bit indicates whether IPEND is triggered after a frame is successfully transmitted.

0: IPEND not triggered.

1: IPEND triggered.

IPEND - Interrupt Pending

This bit indicates whether there is a pending interrupt in the mailbox.

0: No pending interrupt in this mailbox.

1: Pending interrupt in this mailbox. When no other higher priority interrupts are pending, the interrupt identifier in the interrupt register will point to this mailbox.

REMEN - Remote Frame Enable

This bit indicates whether TXREQEN is set after a remote frame is received.

0: TXREQEN is not changed after a remote frame is received

1: TXREQEN is set after a remote frame is received

TXREQEN - Transmit Request Enable

This bit indicates whether there is a pending transmission request in the mailbox.

0: No pending transmission request for this mailbox

1: A pending transmission request exists for this mailbox and transmission is not yet complete

MAILEN - Mailbox Enable

This bit indicates whether the message processor uses this mailbox. When the mailbox is disabled, the message processor ignores it, and it will not be considered for receiving or transmitting CAN messages. When the mailbox is enabled, it will be used by the message processor for receiving or transmitting operations.

Note: This bit may remain high even if the identifier bit fields MIDCFG, EXTIDCFG, MDIRCFG, or DLCCFG change.

0: Disabled, the message processor ignores this mailbox

1: Enabled, the message processor uses this mailbox

MIDCFG - Mailbox Identifier

This bit indicates the identifier of the mailbox.

11-bit identifier - Standard Frame: MIDCFG[28:18]

29-bit identifier - Extended Frame: MIDCFG[28:0]

EXTIDCFG - Extended Identifier

This bit indicates whether the mailbox uses a standard or extended format identifier.

0: The mailbox uses an 11-bit standard identifier

1: The mailbox uses a 29-bit extended identifier

MDIRCFG - Message Valid

This bit indicates the transmission direction of the message.

0: Receive: When TXREQEN=1, a remote frame is transmitted, carrying an identifier that matches this mailbox. When a data frame is received and its identifier matches that of the mailbox, the data frame is stored in the mailbox

1: Transmit: When TXREQEN=1, a data frame is transmitted. When a remote frame is received and its identifier matches that of the mailbox; if REMEN=1, TXREQEN of this mailbox is set to 1.

DLCCFG - Data Length Code

This bit indicates the data length of the data frame.

Note: Mailboxes of different nodes with the same identifier need to be set with the same data length code. When the message processor receives a data frame, it sets the DLC in the corresponding mailbox according to the received message.

0000-1000: The data frame has 0-8 data bytes

1001-1111: The data frame has 8 data bytes

DATDx - Data Byte x of the Data Frame

This bit represents the data portion of the data frame.

The data bytes are arranged in the order in which they are shifted into the shift register of the CAN core during reception, with DATD0 being the first byte and DATD7 being the last byte. When storing a data frame, the message processor writes all 8 data bytes into the mailbox. If there are fewer than 8 data bytes, the remaining byte sections in the mailbox may contain undefined values.

38.5.15 Mailbox Configuration

The configuration of the entire message RAM shall be completed before the initialization ends, but the mailbox configuration can be changed during CAN communication.

38.5.15.1 Transmit Mailbox Configuration

The initialization configuration for each bit field of the transmit mailbox is shown in the following table:

Table 204 Transmit Mailbox Initialization Configuration

Bit band	Value
MAILEN	1
Arbitration	Specified by the user program.
Data	Specified by the user program.
Mask	Specified by the user program.
EOBEN	1
MDIRCFG	1
NDATA	0
MLOST	0
RXIEN	0
TXIEN	Specified by the user program.
IPEND	0

Bit band	Value
REMEN	Specified by the user program.
TXREQEN	0

(1) Arbitration Bit Field: This field defines the identifier and type of the transmitted message, including MIDCFG and EXTIDCFG, and the value of this field is specified by the user program. When using a standard frame with an 11-bit identifier (EXTIDCFG=0), MIDCFG[17:0] is ignored, and the 11-bit identifier is written to MIDCFG[28:18].

(2) Data Bit Field: This field defines the data register information, including DLCCFG and DATD0-7, and the value of this field is specified by the user program. Set the data to valid before setting REMEN and TXREQEN

(3) Mask Bit Field: This field is used to receive groups of data frames with the same identifier to set TXREQEN, including IDMASK, MASKEN, EXTMASK and DIRMASK. When this field is enabled (i.e., MASKEN=1), groups of data frames with the same identifier can be received. The MDIRCFG bit cannot be masked. When remote frames are disabled from setting TXREQEN, this bit field is also disabled, i.e., when REMEN=0, MASKEN=0. For more information, refer to the Receiving Data Frame/Remote Frame section.

(4) TXIEN Bit Field: When set, IPEND is set after the mailbox successfully transmits data.

(5) REMEN Bit Field: When set, TXREQEN is set when the CAN receives a matching remote frame. The CAN will automatically generate a data frame and transmit it as a response to the remote frame.

Transmit Mailbox Remote Frame Configuration

When transmitting remote frames, there is no need to configure the transmit mailbox. When TXREQEN is set for a receiving mailbox, the transmit mailbox will automatically transmit a remote frame with the same identifier as the data frame in the receiving mailbox.

38.5.15.2 Single Receiving Mailbox Configuration - Receiving Data Frame

When the message processor stores a data frame in the mailbox, it will store the received data length code and the corresponding number of bytes. If the data length code is less than 8, the remaining bytes in the mailbox will be filled with undefined values. When configuring a receiving mailbox for receiving data frames, the initialization configuration for each bit field is shown in the following table:

Table 205 Receive Mailbox - Initialization Configuration for Receive Data Frames

Bit band	Value
MAILEN	1
Arbitration	Specified by the user program.
Data	Specified by the user program.
Mask	Specified by the user program.
EOBEN	1
MDIRCFG	0
NDATA	0
MLOST	0
RXIEN	Specified by the user program.
TXIEN	0
IPEND	0
REMEN	0
TXREQEN	0

(1) Arbitration Bit Field: This field defines the identifier and type of the received data frame, including MIDCFG and EXTIDCFG, and the value of this field is specified by the user program. When using a standard frame with an 11-bit identifier (EXTIDCFG=0), MIDCFG[17:0] is ignored, and the 11-bit identifier is written to MIDCFG[28:18]. When a data frame with an 11-bit identifier is received, MIDCFG[17:0] will be set to 0.

(2) Mask Bit Field: This field is used to receive groups of data frames with the same identifier, including IDMASK, MASKEN, EXTMASK and DIRMASK. When this field is enabled (i.e., MASKEN=1), groups of data frames with the same identifier can be received. In typical applications, the MDIRCFG bit cannot be masked. When certain bits in the Mask bit field are set as "invalid", the corresponding bits in the arbitration register will be overwritten by the bits of the stored data frame

(3) RXIEN Bit Field: When set, IPEND is set after a data frame is received and stored in the mailbox.

(4) TXREQEN Bit Field: When set, a remote frame is transmitted with an identifier matching the identifier stored in the arbitration field. When the mask is enabled for receiving filtering, i.e., MASKEN=1, the value in the Arbitration Bit Field may change.

38.5.15.3 Single Receiving Mailbox Configuration - Receiving Remote Frame

Receive mailboxes used for remote frames can monitor remote frames on the CAN bus and can be extended into a FIFO buffer. For information about buffers,

refer to the FIFO Buffer Configuration section for details. Remote frames stored in the receive mailbox will not trigger the transmission of data frames. When configuring a receiving mailbox for receiving remote frames, the initialization configuration for each bit field is shown in the following table:

Table 206 Receive Mailbox - Initialization Configuration for Receive Remote Frames

Bit band	Value
MAILEN	1
Arbitration	Specified by the user program.
Data	Specified by the user program.
Mask	Specified by the user program.
EOBEN	1
MDIRCFG	1
NDATA	0
MLOST	0
RXIEN	Specified by the user program.
TXIEN	0
IPEND	0
REMEN	0
TXREQEN	0

(1) Arbitration Bit Field: This field defines the identifier and type of the received remote frame, including MIDCFG and EXTIDCFG, and the value of this field is specified by the user program. When using a standard frame with an 11-bit identifier (EXTIDCFG=0), MIDCFG[17:0] is ignored, and the 11-bit identifier is written to MIDCFG[28:18]. When a data frame with an 11-bit identifier is received, MIDCFG[17:0] will be set to 0.

(2) Mask Bit Field: This field is used to receive groups of remote frames with the same identifier, including IDMASK, MASKEN, EXTMASK and DIRMASK. In these bits, MASKEN must be set. In typical applications, the MDIRCFG bit cannot be masked. The Mask bit field can be configured as "must match" or "invalid" to enable receiving filtering for remote frame groups with the same identifier. For more information, refer to the Receiving Data Frame/Remote Frame section.

(3) RXIEN Bit Field: When set, IPEND is set after a remote frame is received and stored in the mailbox.

(4) Data Length Code (DLCCFG): When the message processor stores a received remote frame in the mailbox, the data length code is also

saved, while the data bytes in the mailbox remain unchanged. The data length code is specified by the user program.

38.5.15.4 FIFO Buffer Configuration

When connecting multiple mailboxes to the same FIFO buffer, ensure that the identifiers and masks (if used) of these mailboxes are programmed with matching values. Due to the hidden relationship between mailbox numbers and priorities, the mailbox with the smallest number will serve as the first mailbox in the FIFO buffer.

The configuration of receive mailboxes in a FIFO buffer is the same as that of a single receiving mailbox, except for the EOBEN bit. The last mailbox in the FIFO buffer must be configured as 1 to indicate the end of the block, and the EOBEN bit for all other mailboxes except the last one must be configured as 0.

38.5.16 Message Processor

Message Handling

After the CAN module initialization is complete, it can immediately synchronize and communicate with other devices on the CAN bus. Received messages are filtered within the CAN module, and once passing through the reception filter, they are stored in the specified mailbox, where the user can read them.

The user must update the data of the message to be transmitted, enable the transmission, and trigger a transmission request. When a remote frame matching with the identifier is received, the CAN module will automatically request the transmission of the message.

If a new message is received by the mailbox before the user reads the current message, the new message will overwrite the unread one.

Messages can be read in the following two ways:

- Read through interrupt drive
- Read after polling NDATA

Message Processor

The message processor state machine (hereafter referred to as the message processor) is used to control data transmission between the shift register of the CAN core and the message RAM.

The message processor is responsible for performing the following tasks:

(1) Data transmission

- Receiving messages, from the CAN core to the message RAM
- Messages to be transmitted, from the message RAM to the CAN core
- From the CAN core to the receiving filtering unit

(2) Receiving filtering, from scanning the message RAM to finding matched mailbox

(3) When the interface registers (IF1 and IF2) modify the mailbox, it will be recognized and processed by the message processor during the next scan only if its priority is not lowered (i.e., it remains the same or is increased).

(4) Process interrupt flags and transmission request flags

The registers of the message processor contain four types of status flags for all mailboxes: transmission request flags, new data flags, interrupt pending flags, and message valid flags. These registers can only be read.

The message handling registers allow users to efficiently view the status information of all mailboxes without having to individually collect each mailbox's status information using the IFx registers. For example, the message handler registers can provide a single view of all pending transmit requests.

38.5.17 Event-driven Message Transmit

The CAN module evaluates the MVAL bit in the mailbox valid register (CAN_MVAL) and the TXREQEN bit in the transmit request register (CAN_TXREQ) when the following two conditions are met:

(1) The shift register in the CAN core is ready to load data;

(2) There is no data transmission between the IFx registers and the message RAM.

When multiple transmission requests are pending, the message processor selects the valid mailbox with the highest priority to load into the shift register and initiates the transmission. The NDATA bit of the mailbox is cleared after the message is loaded into the shift register.

- If the message is successfully transmitted and no new data has been written to the mailbox since the start of transmission (i.e., NDATA remains 0), the transmit request bit will be cleared;
- If TXIEN is set, IPEND will be set after successful transmission.
- If an error or arbitration loss occurs during transmission, the message will be retransmitted when the CAN bus becomes idle again.
- If a message with higher priority requests transmission while a message is waiting to be transmitted, the messages will be transmitted in order of priority.

If automatic retransmission mode is disabled (i.e., CAN_CTRL[DISARETX]=1), the TXREQEN and NDATA bits in the IFx message control register (CAN_IFxMCTRL) of the interface register bank are configured as follows:

Table 207 TXREQEN and NDATA Bit Configuration

Status	TXREQEN	NDATA
Transmit initiated	Reset	Unchanged
Successfully transmit	Unchanged	Reset
Transmit failed (arbitration loss or error)	Reset again to restart transmit	Unchanged

Additionally, there is no need to allocate a receiving mailbox for remote frames. In this case, if REMEN is set in the transmit mailbox that matches the remote frame, it will automatically trigger the transmission of a data frame.

38.5.18 Transmit/Receive Priority

Transmit/Receive Priority

In the CAN module, the mailbox number determines the transmit/receive priority, and the CAN identifier is not related to priority. The smaller the mailbox number, the higher the priority; conversely, the larger the number, the lower the priority. Therefore, mailbox 1 has the highest priority, while mailbox 32 has the lowest priority.

When there are multiple transmit requests, the CAN module determines which request to process based on the priority of the mailboxes. Therefore, the message with the highest priority shall be stored in the mailbox with the smallest number.

Receive Filtering Priority

The mailbox number also determines the priority for receiving filtering. When performing receiving filtering on received data frames or remote frames, the mailbox with the smaller number will be used for receiving filtering. As such, once a data frame or remote frame is accepted by a mailbox, it will not be accepted by any other mailbox with a larger number.

The last mailbox can be configured to accept any data frame or remote frame not accepted by other mailboxes, ensuring complete logging of messages from nodes on the CAN bus that require communication.

38.5.19 Data transmission

38.5.19.1 Update/Change Transmit Mailbox

Update

When updating the transmit mailbox, the following points need to be noted:

(1) The data bytes in the transmit mailbox can be updated by the CPU at any time through the IF1 and IF2 register banks, with no need to reset

the message valid bit (MVAL) or the transmit request enable bit (TXREQEN) before updating.

(2) If only the data bytes are updated, follow these steps:
- Write 0x87 to bits [23:16] of the IFx command register (CAN_IFxCOM)
- Write the mailbox number to bits [7:0] of the IFx command register (CAN_IFxCOM)
- Update the data bytes and set both TXREQEN and NDATA

(3) Even when only part of the data bytes are updated, all four bytes in the IF1/IF2 data registers CAN_IFxDATAA or CAN_IFxDATAB (x=1/2) must be valid before being transmitted to the mailbox. To ensure data validity, the following methods can be used:
- The CPU directly writes all four bytes into the IF1/IF2 data registers
- After the mailbox's current data is transmitted to the IF1/IF2 data registers, the CPU writes the new data bytes into the registers

(4) When updating data, both NDATA and event-driven TXREQEN in CAN communication must be set simultaneously, ensuring that TXREQEN will not be cleared during an ongoing transmission. When both NDATA and TXREQEN are set, NDATA will be cleared once a new transmission process begins. For more information, refer to the Event-driven Message Transmit section.

Change

When the available number of mailboxes is insufficient to be used exclusively as permanent mailboxes, transmit mailboxes can be managed dynamically.

The CPU can write the entire message, including the arbitration, control, and data fields, into the interface registers. By writing 0xB7 to bits [23:16] of the IFx command register (CAN_IFxCOM), the entire message content can be transmitted to the mailbox without resetting TXREQEN and NDATA in advance.

If a mailbox's transmission request is in progress and not yet completed, modifying the mailbox in the manner described above will allow the mailbox to continue its current transmission. If the transmission process is interrupted due to interference, it will not restart.

To update only the data bytes of the currently transmitting message, write 0x87 to bits [23:16] of the IFx command register (CAN_IFxCOM).

Note: After updating a transmit mailbox, both the updated and non-updated contents will be backed up in the interface register bank.

38.5.20 Data receiving

38.5.20.1 Receiving filtering

Once the arbitration bits and control bits (including identifier, extended identifier, remote transmit request, and data length code) of the message are shifted into the CAN core's shift register, the message processor begins scanning the message RAM for a valid mailbox that matches the message, following these steps:

(1) Load the arbitration bits of the shift register: The receiving filter unit loads the arbitration bits from the CAN core's shift register.

(2) Load arbitration and mask bits of mailbox 1: The receiving filter unit loads the arbitration bits and mask bits of mailbox 1, including MVAL, MASKEN, NDATA, and EOBEN.

(3) Compare the bits loaded in the previous two steps.

The steps mentioned above will be repeated for subsequent mailboxes until a matching mailbox is found or until the end of the message RAM is reached. When a matched mailbox is found, scanning stops, and the message processor continues operations based on the type of frame received (data frame/remote frame).

38.5.20.2 Receive Data Frames/Remote Frame

Data frame

The message processor stores the contents of the message from the CAN core shift register into the corresponding mailbox in the message RAM, including the data bytes, arbitration bits, and data length code. This operation ensures that data bytes can still be associated with identifiers even when arbitration mask registers are used.

When new data is received (data not yet read by the CPU), the NDATA bit is set. After the CPU reads the mailbox, this bit is cleared. If new data is received and the NDATA bit is set, it means that the previous data (data not yet read by the CPU) is lost. In this case, the MLOST bit is set to indicate a message loss. If the receive interrupt is enabled (RXIEN is set), receiving data sets the IPEND bit, and the interrupt register points to this mailbox to trigger an interrupt.

To prevent transmitting a remote frame immediately after receiving the requested data frame, the TXREQEN of the mailbox will be reset after a data frame is received.

Remote frame

When receiving a remote frame, the three configuration scenarios for the matched mailbox are shown in the table below:

The MDIRCFG bit is used to configure the message direction, the REMEN bit is used to enable remote frames, and the MASKEN bit is used to enable the mask.

Table 208 Matched Mailbox Configuration

MDIRCFG	REMEN	MASKEN	Operation
1	1	1/0	When a matched remote frame is received, the TXREQEN bit of the mailbox is set, and the rest of the mailbox remains unchanged.
1	0	0	If the received remote frame is ignored, the mailbox remains unchanged.
1	0	1	When a remote frame is received, it is processed in the same way as a data frame. That is, when a matching remote frame is received, the TXREQEN bit of the mailbox is set. The arbitration and control bits in the CAN core shift register, namely the identifier, extended identifier, remote transmit request, and data length code, are stored in the corresponding mailbox in the message RAM, and the NDATA bit of this mailbox is set, with the mailbox's data bytes remaining unchanged.

38.5.20.3 Read Receive Messages

The message processor ensures data consistency during the read operation, and the CPU can read the received messages through the IFx interface registers at any time. The typical steps to transmit the entire received message from the message RAM to the interface registers are as follows:

- Write 0x7F to bits [23:16] of the IFx command register (CAN_IFxCOM)
- Write the mailbox number to bits [7:0] of the IFx command register (CAN_IFxCOM)

Additionally, this will clear the NDATA and IPEND bits in the message RAM (the NDATA and IPEND bits in the interface registers are not affected). The values of these bits in the message control register always remain in their pre-reset state. When a mailbox uses the mask for receiving filtering, the arbitration bits indicate which matched messages have been received.

Actual values of NDATA and MLOST:

(1) NDATA actual value: whether a new message has been received since the last read of this mailbox;

(2) MLOST actual value: whether multiple messages have been received since the last read of this mailbox, causing message loss. MLOST does not reset automatically.

38.5.20.4 Request New Data Receive

By setting the TXREQEN bit of the mailbox, the CPU can transmit a remote frame to other CAN nodes to request new data. This remote frame carries the identifier of the receive mailbox, triggering another CAN node to transmit a data frame that matches the identifier. If the FIFO receives matched data frame before the remote frame is transmitted, the TXREQEN bit of the mailbox is automatically cleared.

By writing 0x84 to bits [23:16] of the IFx command register (CAN_IFxCOM), the TXREQEN bit can be set without altering the mailbox content.

38.5.21 FIFO Buffer

Several mailboxes can be organized into a FIFO buffer or multiple FIFO buffers.

Store Receive Message

Each FIFO is used to store received messages with a specific identifier. Within the same FIFO buffer, the configurations of arbitration and mask registers are identical for all mailboxes. The last mailbox in the FIFO buffer must be configured as 1 to indicate the end of the block, and the EOBEN bit for all other mailboxes except the last one must be configured as 0.

When a received message matches an identifier in the FIFO buffer, it is stored sequentially in the mailboxes of the FIFO buffer in order from the lowest to the highest mailbox number. While storing the message into a mailbox, the NDATA bit of that mailbox is set, indicating that new data has been received. If the EOBEN bit of the mailbox is 0 (indicating that the mailbox is not the last one in the FIFO buffer), setting the NDATA bit will lock the mailbox, preventing the message processor from writing further data into it until the CPU clears the NDATA bit

FIFO buffers store messages in order from the lowest to the highest mailbox number, filling mailboxes until reaching the last one in the buffer. If all mailboxes are locked (i.e., NDATA is set), and the EOBEN of the last mailbox in the FIFO buffer is 1, indicating that it is the last available mailbox, then subsequent messages can only be written into this last mailbox and will overwrite the previously stored messages (which have not yet been read by the CPU).

Read FIFO Buffer

In a FIFO buffer composed of a group of mailboxes, these mailboxes can store multiple messages, and the user program needs to empty the buffer to prevent data loss. A FIFO buffer with a length of N stores two parts:

- N-1 messages
- The last message received since the buffer was last cleared

The process of reading a message from a FIFO buffer mailbox and clearing the

NDATA bit is the same as that for a single mailbox. In a subroutine, to clear the FIFO buffer, read and reset the NDATA bit of all mailboxes, proceeding in order from the lowest to the highest mailbox number. The steps of the subroutine for interrupt-driven CPU handling of the FIFO buffer are as follows:

(1) Receive the mailbox interrupt and start the process;

(2) Read the interrupt identifier;

(3) Perform a conditional judgment based on the interrupt identifier value:
- If the value is 0x0000, end the process
- If the value is 0x0800, enter the state change interrupt handling routine
- For other values, proceed to the next step;

(4) Write 0x007F into CAN_IFxCOM[31:16], with the mailbox number equal to the value of the interrupt identifier;

(5) Write the mailbox number into the CAN_IFxCOM register, triggering the transfer of the message to the IF1/IF2 register set, and clear the NDATA and IPEND bits.

(6) Read the IF1/IF2 Message Control Field

(7) Determine the NDATA value
- If NDATA = 1, indicating there is new data to be read, proceed to the next step
- If NDATA = 0, indicating there is no new data, restart from step (2)

(8) Read data from the IF1/IF2 data registers CAN_IFxDATAA and CAN_IFxDATAB

(9) Determine the EOBEN value
- If EOBEN = 1, it indicates that all data in the current data block has been read. Restart from step (2) to read the interrupt identifier again
- If EOBEN = 0, it indicates that there is still data left to be read in the current data block. Return to step (5) to read the next mailbox

Note: All mailboxes in the FIFO buffer must be read and cleared before storing the next batch of messages. If this order is not followed, the mailboxes in the buffer that have already been read will be filled in descending priority order, making it impossible to implement proper FIFO function.

38.5.22 Bit timing

CAN supports bit rates ranging from 1 kBit/s to 1000 kBit/s. The clock signal of the CAN module is typically provided by a crystal oscillator, with each CAN node having its own clock signal. As a result, the bit timing parameters of each CAN node can be configured independently. Even if the oscillator periods (Fosc) of the CAN nodes may differ, the same bit rate can still be achieved. However, temperature or voltage changes, as well as component degradation, can cause

small variations in oscillator frequency, meaning that the oscillator frequency is not absolutely stable. As long as these variations do not exceed a specific oscillator tolerance range (df), CAN nodes can compensate for different bit rates by resynchronizing to the bit stream

CAN bit synchronization can correct configuration errors in CAN bit timing, reducing the frequency of error frames. However, during arbitration, if a frame is transmitted simultaneously by two or more nodes, misaligned sampling points may cause one of the transmitting nodes to generate a passive error.

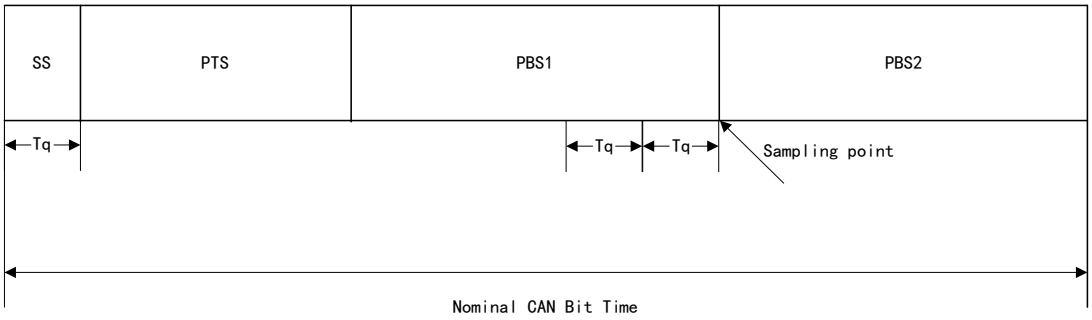
In many cases, CAN bit synchronization will correct errors in CAN bit timing configuration, causing it to only occasionally produce error frames. However, in arbitration situations, when two or more CAN nodes simultaneously attempt to transmit a frame, a misaligned sampling point may cause one of the transmitters to become erroneously recessive.

Understanding the internal CAN bit synchronization and the interaction of CAN nodes on the CAN bus can help analyze these sporadic errors. Even minor errors in CAN bit timing configuration can have a significant impact on the performance of a CAN network, although such minor errors will not immediately lead to failures.

Bit Timing Composition

The basic time unit in bit timing is the Time Quantum (Tq). According to the CAN protocol specifications, bit timing is subdivided into the Synchronization Segment (SS), Propagation Time Segment (PTS), Phase Buffer Segment 1 (PBS1), and Phase Buffer Segment 2 (PBS2), each composed of a specific number of time quanta. The length of the Synchronization Segment is fixed at one time quantum, while the number of time quanta for the other segments can be configured. as shown in the figure below:

Figure 237 Bit Timing



The formula for calculating the Time Quantum (Tq) is as follows:

$$\mathrm{Tq} = \frac{(\text{Baud rate prescale})}{f_{CANCLK}}$$

Where:

Baud rate prescaler: consists of BRPSCEXT and BRPSC, configurable between 1 to 1024 for frequency division.

f_{CANCLK}: CAN clock frequency.

For a given bit rate, there may be multiple bit timing configurations that meet the requirements. The minimum configurable ranges for each time segment in the bit timing are shown in the table below:

Table 209Configurable Ranges of Bit Timing

Parameter	Description	Configurable Range
Synchronization segment	Used for synchronizing bus inputs to CAN_CLK	1 * Tq ()
Propagation time period	Used for compensating physical delay time	(1-8) * Tq
Phase buffer segment 1	Can be temporarily extended during synchronization	(1-8) * Tq
Phase buffer segment 2	Can be temporarily shortened during synchronization	(1-8) * Tq
Synchronization jump width	This parameter's time must not exceed any of the phase buffer segments	(1-4) * Tq

Note: To ensure stable operation of the CAN network, the tolerance range of the oscillator and the physical delay duration must be considered.

38.5.22.1 Synchronization Segment (SS)

In the CAN bit timing, the CAN bus level edge jump is expected to occur in the synchronization segment. If an edge jump occurs outside the synchronization segment, the distance between it and the synchronization segment is referred to as the phase error of that edge jump.

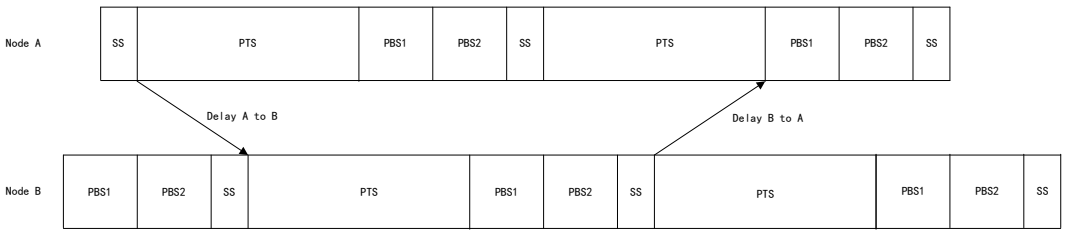
38.5.22.2 Propagation Time Segment (PTS)

In CAN bit timing, the propagation time segment compensates for the physical delay time in the CAN network, including the internal delays within CAN nodes and the signal propagation time on the bus.

Due to varying distances between two nodes, the time for a signal to reach each node also varies. Consequently, any CAN node synchronized with the CAN bus bit stream may be out of phase with the node transmitting the bit stream.

In the CAN protocol, the non-destructive bit-level arbitration mechanism and the dominant response bits provided by receiving nodes require that each node transmitting a bit stream must also be capable of receiving other nodes' transmitted bits synchronized with this bit stream. The phase shift and propagation time between two CAN nodes is illustrated as follows:

Figure 238 Propagation Time Period



Where:

Delay A to B ≥ Node A output delay + (A to B) bus delay + Node B input delay;

Propagation time segment ≥ Delay A to B + Delay B to A

The specific analysis in the example is as follows:

(1) Delay and Bus Arbitration

- Node A and Node B perform arbitration on the CAN bus, both acting as data transmitters;
- Node A transmits the start frame (dominant bit) slightly earlier than Node B, where 'slightly earlier time' is less than one bit time;
- When Node B receives the edge jump from Node A's recessive to dominant bit, it immediately synchronizes;
- Since Node B receives the edge jump after Node A has transmitted the dominant bit, there is a delay (Delay A to B), causing a shift in the bit timing segment of Node B compared with Node S.
- If Node B transmits an identifier with higher priority, exactly when Node A transmits a recessive bit and Node B transmits a dominant bit, Node B will win the arbitration on this identifier bit. At this point, the dominant bit transmitted by Node B is received by Node A after a delay (Delay B to A).

(2) Oscillator Tolerance

- Due to the tolerance of the oscillators, the actual position of the sampling point of Node A may be anywhere within the nominal range of its phase buffer segment;
- Therefore, to ensure that Node A can correctly sample the bit transmitted by Node B, the jump edge transmitted by Node B must reach Node A before starting the Phase Buffer Segment 1 of Node A. Thus, the propagation time length must meet the above requirement.

(3) Sampling Errors

- If the edge jump transmitted by Node B arrives after starting the Phase Buffer Segment 1 of Node A, Node A may incorrectly sample a recessive bit at the sampling point, leading to a bit error and generating an error flag that disrupts the current frame.

- Such errors only occur when both nodes are attempting arbitration on the bus, and their oscillator frequencies are at the extremes of the tolerance range, separated by a longer bus distance. Therefore, in bit timing, occasional bus errors may occur due to the Propagation Time Segment (PTS) being configured too short.

38.5.22.3 Phase Buffer Segments and Synchronization Jump Width

Oscillator tolerance is compensated for by the phase buffer segments and synchronization jump width.

Phase Buffer Segments (PBS1 and PBS2)

Phase buffer segments are located on both sides of the sampling point. During synchronization, these segments can be extended or shortened as needed

Synchronization Jump Width (SJW)

The synchronization jump width is used to define the maximum extent to which the resynchronization mechanism can adjust the sampling point position within the phase buffer segments to compensate for edge phase errors.

Edge detectION

To detect edges, the bus level needs to be sampled at every time quantum, and the current sampled bus level is compared with the previous sampled bus level. The synchronization process occurs during the edge jump from recessive to dominant bit, controlling the distance from the edge to the sampling point. Therefore, synchronization can occur only if the previous sampling point sampled a recessive level and the bus level at the current time quantum is dominant.

Synchronization Edge and Phase Error

(1) When an edge occurs within the synchronization segment, it is defined as the synchronization edge.

(2) When an edge occurs outside the synchronization segment, the distance from this edge to the synchronization segment is defined as the edge phase error, measured in time quanta.
- Phase error is positive: The edge occurs after the synchronization segment
- Phase error is negative: The edge occurs before the synchronization segment

Hard Synchronization and Resynchronization

Synchronization methods are divided into hard synchronization and

resynchronization.

(1) Hard synchronization: Hard synchronization typically occurs before the start of frame transmission. After hard synchronization, edge phase errors are not considered, and the bit timing restarts after the end of the synchronization segment, meaning that the edge triggering hard synchronization falls within the synchronization segment of the restarted bit timing.

(2) Resynchronization: During frame transmission, only resynchronization is allowed. Resynchronization adjusts the distance of the sampling point relative to the edge by shortening or lengthening the bit time.

- If the phase error is positive: Lengthen PBS1. If the phase error is less than the synchronization jump width, PBS1 is lengthened by the width of the phase error; otherwise, PBS1 is lengthened by the width of the synchronization jump;
- If the phase error is negative: Shorten PBS2. If the phase error is less than the synchronization jump width, then PBS2 is shortened by the width of the phase error; otherwise, PBS2 is shortened by the width of the synchronization jump.

Impact of the relationship between the phase error and synchronization jump on synchronization:

- When the magnitude of the phase error is ≤ the synchronization jump width, the outcomes of hard synchronization and resynchronization are the same.
- When the magnitude of the phase error is > the synchronization jump width, the result of resynchronization cannot fully compensate for the phase error, leaving an error equal to (phase error - synchronization jump width).

Only one synchronization can occur between two sampling points. Synchronization operations maintain a minimal distance between the edge and the sampling point, providing time for the bus level to stabilize and filtering out spikes shorter than the first phase segment (PBS1) of the PTS.

Reasons for Synchronization

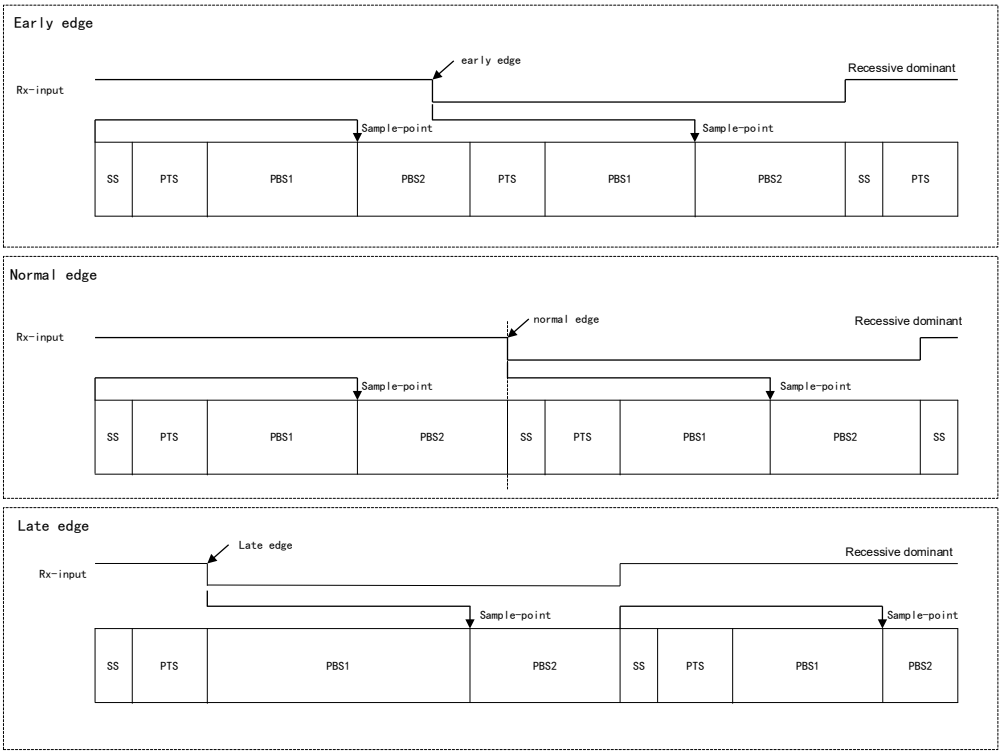
Apart from spike noise, most synchronization processes are caused by arbitration. All nodes transmitting data will produce hard synchronization on the jump edge of the node that starts transmission first. However, due to propagation delays, ideal synchronization across all nodes cannot be achieved. The first node to start transmission (the "leading" node) does not necessarily win the arbitration; therefore, the receivers in the nodes must synchronize with the subsequent "leading" transmitting nodes, which are out of sync with the previously "leading" node.

Acknowledge Field: A similar situation to arbitration occurs in the acknowledge field: The transmit node and some receive nodes must synchronize to the "leading" node that transmits the dominant acknowledge bit.

Oscillator Tolerance: After arbitration ends, oscillator tolerance triggers the synchronization process. Due to the differences in oscillator clock cycles between the transmitter and receiver accumulating over the time between two synchronizations (up to 10 bits), these accumulated differences must not exceed the synchronization jump width, thus limiting the tolerance range of the oscillator.

Using phase buffer segments to compensate for phase errors is illustrated as follows:

Figure 239 Synchronization on "Early" and "Late" Edges



The three figures represent synchronization on "early" edges, no synchronization, and "late" edges, respectively. During synchronization, the phase buffer segments are temporarily lengthened or shortened, and the length of the synchronization segment will return to its preset nominal value in the next bit timing.

"Early" Edge Synchronization

As shown in the figure above, the edge jump (from recessive to dominant) occurs on Phase Buffer Segment 2 (PBS2), earlier than the Synchronization Segment (SS), and is therefore an "early" edge. When an "early" edge is detected, in order to make the distance from the edge jump to the sampling

point the same as the distance from the SS to the sampling point when no edge jump occurs, PBS2 is shortened and SS is ignored. Because the phase error of this "early" edge is less than the synchronization jump width, it can be fully compensated.

"Late" Edge Synchronization

As shown in the figure above, the edge jump (from recessive to dominant) occurs at the end of the Propagation Time Segment (PTS), later than the Synchronization Segment (SS), and is therefore an "late" edge. When an "late" edge is detected, in order to make the distance from the edge jump to the sampling point the same as the distance from the SS to the sampling point when no edge jump occurs, PBS2 is lengthened. Because the phase error of this "late" edge is less than the synchronization jump width, it can be fully compensated. Due to this compensation, a recessive-to-dominant edge jump will occur during the next SS (a nominal bit-time length).

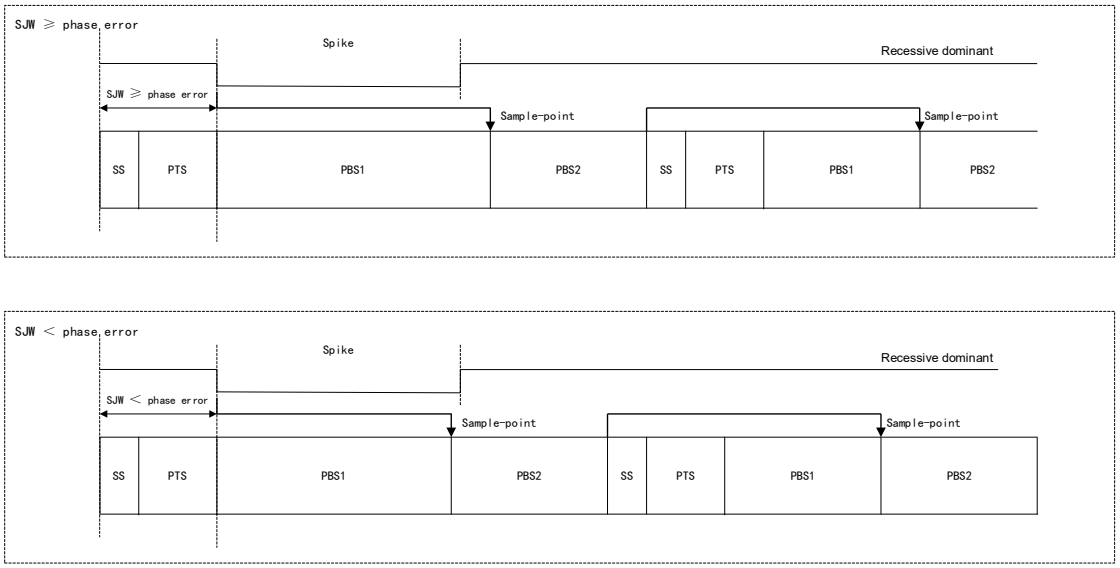
In the above example, the bit timing is analyzed from the perspective of the state machine implemented by the CAN protocol, where the bit time starts at the sampling point and ends at the sampling point. When synchronizing on an "early" edge, the SS is ignored. This is because once the edge jump is detected within PBS2, the state machine cannot reassign that time quantum as SS.

Since the state machine cannot subsequently redefine the time quantum of Phase Buffer Segment 2 where the edge occurs as a synchronization segment.

Dominant Spike Filtering

Using synchronization to filter short spike noise, as shown in the diagram below, where the spike begins at the end of PTS and lasts for (PTS+PBS1)

Figure 240 Short Dominant Spike Filtering



(1) When SJW ≥ phase error: The sampling point is shifted beyond the spike, and the sampled level is recessive (“1”).

(2) When SJW < phase error: Since the sampling point cannot be shifted far enough, the actual sampled bus level is the dominant spike.

38.5.22.4 Oscillator Tolerance

The oscillator tolerance range (the difference between the oscillator frequency fosc and the nominal frequency f_{nom} is defined as shown in the formula:

$$(df) \cdot f_{nom} \leq f_{osc} \leq (1 + df) \cdot f_{nom} \quad (1)$$

Where:

df: Tolerance range

f_{osc}: Oscillator frequency

f_{nom}: Nominal frequency

The maximum tolerance depends on the ratio of Phase Buffer Segment 1, Phase Buffer Segment 2, Synchronization Jump Width, and bit time, and the following two conditions must be met simultaneously

$$df \leq \frac{\min(T_{PBS1}, T_{PBS2})}{2\left((13 \times bit\ time) - T_{PBS2}\right)} \quad (2)$$

$$df \leq \frac{SJW}{20 \times bit\ time}$$

SJW: Synchronization jump width

TPBS1, TPBS2: Phase buffer segment time length

Note:

(1) SJW must be less than or equal to the smaller of the two phase buffer segments.

(2) The available bit time for phase buffer segments is limited by the propagation time segment.

For example, when PTS=1, PBS1=PBS2=SJW=4, the maximum oscillator tolerance based on formula (2) is 1.58%. In this configuration, the propagation time segment occupies only 10% of the bit timing, making it unsuitable for shorter bit times. This configuration is applicable to bit rates up to 125 kBit/s (with a bit time of 8μs) and a bus length of 40m.

38.5.22.5 Bit timing configuration

In CAN modules, the bit timing configuration is implemented through two bytes in the Bit Timing Register (CAN_BTIM):

(1) TSSPVP1 and TSSPVP2 form one byte. TSSPVP1 represents the sum of PTS and PBS1, while TSSPVP2 represents PBS2

(2) SJWP and BRPSC form another byte. In addition, the bit timing register also provides a byte, BRPSCEXT, for baud rate prescaler extension. Together with SJWP and BRPSC, it constitutes one byte.

The data in the bit timing register configures the input of the CAN protocol controller. TSSPVP1, TSSPVP2, and SJWP define the number of time quanta in the bit timing logic circuit, while the baud rate prescaler formed by BRPSC and BRPSCEXT determines the length of a time quantum. A time quantum serves as the basic time unit of bit timing.

In the bit timing register, the field values of TSSPVP1, TSSPVP2, SJWP, and BRPSC correspond to their respective functional values as shown in the following table:

Table 210 Correspondence between Field Values and Function Values

Function Values	Field Values
1	0
2	1
…	…
n	n-1

Therefore, the range of field values is 0-n-1, rather than 1-n. For example, if the function value range of SJWP is 1-4, the field value range is 0-3, meaning that only two bits are needed to represent the SJWP field value.

The write-in value and function value for bit time length are as follows:

(1) (Write-in value) Field value: [TSSPVP1 + TSSPVP2 + 3] * Tq

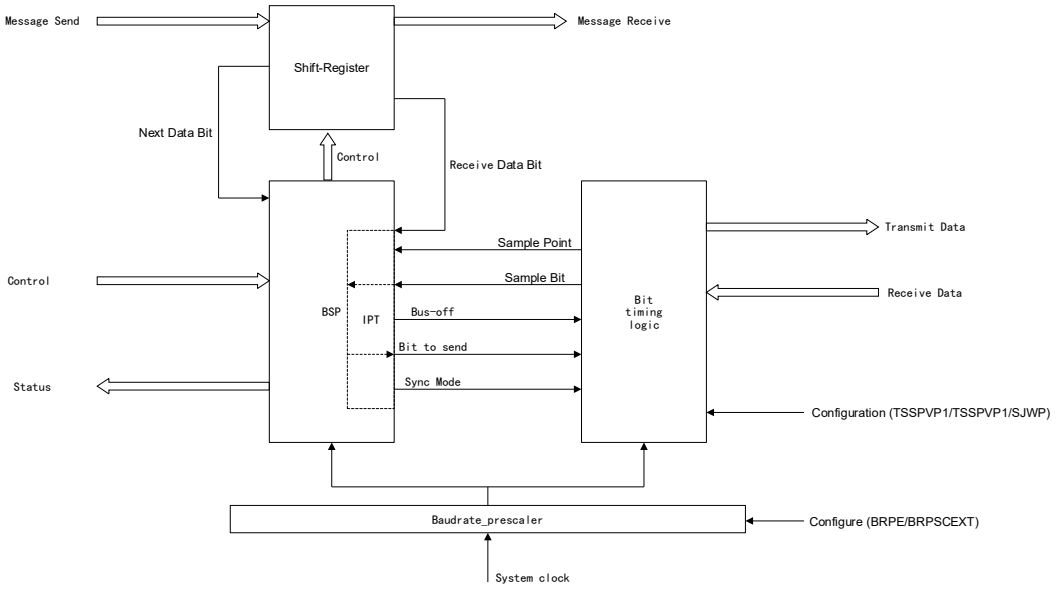
(2) Function value: [SS + PTS + PBS1 + PBS2] * Tq

The bit timing state machine of the CAN protocol controller is responsible for managing bit time processing, determining the sampling point position, and handling occasional synchronization.

Bit Stream Processor (BSP)

The Bit Stream Processor (BSP) state machine in the CAN protocol controller calculates once per sampling point for every bit time.

Figure 241 CAN Protocol Controller Structure



BSP functions are as follows:

(1) Frame conversion: The BSP can convert messages into frames and from frames to messages;

(2) Management of fixed format bits: The BSP can generate or discard fixed-format boundary bits of data.

(3) Stuffing bit insertion or extraction: The BSP inserts stuffing bits when needed, and identifies and removes them upon receiving;

(4) CRC calculation and check: The BSP calculates CRC codes during transmission and checks them during reception to ensure data integrity;

(5) Execution of error management;

(6) Synchronization type selection: The BSP determines which synchronization type to use to maintain clock synchronization

(7) Processing of bus input bits: The BSP evaluates at the sampling point and processes the sampled bus input bits.

(8) Information Processing Time (IPT) calculation: After the sampling point, the BSP must determine the time required to transmit the next bit (e.g., data bit, CRC bit, stuffing bit, error flag, or idle state), which is referred to as Information Processing Time (IPT). In CAN, this time is defined as 0 Time Quanta (Tq).

The IPT value may vary between different CAN controllers, but its length must not exceed 2*Tq. Additionally, the IPT length serves as the lower limit of the configured PBS2 length. During synchronization, the actual value of PBS2 may be smaller than IPT, but bus timing will remain unaffected.

38.5.22.6 Bit Timing Parameter Calculation

Calculating bit timing configuration typically begins with the desired bit rate or bit time, and the resulting bit time must be an integer multiple of the CAN clock cycle. The range for bit time is 8-25 time quanta (Tq). The formula for calculating time quanta is detailed in the Bit Timing section. The relationship between bit time and bit rate is given by the following formula:

$$\text{Bit time} = \frac{1}{\text{Bit rate}}$$

The required bit time can be achieved through various combinations, and the calculation can be repeated iteratively using the following steps:

(1) Calculate Propagation Time Segment (PTS) Length:The length of PTS is determined by the measured delay time in the system. For expandable CAN bus systems, the maximum bus length and maximum node delay time must be defined. The calculated PTS length must be converted into an integer multiple of Tq (rounded up).

(2) Calculate the length of the Synchronization Segment (SS): The SS length is a fixed value of 1*Tq.

(3) Calculate the lengths of the Phase Buffer Segments (PBS1 and PBS2): The total length of the phase buffer segments is (bit time - PTS - SS)*Tq.
- If the remaining value is an even number of Tq: PBS1 = PBS2
- If the remaining value is an odd number of Tq: PBS2 = PBS1 + 1

In addition, since PBS2 must be greater than or equal to the information processing time of any CAN controller in the network, which is determined by the device, the minimum nominal length range of PBS2 must be considered, i.e., (0-2)*Tq.

(1) Calculate the Synchronization Jump Width (SJW) length: The SJW length must be set to the maximum value, which is the lesser of PBS1 length and 4.

(2) Using the parameters obtained from the above steps, calculate the oscillator tolerance range. For details, please refer to the Oscillator Tolerance section. When multiple combinations can achieve the desired bit time, it is recommended to choose the combination with the maximum oscillator tolerance. Among all nodes, the node with the minimum oscillator tolerance range limits the oscillator tolerance range of the CAN system.

When CAN nodes have different clocks, different bit timing configurations must be used to achieve the same bit rate. In the CAN network, the delay time of the CAN network is calculated based on the node with the maximum delay. Across

the entire CAN network, only one delay time calculation is performed.

The above calculations indicate that it is required to lower the bit rate, reduce the bus length, or improve the oscillator frequency stability in order to achieve CAN bit timing that complies with the protocol. The parameters obtained from the above steps are written into the bit timing register:

(PBS2-1)&(PBS1+PTS-1)&(SJW-1)&(PSC-1)

Note: The minimum CAN clock frequency required for operation at 1Mbit/s is 8MHz.

Calculation Example - High Baud Rate

In this example, assume f_{CANCLK} = 10MHz, BRPSC = 0, and bit rate = 1Mbit/s.

Table 211 Example of High Baud Rate Bit Timing Configuration

Parameter	Calculation formula	Calculation result
Tq	tCANCLK	100 ns
Bus Delay (40m)	-	220 ns
Bus Driver Delay	-	90 ns
Receiver Circuit Delay	-	40 ns
tPTS	2 * Delay = 2 * (220 + 90 + 40)ns = 700ns = 7*Tq	700 ns
tSS	1* Tq	100 ns
tSJW	1* Tq	100 ns
tPBS1	tPTS+ tSJW	800 ns
tPBS2	tIPT+1* Tq	100 ns
Bit time	tSS + t PBS1 + t PBS1	1000ns
Oscillator Tolerance	min(tPBS1 , tPBS2)/2((13×bit time)-tPBS2)=100ns/2(13*1000ns-100ns)	0.35 %

Therefore, the time parameters written into the bit timing register are $(4\text{-}1)_3$ & $(8\text{-}1)_4$ & $(1\text{-}1)_2$ & $(1\text{-}1)_6$, i.e., writing 0x0000 0700 into the bit timing register.

Calculation Example - Low Baud Rate

In this example, assume f_{CANCLK} = 2 MHz, BRPSC = 1, and bit rate = 100 KBit/s.

Table 212 Example of Low Baud Rate Bit Timing Configuration

Parameter	Calculation formula	Calculation result
Tq	$2*t_{CANCLK}$	1 µs
Bus Delay (40m)	-	80 µs
Bus Driver Delay	-	200 µs

Parameter	Calculation formula	Calculation result
Receiver Circuit Delay	-	220 μs
t_{PTS}	1* Tq	1 μs
t_{SS}	1* Tq	1 μs
t_{SJW}	4* Tq	4 μs
t_{PBS1}	t_{PTS}+ t_{SJW}	5 μs
t_{PBS2}	t_{IPT}+4* Tq	4 μs
Bit time	t_{SS} + t_{PBS1} + t_{PBS1}	10 μs
Oscillator Tolerance	min(t_{PBS1} , t_{PBS2})/2((13×bit time)- t_{PBS2})= 4 μs /2(13*10μs -4μs)	1.58%

Therefore, the time parameters written into the bit timing register are (4-1)3 & (5-1)4 & (4-1)2 & (2-1)6, i.e., writing 0x0000 34C1 into the bit timing register.

38.5.23 Message Interface Register Banks

In CAN, there are three interface register banks used to control the CPU's read and write access to the message RAM. Where:

- The IF1 and IF2 register banks are used for both read and write access
- The IF3 register bank is used solely for read access

Due to the structural limitations of the message RAM, individual bits or bytes within a mailbox cannot be directly modified. When updating part of a mailbox, the entire mailbox in the message RAM must be accessed. Therefore, the message processor needs to perform read-modify-write operations on data transmitted from the IF1/IF2 registers to the message RAM. For parts of the mailbox that need to remain unchanged, the contents are read from the message RAM into the IF1 and IF2 interface register banks, and after updating, the entire interface register bank is written to the mailbox.

Partial write: After performing a partial write to the mailbox, the unselected parts of the interface register bank, not chosen by the command register, will be set to the actual content of the selected mailbox.

Partial read: After performing a partial read from the mailbox, the unselected parts of the interface register bank, not chosen by the command register, will remain unchanged.

Data Caching

To prevent conflicts between concurrent CAN message receive and transmit and CPU access to the message RAM, the data to be transmitted is cached. During a single transfer between the message RAM and the IF1/IF2 register banks, either all the contents or only part of a mailbox can be transmitted. By

transmitting all the selected parts of the data in parallel, the consistency of data within CAN messages is ensured.

Write Access Loss

Cause: While receiving messages from the same mailbox, an IFx register write to the message RAM occurs between the read - modify - write access of the master message processor. Therefore, when the accessed mailbox's MAILEN=1 (indicating that this mailbox is enabled) and CAN communication is ongoing, data transmission from the IFx registers to the message RAM may be lost.

How to avoid write access loss:

- For receive mailboxes, reset MsgVal before changing the MIDCFG, EXTIDCFG, MDIRCFG, DLCCFG, RXIEN, TXIEN, REMEN, EOBEN, MASKEN, IDMASK, EXTMASK, and DIRMASK fields
- For transmit mailboxes, reset MsgVal before changing the MDIRCFG, RXIEN, TXIEN, REMEN, EOBEN, MASKEN, IDMASK, EXTMASK, and DIRMASK fields

No write access loss to message RAM occurs when modifying fields other than those listed above.

38.5.23.1 IF1/12

The IF1 and IF2 register banks can interact with the mailboxes. These register interfaces can transmit either the entire mailbox or individual parts of the mailbox.

The IFx command register (CAN_IFxCOM) within the register bank controls the direction of data transfer:

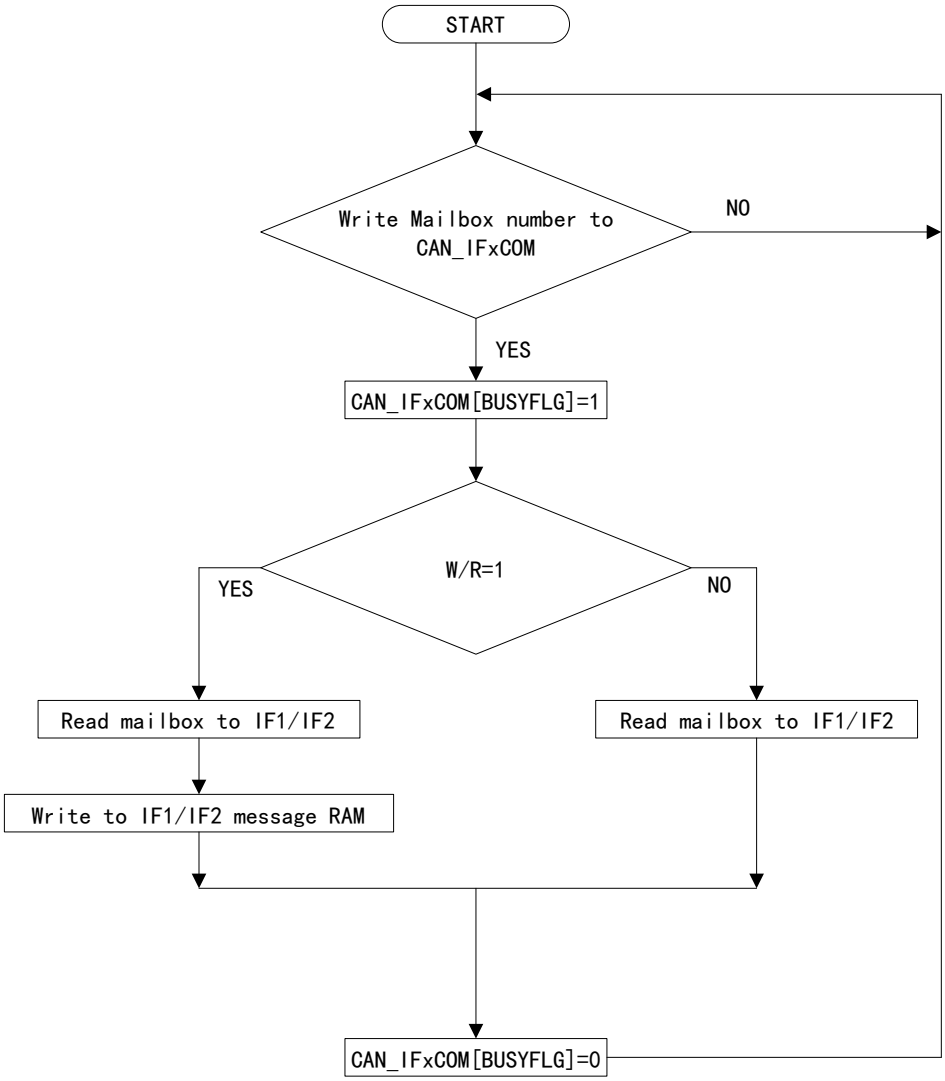
(1) If the CAN_IFxCOM register is set to read, the mailbox fields selected by the CAN_IFxCOM register are copied from the mailbox to the other IFx registers;

(2) If the CAN_IFxCOM register is set to write, the mailbox fields selected by the CAN_IFxCOM register are overwritten by the values from other IFx registers.

To initiate data transmission, write the target mailbox number into bits [7:0] of the CAN_IFxCOM register.

When the CPU initiates data transmission between the IF1/IF2 registers and the message RAM, the message processor sets the corresponding CAN_IFxCOM [BUSYFLG] bit to 1. Once the transmission is complete, the CAN_IFxCOM [BUSYFLG] bit is reset to 0, as shown below:

Figure 242 Data Interaction Process



38.5.23.2 **IF3**

In the IF3 register, an automatic update function can be configured for each mailbox (refer to the CAN_IF3UDEN Register for details). The IF3 register bank can automatically update the received mailbox into the registers without requiring the CPU to initiate message RAM transmission.

In all valid mailboxes within the message RAM, when the mailbox auto-update function is enabled, the NDATA flags of those mailboxes are checked to see if they have been activated. If a mailbox with an activated NDATA flag is detected and no DMA transmission is ongoing, that mailbox will be transmitted to the IF3 register, and controlled by the IF3 observation register (CAN_IF3OBS). If multiple mailboxes with activated NDATA flags are detected, the priority for automatic updates depends on mailbox numbers: The lower the mailbox number, the higher the priority; conversely, the higher the mailbox number, the lower the priority. Once a mailbox is transmitted to the IF3 register, the NDATA flag in that mailbox is reset.

The CAN DMA function can be enabled by setting CAN_CTRL[IFDEN3]. After the internal IF3 register bank in CAN is updated, a DMA request is triggered, and the DMA request remains active until the first read access to any register in the IF3 register bank.

Note: The IF3 register bank is only intended for read access and cannot be used to transmit data to mailboxes.

38.6 Register bank address

Table 213 CAN Register Bank Address

Device register	Register bank	Start address	End address
CanRegs	CAN_REGS	0x4000_2400	0x4000_2BFF

38.7 Register address mapping

Table 214 CAN_REGS Offset Address

Register name	Register description	Offset address	WRPRT
CAN_CTRL	Control register	0x00	-
CAN_EFLG	Error flag register	0x04	-
CAN_ECNT	Error count register	0x08	-
CAN_BTIM	Bit timing register	0x0C	-
CAN_INT	Interrupt register	0x10	-
CAN_TEST	Test Register	0x14	-
CAN_PE	Parity Error Register	0x1C	-
CAN_RAMINIT	RAM Initialization Register	0x40	-
CAN_GLBINTEN	Global interrupt enable register	0x50	-
CAN_GLBINTFLG	Global interrupt flag register	0x54	-
CAN_GLBINTCLR	Global interrupt clear register	0x58	-
CAN_ABOTMR	Auto-Bus-On Timing Register	0x80	-
CAN_TXREQFLG	Transmit Request Flag Register	0x84	-
CAN_TXREQ	Transmit request register	0x88	-
CAN_NDATAFLG	New Data Flag Register	0x98	-
CAN_NDATA	New data register	0x9C	-
CAN_IPENDFLG	Interrupt pending flag register	0xAC	-
CAN_IPEND	Interrupt pending register	0xB0	-
CAN_MVALFLG	Message Valid Flag Register	0xC0	-
CAN_MVAL	Mailbox Valid Register	0xC4	-

Register name	Register description	Offset address	WRPRT
CAN_INTMAIL	Interrupt mailbox register	0xD8	-
CAN_IFxCOM (x=1…2)	IFx Command Register	0x100+(x-1)*0x40	-
CAN_IFxMASK (x=1…3)	IFx Mask Register	0x104+(x-1)*0x40	-
CAN_IFxARB (x=1…3)	IFx Arbitration Register	0x108+(x-1)*0x40	-
CAN_IFxMCTRL (x=1…3)	IFx Message Control Register	0x10C+(x-1)*0x40	-
CAN_IFxDATAA (x=1..3)	IFx Data A Register	0x110+(x-1)*0x40	-
CAN_IFxDATAB (x=1…3)	IFx Data B Register	0x114+(x-1)*0x40	-
CAN_IF3OBS	IF3 Observation Register	0x140	-
CAN_IF3UDEN	IF3 Update Enable Register	0x160	-

38.8 Register functional description

38.8.1 Control register (CAN_CTRL)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	INIT	R/W	Initialization Mode This bit is automatically set during Bus-off events and used to keep the CAN module inactive during message RAM initialization and bit timing configuration. Clearing this bit will not shorten the bus recovery time. 0: Normal message processing 1: Ignore bus activity	0h
1	IEN0	R/W	Interrupt Line 0 Enable When enabled, CANINT0 will be set to 1 upon an interrupt and remain active until the interrupt event is handled. 0: Disable CANINT0 1: Enable CANINT0	0h
2	SCHIEN	R/W	Status Change Interrupt Enable This bit indicates whether TXCFLG, RXCFLG, and LEC can generate an interrupt on the CANINT0 interrupt line. 0: Disabled, interrupts cannot be generated on CANINT0 1: Enabled, interrupts can be generated on CANINT0	0h
3	ERRIEN	R/W	Error interrupt Enable This bit indicates whether PEFLG, BUSOFLG, and EWFLG can generate an interrupt on the CANINT0 interrupt line. 0: Disabled, interrupts cannot be generated on CANINT0	0h

Field	Name	R/W	Description	Reset value
			1: Enable; interrupts can be generated on CANINT0 and affect the CAN_INT register	
4	Reserved			0h
5	DISARETX	R/W	Disable Automatic ReTransmit 0: Enable automatic retransmission, and the message will be retransmitted automatically until it is transmitted successfully 1: Disable automatic retransmission	0h
6	CFGWACCEN	R/W	Configure Write Access Enable 0: CPU cannot write to configuration registers 1: When INIT is set to 1, the CPU can write to configuration registers	0h
7	TESTEN	R/W	Test Mode Enable 0: Disable (normal operation) 1: Enable	0h
8	IDBGSEN	R/W	Interrupt Debug Support Enable This bit indicates whether to interrupt ongoing transmit or receive when a debug mode request is received. 0: Wait until the currently initiated transmit or receive is complete before entering debug mode 1: Interrupt any ongoing transmit or receive and enter debug mode immediately	0h
9	ABOEN	R/W	Auto-Bus-On Enable 0: Disable 1: Enable	0h
13:10	PEN	R/W	Parity Enable 0101: Disable Other: Enable	0h
14	Reserved			0h
15	SWRSTEN	RC_W1	Software Reset Enable Software reset steps: Set the INIT bit to disable CAN communication, then set the SWRSTEN bit. Once set, this bit will be automatically cleared after one clock cycle during the software reset. Note: This bit is write-protected by the INIT bit. When the SWRSTEN bit is used for module reset, user configurations are not lost; only the status bits and logic that need to be reset for the next CAN transaction are reset. When the SOFTPRESx register is used for module reset, the entire CAN module, including configuration registers, is reset. 0: Normal operation 1: Force the CAN module to perform the software reset	0h
16	DBGFLG	R	Debug Mode Flag This bit indicates the internal initialization state of the debug mode.	0h

Field	Name	R/W	Description	Reset value
			0: Not in debug mode, or debug mode has been requested but not yet entered 1: Debug mode has been requested, and the CAN module is ready for debug access	
17	IEN1	R/W	Interrupt Line 1 Enable When enabled, CANINT1 will be set to 1 upon an interrupt and remain active until the interrupt event is handled. 0: Disable CANINT1 1: Enable CANINT1	0h
18	IFDEN1	R/W	IF1 DMA Request Line Enable Note: When this bit is set to 1, the pending DMA request is only activated on the first access to the IF1 registers. 0: Disable 1: Enable	0h
19	IFDEN2	R/W	IF2 DMA Request Line Enable Note: When this bit is set to 1, the pending DMA request is only activated on the first access to the IF2 registers. 0: Disable 1: Enable	0h
20	IFDEN3	R/W	IF3 DMA Request Line Enable Note: When this bit is set to 1, the pending DMA request is only activated on the first access to the IF3 registers. 0: Disable 1: Enable	0h
31:21	Reserved			0h

38.8.2 Error Flag Register (CAN_EFLG)

Offset address: 0x04

Reset type: SYSRSn

This register indicates the type of error in the CAN module. When CAN_CTRL[SCHIEN] is set, the TXCFLG, RXCFLG, and LEC bits can generate interrupts; when CAN_CTRL[ERRIEN] is set, the PEFLG, BUSOFLG, and EWFLG bits can generate interrupts. Changes in the EPFLG bit do not generate interrupts. The next lower-priority interrupt value will replace the status interrupt value (0x8000) in the interrupt register.

Reading this register automatically clears the TXCFLG, RXCFLG, and PEFLG bits and resets LEC to 0x07. In debug mode or suspension mode, the automatic clearing of this register (i.e., clearing the flag bits upon reading the register) is disabled.

Field	Name	R/W	Description	Reset value
2:0	LEC	R	Last Error Code This field indicates the type of the last error on the CAN bus. When a message is transmitted (received or transmitted) without errors, this field is cleared to 0. Each time the CPU reads the register, this field is reset to 0x07. 000: No error 001: Stuff error - More than five consecutive identical bits are detected in a received message (not allowed) 010: Form error - Format error in the fixed format part of a received frame 011: Acknowledge error - A message transmitted by the CAN core was not acknowledged by another node 100: Bit 1 error - During message transmit (except in the arbitration field), the device tried to transmit a recessive level (logic “1” bit), but the monitored bus level was dominant 101: Bit 0 error - During message transmit (or in the acknowledge bit, active error flag, or overrun flag), the device tried to transmit a dominant level (logic “0” bit), but the monitored bus level was recessive. During bus recovery, this state is set when a sequence of 11 consecutive recessive bits is detected. Thus, the CPU can monitor the progress of the bus recovery sequence (indicating the bus is not stuck at a dominant state or subjected to continuous interference) 110: CRC Error - CRC checksum error in the received message (the CRC of the received message does not match the calculated CRC for the received data) 111: No CAN bus event was detected since the last CPU read of the CAN_EFLG register. Any read access to the CAN_EFLG register will reset this bit to 111	7h
3	TXCFLG	R	Transmit Completion Flag This bit indicates the transmit status. This bit is not reset by internal events of CAN. 0: No message has been successfully transmitted since the last CPU read of TXCFLG 1: A message has been successfully transmitted (without errors) and acknowledged by at least one node since the last CPU read that cleared the TXCFLG bit	0h
4	RXCFLG	R	Reception Completion Flag This bit indicates the receive status. This bit is not reset by internal events of CAN. 0: No message has been successfully received since the last CPU read of RXCFLG 1: A message has been successfully received since the last CPU read that cleared the RXCFLG bit. This bit is set upon successful message receive regardless of the result of the receive filtering	0h

Field	Name	R/W	Description	Reset value
5	EPFLG	R	Error Passive Flag 0: When the CAN bus is in error, CAN can transmit active error frames 1: The CAN core is in the passive error state as defined by the CAN specification	0h
6	EWFLG	R	Error Warning Flag This bit indicates the error warning status, showing whether the error counters have reached the error warning limit of 96. 0: Neither error counter has reached the error warning limit 1: At least one error counter has reached the error warning limit	0h
7	BUSOFLG	R	Bus-off Flag This bit indicates whether the CAN module is in a Bus-off state. 0: Not in Bus-off state 1: In Bus-off state	0h
8	PEFLG	R	Parity Error Flag This bit indicates whether a parity error has been detected. 0: No parity error has been detected since the last CPU read access 1： In the message RAM, a parity check error was detected by the parity mechanism	0h
31:9	Reserved			0h

38.8.3 Error count register (CAN_ECNT)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXECNT	R	Transmit Error Counter This field indicates the actual value of the transmit error counter (0-255).	0h
14:8	RXECNT	R	Receive Error Counter This field indicates the actual value of the receive error counter (0-127).	0h
15	RXEP	R	Receive Error Passive The passive error level is defined by the CAN specification, and this bit indicates whether the receive error counter has exceeded the passive error level. 0: The receive error counter is below the passive error level 1: The receive error counter has reached the passive error level	0h
31:16	Reserved			0h

38.8.4 Bit timing register (CAN_BTIM)

Offset address: 0x0C

Reset type: SYSRSn

This field is used to configure the CAN bit timing parameters. This register can only be written to when CAN_CTRL[INIT] and [CFGWACCEN] are set.

The configurable value of CAN bit time: 8-25 bit time quanta.

The configurable value of CAN bit time quantum: 1-1024 CAN_CLK cycles

Field	Name	R/W	Description	Reset value
5:0	BRPSC	R/W	Baud Rate Prescaler This bit is used to generate the bit time quantum, with CAN_CLK divided by BRPSC's value equal to one bit time quantum. Bit time is a multiple of the bit time quantum. Valid programming values for BRPSC are 0-63. The actual value of BRPSC = BRPSC programming value + 1 Note: This bit is write-protected by the CAN_CTRL[CFGWACCEN] bit.	1h
7:6	SJWP	R/W	Synchronization Jump Width Valid Programmed Values This bit is used to configure the synchronization jump width, with the valid programming values of 0 to 3. SJWP actual value = SJWP programming value + 1 Note: This bit is write-protected by the CAN_CTRL[CFGWACCEN] bit.	0h
11:8	TSSPVP1	R/W	Time Segment 1 before the Sample Point Valid Programmed Values This bit is used to configure the valid programming value of Time Segment 1 before the sampling point, with valid values of 1 to 15. TSSPVP1 actual value = TSSPVP1 programming value + 1 Note: This bit is write-protected by the CAN_CTRL[CFGWACCEN] bit.	3h
14:12	TSSPVP2	R/W	Time Segment 2 before the Sample Point Valid Programmed Values This bit is used to configure the valid programming value of Time Segment 2 before the sampling point, with valid values of 0 to 7. TSSPVP2 actual value = TSSPVP2 programming value + 1 Note: This bit is write-protected by the CAN_CTRL[CFGWACCEN] bit.	2h
15	Reserved			0h
19:16	BRPSCEXT	R/W	Baud Rate Prescaler Extension This bit is used to configure the valid programming value of the extended baud rate prescaler, with valid values of 0 to 15. With extension, the baud rate prescaler value can reach 1024. BRPSCEXT actual value = BRPSCEXT programmed value + 1	0h

Field	Name	R/W	Description	Reset value
			Note: This bit is write-protected by the CAN_CTRL[CFGWACCEN] bit.	
31:17	Reserved			0h

38.8.5 Interrupt Register (CAN_INT)

Offset address: 0x10

Reset type: SYSRSn

This register is used to identify the interrupt source.

Field	Name	R/W	Description	Reset value
15:0	INT0	R	Interrupt 0 This field indicates the type of pending interrupt. When multiple interrupts are pending, the CAN_INT register points to the pending interrupt with the highest priority. Note: The state interrupt has the highest priority, and among message interrupts, the priority of mailbox interrupts decreases as mailbox numbers increase. The CANINT0 interrupt line remains active until CAN_INT[INT0]=0 or CAN_CTRL[IEN0] is cleared. 0x0000: No pending interrupt 0x0001-0x0020: Mailbox number that generated the interrupt 0x8000: CAN_EFLG register value is not 0x07 Others: Reserved	0h
23:16	INT1	R	Interrupt 1 This field indicates the type of pending interrupt. When multiple interrupts are pending, the CAN_INT register points to the pending interrupt with the highest priority. Note: Among message interrupts, the priority of mailbox interrupts decreases as mailbox numbers increase. Clear the message interrupt by clearing the CAN_IFxMCTRL[IPEND] bit. The CANINT1 interrupt line remains active until CAN_INT[INT1]=0 or CAN_CTRL[IEN1] is cleared. 0x00: No pending interrupt 0x01-0x20: Mailbox number that generated the interrupt Others: Reserved	0h
31:24	Reserved			0h

38.8.6 Test Register (CAN_TEST)

Offset address: 0x14

Reset type: SYSRSn

This register is used to configure various test options in the CAN module. Set CAN_CTRL[TESTEN]=1 to enable test mode. Once set, all fields in the CAN_TEST register, except RXMON, can be written to. Since the RXMON bit is used to monitor the state of the CANRX pin, it can only be read. Clearing the CAN_CTRL[TESTEN] bit disables all functions of the CAN_TEST register.

Note:

(1) When internal loopback mode (LBEN=1) is activated, the EXTLBEN setting is ignored;

(2) Setting TXCTRL to a value other than 00 will interfere with message transmission.

Field	Name	R/W	Description	Reset value
2:0	Reserved			0h
3	SLTEN	R/W	Silent Mode Enable 0: Disable 1: Enable	0h
4	LBEN	R/W	Loop Back Mode Enable 0: Disable 1: Enable	0h
6:5	TXCTRL	R/W	TX Pin Control This bit controls the behavior of the transmit pin. 00: Normal operation; the CAN core controls the CANTX pin 01: The sampling point can be detected on the CANTX pin 10: The CANTX pin drive is a dominant level (logic “0”) 11: The CANTX pin drive is a recessive level (logic “1”)	0h
7	RXMON	R	RX Pin Monitors This bit is used to monitor the actual value of the receive pin. 0: The CAN bus is a dominant level (logic “0”). 1: The CAN bus is a recessive level (logic “1”)	0h
8	EXTLBEN	R/W	External Loop Back Mode Enable 0: Disable 1: Enable	0h
9	RDAEN	R/W	RAM Direct Access Enable 0: Normal operation 1: Enable direct access to RAM	0h
31:10	Reserved			0h

38.8.7 Parity Error Register (CAN_PE)

Offset address: 0x1C

Reset type: SYSRSn

This register indicates the word number or mailbox number where a parity error was detected.

If a parity error occurs, CAN_EFLG[PEFLG] is set. CAN_EFLG[PEFLG] is not cleared by the parity mechanism and must be cleared by reading the CAN_EFLG register.

Additionally, this register indicates the memory region where the parity error occurred. When multiple words with parity errors are detected, this register only shows the highest-numbered word. Once a parity error is detected, the last error information remains in this register until the system powers down.

Field	Name	R/W	Description	Reset value
7:0	MAILNUM	R	Parity Error Mailbox Number 0x01-0x21: Mailbox with a parity error; the field value represents the mailbox number with the parity error.	0h
10:8	WORDNUM	R	Parity Error Word Number 0x01-0x21: Word with a parity error; the field value represents the word number with the parity error. According to the representation of message RAM in RAM Direct Access (RDA) mode, the RDA word numbers for mailboxes are 1-5.	1h
31:11	Reserved			0h

38.8.8 RAM Initialization Register (CAN_RAMINIT)

Offset address: 0x40

Reset type: SYSRSn

This register is used to initialize the mailbox RAM. During initialization, the entire mailbox RAM, including the MVAL bit, is cleared.

Field	Name	R/W	Description	Reset value
0	KEY0	R/W	Key 0 Write access to this register is valid only when KEY3-KEY0 is set to 1010. Note: Once the CAN RAM initialization is complete, KEY3-KEY0 will return to the reset value.	0h
1	KEY1	R/W	Key 1 Refer to KEY0 for details.	0h
2	KEY2	R/W	Key 2 Refer to KEY0 for details.	1h
3	KEY3	R/W	Key 3 Refer to KEY0 for details.	0h
4	RAMINIT	R/W	Initiate CAN Mailbox RAM Initialization This bit is used to read the initialization status of the CAN mailbox RAM and trigger initialization. Read operation: 0: Initialization completed or not started 1: Initialization is in progress Write operation: 0: No operation 1: Start the initialization of CAN mailbox RAM. After initialization, the RAMINIT bit will be automatically cleared.	0h
5	RAMINITCFLG	R	CAN Mailbox RAM Initialization Completion Flag 0: Initialization is in progress or not started 1: Initialized	0h
31:6	Reserved			0h

38.8.9 Global Interrupt Enable Register (CAN_GLBINTEN)

Offset address: 0x50

Reset type: SYSRSn

This register is used to enable global interrupts for CANINT0 and CANINT1.

Field	Name	R/W	Description	Reset value
0	GLBINT0EN	R/W	CANINT0 Global Interrupt Enable 0: CANINT0 does not issue interrupt requests to the NVIC 1: When an interrupt condition is detected, CANINT0 issues an interrupt request to the NVIC	0h
1	GLBINT1EN	R/W	CANINT1 Global Interrupt Enable 0: CANINT1 does not issue interrupt requests to the NVIC 1: When an interrupt condition is detected, CANINT1 issues an interrupt request to the NVIC	0h
31:2	Reserved			0h

38.8.10 Global Interrupt Flag Register (CAN_GLBINTFLG)

Offset address: 0x54

Reset type: SYSRSn

This register is used to indicate the interrupt status of CANINT0 and CANINT1.

Field	Name	R/W	Description	Reset value
0	GLBINT0FLG	R	CANINT0 Global Interrupt Flag 0: No CANINT0 interrupt 1: CANINT0 interrupt occurred	0h
1	GLBINT1FLG	R	CANINT1 Global Interrupt Flag 0: No CANINT1 interrupt 1: CANINT1 interrupt occurred	0h
31:2	Reserved			0h

38.8.11 Global Interrupt Clear Register (CAN_GLBINTCLR)

Offset address: 0x58

Reset type: SYSRSn

This register is used to clear the global interrupt flags for CANINT0 and CANINT1 in the CAN_GLBINTFLG register.

Field	Name	R/W	Description	Reset value
0	GLBINT0CLR	W	CANINT0 Global Interrupt Flag Clear 0: No effect 1: Clear CAN_GLBINTFLG[GLBINT0FLG] and allow the NVIC to receive another interrupt from CANINT0	0h
1	GLBINT1CLR	W	CANINT1 Global Interrupt Flag Clear 0: No effect 1: Clear CAN_GLBINTFLG[GLBINT1FLG] and allow the NVIC to receive another interrupt from CANINT1	0h

Field	Name	R/W	Description	Reset value
31:2	Reserved			0h

38.8.12 Auto-Bus-On Timer Register (CAN_ABOTMR)

Offset address: 0x80

Reset type: SYSRSn

This register is used to introduce a delay before starting the Bus-off recovery sequence, and the delay value is variable.

Field	Name	R/W	Description	Reset value
31:0	ABOTMR	R/W	Auto-Bus-On Timer The INIT bit is cleared to start the Bus-off recovery sequence. This field is used to specify the number of clock cycles to wait before the sequence begins. The clock refers to the input clock of the CAN module. The Auto-Bus-On timer function is enabled by setting CAN_CTRL[ABOEN], and the timer is implemented using a 32-bit counter. When the CAN enters the Bus-off state, the counter begins counting down from the preload value to zero. Once the countdown finishes, the counter is reloaded with the preload value of CAN_ABOTMR. Note: If the CAN module is in debug mode, the ABO timer pauses. If the CAN_CTRL register is written to while the ABO timer is running, the ABO process is aborted.	0h

38.8.13 Transmit Request Flag Register (CAN_TXREQFLG)

Offset address: 0x84

Reset type: SYSRSn

This register is used to indicate whether one or more bits in the CAN_TXREQ register are set. Each bit in this register represents one byte in the CAN_TXREQ register (i.e., 8 mailboxes per bank). When at least one bit in TXREQ is set, the corresponding bit in CAN_TXREQFLG is also set.

Field	Name	R/W	Description	Reset value
1:0	TXREQ1FLG	R	Transmit Request Register 1 Flag [0]: Represents the first byte (byte 0) in the CAN_TXREQ register. If at least one bit of byte 0 in the CAN_TXREQ register is set, this bit is also set [1]: Represents the second byte (byte 1) in the CAN_TXREQ register. If at least one bit of byte 1 in the CAN_TXREQ register is set, this bit is also set	0h
3:2	TXREQ2FLG	R	Transmit Request Register 2 Flag [2]: Represents the third byte (byte 2) in the CAN_TXREQ register. If at least one bit of byte 2 in the CAN_TXREQ register is set, this bit is also set	0h

Field	Name	R/W	Description	Reset value
			[3]: Represents the fourth byte (byte 13) in the CAN_TXREQ register. If at least one bit of byte 3 in the CAN_TXREQ register is set, this bit is also set	
31:4	Reserved			0h

38.8.14 Transmit Request Register (CAN_TXREQ)

Offset address: 0x88

Reset type: SYSRSn

This register is used to indicate whether a mailbox has a pending transmission request.

The TXREQ bit for a specific mailbox is set/reset as follows:

(1) The CPU is set or cleared by using the IF1/2 message interface register

(2) After successful transmission, it is set or cleared using the message processor

(3) After the remote frame is received, it is set or cleared using the message processor

Field	Name	R/W	Description	Reset value
31:0	TXREQ	R	Transmit Request Bit 0 corresponds to mailbox 1 Bit 1 corresponds to mailbox 2 … Bit 31 corresponds to mailbox 32 0: No transmission request for this mailbox 1: A pending transmission request exists for this mailbox and transmission is not yet complete	0h

38.8.15 New Data Flag Register (CAN_NDATAFLG)

Offset address: 0x98

Reset type: SYSRSn

This register is used to indicate whether one or more bits in the CAN_NDATA register are set. Each bit in this register represents one byte in the CAN_NDATA register (i.e., 8 mailboxes per bank). When at least one bit in NDATA is set, the corresponding bit in CAN_NDATAFLG is also set.

Field	Name	R/W	Description	Reset value
1:0	NDATA1FLG	R	New Data Register 1 Flag [0]: Represents the first byte (byte 0) in the CAN_DATA register. If at least one bit of byte 0 in the CAN_DATA register is set, this bit is also set [1]: Represents the first byte (byte 1) in the CAN_DATA register. If at least one bit of byte 1 in the CAN_DATA register is set, this bit is also set	0h

Field	Name	R/W	Description	Reset value
3:2	NDATA2FLG	R	New Data Register 2 Flag [2]: Represents the first byte (byte 2) in the CAN_DATA register. If at least one bit of byte 2 in the CAN_DATA register is set, this bit is also set [3]: Represents the first byte (byte 3) in the CAN_DATA register. If at least one bit of byte 3 in the CAN_DATA register is set, this bit is also set	0h
31:4	Reserved			0h

38.8.16 New data register (CAN_NDATA)

Offset address: 0x9C

Reset type: SYSRSn

This register indicates which data bytes of the mailbox have been updated.

The NDATA bit for a specific mailbox is set/reset as follows:

(1) The CPU is set or cleared by using the IFx message interface register

(2) After successful transmission, it is set or cleared using the message processor

(3) After the remote frame is received, it is set or cleared using the message processor

Field	Name	R/W	Description	Reset value
31:0	NDATA	R	New Data Bit 0 corresponds to mailbox 1 Bit 1 corresponds to mailbox 2 … Bit 31 corresponds to mailbox 32 0: Data not updated (no new data has been written to the data section of this mailbox by the message processor since this flag was last cleared). 1: New data (written by the CPU or message processor) is present in the data section of this mailbox.	0h

38.8.17 Interrupt Pending Flag Register (CAN_IPENDFLG)

Offset address: 0xAC

Reset type: SYSRSn

This register is used to indicate whether one or more bits in the CAN_IPEND register are set. Each bit in this register represents one byte in the CAN_IPEND register (i.e., 8 mailboxes per bank). When at least one bit in NDATA is set, the corresponding bit in CAN_IPENDFLG is also set.

Field	Name	R/W	Description	Reset value
1:0	IPEND1FLG	R	Interrupt Pending Register 1 Flag	0h

Field	Name	R/W	Description	Reset value
			[0]: Represents the first byte (byte 0) in the CAN_IPEND register. If at least one bit of byte 0 in the CAN_IPEND register is set, this bit is also set [1]: Represents the second byte (byte 1) in the CAN_IPEND register. If at least one bit of byte 1 in the CAN_IPEND register is set, this bit is also set	
3:2	IPEND2FLG	R	Interrupt Pending Register 2 Flag [2]: Represents the third byte (byte 2) in the CAN_IPEND register. If at least one bit of byte 2 in the CAN_IPEND register is set, this bit is also set [3]: Represents the fourth byte (byte 3) in the CAN_IPEND register. If at least one bit of byte 3 in the CAN_IPEND register is set, this bit is also set	0h
31:4	Reserved			0h

38.8.18 Interrupt pending register (CAN_IPEND)

Offset address: 0xB0

Reset type: SYSRSn

This register indicates which mailboxes have pending interrupts.

The IPEND bit for a specific mailbox is set/reset as follows:

(1) The CPU is set or cleared by using the IFx message interface register

(2) After successful transmission, it is set or cleared using the message processor

(3) After the remote frame is received, it is set or cleared using the message processor

Field	Name	R/W	Description	Reset value
31:0	IPEND	R/W	Interrupt Pending Bit 0 corresponds to mailbox 1 Bit 1 corresponds to mailbox 2 … Bit 31 corresponds to mailbox 32 0: No pending interrupt for this mailbox 1: Pending interrupt for this mailbox	0h

38.8.19 Mailbox Valid Flag Register (CAN_MVALFLG)

Offset address: 0xC0

Reset type: SYSRSn

This register is used to indicate whether one or more bits in the CAN_MVAL register are set. Each bit in this register represents one byte in the CAN_MVAL register (i.e., 8 mailboxes per bank). When at least one bit in MVAL is set, the corresponding bit in CAN_MVALFLG is also set.

Field	Name	R/W	Description	Reset value
1:0	MVAL1FLG	R	Message Valid Register 1 Flag [0]: Represents the first byte (byte 0) in the CAN_MVAL register. If at least one bit of byte 0 in the CAN_MVAL register is set, this bit is also set [1]: Represents the second byte (byte 1) in the CAN_MVAL register. If at least one bit of byte 1 in the CAN_MVAL register is set, this bit is also set	0h
3:2	MVAL2FLG	R	Message Valid Register 2 Flag [2]: Represents the third byte (byte 2) in the CAN_MVAL register. If at least one bit of byte 2 in the CAN_MVAL register is set, this bit is also set [3]: Represents the fourth byte (byte 3) in the CAN_MVAL register. If at least one bit of byte 3 in the CAN_MVAL register is set, this bit is also set	0h
31:4	Reserved			0h

38.8.20 Mailbox Valid Register (CAN_MVAL)

Offset address: 0xC4

Reset type: SYSRSn

This register is used to indicate whether a mailbox is valid.

The CPU sets or clears the MVAL bit of the specific mailbox by using the IF1/2 message interface register

Field	Name	R/W	Description	Reset value
31:0	MVAL	R	Mailbox Valid Bit 0 corresponds to mailbox 1 Bit 1 corresponds to mailbox 2 … Bit 31 corresponds to mailbox 32 0: This mailbox is ignored by the message processor 1: This mailbox is configured and processed by the message processor	0h

38.8.21 CAN Interrupt Mailbox Register (CAN_INTMAIL)

Offset address: 0xD8

Reset type: SYSRSn

This register determines which interrupt line (CANINT0 or CANINT1) is asserted when each mailbox sets CAN_IPEND[IPEND]. When CAN_CTRL[IEN0] is set, CANINT0 is globally enabled; when CAN_CTRL[IEN0] = 0 is cleared, CANINT0 is globally disabled. The same applies to CANINT1.

Interrupts in the CAN_INT register. When the interrupt line CANINT0 or CANINT1 is triggered, CAN_INT[IINT0] or CAN_INT[IINT1] will also be set to the value corresponding to the mailbox number.

Field	Name	R/W	Description	Reset value
31:0	INTMAIL	R/W	Interrupt Mailbox Bit 0 corresponds to mailbox 1 Bit 1 corresponds to mailbox 2 … Bit 31 corresponds to mailbox 32 0: When the corresponding CAN_IPENDFLG[IPENDFLG] = 1, the interrupt line CANINT0 is asserted 1: When the corresponding CAN_IPENDFLG[IPENDFLG] = 1, the interrupt line CANINT1 is asserted	0h

38.8.22 IFx Command Register (CAN_IFxCOM) (x=1…2)

Offset Address: 0x100 + (x-1) * 0x40

Reset type: SYSRSn

This register is used to configure and initiate data transmission between the message RAM and the IF1/IF2 register banks, allowing configuring mailboxes that need to be transmitted.

Initiate Transmission: A transmission is initiated when the CPU writes a mailbox number to CAN_IFxCOM[VALMNUM]. This write operation automatically sets the CAN_IFxCOM[BUSYFLG] bit to indicate that the transmission is in progress.

Transmission Completion: After initiating the transmission, the transmission between the message RAM and the interface register will complete in 4-14 clock cycles, and the CAN_IFxCOM[BUSYFLG] bit will be cleared. When message transmission overlaps with transmitting, receiving, filtering, or storing messages, the required clock cycles increase to the maximum.

Continuous Write: If you continuously write mailbox numbers to the CAN_IFxCOM register (i.e., request the second transmission while the first one is still in progress), the second transmission will start after the first one completes.

Note:

(1) This register shall be written as a 32-bit value or by writing the high 16 bits first, followed by the low 16 bits.

(2) Writing all zeros to the entire register is prohibited.

(3) Debug Support: In debug mode or suspension mode, the automatic clearing of this register (i.e., clearing the REQDMAIFx bit upon reading/writing the register) is disabled.

(4) Write Protection: When CAN_IFxCOM[BUSYFLG] is set, data cannot be written to the IF1/IF2 register bank (write protection).

(5) Invalid Mailbox Number: If an invalid mailbox number is written to CAN_IFxCOM[VALMNUM], the message processor may access a implemented (valid) mailbox.

Field	Name	R/W	Description	Reset value
7:0	VALMNUM	R/W	Valid Mailbox Number This field indicates the mailbox number in the message RAM used for data transmission. This field is write-protected by the BUSYFLG bit. 0x01-0x20: Valid mailbox numbers for data transmission Others: Invalid mailbox numbers	1h
13:8	Reserved			0h
14	REQDMAIFx	R/W	Request DMA after IFx Update This field indicates the trigger state of a DMA request after the IFx update. When this bit is set to 1, writing to the VALMNUM bit will not cancel the active DMA request state. This field is write-protected by the BUSYFLG bit. Auto-reset Feature: This bit has the auto-reset function. That is, after each DMA request is completed, this bit will be automatically reset to 0, so it must be set individually for each DMA cycle. 0: No activated IFx DMA request 1: After the message RAM to IFx transmission is complete, the DMA request is activated. Once the DMA request is activated, it remains active until the first read or write operation on the IFx register	0h
15	BUSYFLG	R	Busy Flag When the CPU writes a mailbox number to VALMNUM, this bit is automatically set, enabling write protection on the IFx register bank. It is automatically cleared upon completion of the transmission. 0: No data transmission is in progress between the message RAM and the IFx register bank 1: Data transmission is in progress between the message RAM and the IFx register bank	0h
16	DATABYTEB	R/W	Access Data Bytes 4-7 This field is write-protected by the BUSYFLG bit. Note: The duration of message transmission is not related to the number of bytes being transmitted. 0: No operation 1: Read operation: Data bytes 4-7 are transmitted from the mailbox addressed by the mailbox number written to VALMNUM to the corresponding IFx register bank. Write operation: Data bytes 4-7 are transmitted from the IFx register bank to the mailbox addressed by the mailbox number written to VALMNUM	0h
17	DATABYTEA	R/W	Access Data Bytes 0-3 This field is write-protected by the BUSYFLG bit. Note: The duration of message transmission is not related to the number of bytes being transmitted. 0: No operation	0h

Field	Name	R/W	Description	Reset value
			1: Read operation: Data bytes 0-3 are transmitted from the mailbox addressed by the mailbox number written to VALMNUM to the corresponding IFx register bank. Write operation: Data bytes 0-3 are transmitted from the IFx register bank to the mailbox addressed by the mailbox number written to VALMNUM	
18	TXREQCFG	R/W	Transmit Request/ New Data Configure This field is write-protected by the BUSYFLG bit. Note: When reading a mailbox, this can be combined with resetting the control bits CAN_IPEND[IPEND] and CAN_NDATA[NDATA]. The values of these bits transmitted to the IFx message control register always reflect the pre-reset state. When requesting a CAN transmission by setting TXREQCFG in this register, the CAN_TXREQ[TXREQ]/CAN_NDATA[NDATA] bits in the mailbox will be set to 1, regardless of the value in the IFx message control register. Read operation: 0: No operation 1: Clear the CAN_NDATA[NDATA] bit in the mailbox Write operation: 0: Process the CAN_TXREQ[TXREQ]/CAN_NDATA[NDATA] bits according to the ACCCTRL bit configuration 1: Set the CAN_TXREQ[TXREQ]/CAN_NDATA[NDATA] bits in the mailbox	0h
19	IPENDCLR	R/W	Interrupt Pending Clear This field is write-protected by the BUSYFLG bit. 0: No operation 1: Read Operation: Clear the CAN_IPEND[IPEND] bit in the mailbox; Write Operation: This bit is ignored	0h
20	ACCCTRL	R/W	Access Control When TXREQCFG is set, the CAN_IFxMCTRL[TXREQEN] and CAN_IFxMCTRL[NDATA] bits are ignored This field is write-protected by the BUSYFLG bit. 0: No operation 1: Read operation: The bits in CAN_IFxMCTRL are transmitted from the mailbox addressed by the mailbox number written to VALMNUM to the corresponding IFx register bank. Write operation: The bits in CAN_IFxMCTRL are transmitted from the IFx register bank to the mailbox addressed by the mailbox number written to VALMNUM	0h
21	ACCARB	R/W	Access Arbitration This field is write-protected by the BUSYFLG bit. 0: No operation 1: Read operation: The bits in CAN_IFxARB are transmitted from the mailbox addressed by the mailbox number written to VALMNUM to the corresponding IFx	0h

Field	Name	R/W	Description	Reset value
			register bank. Write operation: The bits in CAN_IFxARB are transmitted from the IFx register bank to the mailbox addressed by the mailbox number written to VALMNUM	
22	ACCMASK	R/W	Access Mask This field is write-protected by the BUSYFLG bit. 0: No operation 1: Read operation: The bits in CAN_IFxMASK are transmitted from the mailbox addressed by the mailbox number written to VALMNUM to the corresponding IFx register bank. Write operation: The bits in CAN_IFxMASK are transmitted from the IFx register bank to the mailbox addressed by the mailbox number written to VALMNUM	0h
23	RWCFG	R/W	Read/Write Function Configure This bit indicates the data transmission direction, and the functions of other bits in the CAN_IFxCOM register are related to the data transmission direction. This field is write-protected by the BUSYFLG bit. 0: Perform a read operation, i.e., transmitting data from the mailbox to the selected IFx message buffer registers 1: Perform a write operation, transmitting data from the IFx message buffer registers to the mailbox addressed by the mailbox number written to VALMNUM	0h
31:24	Reserved			0h

38.8.23 IFx Mask Register (CAN_IFxMASK) (x=1…2)

Offset address: 0x104+(x-1)*0x40

Reset type: SYSRSn

This register is used to configure mailbox masks.

Note: When CAN_IFxCOM[BUSYFLG]=1 (x=1…2), write access to the IFx (x=1…2) register bank is not allowed.

Field	Name	R/W	Description	Reset value
28:0	IDMASK	R/W	Identifier Mask This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: The corresponding bit in the mailbox identifier is not used for receive filtering (i.e., it can be any value) 1: The corresponding bit in the mailbox identifier is used for receive filtering	1FFFFFFFh
29	Reserved			1h
30	DIRMASK	R/W	Message Direction Mask This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: The CAN_IFxARB[MDIRCFG] bit is not used for receive filtering	1h

Field	Name	R/W	Description	Reset value
			1: The CAN_IFxARB[MDIRCFG] bit is used for receive filtering	
31	EXTMASK	R/W	Extended Identifier Mask When the mailbox uses an 11-bit standard format identifier, the 11-bit identifier of the received data frame is placed in MIDCFG[28:18]. During receive filtering, only MIDCFG[28:18] and the corresponding IDMASK[28:18] bits are considered. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: The CAN_IFxARB[EXTIDCFG] bit is not used for receive filtering 1: The CAN_IFxARB[EXTIDCFG] bit is used for receive filtering	1h

38.8.24 IFx Arbitration Register (CAN_IFxARB) (x=1…2)

Offset Address: 0x108+(x-1)*0x40

Reset type: SYSRSn

This register is used to configure the arbitration information for mailboxes. The MIDCFG, MDIRCFG, and EXTIDCFG bits define the type and identifier of the message to be transmitted and are used together with the corresponding bits in the CAN_IFxMASK register for receive filtering of incoming messages.

When a received message is stored in a valid mailbox with a matching identifier, it is the Data Frame if the transmission is received; it is the Remote Frame if the transmission is transmitted.

When a received message (Data Frame or Remote Frame) matches multiple valid mailboxes, it will be stored in the mailbox with the lowest number.

Note: When CAN_IFxCOM[BUSYFLG]=1 (x=1...2), write access to the IFx (x=1...2) register bank is not allowed.

Field	Name	R/W	Description	Reset value
28:0	MIDCFG	R/W	Message Identifier Configure 11-bit identifier - Standard Frame: MIDCFG[28:18] 29-bit identifier - Extended Frame: MIDCFG[28:0] This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit.	0h
29	MDIRCFG	R/W	Message Direction Configure This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Receive: When TXREQEN=1, a remote frame is transmitted, carrying an identifier that matches this mailbox. When a data frame is received and its identifier matches that of the mailbox, the data frame is stored in the mailbox 1: Transmit: When TXREQEN=1, a data frame is transmitted. When a remote frame is received and its	0h

Field	Name	R/W	Description	Reset value
			identifier matches that of the mailbox; if REMEN=1, TXREQEN of this mailbox is set to 1.	
30	EXTIDCFG	R/W	Extended Identifier Configure This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: The mailbox uses an 11-bit standard identifier 1: The mailbox uses a 29-bit extended identifier	0h
31	MAILEN	R/W	Mailbox Enable When the mailbox is disabled, the message processor ignores it, and it will not be considered for receiving or transmitting CAN messages. When the mailbox is enabled, it will be used by the message processor for receiving or transmitting operations. During CAN initialization, the CPU shall clear the CAN_MVAL[MVAL] bit for all unused mailboxes. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Disable 1: Enable	0h

38.8.25 IFx Message Control Register (CAN_IFxMCTRL) (x=1…2)

Offset Address: 0x10C+(x-1)*0x40

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	DLCCFG	R/W	Data Length Code Configure Note: Mailboxes of different nodes with the same identifier need to be set with the same data length code. When the message processor receives a data frame, it sets the DLC in the corresponding mailbox according to the received message. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0000-1000: The data frame has 0-8 data bytes 1001-1111: The data frame has 8 data bytes	0h
6:4	Reserved			0h
7	EOBEN	R/W	End of Block Enable This field is used to cascade multiple mailboxes to construct a FIFO buffer. When used as a standalone mailbox, this bit must always be set to 1. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: The current mailbox is part of a FIFO buffer block but not the last mailbox in the block. 1: The current mailbox is a standalone mailbox, not belonging to any FIFO buffer block, or the current mailbox is the last mailbox in a FIFO buffer block	0h
8	TXREQEN	R/W	Transmit Request Enable	0h

Field	Name	R/W	Description	Reset value
			This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: No pending transmission request for this mailbox 1: A pending transmission request exists for this mailbox and transmission is not yet complete	
9	REMEN	R/W	Remote Enable This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: TXREQEN is not changed after a remote frame is received 1: TXREQEN is set after a remote frame is received	0h
10	RXIEN	R/W	Receive Interrupt Enable This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Do not trigger IPEND after successfully receiving frame data 1: Trigger IPEND after successfully receiving frame data	0h
11	TXIEN	R/W	Transmit Interrupt Enable This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Do not trigger IPEND after successfully transmitting frame data 1: Trigger IPEND after successfully transmitting frame data	0h
12	MASKEN	R/W	Mask Enable When this bit is set to 1, the mailbox's MASK fields (IDMASK, EXTMASK, DIRMASK) must be configured during mailbox initialization before setting CAN_MVAL[MVAL]. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Ignore the mask 1: Use the mask (i.e., the field in the CAN_IFxMASK register) for receive filtering	0h
13	IPEND	R/W	Interrupt Pending This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: No pending interrupt for this mailbox 1: This mailbox has a pending interrupt. When no other interrupt source has a higher priority than this mailbox, the interrupt register's interrupt identifier will point to this mailbox	0h
14	MLOST	R/W	Message Lost This field applies only to mailboxes configured for the receive direction. This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: No message loss occurred since the CPU last cleared this bit.	0h

Field	Name	R/W	Description	Reset value
			1: A message loss occurred, i.e., the message processor received a new message while CAN_NDATA[NDATA] was set and stored it in this mailbox, causing the previous message to be overwritten by the new one.	
15	NDATA	R/W	New Data This bit is write-protected by the CAN_IFxCOM[BUSYFLG] bit. 0: Since the CPU last cleared this bit, no new data has been written to the data section of this mailbox 1: New data has been written, i.e., the data section of this mailbox contains new data (written by the message processor or CPU)	0h
31:16	Reserved			0h

38.8.26 IFx Data A Register (CAN_IFxDATAA) (x=1…2)

Offset address: 0x110+(x-1)*0x40

Reset type: SYSRSn

This register is used to provide a window for accessing CAN message data bytes. The data bytes are arranged in the transmit and receive order of the CAN data frame. DATD0 is the first byte, and DATD7 is the last one. In the CAN serial bit stream, the most significant bit (MSB) of each byte is transmitted first.

Note: When CAN_IFxCOM[BUSYFLG]=1 (x=1…2), no write access is allowed to all bits in this register.

Field	Name	R/W	Description	Reset value
7:0	DATD0	R/W	Data Byte 0	0h
15:8	DATD1	R/W	Data Byte 1	0h
23:16	DATD2	R/W	Data Byte 2	0h
31:24	DATD3	R/W	Data Byte 3	0h

38.8.27 IFx Data B Register (CAN_IFxDATAB) (x=1…2)

Offset address: 0x114+(x-1)*0x40

Reset type: SYSRSn

This register is used to provide a window for accessing CAN message data bytes. The data bytes are arranged in the transmit and receive order of the CAN data frame. DATD0 is the first byte, and DATD7 is the last one. In the CAN serial bit stream, the most significant bit (MSB) of each byte is transmitted first.

Note: When CAN_IFxCOM[BUSYFLG]=1 (x=1…2), no write access is allowed to all bits in this register.

Field	Name	R/W	Description	Reset value
7:0	DATD4	R/W	Data Byte 4	0h
15:8	DATD5	R/W	Data Byte 5	0h

Field	Name	R/W	Description	Reset value
23:16	DATD6	R/W	Data Byte 6	0h
31:24	DATD7	R/W	Data Byte 7	0h

38.8.28 IF3 Observation Register (CAN_IF3OBS)

Offset address: 0x140

Reset type: SYSRSn

The IF3 register bank is automatically updated with newly received mailboxes without requiring CPU-initiated transmission from the message RAM to the IF3 register bank.

Bits [4:0] in this register serve as observation flags, indicating which data sections of the IF3 register bank need to be read to complete a DMA read cycle. When all marked data sections have been read, enable CAN and update the IF3 register bank with the new data. Bits [12:8] are used to observe the status of the current read cycle. Applications can use Bits [12:8] and UDDATA to receive notifications about new IF3 content in either polling mode or interrupt mode. Writing to the IF3 register interrupts an ongoing DMA cycle, resets the DMA request line, and allows the IF3 interface register bank to be updated with new data. To ensure data consistency, disable DMA before reconfiguring the CAN_IFxOBS register. When the IF3 update enable mechanism is used and observation flags are not set, the corresponding mailbox is copied to IF3 without activating the DMA request line or waiting for DMA read access.

Any sequence of single-byte or half-word accesses is allowed. When accessing by byte or half-word, if all bytes of a data section are read, that data section is marked as completed.

Field	Name	R/W	Description	Reset value
0	MASKRD	R/W	Mask Data Read 0: Do not read mask data 1: Read mask data for the next IF3 update	0h
1	ARBRD	R/W	Arbitration Data Read 0: Do not read arbitration data 1: Read arbitration data for the next IF3 update	0h
2	CTRLRD	R/W	Control Bits Read 0: Do not read control bit data 1: Read control bit data for the next IF3 update	0h
3	DATAARD	R/W	Data A Read 0: Do not read Data A 1: Read Data A for the next IF3 update	0h
4	DATABRD	R/W	Data B Read 0: Do not read Data B 1: Read Data B for the next IF3 update	0h
7:5	Reserved			0h

Field	Name	R/W	Description	Reset value
8	MASKRDACC	R	Mask Data Read Access Status 0: All mask data bytes have been read, or the mask data bytes are not marked for reading 1: There is data in the mask data bytes that still needs to be read	0h
9	ARBRDACC	R	Arbitration Data Read Access Status 0: All mask data bytes have been read, or the mask data bytes are not marked for reading 1: There is data in the mask data bytes that still needs to be read	0h
10	CTRLRDACC	R	Data Bits Read Access Status 0: All control bit data bytes have been read, or the control bit data bytes are not marked for reading 1: There is data in the control bit data bytes that still needs to be read	0h
11	DATAARDACC	R	Data A Read Access Status 0: All Data A bytes have been read, or Data A bytes are not marked for reading 1: There is data in Data A bytes that still needs to be read	0h
12	DATABRDACC	R	Data B Read Access Status 0: All Data B bytes have been read, or Data B bytes are not marked for reading 1: There is data in Data B bytes that still needs to be read	0h
14:13	Reserved			0h
15	UDDATA	R	Update Data This field indicates whether new data has been loaded into the IF3 register bank since the last time it was read 0: No new data loaded 1: New data loaded	0h
31:16	Reserved			0h

38.8.29 IF3 Mask Register (CAN_IF3MASK)

Offset address: 0x144

Reset type: SYSRSn

This register is used to configure mailbox masks.

Field	Name	R/W	Description	Reset value
28:0	IDMASK	R/W	Identifier Mask 0: The corresponding bit in the mailbox identifier is not used for receive filtering (i.e., it can be any value) 1: The corresponding bit in the mailbox identifier is used for receive filtering	1FFFFFFFh
29	Reserved			1h

Field	Name	R/W	Description	Reset value
30	DIRMASK	R/W	Message Direction Mask 0: The CAN_IF3ARB[MDIRCFG] bit is not used for receive filtering 1: The CAN_IF3ARB[MDIRCFG] bit is used for receive filtering	1h
31	EXTMASK	R/W	Extended Identifier Mask When the mailbox uses an 11-bit standard format identifier, the 11-bit identifier of the received data frame is placed in MIDCFG[28:18]. During receive filtering, only MIDCFG[28:18] and the corresponding IDMASK[28:18] bits are considered. 0: The CAN_IF3ARB[EXTIDCFG] bit is not used for receive filtering 1: The CAN_IF3ARB[EXTIDCFG] bit is used for receive filtering	1h

38.8.30 IF3 Arbitration Register (CAN_IF3ARB)

Offset address: 0x148

Reset type: SYSRSn

This register is used to configure the arbitration information for mailboxes.

Field	Name	R/W	Description	Reset value
28:0	MIDCFG	R/W	Message Identifier Configure 11-bit identifier - Standard Frame: MIDCFG[28:18] 29-bit identifier - Extended Frame: MIDCFG[28:0]	0h
29	MDIRCFG	R/W	Message Direction Configure 0: Receive: When TXREQEN=1, a remote frame is transmitted, carrying an identifier that matches this mailbox. When a data frame is received and its identifier matches that of the mailbox, the data frame is stored in the mailbox 1: Transmit: When TXREQEN=1, a data frame is transmitted. When a remote frame is received and its identifier matches that of the mailbox; if REMEN=1, TXREQEN of this mailbox is set to 1.	0h
30	EXTIDCFG	R/W	Extended Identifier Configure 0: The mailbox uses an 11-bit standard identifier 1: The mailbox uses a 29-bit extended identifier	0h
31	MAILEN	R/W	Mailbox Enable When the mailbox is disabled, the message processor ignores it, and it will not be considered for receiving or transmitting CAN messages. When the mailbox is enabled, it will be used by the message processor for receiving or transmitting operations. During CAN initialization, the CPU shall clear the CAN_MVAL[MVAL] bit for all unused mailboxes. 0: Disable 1: Enable	0h

38.8.31 IF3 Message Control Register (CAN_IF3MCTRL)

Offset address: 0x14C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
3:0	DLCCFG	R/W	Data Length Code Configure Note: Mailboxes of different nodes with the same identifier need to be set with the same data length code. When the message processor receives a data frame, it sets the DLC in the corresponding mailbox according to the received message. 0000-1000: The data frame has 0-8 data bytes 1001-1111: The data frame has 8 data bytes	0h
6:4	Reserved			0h
7	EOBEN	R/W	End of Block Enable This field is used to cascade multiple mailboxes to construct a FIFO buffer. When used as a standalone mailbox, this bit must always be set to 1. 0: The current mailbox is part of a FIFO buffer block but not the last mailbox in the block. 1: The current mailbox is a standalone mailbox, not belonging to any FIFO buffer block, or the current mailbox is the last mailbox in a FIFO buffer block	0h
8	TXREQEN	R/W	Transmit Request Enable 0: No pending transmission request for this mailbox 1: A pending transmission request exists for this mailbox and transmission is not yet complete	0h
9	REMEN	R/W	Remote Enable 0: TXREQEN is not changed after a remote frame is received 1: TXREQEN is set after a remote frame is received	0h
10	RXIEN	R/W	Receive Interrupt Enable 0: Do not trigger IPEND after successfully receiving frame data 1: Trigger IPEND after successfully receiving frame data	0h
11	TXIEN	R/W	Transmit Interrupt Enable 0: Do not trigger IPEND after successfully transmitting frame data 1: Trigger IPEND after successfully transmitting frame data	0h
12	MASKEN	R/W	Mask Enable When this bit is set to 1, the mailbox's MASK fields (IDMASK, EXTMASK, DIRMASK) must be configured during mailbox initialization before setting CAN_MVAL[MVAL]. 0: Ignore the mask 1: Use the mask (i.e., the field in the CAN_IF3MASK register) for receive filtering	0h
13	IPEND	R/W	Interrupt Pending	0h

Field	Name	R/W	Description	Reset value
			0: No pending interrupt for this mailbox 1: This mailbox has a pending interrupt. When no other interrupt source has a higher priority than this mailbox, the interrupt register's interrupt identifier will point to this mailbox	
14	MLOST	R/W	Message Lost This field applies only to mailboxes configured for the receive direction. 0: No message loss occurred since the CPU last cleared this bit. 1: A message loss occurred, i.e., the message processor received a new message while CAN_NDATA[NDATA] was set and stored it in this mailbox, causing the previous message to be overwritten by the new one.	0h
15	NDATA	R/W	New Data 0: Since the CPU last cleared this bit, no new data has been written to the data section of this mailbox 1: New data has been written, i.e., the data section of this mailbox contains new data (written by the message processor or CPU)	0h
31:16	Reserved			0h

38.8.32 IF3 Data A Register (CAN_IF3DATAA)

Offset address: 0x110+(x-1)*0x40

Reset type: SYSRSn

This register is used to provide a window for accessing CAN message data bytes.

Field	Name	R/W	Description	Reset value
7:0	DATD0	R/W	Data Byte 0	0h
15:8	DATD1	R/W	Data Byte 1	0h
23:16	DATD2	R/W	Data Byte 2	0h
31:24	DATD3	R/W	Data Byte 3	0h

38.8.33 IF3 Data B Register (CAN_IF3DATAB)

Offset address: 0x114+(x-1)*0x40

Reset type: SYSRSn

This register is used to provide a window for accessing CAN message data bytes.

Field	Name	R/W	Description	Reset value
7:0	DATD4	R/W	Data Byte 4	0h
15:8	DATD5	R/W	Data Byte 5	0h
23:16	DATD6	R/W	Data Byte 6	0h

Field	Name	R/W	Description	Reset value
31:24	DATD7	R/W	Data Byte 7	0h

38.8.34 IF3 Update Enable Register (CAN_IF3UDEN)

Offset address: 0x160

Reset type: SYSRSn

This register is used to enable the automatic update function of the IF3 register bank for each mailbox individually. When UDEN=1 for a mailbox, that mailbox automatically updates the IF3 register bank. At this point, if the NDATA bit for a mailbox is activated (for example, due to a received CAN frame), it triggers the operation to automatically copy the entire mailbox into the IF3 register bank.

Note: Avoid enabling IF3 updates for transmit mailboxes.

Field	Name	R/W	Description	Reset value
31:0	UDEN	R/W	F3 Update Enable This field is used to enable the automatic update function of the IF3 register bank for each mailbox individually. 0: Disable automatic updates to the IF3 register bank for this mailbox 1: Enable automatic updates to the IF3 register bank for this mailbox	0h

39 Controller Area Network (CAN FD)

39.1 Full Name and Abbreviation Description of Terms

Table 215 Full Name and Abbreviation Description of Terms

Full Name	Abbreviation
Bit Rate Prescaler	BRP
Bit Stream Processor	BSP
Bit Timing Logic	BTL
Controller Area Network	CAN
Controller Area Network with Flexible Data-rate	CAN FD
Cyclic Redundancy Check	CRC
Data Length Code	DLC
Error Correction Code	ECC
Electronic Control Unit	ECU
Error Management Logic	EML
End of Frame	EOF
Finite State Machine	FSM
minimum time quantum = CAN clock period (canfdx_cclk)	mtq
Start of Frame	SOF
Secondary Sample Point	SSP
Transmitter Delay Compensation	TDC
Time quantum	Tq
Time Segment before Sample Point	TSEG1
Time Segment after Sample Point	TSEG2

39.2 Introduction

The CAN FD module supports the standard of ISO 11898-1:2015 and the protocol of BOSCH CAN FD (Version: V1.0). To be connected to the physical layer, an additional transceiver hardware is required.

The Message RAM is implemented as a single-port SRAM and can be accessed by either the CPU or the CAN FD module.

All message handling-related functions are implemented by the Receive Handler (Rx Handler) and the Transmit Handler (Tx Handler). The Receive Handler manages message reception filtering, transfers received messages

from the CAN FD core to the Message RAM, and provides the status information on received messages. The Transmit Handler is responsible for transferring messages to be sent from the Message RAM to the CAN FD core and provides transmission status information.

The reception filtering function is achieved through up to 128 filter combinations. Each filter can be configured as a range filter, a bit mask filter, or a specific ID filter.

The CAN FD module canbe connected to the CPU via a 32-bit generic slave interface.

39.3 Main Characteristics

(1) Conforms to ISO 11898-1:2015

(2) Supports the CAN FD protocol with data payloads of up to 64 bytes

(3) Features CAN error logging functionality

(4) Supports AUTOSAR (Automotive Open System Architecture)

(5) Supports the SAE J1939 protocol (a communication protocol for commercial vehicles)

(6) Improved acceptance filtering capability

(7) Two configurable receive First-In-First-Out (FIFO) buffers

(8) Provides a dedicated signal upon reception of a high-priority message

(9) Up to 64 dedicated receive buffers

(10) Up to 32 dedicated transmit buffers

(11) Configurable transmit FIFO buffers

(12) Configurable transmit queue

(13) Configurable transmit event FIFO buffer

(14) CPU has direct access to the Message RAM

(15) Programmable loopback test mode

(16) Maskable module interrupts

(17) Dual clock domains (CAN FD protocol clock and CAN FD peripheral clock)

(18) Supports power-down mode and wake-up functionality

(19) Supports SRAM ECC check; ECC errors CAN FD trigger an interrupt and the error address can be read

39.4 Structure Block Diagram

Figure 243 CAN FD Structure Block Diagram

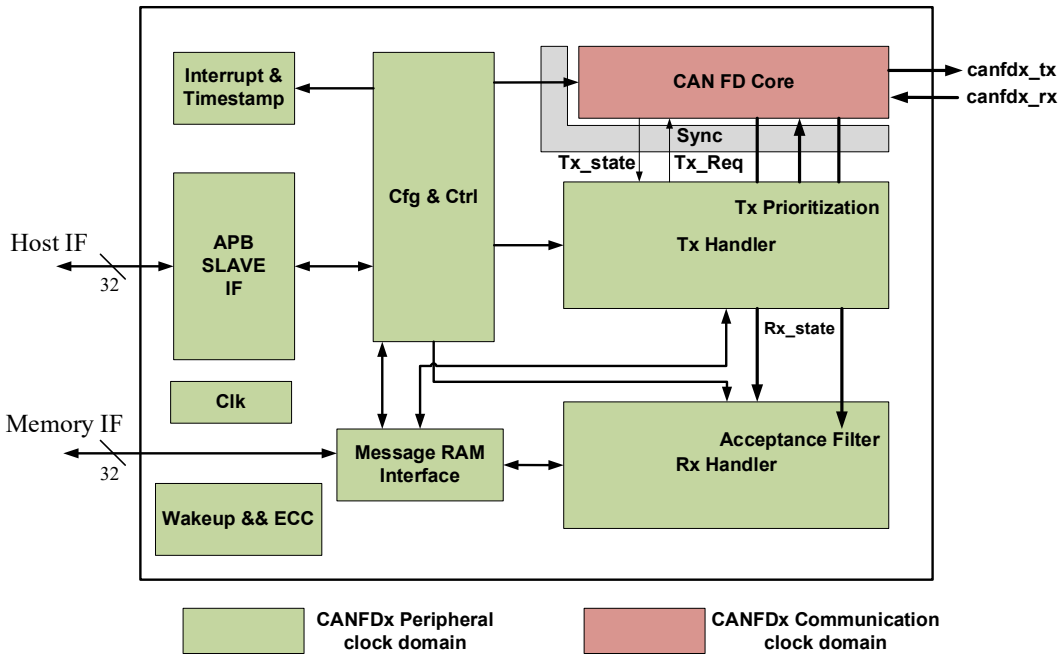


Table 216 Description of CAN FD Controller Structure Block Diagram

Name	Meaning	Description
CAN FD Core	CAN FD core	The CAN FD protocol controller and the Receive/Transmit Shift Registers are responsible for handling all protocols compliant with the ISO 11898-1:2015, supporting both 11-bit and 29-bit identifiers.
Sync	Synchronization	It synchronizes signals from the CAN FD peripheral clock domain to the CAN FD protocol clock domain, and vice versa.
CLK	Clock	It synchronizes the CAN FD reset signal to both the CAN FD peripheral clock domain and the CAN FD protocol clock domain.
CFG & CTRL	Configuration and control	This includes configuration and control bits related to the CAN FD core.
Interrupt & Timestamp	Interrupt and timestamp	It features interrupt control and a 16-bit CAN FD bit time counter for generating receive and transmit timestamps. Alternatively, an externally generated 16-bit vector can be used instead of the internal 16-bit CAN FD bit time counter to generate these timestamps.
Tx Handler	Transmit Handler	It controls the transfer of messages from the external message RAM to the CAN FD core. Up to 32 transmit buffers can be configured for message transmission. These buffers can be used as dedicated transmit buffers, as part of a transmit FIFO buffer, as part of a transmit queue, or in a combination of these modes. The Transmit Event FIFO stores the transmit timestamp along with the corresponding message ID. Transmit abort functionality is also supported.

Name	Meaning	Description
Rx Handler	Receive Handler	It controls the transfer of received messages from the CAN FD core to the external message RAM. The Receive Handler supports two configurable Receive FIFO buffers, as well as up to 64 dedicated receive buffers for storing all messages that pass the acceptance filtering. Unlike the Receive FIFO buffers, dedicated receive buffers are used to store only messages with specific identifiers. Each message is stored along with a receive timestamp. A maximum of 128 filters can be defined for 11-bit ID, and up to 64 filters for 29-bit ID.
APB Slave Interface	APB Slave Interface	The CAN FD module is connected to the APB bus. Its APB Slave Interface supports connection only to a 32-bit bus.
Message RAM Interface	Message storage interface	The external 32-bit message RAM is accessible by both the CPU and the CAN FD controller.
Wakeup & ECC	Wake-up and ECC check	The CAN FD module supports waking the system from a low-power mode (STOP mode) and also features ECC check.
canfdx_pclk & canfdx_cclk	Double clock source	The CAN FD module operates with two clock domains: the CAN FD peripheral clock (canfdx_pclk) and the CAN FD protocol clock (canfdx_cclk), which correspond to the APB CLK and CAN FD clock described in the section of “Clock Tree”, respectively.
Interrupt	Interrupt	This module provides five interrupt lines, and the interrupt signal can be routed to CANFDA_INT0, CANFDA_INT1, CANFDA_BEC_INT, CANFDA_BEU_INT and CANFDA_WAKE_IT. For the CANFDA controller, all interrupts can be routed to the interrupt lines CANFDA_INT0 and CANFDA_INT1. By programming ILE.EINT0 and ILE.EINT1, these two interrupt lines can be enabled or disabled respectively. For ECC check, interrupts can be routed to the interrupt line CANFDA_BEC_INT/CANFDA_BEU_INT; For wake-up, interrupts can be routed to the interrupt line CANFDA_WAKE_IT.

39.5 Functional Description

39.5.1 Operation mode

39.5.1.1 Software initialization

Software initialization is triggered by setting the CCCR [INIT] bit. It CAN FD also be initiated by a software setting, a hardware reset, or when the CAN FD controller enters the Bus-Off state. When CCCR [INIT] bit is set, message transmission on the CAN bus is halted, and the CAN bus output canfdx_tx becomes recessive (high level). The counter of the Error Management Logic (EML) remains unchanged. Setting CCCR [INIT] does not alter any configuration registers. Resetting CCCR [INIT] completes the software initialization.

Following this, the Bit Stream Processor (BSP) must synchronize with the data transmission on the CAN bus by waiting for the occurrence of 11 consecutive

recessive bits (indicating bus idle, Bus_Idle). Only after this synchronization CAN FD it participate in bus activities and begin message transmission.

Access to the CAN FD configuration registers is permitted only when both the CCCR [INIT] and CCCR [CCE] bits are set (via a protected write).

The CCCR [CCE] bit CAN FD only be set or reset when CCCR [INIT] = 1. When CCCR [INIT] is reset, CCCR [CCE] is automatically reset.

When the CCCR [CCE] bit is set, the following registers are reset:

- HPMS (High Priority Message Status)
- RXF0S (Rx FIFO 0 Status)
- RXF1S (Rx FIFO 1 Status)
- TXFQS (Tx FIFO/Queue Status)
- TXBRP (Tx Buffer Request Pending)
- TXBTO (Tx Buffer Transmission Occurred)
- TXBCF (Tx Buffer cancellation Finished)
- TXEFS (Tx Event FIFO Status)

When CCCR [CCE] is set, the timeout counter value TOCV [TOC] is preloaded with the value configured in TOCC [TOP].

Furthermore, when CCCR [CCE] = 1, the state machines of the Tx Handler and the Rx Handler remain in an idle state.

The following registers are writable only when CCCR [CCE] = 0:

- TXBAR (Tx Buffer Add Request)
- TXBCR (Tx Buffer cancellation Request)

The CCCR [TEST] and CCCR [MON] bits CAN FD only be set by the host when CCCR [INIT] = 1 and CCCR [CCE] = 1. Both bits can be reset at any time. The CCCR [DAR] bit CAN FD only be set/reset when CCCR [INIT] = 1 and CCCR [CCE] = 1.

Note: The message RAM is equipped with ECC functionality. It is recommended to initialize the message RAM after a hardware reset by writing, for example, 0x0000 0000 to each message RAM word to generate a valid ECC checksum. This prevents the activation of ECC interrupts when reading from uninitialized sections of the message RAM.

39.5.1.2 **Normal operation**

Once the CAN FD controller completes initialization and the CCCR [INIT] bit is reset to 0, it synchronizes with the CAN bus and is ready for communication.

- The received messages (including the DLC and message ID) are stored into either a dedicated Rx Buffer, Rx FIFO 0, or Rx FIFO 1 after passing the acceptance filtering.
- The messages to be transmitted can be used for the initialization or update of the dedicated Tx Buffers, Tx FIFO, or Tx Queue. Automatic transmission of remote frames is not supported.

39.5.1.3 CAN FD operation

There are two variants of CAN FD frame transmission. The first is a CAN FD frame with bit rate switching disabled. The second is a CAN frame where the Control Field, Data Field, and CRC Field are transmitted at a faster bit rate compared to the beginning and end parts of the frame.

In a CAN frame with an 11-bit identifier, the previously reserved bits, and the first reserved bit in a CAN frame with a 29-bit identifier, will now be decoded as the FDF (FD Frame) bits. FDF recessive indicates a CAN FD frame, while FDF dominant indicates a classic CAN frame.

In a CAN FD frame, the two bits following FDF, res (reserved) and BRS (Bit Rate Switch), determine whether the bit rate is switched within this CAN FD frame. CAN FD bit rate switching is indicated by res = dominant and BRS = recessive. The recessive (res) code is reserved for future protocol extensions. If the CAN controller receives a frame with FDF = recessive and res = recessive, it signals a Protocol Exception Event by setting the PSR[PXE] (Protocol Exception Event) bit. When protocol exception handling is enabled (CCCR [PXHD] = 0), this causes the module's active state to change from Receiver (PSR[ACT] = 10) to Synchronizing (PSR[ACT] = 00) at the next sampling point. When protocol exception handling is disabled (CCCR [PXHD] = 1), the CAN controller treats the recessive res bit as a format error and responds with an error frame.

CAN FD operation is enabled by programming the CCCR [FDOE] bit.If CCCR [FDOE] = 1, the transmission and reception of CAN FD frames are enabled. The controller CAN FD always transmit and receive classic CAN frames. It determines whether a CAN FD frame or a classic CAN frame is transmitted according to the configuration of the FDF bit in the corresponding Tx Buffer Element.If CCCR [FDOE] = 0, received frames are interpreted as classic CAN frames. This will cause an error frame to be transmitted if a CAN FD frame is received. When CAN FD operation is disabled, CAN FD frames are not transmitted, even if the FDF bit in the Tx Buffer Element is set. The CCCR [FDOE] and CCCR [BRSE] bits CAN FD only be changed when both CCCR [INIT] and CCCR [CCE] are set.

If CCCR [FDOE] = 0, the settings of the FDF and BRS bits are ignored, and the frames are transmitted in the format of classic CAN. If CCCR [FDOE] = 1 and CCCR [BRSE] = 0, only the FDF bit in the Tx Buffer Element is evaluated. If CCCR [FDOE] = 1 and CCCR [BRSE] = 1, the transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer Elements that have both the FDF and BRS bits set are transmitted in the CAN FD format with bit rate switching.

During CAN operation, a mode change is recommended only under the following circumstances:

(1) During system startup, all nodes transmit classic CAN frames until it is verified that they are capable of communicating in the CAN FD format. If so, all nodes switch to CAN FD operation.

(2) Wake-up messages in a partial CAN network must be transmitted in the classic CAN format.

(3) The fault rate during the CAN FD data phase is significantly higher than that during the CAN FD arbitration phase. In this case, the CAN FD bit rate switching option for transmission is disabled.

(4) End-of-line programming is implemented if not all nodes support CAN FD operation. Non-CAN FD nodes remain in mute mode until programming is completed. Afterwards, all nodes switch back to classic CAN communication mode.

In the CAN FD format, the DLC code differs from the standard CAN FD format. DLC codes 0 to 8 are encoded the same as in standard CAN FD, while codes 9 to 15, which are all encoded as an 8-byte data field in standard CAN FD, are encoded according to the following table in CAN FD.

Table 217 DLC Codes

DLC	9	10	11	12	13	14	15
Number of data bytes	12	16	20	24	32	48	64

In a CAN FD frame, if the BRS (Bit Rate Switch) bit is recessive, the bit timing switches within the frame after this bit. Before the BRS bit, the nominal CAN FD bit timing defined by the nominal bit timing and Prescaler Register (NBTP) is used during the CAN FD arbitration phase. In the subsequent CAN FD data phase, the data phase bit timing defined by the Data Bit Timing and Prescaler Register (DBTP) is used. The bit timing switches back from the data phase timing at the CRC delimiter or upon error detection, whichever occurs first.

The maximum configurable bit rate for the CAN FD data phase depends on the CAN FD protocol clock frequency (canfdx bit clk). For example: If the CAN FD protocol clock frequency is 20 MHz and the minimum configurable bit time is 4 tq, the resulting bit rate for the data phase is 5 Mbit/s.

In both data frame formats (CAN FD and CAN FD with bit rate switching), the value of the ESI (Error State Indicator) bit is determined by the error state of the transmitter at the start of the transmission. If the transmitter is in the error-passive state, the ESI bit is transmitted in recessive format; otherwise, it is transmitted in dominant format.

39.5.1.4 Transmitter delay compensation

During the data phase of a CAN FD transmission, only one node acts as the transmitter, while all other nodes are receivers. The bus length has no impact in

this phase. When transmitting via the canfdx_tx pin, the CAN FD controller simultaneously receives the transmitted data from the local CAN FD transceiver via the canfdx_rx pin. The received data is delayed due to transmitter loop delay. If this delay exceeds TSEG1 (the time segment before the sampling point), a bit error will be detected. To achieve data phase timings shorter than this transmitter loop delay, a transmitter delay compensation (TDC) mechanism is introduced. Without transmitter delay compensation, the achievable bit rate in the data phase of a CAN FD frame would be limited by the transmitter loop delay.

Description

The protocol unit of the CAN FD controller implements a Transmitter Delay Compensation (TDC) mechanism to compensate for the transmitter loop delay, thereby enabling higher bit rates in the CAN FD data phase without being limited by the specific delay of a CAN transceiver.

To check for bit errors in the data phase at the transmitting node, the delayed transmitted data is compared with the received data at a Secondary Sampling Point (SSP). If a bit error is detected, the transmitter will react to this bit error at the next regular sampling point. The delay compensation is always disabled during the arbitration phase.

The transmitter delay compensation allows for the configuration of data bit times that are shorter than the transmitter loop delay, as detailed in ISO 11898-1:2015. This feature is enabled by setting the DBTP [TDC] bit.

At the SSP, the receiving bit is compared with the transmitting bit. The SSP position is defined as the sum of the measured loop delay from the controller's transmit output canfdx_tx through the transceiver to its receive input canfdx_rx, and the configured Transmitter Delay Compensation Offset TDCR [TDCO]. This offset is used to adjust the position of the SSP within the receiving bit (for example, to half of the data phase bit time). The calculated position of the secondary sampling point is rounded down to the next integer multiple of the minimum time quantum (mtq).

The PSR [TDCV] register displays the actual measured transmitter delay compensation value. PSR [TDCV] is cleared when CCCR [INIT] is set and is updated each time an FD frame is transmitted while DBTP [TDC] is set.

The transmitter delay compensation mechanism in the CAN FD controller must adhere to the following boundary conditions:

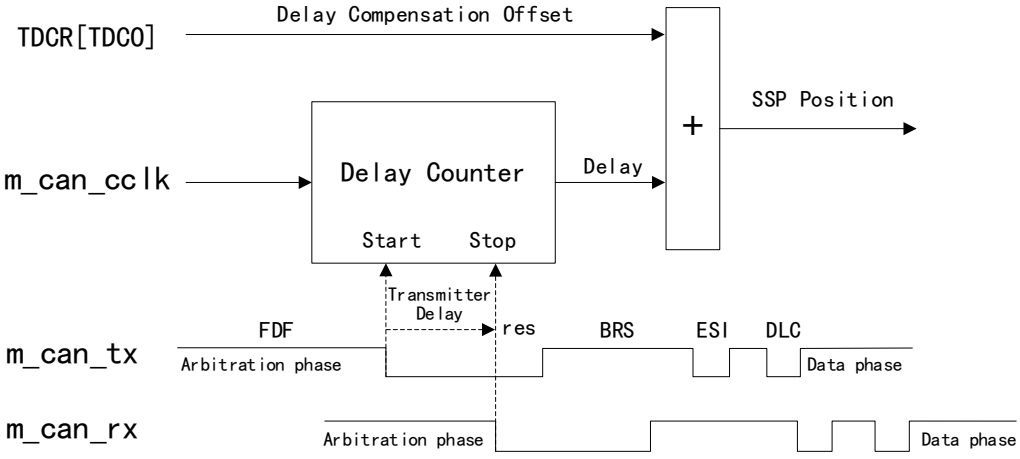
- The sum of the measured loop delay from canfdx_tx to canfdx_rx and the configured transmitter delay compensation offset TDCR[TDCO] must be less than or equal to 127 mtq. If this sum exceeds 127 mtq, the maximum value of 127 mtq is used for transmitter delay compensation.

- During the data phase, the sum of the measured loop delay from canfdx_tx to canfdx_rx and the configured transmitter delay compensation offset TDCR[TDCO] must be less than 6-bit time.
- The data phase ends at the sampling point of the CRC delimiter, at which point the checking of receiving bits at the SSP ceases.

Transmitter delay compensation measurement

If transmitter delay compensation is enabled by programming the DBTP[TDC] bit to 1, the measurement starts at the falling edge from the FDF bit to the res bit in each transmitted CAN FD frame. The measurement stops when this edge is detected at the transmitter's receive input, canfdx_rx. The resolution of this measurement is one minimum time quantum (mtq).

Figure 244 Transmitter Delay Measurement



To prevent a dominant glitch within the received FDF bit from prematurely ending the delay compensation measurement before the falling edge of the received res bit – which would cause the SSP to be positioned too early – a transmitter delay compensation filter window can be enabled by programming TDCR [TDCF]. This defines a minimum value for the SSP position. For the transmitter delay measurement, dominant edges on canfdx_rx that would result in an earlier SSP position are ignored. The measurement stops when the calculated SSP position is at least TDCR [TDCF] and canfdx_rx is at a low level.

Note: The CAN FD controller always performs a dynamic transmitter delay measurement; the configuration of static (fixed) transmitter delay is not supported.

39.5.1.5 Restricted operation mode

In Restricted Operation Mode, a node is capable of receiving data and remote frames and acknowledging valid frames, but it will not transmit data frames, remote frames, active error frames, or overload frames. In the event of an error condition or overload condition, instead of transmitting dominant bits, it waits for the occurrence of a bus idle state to resynchronize with the CAN

communication. While error logging (ECR[CEL]) is active, the error counters (ECR[REC], ECR[TEC]) are frozen. The host CAN FD set the CAN FD controller into Restricted Operation Mode by setting the CCCR[ASM] bit. This bit CAN FD only be set by the host when both CCCR [CCE] and CCCR [INIT] bits are set to 1. The host CAN FD reset this bit at any time.

The controller automatically enters Restricted Operation Mode when the Tx Handler fails to read data from the message RAM in time. To exit Restricted Operation Mode, the host CPU must reset CCCR [ASM].

Restricted Operation Mode can be used in applications that need to adapt to different CAN bit rates. In such cases, the APP tests different bit rates and exits Restricted Operation Mode upon receiving a valid frame.

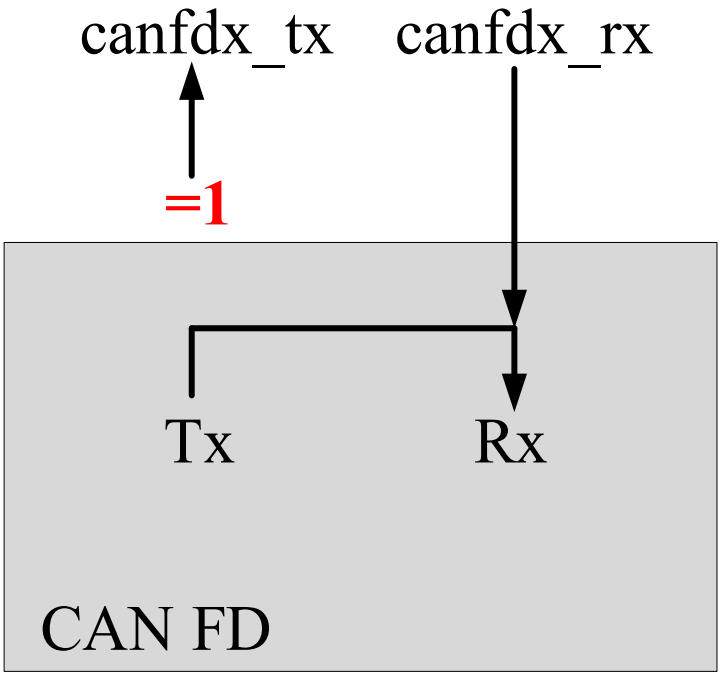
Note: Restricted Operation Mode must not be used in combination with Loopback Mode (internal or external).

39.5.1.6 Bus monitoring mode

The CAN FD controller can be set to Bus Monitoring Mode by programming the CCCR [MON] bit to 1. In Bus Monitoring Mode (see “10.14 Bus Monitoring” in the ISO 11898-1:2015), the CAN FD controller is able to receive valid data frames and valid remote frames but can not initiate a transmission. In this mode, it transmits only recessive bits on the CAN bus. If the controller needs to send a dominant bit (such as an ACK bit, overload flag, or active error flag), this bit is internally rerouted so that the controller CAN FD monitor it, even though the actual CAN bus may remain in the recessive state. In Bus Monitoring Mode, the TXBRP register remains in its reset state.

Bus Monitoring Mode can be used to analyze the traffic on the CAN bus without affecting it by transmitting dominant bits. The figure below shows the connection of the canfdx_tx and canfdx_rx signals when the CAN FD controller is in Bus Monitoring Mode.

Figure 245 Pin control in bus monitoring mode



39.5.1.7 Disable automatic retransmission

According to the CAN FD specification (see “8.3.4 Recovery Management” in the ISO 11898-1:2015), the CAN FD controller provides a method for the automatic retransmission of frames that have lost arbitration or have been disturbed by errors during their transmission. By default, automatic retransmission is enabled. To support the time-triggered communication as described in Section 9.2 of ISO 11898-1:2015, automatic retransmission can be disabled via the CCCR [DAR] bit.

Frame transmission in DAR mode

In DAR (Disable Automatic Retransmission) mode, all transmissions initiated on the CAN bus are automatically cancelled. Upon a successful transmission, the Tx request pending bit TXBRP [TRPx] in the Tx Buffer will be cleared; this bit is also cleared if the transmission has not started by the cancellation point, is aborted due to arbitration loss, or if an error occurs during frame transmission.

(1) When transmission is successful

- The corresponding Tx Buffer Transmission Occurred bit TXBTO[TOx] is set,
- The corresponding Tx Buffer cancellation Finished bit TXBCF[CFx] is not set.

(2) When a frame is successfully transmitted despite cancellation

- The corresponding Tx Buffer Transmission Occurred bit TXBTO [TOx] is set

- The corresponding Tx Buffer cancellation Finished bit TXBCF[CFx] is set.

(3) When arbitration is lost or the frame transmission is disturbed

- The corresponding Tx Buffer Transmission Occurred bit TXBTO [TOx] is not set
- The corresponding Tx Buffer cancellation Finished bit TXBCF[CFx] is set.

If a frame is successfully transmitted and Tx Event storage is enabled, an event with Event Type ET = 10 (transmission occurred despite cancellation) is written into the Tx Event FIFO element.

39.5.1.8 Power-down (sleep mode)

The controller enters power-down mode via an externally input clock stop request signal or by setting the CCCR [CSR] bit. The CCCR [CSR] bit reads as 1 as long as the external clock stop request signal is active.

After all pending transmission requests are completed, the CAN FD controller waits until a bus idle state is detected. It then sets CCCR [INIT] bit to 1 to prevent any further CAN FD transmissions. Next, the CAN FD controller acknowledges it is ready for power-down by setting the output clock stop acknowledge signal to 1 and setting the CCCR[CSA] bit to 1. Further register accesses are possible before deactivating the module clocks canfdx_hclk and canfdx bit clk, with the exception of the CCCR[INIT] bit, which remains set to 1.

Note: In cases of severe disturbance on the CAN bus, a bus idle state might not be reached, and consequently, the CAN FD controller will not set CCCR [INIT] bit. This condition can be detected by polling the PSR[ACT] register. If the CAN FD controller does not enter the idle state, software CAN FD write CCCR [INIT] = 1, which will immediately halt the controller's CAN communication, regardless of any ongoing transmission or reception.

To exit power-down mode, the APP must open the module clock before resetting the external clock stop request signal or the CCCR [CSR] bit. The CAN FD controller will acknowledge this action by resetting the output clock stop acknowledge signal and clearing the CCCR[CSA] bit. Then the APP restarts CAN communication by resetting CCCR [INIT] bit.

39.5.1.9 Wake-up Request

When placing the CAN FD module into power-down mode, ensure that the SYSC.CANFDACLKSTOPREQ bit is set.

Whilst the CAN FD module remains in power-down mode, it will issue a wake-up request upon the occurrence of the following wake-up events:

(1) The RX pin is in the dominant state (logic 0)

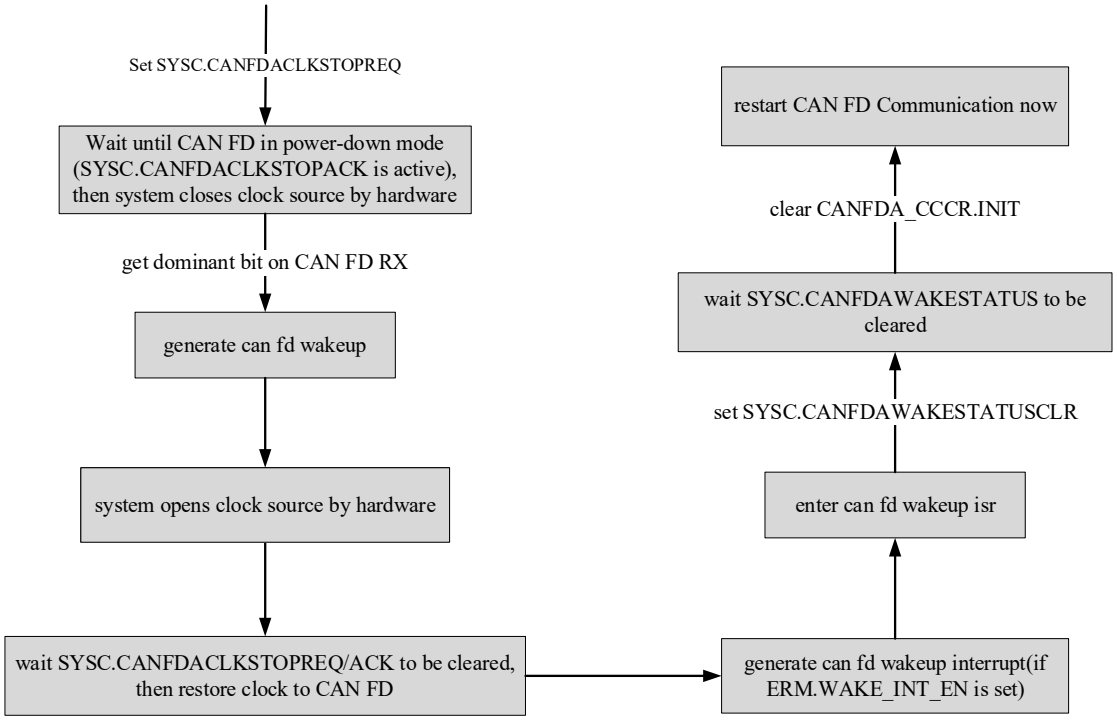
The wake-up request will be cancelled when any of the following conditions are

met:

(1) The clock stop request is removed and the clock stop acknowledgement is cancelled

(2) The CAN FD module is reset

Figure:246 CAN FD Wake-up Flowchart



39.5.1.10 Test Mode

Write access to the TEST register is only permitted when the CCCR[TEST] bit is set to 1. This enables the configuration of test mode and test functions.

By programming the TEST[TX] bits, the CAN transmit pin canfdx_tx can be configured for one of four output functions. In addition to its default function as a serial data output, it can be driven to output the CAN sampling point signal for monitoring the CAN FD bit timing, or to output a constant dominant or recessive value. The current logic level of the receive pin canfdx_rx can be read from the TEST[RX] bit. These two functions can be used to check the physical layer of the CAN bus.

Due to the synchronization mechanism between the CAN FD communication clock and the CAN FD peripheral clock domains, a latency of a few host clock cycles may occur from writing to TEST[TX] bit until the newly configured value appears on the output pin canfdx_tx. The same applies when reading the input pin canfdx_rx via the TEST[RX] bit.

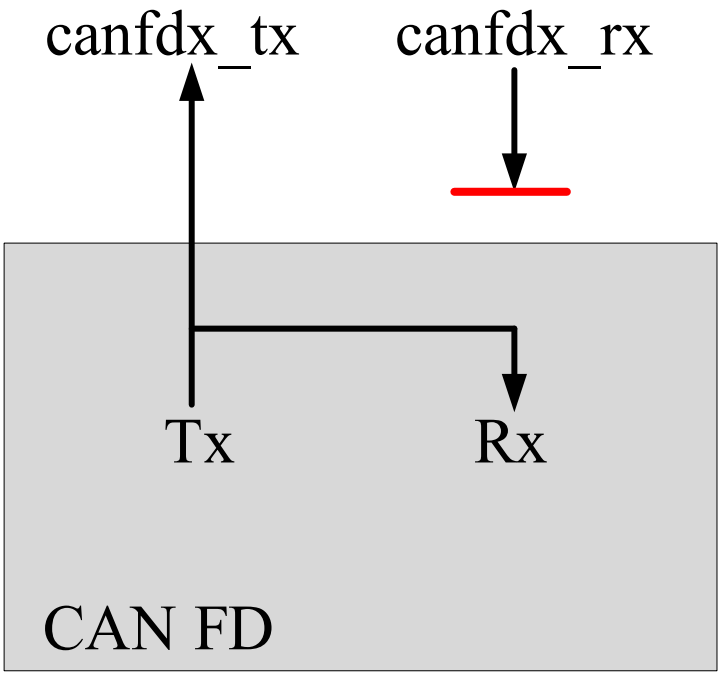
Note: The test mode should only be applicable for production testing or self-test. Software control of the canfdx_tx pin interferes with all CAN FD protocol functions. The test mode is not recommended to be used in applications.

External Loopback Mode

The CAN FD controller can be set to External Loopback Mode by programming the TEST[LBCK] bit to 1. In Loopback Mode, the CAN FD controller treats the messages it transmits as received messages and stores them (if they pass acceptance filtering) into the Rx Buffer or Rx FIFO. The diagram below illustrates the connection of the signal input pins canfdx_tx and canfdx_rx to the CAN FD controller in External Loopback Mode.

This mode is used for hardware self-test. To remain independent of external influences, the CAN FD controller ignores acknowledge errors (sampling a recessive bit in the acknowledge slot of a data/remote frame) in Loopback Mode. In this mode, the CAN FD controller internally feeds back its Tx output to its Rx input. The actual value of the canfdx_rx input pin is ignored by the CAN FD controller. The transmitted messages can be monitored on the canfdx_tx pin.

Figure 247 Pin control for external loopback mode

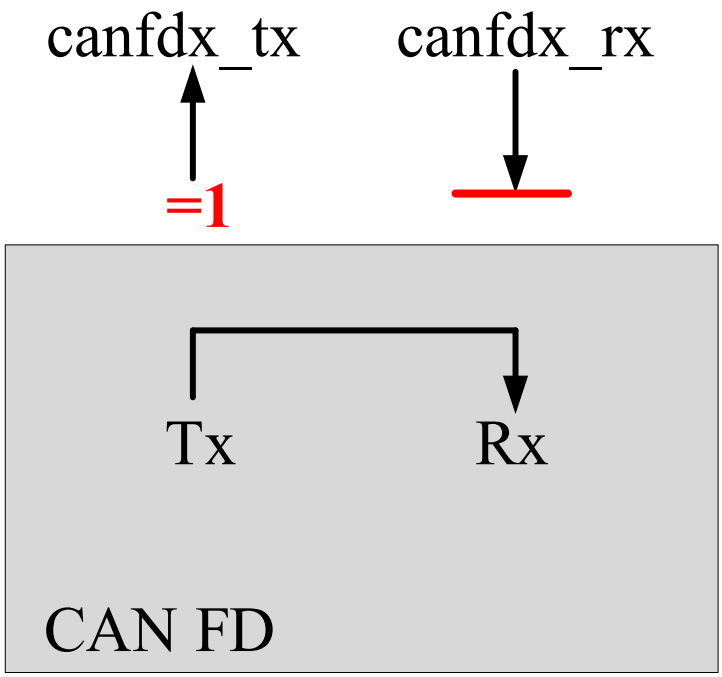


Internal Loopback Mode

The CAN FD controller CAN FD enter Internal Loopback Mode by setting the TEST[LBCK] and CCCR[MON] bits to 1. This mode can be used for a “hot self-test,” allowing the CAN FD controller to be tested without affecting the operational CAN system connected to the canfdx_tx and canfdx_rx pins. In this mode, the canfdx_rx pin is disconnected from the CAN FD controller, while the

canfdx_tx pin is held at the recessive state. The diagram below illustrates the connection of the canfdx_tx and canfdx_rx pins to the CAN FD controller in Internal Loopback Mode.

Figure 248 Pin configuration for internal loopback mode



39.5.2 Timestamp generation

39.5.2.1 Internal timestamp generation

The CAN FD controller implements an internal timestamp generation mechanism using a 16-bit wrapping counter. This counter can be configured via the prescaler TSCC[TCP] to increment at multiples (1…16) of the CAN bit time. The current value of the counter can be read from the TSCV[TSC] register. Writing to the TSCV register resets the counter to zero. The interrupt flag IR[TSW] is set whenever the timestamp counter wraps around.

At the start of a frame's reception or transmission (SOF), the value of the counter is captured. This captured value is stored in the timestamp section (for example, RXTS[15:0]) of the respective Rx buffer, Rx FIFO element, or Tx Event FIFO element (for example, TXTS[15:0]).

39.5.2.2 External timestamp generation

As an alternative to using the internal 16-bit counter, an external 16-bit time base vector input of CAN FD can be captured as the timestamp by programming the TSCC[TSS] bits.

39.5.3 Timeout counter

The CAN FD controller is provided with a 16-bit timeout counter to signal

timeout for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO. The timeout counter is configured via the TOCC register, and its current value can be read from TOCV[TOC].

This timeout counter operates as a decrementing counter and uses the same prescaler controlled by TSCC[TCP] as the timestamp counter. The timeout counter CAN FD only start when CCCR[INIT] = 0. It stops when CCCR[INIT] = 1, for example, when the CAN FD controller enters the Bus-Off state.

The interrupt flag IR[TOO] is set when the value of the counter reaches zero. In continuous mode, the counter immediately restarts from the value configured in TOCC[TOP].

The operating mode is selected via TOCC[TOS]. When operating in continuous mode, the counter begins running when CCCR[INIT] is reset. Writing to the TOCV register presets the counter to the value configured in TOCC[TOP], and it continues decrementing.

When the timeout counter is controlled by a specific FIFO, an empty FIFO causes the counter to be preset to the value configured in TOCC[TOP]. Decrementing begins when the first FIFO element is stored. In this mode, writing to the TOCV register has no effect.

Note: The clock signal for the timeout counter is derived from the sampling point signal of the CAN FD core. Consequently, the exact timing of the counter's decrements may vary due to the synchronization/resynchronization mechanisms of the CAN FD core. If the bit rate switching of CAN FD is enabled, the timeout counter will tick at different rates in the arbitration and data fields.

39.5.4 Rx handler

The Rx Handler controls the acceptance filtering, the transfer of received messages to either the Rx buffer or one of the two Rx FIFOs, and manages the put and get indices for the Rx FIFOs.

Note: In the current design, it is not possible to block the reception of debug messages or prevent the toggling of their associated status bits. However, this does not affect the normal usage of the CAN FD controller.

39.5.4.1 Acceptance filtering

The CAN FD controller can be configured with two sets of acceptance filters: one for standard identifiers and another for extended identifiers. These filters can be assigned to either the Rx buffer or to Rx FIFO 0/1. During acceptance filtering, each filter list is executed starting from Element 0 until the first matching element is found. The acceptance filtering process stops upon finding the first match, and subsequent filter elements in the list are not evaluated.

The key features are as follows:

- Each filter element can be configured for either acceptance or rejection filtering.

- Each filter element can be individually enabled or disabled.
- Each filter element can be configured as:
  - A filter for one or two dedicated IDs
  - A classic bit mask filter
  - A range filter
- Filters are checked sequentially. The execution stops as soon as the first matching filter element is found.

Related configuration registers:

- Standard ID Filter Configuration (SIDFC)
- Extended ID and Mask (XIDAM)
- Extended ID Filter Configuration (XIDFC)
- Global Filter Configuration (GFC)

Depending on the configuration of the filter element (SFEC/EFEC), a match triggers one of the following actions:

- Set the high-priority message interrupt flag IR[HPM]
- Set the high-priority message interrupt flag IR[HPM] and store the received frame in FIFO 0 or FIFO 1
- Store the received frame in FIFO 0 or FIFO 1
- Store the received frame in the Rx Buffer
- Reject the received frame

Acceptance filtering begins after the complete identifier has been received. Once acceptance filtering is complete and a matching Rx Buffer or Rx FIFO is found, the message handler writes the received message data in 32-bit units to the matched Rx Buffer or Rx FIFO. If the CAN FD protocol controller detects an error (for example, a CRC error), the message is discarded. This has the following consequences for the affected Rx buffer or Rx FIFO:

- Rx Buffer: The new data flag for the matched Rx buffer is not set, but the Rx buffer is (partially) overwritten by the received data. The error type can be checked in PSR[LEC] or PSR[DLEC].
- Rx FIFO: The put index for the matched Rx FIFO is not updated, but the associated Rx FIFO element is (partially) overwritten by the received data. The error type can be checked in PSR[LEC] or PSR[DLEC]. If the matched Rx FIFO operates in overwrite mode, the boundary conditions of the Rx FIFO in this mode must be considered.

Note: When a received message that has passed acceptance filtering is written to one of the two Rx FIFOs or to a Rx Buffer, the unmodified received identifier is stored, regardless of which filters were used. The outcome of the acceptance filtering process depends heavily on the order of the configured filter elements.

Range filter

This filter matches all received frames whose message IDs fall within the range defined by SF1ID/SF2ID or EF1ID/EF2ID.

When range filter works with extended frames, there are two situations:

EFT = 00: The AND operation of received frame's message ID and the Extended ID and Mask (XIDAM) will be implemented before the range filter is applied.

EFT = 11: The range filter is applied without using the Extended ID and Mask (XIDAM).

Specific message ID filter

A single filter element can be configured to filter one or two specific message IDs. To filter a single specific message ID, the filter element must be configured such that SF1ID = SF2ID or EF1ID = EF2ID.

Classic bit mask filter

Classic bit mask filter is designed to filter groups of message IDs by masking individual bits of the received message IDs. In classic bit mask filter, SF1ID/EF1ID serves as the message ID filter, while SF2ID/EF2ID serves as the filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter. For example, the value of the received message ID in that bit position is irrelevant for acceptance filtering. Only those bit positions where the mask bit is "1" in the received message IDs are significant for acceptance filtering.

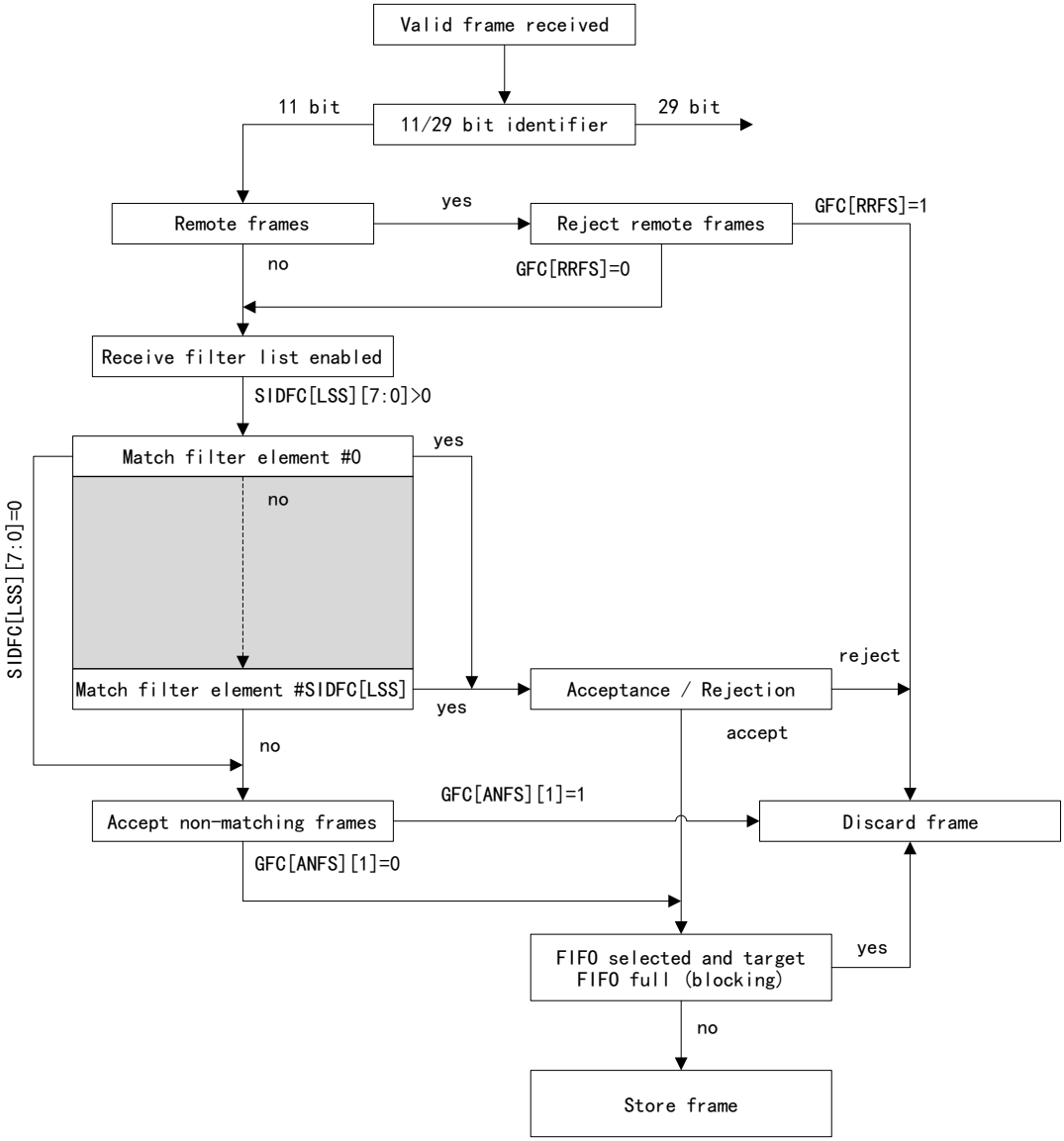
If all mask bits are "1", a match occurs only when the received message ID is identical to the message ID filter. If all mask bits are "0", any message ID will match.

Standard message ID filter

The following figure illustrates the filtering flow for a standard message ID (11-bit identifier). The standard message ID filter elements are described in other sections.

Under the control of the Global Filter Configuration (GFC) and Standard ID Filter Configuration (SIDFC) registers, the received frame's message ID, Remote Transmission Request (RTR) bit, and Identifier Extension (IDE) bit are compared against the configured list of filter elements.

Figure 249 Standard Message ID (11-bit identifier) Filtering Flow Chart



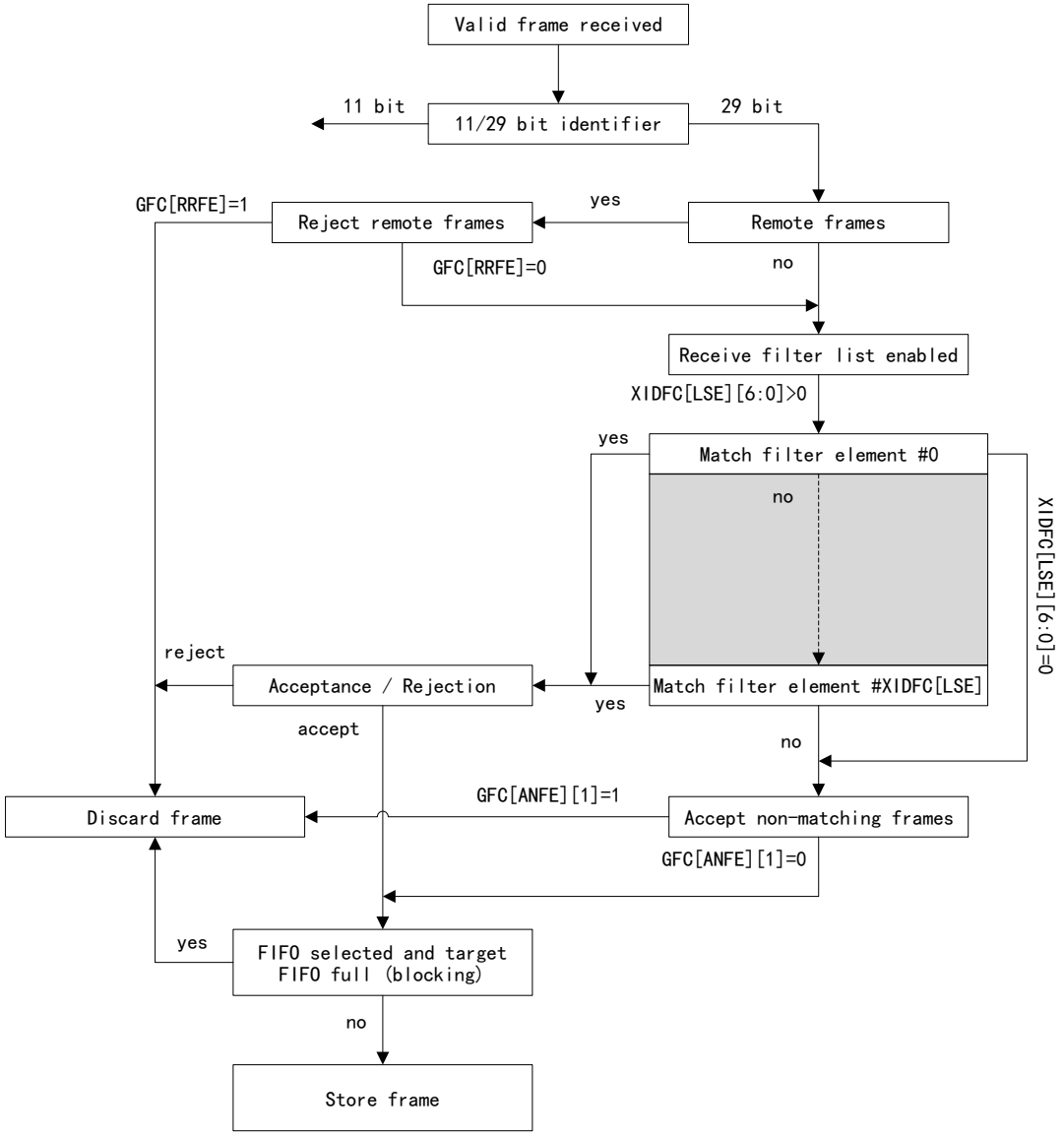
Extension message ID filter

The following figure illustrates the filtering flow for an extension message ID (29-bit identifier). The extension message ID filter elements are described in other sections.

Under the control of the Global Filter Configuration (GFC) and Extension ID Filter Configuration (XIDFC) registers, the received frame's message ID, Remote Transmission Request (RTR) bit, and Identifier Extension (IDE) bit are compared against the configured list of filter elements.

The AND operation of the Extended ID and Mask (XIDAM) and the received identifier will be implemented before the filter list is applied.

Figure 250 Extension Message ID (29-bit identifier) Filtering Flow Chart

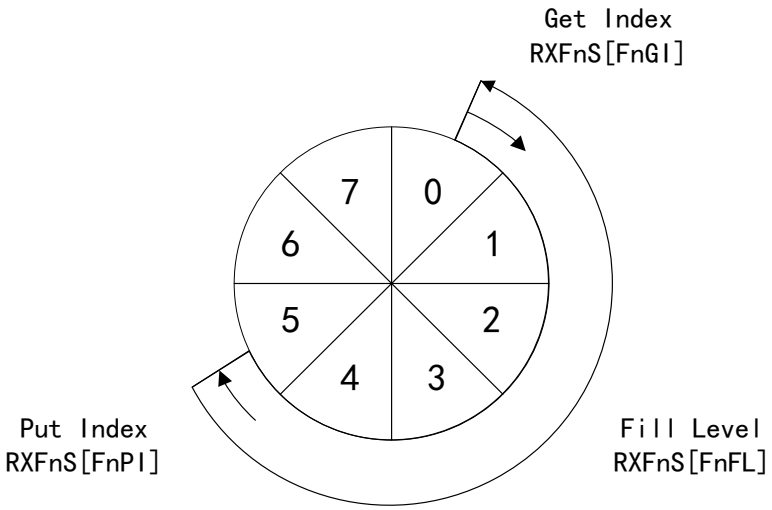


39.5.4.2 **Rx FIFO**

Both Rx FIFO 0 and Rx FIFO 1 can be individually configured to hold up to 64 elements. The two Rx FIFOs are configured via the RXF0C and RXF1C registers. Received messages that pass the acceptance filtering are transferred to the Rx FIFO based on the configuration of the matching filter element. For a description of the Rx FIFO element structure and the filtering mechanisms available for Rx FIFO 0 and Rx FIFO 1, refer to the relevant sections.

To prevent Rx FIFO overflow, an Rx FIFO watermark can be used. When the fill level of the Rx FIFO reaches the watermark value configured by RXFnC[FnWM], the interrupt flag IR[RFnW] is set. When the number of Rx FIFO's put indexes reaches the get level, RXFnS[FnF] signals that the Rx FIFO is full. Additionally, the interrupt flag IR[RFnF] is set.

Figure 251 Rx FIFO Status



When reading data from Rx FIFO, the Rx FIFO get index RXFnS[FnGI] must be multiplied by the FIFO element size and then added to the corresponding Rx FIFO start address RXFnC[FnSA].

Table 218 Rx FIFO and FIFO Element Size

RXESC[RBDS][2:0] RXESC[FnDS][2:0]	Data Field (bytes)	FIFO Element Size (RAM words)
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

Rx FIFO Blocking Mode

Rx FIFO Blocking Mode is configured by setting RXFnC[FnOM] to 0. This is the default operating mode of the Rx FIFO.

When the condition for a full Rx FIFO is met (RXFnS[FnPI] = RXFnS[FnGI]), no further messages will be written to the corresponding Rx FIFO until at least one message is read, causing the Rx FIFO get index to increment. RXFnS[FnF] = 1 signals that the Rx FIFO is full. Additionally, the interrupt flag IR[RFnF] is set.

If a message is received while the corresponding Rx FIFO is full, the message is discarded, and RXFnS[RFnL] = 1 signals a message loss. Additionally, the interrupt flag IR[RFnL] is set.

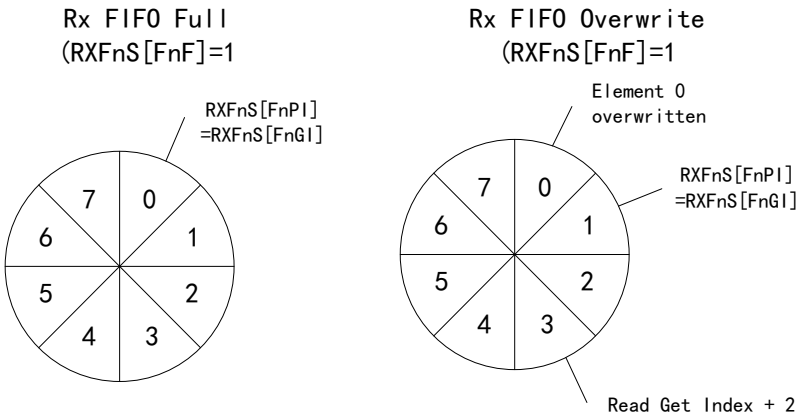
Rx FIFO Overwrite Mode

Rx FIFO Overwrite Mode is configured by setting RXFnC[FnOM] to 1.

When RXFnS[FnF] = 1 and it signals that the Rx FIFO is full (RXFnS[FnPI] = RXFnS[FnGI]), the next message accepted by the FIFO will overwrite the oldest FIFO message. Both the put index and the get index are incremented by 1.

When Rx FIFO operates in Overwrite Mode and signals that it is full, reading the Rx FIFO elements should start from at least the get index + 1. This is necessary because a situation may arise where the CPU is reading data from the message RAM (at the get index) while a received message is simultaneously being written to the message RAM (at the put index). In this case, inconsistent data might be read from the corresponding Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The required offset depends on the speed of the CPU's access to the Rx FIFO. The figure below illustrates a scenario where an offset of 2 relative to the get index is used during FIFO read operations. In this case, the two messages stored in Element 1 and 2 are lost.

Figure 252 Rx FIFO Overflow Processing



After reading data from an Rx FIFO, it must write the element number of the last read element to the Rx FIFO Acknowledge Index RXFnA[FnAI]. This action increments the get index to that element number. If the put index has not yet incremented to this Rx FIFO element, the Rx FIFO full condition will be reset (RXFnS[FnF] = 0).

39.5.4.3 Dedicated Rx buffer

The CAN FD controller supports up to 64 dedicated Rx buffers. The start address of the dedicated Rx buffer is configured via RXBC[RBSA].

For each Rx buffer, a standard or extension message ID filter element must be configured with SFEC/EFEC = 111 and SFID2/EFID2[10:9] = 00.

After a received message is accepted by a filter element, it is stored into the Rx buffer within the message RAM that is referenced by that filter element. Its

format is identical to that of the Rx FIFO element. Furthermore, the flag IR[DRX] (message stored to a dedicated Rx buffer) in the interrupt register is set.

Table 219 Rx Buffer Filter Configuration Example

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[5:0] EFID2[5:0]	SFID2[10:9] EFID2[10:9]
0	ID Message 1	00 0000	00
1	ID Message 2	00 0001	00
2	ID Message 3	00 0010	00

After the last word of the matching message received is written into the message RAM, the corresponding new data flag in the NDAT1 or NDAT2 register is set. As long as the new data flag is set, the corresponding Rx buffer is locked to prevent newly received matching frames from being updated. The host must reset the new data flag by writing “1” to the corresponding bit position.

When the new data flag of a Rx buffer is set, the message ID filter element referencing that specific Rx buffer will not produce a match, allowing the acceptance filtering process to continue. Subsequent message ID filter elements may cause the received message to be stored in another Rx buffer or the Rx FIFO, or rejected, according to the filter configuration.

Rx buffer handling

- Reading the message from the message RAM
- Resetting the interrupt flag IR[DRX]
- Resetting the new data flags for the processed messages
- Reading the new data register

39.5.5 Tx Handler

The Tx Handler is responsible for handling transmission requests from dedicated Tx buffers, Tx FIFOs, and Tx queues. It controls the transfer, the write and read indices of messages to be transmitted to the CAN FD core, as well as the Tx Event FIFO. Up to 32 Tx buffers can be configured for message transmission. The transmission CAN mode (classic CAN or CAN FD) for each Tx buffer element can be individually configured. The table below illustrates the possible configurations for frame transmission.

Table 220 Configurations for Frame Transmission

Tx Buffer Element		CCCR		Frame Transmission
FDF	BRS	BRSE	FDOE	
ignored	ignored	ignored	0	Classic CAN
0	ignored	0	1	Classic CAN
1	ignored	0	1	FD without bit rate switching
0	ignored	1	1	Classic CAN

Tx Buffer Element		CCCR		Frame Transmission
FDF	BRS	BRSE	FDOE	
1	0	1	1	FD without bit rate switching
1	1	1	1	FD with bit rate switching

Note: AUTOSAR requires at least three Tx queue buffers and supports transmission cancellation.

When the Tx Buffer Request Pending Register (TXBRP) is updated, the Tx Handler performs a Tx scanning of the message RAM to check for the highest-priority pending Tx request (the Tx buffer with the lowest message ID). TXBRP is updated when a transmission is completed, or after the host requests transmission by writing to TXBAR, or after the host cancels a pending transmission by writing to TXBCR.

When new scan results are available, temporary buffers are updated. These temporary buffers store the preloaded first four RAM words of the two messages with the highest Tx priority. The time required for Tx scanning and preloading depends on the host clock frequency, the number of configured Tx buffers, and the number of CAN FD modules connected to the message RAM.

Since the Tx scanning of the message RAM takes some time, the following situations may occur:

- Preloading of the temporary buffers is not completed before an ongoing transmission/ reception finishes: The Tx message cannot start transmission at the first opportunity. In this case, another node may start transmission without having to arbitrate with the CAN FD's transmission message. If this other message has a lower priority, this may be considered external priority inversion.
- Preloading of the temporary buffers is completed before an ongoing transmission/reception finishes: The preloaded Tx message CAN FD start transmission at the first opportunity.

39.5.5.1 Transmission pausing

The Transmission Pausing feature is applicable to CAN systems where CAN message identifiers are (permanently) assigned specific values and are not easily changed. These message identifiers might have a higher CAN FD arbitration priority than other defined messages, whereas in specific applications, their relative arbitration priority should be the opposite. This CAN FD lead to a situation where one ECU sends a series of CAN messages, causing CAN messages from another ECU to be delayed because these other messages have a lower CAN arbitration priority.

For example, if CAN FD ECU-1 enables the Transmission Pausing feature and its application software requests the transmission of four messages, then after the first successful message transmission, it will wait for two CAN FD bit times

of bus idle time before allowing the next requested message transmission to begin. If other ECUs have messages pending for transmission, these messages will start transmission during the idle time and do not need to arbitrate with ECU-1's next message. After receiving a message, ECU-1 CAN FD start its next transmission once the received message has released the CAN bus.

The Transmission Pausing feature is controlled by the CCCR[TXP] bit. If this bit is set to 1, the CAN FD will pause for the time of two CAN bits after each successful message transmission before starting the next transmission. This allows other CAN FD nodes in the network to transmit messages, even if their messages have lower-priority identifiers. By default, the Transmission Pausing feature is disabled (CCCR[TXP] = 0).

This feature mitigates burst transmissions from a single node and prevents "erroneous instructions," where the application incorrectly requests an excessive number of transmissions.

39.5.5.2 Dedicated Tx buffer

Dedicated Tx buffers are used for message transmission under the full control of the host CPU. Each dedicated Tx buffer is configured with a specific message ID. When multiple Tx buffers are configured with the same message ID, the Tx buffer with the lowest buffer number implements the transmission first. This can be achieved in one of two ways:

- Tx buffers should be requested in ascending numerical order, starting with the lowest-numbered buffer.
- All Tx buffers configured with the same message ID can be requested simultaneously via a single write access to the TXBAR register.

A transmission is requested by sending an Add Request via the TXBAR[Arn] bit, typically after the message data has been updated. The requested message arbitrates internally with messages from the optional Tx FIFO or Tx queues, and externally with messages on the CAN bus, before being transmitted based on their message IDs.

A dedicated Tx buffer is allocated an element in the message RAM consisting of 32-bit words (see the table below). Consequently, the start address of a dedicated Tx buffer in the message RAM is calculated by multiplying the Tx Buffer Index (0…31) by the element size and adding it to the Tx buffer start address (TXBC[TBSA]).

Table 221 Tx Buffer, FIFO and Queue Element Size

TXESC[TBDS][2:0]	Data Field (bytes)	Element Size (RAM words)
000	8	4
001	12	5
010	16	6

TXESC[TBDS][2:0]	Data Field (bytes)	Element Size (RAM words)
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

39.5.5.3 Tx FIFO

Tx FIFO operation is configured by programming TXBC[TFQM] to 0. Messages stored in the Tx FIFO are transmitted starting from the message referenced by the read index TXFQS[TFGI]. After each transmission, the read index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO allows the transmission of messages with the same message ID from different Tx buffers in the order they were written into the Tx FIFO. The CAN FD core calculates the Tx FIFO Free Level TXFQS[TFFL] as the difference between the write index and the read index. It indicates the number of available (free) Tx FIFO elements.

New transmission messages must be written into the Tx FIFO starting from the Tx buffer referenced by the write index TXFQS[TFQPI]. An Add Request increments the write index to the next free Tx FIFO element. When the write index reaches the read index, a Tx FIFO Full condition (TXFQS[TFQF]=1) is signaled. In this case, no further messages should be written to the Tx FIFO until the next message is transmitted and the read index is incremented.

When adding a single message to the Tx FIFO, transmission is requested by writing “1” to the TXBAR bit associated with the Tx buffer referenced by the Tx FIFO’s write index.

When adding multiple (n) messages to the Tx FIFO, they are written into n consecutive Tx buffers starting from the write index. Transmission is then requested via TXBAR. The write index is cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx buffers indicated by the Tx FIFO Free Level.

When a transmission request is cancelled for the Tx buffer referenced by the read index, the read index is incremented to the next Tx buffer with a pending transmission request, and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx buffer, the read index and FIFO free level remain unchanged.

A Tx FIFO element is allocated in the Message RAM with an element size of 32-bit words. Therefore, the start address of the next available (free) Tx FIFO buffer is calculated by multiplying the Tx FIFO/Queue Write Index TXFQS[TFQPI] (0…31) by the element size and adding it to the Tx Buffer Start Address TXBC[TBSA].

39.5.5.4 Tx Queue

Tx queue operation is configured by programming TXBC[TFQM] to 1. Messages stored in the Tx queue are transmitted starting from the message with the lowest Message ID (highest priority). If multiple queue buffers are configured with the same Message ID, the transmission order depends on the buffer numbers where the messages are stored for transmission. Since these buffer numbers depend on the current state of the write index, the transmission order cannot be predicted.

New messages must be written to the Tx buffer referenced by the write index TXFQS[TFQPI]. The write index always points to the lowest-numbered free buffer in the Tx queue. If the Tx queue is full (TXFQS[TFQF]=1), the write index becomes invalid, and no further messages should be written to the Tx queue until at least one requested message has been transmitted or a pending transmission request has been cancelled.

The APP CAN FD use the TXBRP register instead of the write index and CAN FD place messages into any Tx buffer that does not have a pending transmission request.

A Tx queue buffer is allocated in the Message RAM with an element size of 32-bit words. Therefore, the start address of the next available (free) Tx queue buffer is calculated by multiplying the Tx FIFO/Queue Write Index TXFQS[TFQPI] (0…31) by the element size and adding it to the Tx Buffer Start Address TXBC[TBSA].

39.5.5.5 Mixed Dedicated Tx Buffers / Tx FIFO

In this configuration, the Tx buffer section in the Message RAM is divided into a set of Dedicated Tx Buffers and one Tx FIFO.

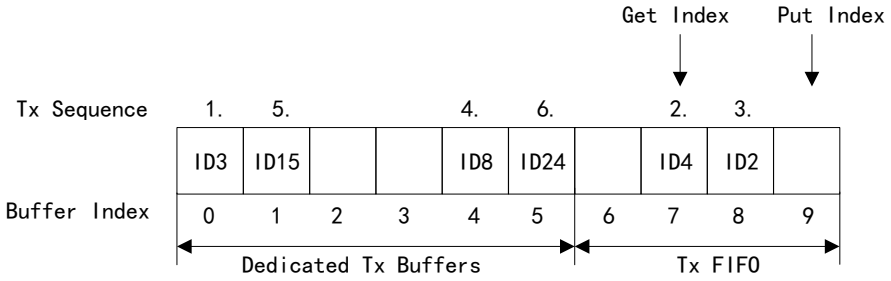
- The number of Dedicated Tx Buffers is configured by TXBC[NDTB].
- The number of Tx Buffers allocated to the Tx FIFO is configured by TXBC[TFQS].

Note: If TXBC[TFQS] is programmed to zero, only Dedicated Tx Buffers are used.

Tx Priority:

- The buffer with the lowest Message ID has the highest priority and will be transmitted next.
- Scans both the Dedicated Tx Buffers and the oldest pending Tx FIFO buffer (referenced by TXFQS[TFGI]) during next transmission.

Figure 253 Example of a Mixed Configuration with Dedicated Transmission Buffers / Transmission FIFO



39.5.5.6 **Mixed Dedicated Tx Buffers / Tx Queue**

In this configuration, the Tx buffer section in the Message RAM is divided into a set of Dedicated Tx Buffers and one Tx Queue.

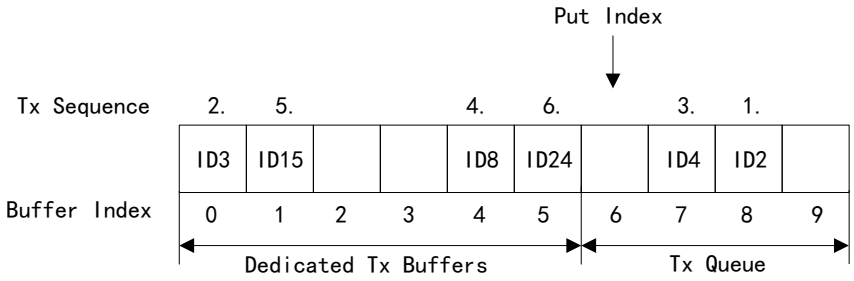
- The number of Dedicated Tx Buffers is configured by TXBC[NDTB].
- The number of Tx Queue Buffers is configured by TXBC[TFQS].

Note: If TXBC[TFQS] is programmed to zero, only Dedicated Tx Buffers are used.

Tx Priority:

- The buffer with the lowest Message ID has the highest priority and will be transmitted next.
- Scan all Tx Buffers with the transmission enabling request during next transmission.

Figure 254 Example of a Mixed Configuration with Dedicated Transmission Buffers / Transmission Queues



39.5.5.7 **Transmission cancellation**

CAN FD supports transmission cancellation. This feature is particularly useful for gateway applications and AUTOSAR-based applications.

To cancel a transmission request from the Dedicated Tx Buffer or Tx Queue Buffer, the host must write “1” to the corresponding bit (= Tx Buffer number) of the TXBCR register. A successful cancellation is indicated by setting the corresponding bit in the TXBCF register to “1”.

Transmission cancellation does not apply to Tx FIFO operation. If transmission

cancellation is requested for a Tx Buffer while its transmission is already ongoing, the corresponding TXBRP bit will remain set as long as the transmission is still in progress. If the transmission is successful, the corresponding TXBTO and TXBCF bits will be set. If the transmission fails, the message will not be retransmitted and only the corresponding TXBCF bit will be set.

Note: There is a short time window where, if a pending transmission is cancelled immediately before its transmission is about to start, no transmission will be initiated by this node even if it has other pending messages. This may allow another node to transmit a message whose priority is lower than that of the second message in this node.

39.5.5.8 Tx event handling

To support Tx event handling, the CAN FD module implements a Tx Event FIFO. After a message is transmitted on the CAN bus, the Message ID and the timestamp are stored in the Tx Event FIFO element. To link the Tx event to the Tx Event FIFO element, the Message Marker from the transmitted Tx buffer is copied into the Tx Event FIFO element.

The configuration of CCCR[WMM] determines whether an 8-bit or 16-bit Message Marker is used. When CCCR[WMM] = 1, the internal timestamp function is disabled.

The Tx Event FIFO can be configured for up to 32 elements.

The purpose of the Tx Event FIFO is to decouple the handling of transmission status information from the handling of the transmitted messages themselves. That is, the Tx buffers only hold the messages to be transmitted, while the transmission status information is stored separately in the Tx Event FIFO. This offers some advantages, particularly during dynamic management of transmission queues, as the Tx buffers can be immediately reused for a new message after a successful transmission. There is no need to preserve the transmission status information of the Tx buffers before overwriting them.

When IR[TEFF] signals that the Tx Event FIFO is full, no new elements are written to it until at least one element is read and the Tx Event FIFO read index is incremented. If a Tx event occurs while the Tx Event FIFO is full, the event is discarded and the interrupt flag IR[TEFL] is set.

To prevent Tx Event FIFO overflow, a Tx Event FIFO watermark can be used. The interrupt flag IR[TEFW] is set when the fill level of the Tx Event FIFO reaches the Tx Event FIFO watermark configured by TXEFC[EFWM].

When reading data from the Tx Event FIFO, twice the Tx Event FIFO read index TXEFS[EFGI] must be added to the Tx Event FIFO Start Address TXEFC[EFSA].

39.5.6 FIFO Acknowledge Handling

The read indices for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index. Writing to the FIFO Acknowledge Index sets the FIFO read index to the Acknowledge Index value plus one, thereby updating the FIFO fill level. There are two primary use cases:

When reading only a single element (the element pointed to by the read index) from the FIFO, write this read index value to the FIFO Acknowledge Index.

When reading a sequence of elements from the FIFO, write the FIFO Acknowledge Index only once at the end of the read sequence (using the index of the last read element) to update the FIFO read index.

Since the CPU has free access to the CAN FD's Message RAM, special care must be taken when reading FIFO elements in an arbitrary order (disregarding the read index). This might be useful, for example, when reading a high-priority message from one of the two Rx FIFOs. In such a case, the FIFO's Acknowledge Index should not be written, as this would set the read index to an incorrect position and also incorrectly change the FIFO fill level. Consequently, some older FIFO elements would be lost.

Note: The application must ensure that the value written to the FIFO Acknowledge Index is valid. The CAN FD module does not perform checks for erroneous values.

39.5.7 Message RAM

To store received (Rx)/transmitted (Tx) messages and filter configurations, the CAN FD module must be connected to a 4 KB single-port SRAM, referred to as the Message RAM.

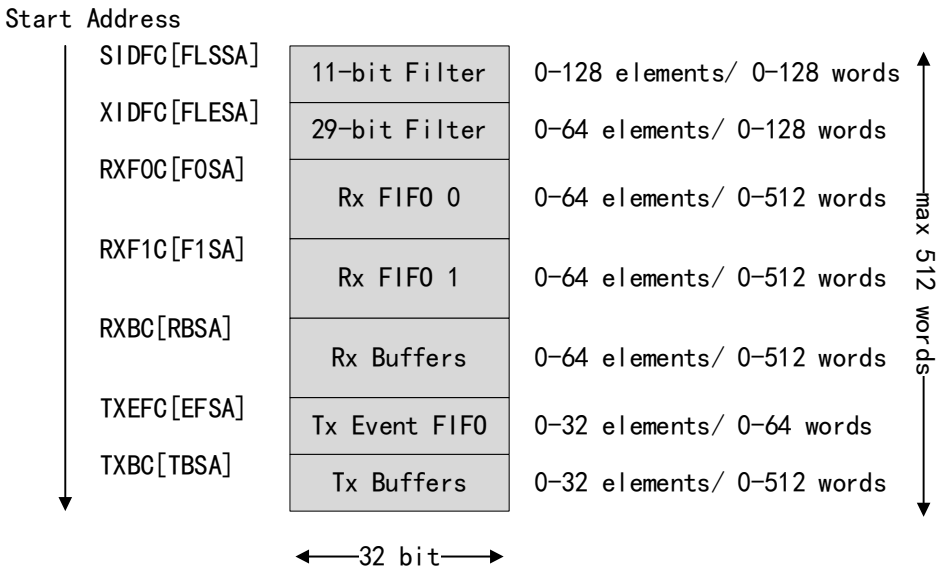
Note: The Message RAM supports ECC (Error Correcting Code) function. It is recommended to initialize the Message RAM after hardware reset. For example, by writing 0x00000000 to each word of the Message RAM to generate valid check bits. This avoids falsely triggering ECC interrupts when reading data from uninitialized sections of the Message RAM, in case the ECC check feature is enabled.

39.5.7.1 Message RAM configuration

The Message RAM has a width of 32 bits. The CAN FD module can be configured to allocate up to 1024 words in the Message RAM. It is not necessary to configure every section listed in the figure below, and there are no restrictions on the order in which the sections are arranged.

In CAN FD mode, the required Message RAM size depends heavily on the element sizes configured via RXESC[F0DS], RXESC[F1DS], RXESC[RBDS], and TXESC[TBDS] for the Receive FIFO 0 (Rx FIFO 0), Receive FIFO 1 (Rx FIFO 1), Receive Buffers (Rx Buffers), and Transmit Buffers (Tx Buffers), respectively.

Figure 255 Message RAM Configuration



When the CAN FD module addresses the Message RAM, it accesses in 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses. This means that only bits [15:2] of the address are evaluated, and the two least significant bits are ignored.

Note: The CAN FD module does not check for errors in the Message RAM configuration. Particular care must be taken when configuring the start addresses of different sections and the number of elements per section to avoid data corruption or loss.

39.5.7.2 Rx Buffers and FIFO Elements

Up to 64 Receive Buffers (Rx Buffers) and two Receive FIFOs can be configured in the Message RAM. Each Receive FIFO can be configured to store up to 64 received messages. The table below shows the structure of a Receive Buffer / FIFO element. The element size can be configured via the RXESC register to store CAN FD messages with data fields up to 64 bytes.

Table 222 Rx Buffers and FIFO Elements

Name	31:24			23:16				15:8	7:0
R0	ESI	XTD	RTR	ID[28:0]					
R1	ANMF	FIDX[6:0]		Reserved	FDF	BRS	DLC[3:0]	RXTS[15:0]	
R2	DB3[7:0]			DB2[7:0]				DB1[7:0]	DB0[7:0]
R3	DB7[7:0]			DB6[7:0]				DB5[7:0]	DB4[7:0]
…	…			…				…	…
Rn	DBm[7:0]			DBm-1[7:0]				DBm-2[7:0]	DBm-3[7:0]

R0 Register Bit Description

Table 223 R0 Register Bit Description

Field	Name	Description
31	ESI	Error State Indicator This bit indicates the error state of the transmitting node. 0: Indicates that the transmitting node is in the error active state, meaning that this node has not accumulated a critical number of errors and CAN FD participate in bus communication normally. 1: Indicates the transmitting node is in the error passive state, meaning that this node has accumulated a significant number of errors and its participation in bus communication is restricted.
30	XTD	Extended Identifier This bit informs the host whether the received frame uses a standard or extended identifier. 0: Indicates that the received frame uses an 11-bit standard identifier which is commonly used in CAN bus communication. 1: Indicates that the received frame uses a 29-bit extended identifier which provides a larger address space for more complex networks or a higher number of nodes.
29	RTR	Remote Transmission Request This bit informs the host whether the received frame is a data frame or a remote frame. 0: Indicates that the received frame is a data frame which contains actual data payload. 1: Indicates that the received frame is a remote frame which does not contain data but requests another node to transmit a data frame with the same identifier. Note: Remote frames do not exist in the CAN FD format. In a CAN FD frame (FDF=1), a dominant RRS (Remote Request Substitute) bit replaces the RTR bit.
28:0	ID	Identifier The type of identifier (standard or extended) is determined by the XTD bit. If XTD = 0 (Standard Identifier), the Standard Identifier is stored in ID[28:18]. If XTD = 1, then use Extended Identifier.

R1 Register Bit Description

Table 224 R1 Register Bit Description

Field	Name	Description
31	ANMF	Accepted non-matching frame Acceptance of non-matching frames can be enabled via GFC[ANFS] and GFC[ANFE]. 0: Indicates that the received frame matched a filter index FIDX. 1: Indicates that the received frame did not match any receive filter element.
30:24	FIDX[6:0]	Filter Index This value, in the range of 0-127, represents the index of the matching Receive Accept Filter element (this value is invalid if ANMF=1). The valid range is 0 to SIDFC[LSS]-1 (for Standard Identifiers) or 0 to XIDFC[LSE]-1 (for Extended Identifiers).
21	FDF	FD Format This bit indicates the frame format. 0: Legacy CAN frame format. 1: CAN FD frame format (uses new DLC code and CRC).

Field	Name	Description
20	BRS	Bit Rate Switch This bit indicates whether the received frame performed bit rate switching. 0: Indicates that the received frame did not use bit rate switching; 1: Indicates that the received frame did use bit rate switching
19:16	DLC[3:0]	Data Length Code In both CAN and CAN FD, a value of 0-8 indicates the received frame has 0-8 data bytes. In CAN, a value of 9-15 indicates the received frame has 8 data bytes. In CAN FD, a value of 9-15 indicates the received frame has 12/16/20/24/32/48/64 data bytes respectively.
15:0	RXTS[15:0]	Receive Timestamp This field captures the value of the Timestamp Counter when the frame's reception started. Its resolution depends on the configuration of the Timestamp Counter Prescaler TSCC[TCP].

R2-Rn Register Bit Description

Table 225 R2 Register Bit Description

Field	Name	Description
31:24	DB3[7:0]	Data Byte 3
23:16	DB2[7:0]	Data Byte 2
15:8	DB1[7:0]	Data Byte 1
7:0	DB0[7:0]	Data Byte 0

This pattern continues in R3 register and subsequent registers, used for storing the data bytes of the CAN message.

Note: Depending on the configured element size (RXESC), 2 to 16 words of 32 bits (Rn=2…17) are used to store the data field of CAN messages.

39.5.7.3 Tx buffer elements

The Transmit Buffer section can be configured to Dedicated Transmit Buffers as well as a Transmit FIFO (Tx FIFO) / Transmit Queue. If the Transmit Buffer section is shared by Dedicated Transmit Buffers and a Tx FIFO / Tx Queue, then the Dedicated Transmit Buffers start at the beginning of the Transmit Buffer section, followed by the buffers allocated to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between Dedicated Transmit Buffers and the Tx FIFO / Tx Queue by evaluating the transmit buffer configuration registers TXBC[TFQS] and TXBC[NDTB]. The element size can be configured via the TXESC register to store CAN FD messages with data fields up to 64 bytes in length.

Table 226 Tx Buffer Elements

Name	31:24			23:16	15:8	7:0
T0	ESI	XTD	RTR	ID[28:0]		

Name	31:24	23:16					15:8	7:0
T1	MM[7:0]	EFC	Reserved	FDF	BRS	DLC[3:0]	MM[15:8]	Reserved
T2	DB3[7:0]	DB2[7:0]					DB1[7:0]	DB0[7:0]
T3	DB7[7:0]	DB6[7:0]					DB5[7:0]	DB4[7:0]
…	…	…					…	…
Tn	DBm[7:0]	DBm-1[7:0]					DBm-2[7:0]	DBm-3[7:0]

T0 Register Bit Description

Table 227 T0 Register Bit Description

Field	Name	Description
31	ESI	Error State Indicator 0: In CAN FD format, the ESI bit depends solely on the Error Passive Flag. 1: The transmitted ESI bit in CAN FD format is set to recessive. Note: The “OR” operation between ESI bit in the Transmit Buffer and the Error Passive Flag is implemented to determine the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, a node in the error active state may optionally transmit a recessive ESI bit, but a node in the error passive state will always transmit a recessive ESI bit.
30	XTD	Extended Identifier 0: 11-bit standard identifier. 1: 29-bit extended identifier.
29	RTR	Remote Transmission Request 0: Transmit a data frame. 1: Transmit a remote frame. Note: When RTR=1, even if CCCR[FDOE] enables transmission in CAN FD format, the CAN FD module will transmit a remote frame according to the ISO 11898-1:2015.
28:0	ID	Identifier The type of identifier (standard or extended) is determined by the XTD bit. Standard identifiers must be written into ID[28:18].

T1 Register Bit Description

Table 228 T1 Register Bit Description

Field	Name	Description
31:24	MM [7:0]	Message Marker Written by the CPU during Tx Buffer configuration. It is copied into the Tx Event FIFO element to identify the status of the transmitted message.
23	EFC	Event FIFO Control 0: Do not store a Tx event. 1: Store a Tx event.
21	FDF	FD Format 0: Transmit the frame in classic CAN format. 1: Transmit the frame in CAN FD format.

Field	Name	Description
20	BRS	Bit Rate Switch 0: The CAN FD frame is transmitted without bit rate switching. 1: The CAN FD frame is transmitted with bit rate switching. Note: The ESI, FDF, and BRS bits are only evaluated when CAN FD operation is enabled (CCCR[FDOE]=1). Additionally, the BRS bit is only evaluated when CCCR[BRSE]=1.
19:16	DLC [3:0]	Data Length Code 0-8: In both CAN and CAN FD, the transmitted frame has 0-8 data bytes. 9-15: In CAN, the transmitted frame has 8 data bytes; in CAN FD, the transmitted frame has 12/16/20/24/32/48/64 data bytes.
15:8	MM[15:8]	Message Marker The high bytes of the Message Marker, are written by the CPU during Tx Buffer configuration. It is copied into the Tx Event FIFO element to identify the status of the transmitted message. Only available when CCCR[WMM]=1.

T2-Tn Register Bit Description

Table 229 T2 Register Bit Description

Field	Name	Description
31:24	DB3[7:0]	Data Byte 3
23:16	DB2[7:0]	Data Byte 2
15:8	DB1[7:0]	Data Byte 1
7:0	DB0[7:0]	Data Byte 0

This pattern continues in T3 register and subsequent registers, used for storing the data bytes of the CAN message.

Note: Depending on the configured element size (TXESC), 2-16 words of 32 bits (Tn=2…17) are used to store the data field of CAN messages.

39.5.7.4 Tx Event FIFO elements

Each element stores information about a transmitted message. The host CPU CAN FD retrieve this information in the order the messages were sent by reading the Tx Event FIFO. The status information on the Tx Event FIFO can be obtained from the TXEFS register.

E1A: When CCCR[WMM] = 0, E1A[TXTS][15:0] holds the 16-bit timestamp generated by the CAN FD internal timestamp logic.

E1B: When the 16-bit Message Marker is enabled (CCCR[WMM] = 1), E1B[MM][15:8] holds the upper 8 bits of the wide Message Marker.

Table 230 Tx Event FIFO Elements

Name	31:24			23:16	15:8	7:0
E0	ESI	XTD	RTR	ID[28:0]		

Name	31:24	23:16				15:8	7:0
E1A	MM[7:0]	ET[1:0]	FDF	BRS	DLC[3:0]	TXTS[15:0]	
E1B	MM[7:0]	ET[1:0]	FDF	BRS	DLC[3:0]	MM[15:8]	Reserved

E0 Register Bit Description

Table 231 E0 Register Bit Description

Field	Name	Description
31	ESI	Error State Indicator 0: The transmitting node is in the error active state. 1: The transmitting node is in the error passive state.
30	XTD	Extended Identifier 0: 11-bit standard identifier. 1: 29-bit extended identifier.
29	RTR	Remote Transmission Request 0: A data frame was transmitted. 1: A remote frame was transmitted.
28:0	ID	Identifier The type of identifier (standard or extended) is determined by the XTD bit. Standard identifiers must be stored into ID[28:18].

E1A/B Register Bit Description

Table 232 E1A/B Register Bit Description

Field	Name	Description
31:24	MM [7:0]	Message Marker Copied from the Transmit Buffer into the Tx Event FIFO element to identify the status of the transmitted message.
23:22	ET	Event Type 00: Reserved. 01: Transmission Event. 10: Transmission occurred despite cancellation (Transmissions in DAR mode always set this status). 11: Reserved.
21	FDF	FD Format 0: Legacy CAN frame format. 1: CAN FD frame format (uses new Data Length Code (DLC) encoding and Cyclic Redundancy Check (CRC).
20	BRS	Bit Rate Switch 0: The frame was transmitted without bit rate switching. 1: The frame was transmitted with bit rate switching.
19:16	DLC[3:0]	Data Length Code 0-8: In both CAN and CAN FD, the transmitted frame has 0-8 data bytes. 9-15: In CAN, the transmitted frame has 8 data bytes; in CAN FD, the transmitted frame has 12/16/20/24/32/48/64 data bytes.
15:0	TXTS[15:0]	Transmit Timestamp

Field	Name	Description
		Only E1A is supported. Captures the value of the Timestamp Counter when the frame's transmission started. Its resolution depends on the configuration of the Timestamp Counter Prescaler TSCC[TCP].
15:8	MM[15:8]	Message Marker Only E1B is supported. The high byte of the Message Marker, written by the CPU during Tx Buffer configuration. It is copied into the Tx Event FIFO element to identify the status of the transmitted message.

39.5.7.5 Standard message ID filter element

The 11-bit IDs can be configured with up to 128 filter elements. When accessing a standard message ID filter element, its address is the start address SIDFC[FLSSA] of Message Filter List plus the filter element index (ranging from 0 to 127).

Table 233 Standard Message ID Filter Elements

Name	31:24		23:16	15:8	7:0
S0	SFT[1:0]	SFEC[2:0]	SFID1[10:0]	Reserved	SFID2[10:0]

Table 234 Register Bit Description

Field	Name	Description
31:30	SFT [1:0]	Standard Filter Type 00: Range filter from SFID1 to SFID2 (requires SFID2≥SFID1). 01: Dual ID filter for either SFID1 or SFID2. 10: Classic filter: SFID1 serves as the filter, SFID2 serves as the mask. 11: Filter element disabled. Note: When SFT = "11", the filtered elements are disabled, and acceptance filtering continues (behaving the same as SFEC = "000")
29:27	SFEC [2:0]	Standard Filter Element Configuration All enabled filter elements are used for acceptance filtering of frames with 11-bit IDs. Acceptance filtering stops at the first matching enabled filter element or at the end of the filter list. If SFEC is "100", "101", or "110", a match sets the interrupt flag IR[HPM] and, if the interrupt feature is enabled, an interrupt will be generated. In this case, the HPMS register is updated with the priority match status. 000: Disable filter element. 001: If filter matches, the register will store message in Rx FIFO 0. 010: If filter matches, the register will store message in Rx FIFO 1. 011: If filter matches, the register will reject this ID (not suitable for synchronous messages). 100: If filter matches, the register will set priority (not suitable for synchronous messages, no storage). 101: If filter matches, the register will set priority and store message in FIFO 0. 110: If filter matches, the register will set priority and store message in FIFO 1.

Field	Name	Description
		111: The message is store into a Receive Buffer or as a Debug message (the SFT[1:0] configuration is ignored in this case).
26:16	SFID1 [10:0]	Standard Filter ID 1 This is the first ID for the standard ID filter element. When filtering for a Receive Buffer, synchronous message, or debug message, this field defines the ID of the message to be stored. The received identifier must match exactly; no masking mechanism is used.
10:0	SFID2 [10:0]	Standard Filter ID 2 The meaning of this bit field depends on the SFEC configuration: When SFEC has a value between “001” and “110”, it is the second ID for the standard ID filter element. When SFEC = “111”, it is used for filtering for a Receive Buffer or Debug message: SFID2[10:9] determine whether the received message is stored into a receive buffer or handled as Message A, B, or C in a debug message sequence: 00: Store message into the receive buffer. 01: Debug Message A. 10: Debug Message B. 11: Debug Message C. Note: CAN FD module does not support debug messages. SFID2[8:6]: Reserved. SFID2[5:0] define the offset, relative to the Receive Buffer Start Address RXBC[RBSA], used for storing the matching messages.

39.5.7.6 Extended message ID filter element

The 29-bit IDs can be configured with up to 64 filter elements. The address for accessing an extended message ID filter element is calculated as the extended filter list start address XIDFC[FLESA] plus double the filter element index (range: 0 to 63).

Table 235 Extended Message ID Filter Elements

Name	31:24		23:16	15:8	7:0
F0	EFEC[2:0]		EFID1[28:0]		
F1	EFT[1:0]	Reserved	EFID2[28:0]		

F0 Register Bit Description

Table 236 F0 Register Bit Description

Field	Name	Description
31:29	EFEC [2:0]	Extended Filter Element Configuration All enabled filter elements are used for acceptance filtering of frames with 29-bit IDs. Acceptance filtering stops at the first matching enabled filter element or at the end of the filter list. If EFEC is “100”, “101”, or “110”, a match sets the interrupt flag IR[HPM] and, if the interrupt feature is enabled, interrupts will be generated. In this case, the HPMS register is updated with the priority match status.

Field	Name	Description
		000: Disable filter element. 001: If filter matches, the message will be stored in Rx FIFO 0. 010: If filter matches, the message will be stored in Rx FIFO 1. 011: If filter matches, the ID will be rejected (not suitable for synchronous messages). 100: If filter matches, the priority will be set (not suitable for synchronous messages, no storage). 101: If filter matches, the priority is set and the message is stored in FIFO 0. 110: If filter matches, the priority is set and the message is stored in FIFO 1. 111: The message is stored into a Receive Buffer or as a Debug message (the EFT[1:0] configuration is ignored in this case).
28:0	EFID1 [28:0]	Extended Filter ID 1 This is the first ID for the extended ID filter element. When filtering for a Receive Buffer, synchronous message, or debug message, this field defines the ID of the extended message to be stored. The received identifier must match exactly; only XIDAM masking mechanism is used.

F1 Register Bit Description

Table 237 F1 Register Bit Description

Field	Name	Description
31:30	EFT [1:0]	Extended Filter Type 00: Range filter from EFID1 to EFID2 (requires EFID2 ≥ EFID1). 01: Dual ID filter for either EFID1 or EFID2. 10: Classic filter: EFID1 serves as the filter, EFID2 serves as the mask. 11: Range filter from EFID1 to EFID2 (requires EFID2 ≥ EFID1), XIDAM mask is not applied.
28:0	EFID2 [28:0]	Extended Filter ID 2 The meaning of this bit field depends on the EFEC configuration: When EFEC has a value between “001” and “110”, it is the second ID for the extended ID filter element. When EFEC = “111”, it is used for filtering for Receive Buffer or Debug message: EFID2[10:9] determine whether the received message is stored into a receive buffer or handled as Message A, B, or C in a debug message sequence: 00: Store message into the receive buffer. 01: Debug Message A. 10: Debug Message B. 11: Debug Message C. Note: CAN FD module does not support debug messages. EFID2[8:6]: Reserved. EFID2[5:0] define the offset, relative to the Receive Buffer Start Address RXBC[RBSA], used for storing the matching messages.

39.6 Register bank address

Table 238 CAN FD Register Bank Address

Device register	Register bank	Start address	End address
CanfdaRegs	CANFDA_REGS	0x4000_2C00	0x4000_33FF

39.7 Register Address Mapping

Table 239 CAN FD Register Address Mapping

Register name	Description	Offset address
CREL	Core Release Register	0x00
ENDN	Endianness Register	0x04
-	Reserved	0x08
DBTP	Data Bit Timing and Prescaler Register	0x0C
TEST	Test Register	0x10
RWD	RAM Watchdog	0x14
CCCR	CC Control Register	0x18
NBTP	Nominal Bit Timing and Prescaler Register	0x1C
TSCC	Timestamp Counter Configuration Register	0x20
TSCV	Timestamp Counter Value Register	0x24
TOCC	Timeout Counter Configuration Register	0x28
TOCV	Timeout Counter Value Register	0x2C
-	Reserved	0x30-0x3C
ECR	Error Counter Register	0x40
PSR	Protocol Status Register	0x44
TDCR	Transmitter Delay Compensation Register	0x48
-	Reserved	0x4C
IR	Interrupt register	0x50
IE	Interrupt enable	0x54
ILS	Interrupt Line Selection Register	0x58
ILE	Interrupt Line Enable Register	0x5C
-	Reserved	0x60-0x7C
GFC	Global Filter Configuration Register	0x80
SIDFC	Standard ID Filter Configuration Register	0x84
XIDFC	Extended ID Filter Configuration Register	0x88

Register name	Description	Offset address
-	Reserved	0x8C
XIDAM	Extended ID and Mask Register	0x90
HPMS	High Priority Message Status Register	0x94
NDAT1	New Data 1	0x98
NDAT2	New Data 2	0x9C
RXF0C	Rx FIFO 0 Configuration	0xA0
RXF0S	Rx FIFO 0 Status	0xA4
RXF0A	Rx FIFO 0 Acknowledge	0xA8
RXBC	Rx Buffer Configuration	0xAC
RXF1C	Rx FIFO 1 Configuration	0xB0
RXF1S	Rx FIFO 1 Status	0xB4
RXF1A	Rx FIFO 1 Acknowledge	0xB8
RXESC	Rx Buffer/ FIFO Element Size Configuration	0xBC
TXBC	Tx Buffer Configuration	0xC0
TXFQS	Tx FIFO/ Queue Status	0xC4
TXESC	Tx Buffer Element Size Configuration	0xC8
TXBRP	Tx Buffer Request Pending	0xCC
TXBAR	Tx Buffer Add Request	0xD0
TXBCR	Tx Buffer cancel Request	0xD4
TXBTO	Tx Buffer Transmission Occurred	0xD8
TXBCF	Tx Buffer cancellation Finished	0xDC
TXBTIE	Tx Buffer Transmission Interrupt Enable	0xE0
TXBCIE	Tx Buffer cancellation Interrupt Enable	0xE4
-	Reserved	0xE8-0xEC
TXEFC	Tx Event FIFO Configuration	0xF0
TXEFS	Tx Event FIFO Status	0xF4
TXEFA	Tx Event FIFO Acknowledge	0xF8
-	Reserved	0xFC-0xFF
ERM_CTRL	ERM and Low-Power Control Register	0x100
ERM_STS	ERM and Low-Power Status Register	0x104
ERM_ECC_ADDR_LOG	ERM error address register	0x108
ERM_ECC_DATA_LOG	ECC error address log	0x10C
ERM_RAM_INIT	RAM hardware initialisation	0x110

Register name	Description	Offset address
ERM_EXTC_CTRL	External timestamp configuration	0x114
ERM_EXTC_VAL	External timestamp value	0x118
-	Reserved	0x11C-0x7FF

39.8 Register Functional Description

The CAN FD register only supports 32-bit access.

39.8.1 Core Release Register (CREL)

Offset address: 0x000

Reset value: 0x3315 0415

Field	Name	R/W	Description
7:0	DAY	R	Timestamp Day A number, BCD code
15:8	MON	R	Timestamp Month A number, BCD code
19:16	YEAR	R	Timestamp Year A number, BCD code
23:20	SUBSTEP	R	Core Release Sub-step A number, BCD code
27:24	STEP	R	Core Release Step A number, BCD code
31:28	REL	R	Core Version A number, BCD code

R = Read; -r = Release; -d = Timestamp; a value defined by general parameters during synthesis.

39.8.2 Endianness Register (ENDN)

Offset address: 0x0004

Reset value: 0x8765 4321

Field	Name	R/W	Description
31:0	ETV	R	Endianness Test Value Test value: 0x8765 4321

39.8.3 Data Bit Timing and Prescaler Register (DBTP)

Offset address: 0x00C

Reset value: 0x0000 0A33

This register is writable only when both the CCCR[CCE] and CCCR[INIT] bits are set. The CAN bit time can be programmed within a range of 4 to 49 time quanta (tq). The CAN time quantum can be programmed within a range of 1 to 32 canfdx bit clk cycles. Tq = (DBRP + 1) mtq.

Therefore, the length of the bit time is (programmed value) [DTSEG1 + DTSEG2 + 3] tq, or (functional value) [Sync_Seg + Prop_Seg + Phase_Seg1 +

Phase_Seg2] tq. The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sampling point.

Field	Name	R/W	Description
3:0	DSJW	R/W	Data (Re)Synchronization Jump Width 0x00 – 0x0F: Valid values are 0 to 15. The hardware interprets this value as the programmed value plus one.
7:4	DTSEG2	R/W	Data Time Period after Sampling Point 0x00 – 0x0F: Valid values are 0 to 15. The hardware interprets this value as the programmed value plus one.
12:8	DTSEG1	R/W	Data Time Period before Sampling Point 0x00 – 0x1F: Valid values are 0 to 31. The hardware interprets this value as the programmed value plus one.
15:13	Reserved		
20:16	DBRP	R/W	Data Bit Rate Prescaler 0x00 – 0x1F: This value is used to divide the oscillator frequency to generate the bit time quantum (tq). The bit time is composed of multiples of this quantum. The valid range for the bit timing prescaler is 0 to 31. When TDC = 1, the range is restricted to 0 or 1. The hardware interprets this value as the programmed value plus one.
22:21	Reserved		
23	TDC	R/W	Transmitter Delay Compensation (TDC) 0: Transmitter delay compensation disabled. 1: Transmitter delay compensation enabled.
31:24	Reserved		

R = Read, P = Protected write; -n = value after reset

Note: With a CAN FD communication clock (canfdx bit clk) of 8 MHz, the reset value 0x0000 0A33 configures the CAN FD for a data phase bit rate of 500 kbit/s. The bit rate configured for the CAN FD data node via DBTP must be greater than or equal to the bit rate configured for the arbitration phase via NBTP.

39.8.4 Test Register (TEST)

Offset address: 0x10

Reset value: 0x0000 0000

Write access to the test registers is only enabled after setting the CCCR[TEST] bit to 1. When the CCCR[TEST] bit is cleared, all test register functions are reset to their default values.

Loopback mode and software control of the canfdx_tx pin are hardware test modes. Programming the Tx control field to a value other than “00” may interfere with message transmission on the CAN bus.

Field	Name	R/W	Description
3:0	Reserved		
4	LBCK	R/W	Loopback Mode 0: Reset value, Loopback mode disabled; 1: Loopback mode enabled.

Field	Name	R/W	Description
6:5	TX	R/W	Transmit Pin Control (TX) 00: Reset value; the canfdx_tx pin is controlled by the CAN FD core and is updated at the end of the CAN bit timing. 01: The sample point can be monitored on the canfdx_tx pin. 10: Drive dominant (0) level on the canfdx_tx pin. 11: Drive recessive (1) level on the canfdx_tx pin.
7	RX	R	Receive Pin Monitor the actual value of canfdx_rx pin 0: CAN bus is dominant (canfdx_rx = 0) 1: CAN bus is recessive (canfdx_rx = 1)
31:8	Reserved		

R = Read, P = Protected write; -n = value after reset

39.8.5 RAM Watchdog (RWD)

Offset address: 0x14

Reset value: 0x0000 0000

RAM watchdog monitors the ready output of the message RAM. When accessing the message RAM via the CAN FD message interface, the message RAM watchdog counter starts with the value configured in RWD[WDC]. When the message RAM indicates that the access has been successfully completed by activating its ready output signal, the counter reloads the value from RWD[WDC]. If the message RAM does not respond before the counter decrements to zero, the counter stops counting and the interrupt flag IR[WDI] is set. The RAM watchdog counter is clocked by the host clock (canfdx_pclk).

This register is writable only when the CCCR[CCE] and CCCR[INIT] bits are set.

Field	Name	R/W	Description
7:0	WDC	R/W	Watchdog Configuration Start value of Message RAM Watchdog Counter. When this value is “00”, the counter is disabled.
15:8	WDV	R	Watchdog Value Actual count value of Message RAM Watchdog
31:16	Reserved		

39.8.6 CC control register (CCCR)

Offset address: 0x18

Reset value: 0x0000 0001

Refer to the “Software Initialization” in this section for details.

Field	Name	R/W	Description
0	INIT	R/W	Initialization 0: Normal running; 1: Initialization start.

Field	Name	R/W	Description
			Note: Due to the synchronization mechanism between the two clock domains, there may be a delay before the value written to INIT can be read back. Therefore, the programmer must ensure that the previously written value to INIT has been accepted by reading INIT before setting a new value to INIT.
1	CCE	R/W	Configuration Change Enable 0: The CPU has no write access to protected configuration registers; 1: The CPU has write access to protected configuration registers (when CCCR[INIT] = 1).
2	ASM	R/W	Restricted Operation Mode The host can set the ASM bit only when both CCE and INIT are set to “1”. This bit can be reset by the host at any time. 0: Normal CAN operation; 1: Restricted operation mode activated.
3	CSA	R	Clock Stop Acknowledge 0: No clock stop acknowledge; 1: CAN FD can set this by stopping can_fd_pclk and can_fd_cclk.
4	CSR	R/W	Clock Stop Request 0: No clock stop request; 1: Clock stop requested. When there is a clock stop request, after all pending transmission requests are completed and the CAN bus becomes idle, first INIT is set, and then CSA is set.
5	MON	R/W	Bus Monitoring Mode The host CAN FD set the MON bit only when both CCE and INIT are set to “1”. This bit can be reset by the host at any time. 0: Bus Monitoring Mode disabled 1: Bus Monitoring Mode enabled
6	DAR	R/W	Disable Automatic Retransmission 0: Enable automatic retransmission of unsuccessfully transmitted messages; 1: Disable automatic retransmission.
7	TEST	R/W	Enable Test Mode 0: Normal operation, TEST register remains at its reset value; 1: Test mode, write access to the TEST register is enabled.
8	FDOE	R/W	Enable FD Operation 0: Disable 1: Enable
9	BRSE	R/W	Bit Rate Switch Enable 0: Disable transmission bit rate switching; 1: Enable transmission bit rate switching. Note: When CAN FD operation is disabled (FDOE = 0), BRSE is not considered.
10	Reserved		
11	WMM	R/W	Wide Message Marker (WMM)

Field	Name	R/W	Description
			Enables the 16-bit wide message marker. When the 16-bit wide message marker is used (WMM=1), the 16-bit internal timestamp function of the Tx Event FIFO is disabled. 0: Use 8-bit message marker 1: Use 16-bit message marker, replacing the 16-bit timestamp in the Tx Event FIFO
12	PXHD	R/W	Protocol Exception Handling Disable 0: Enable protocol exception handling 1: Disable protocol exception handling Note: When protocol exception handling is disabled, the CAN FD will transmit an error frame upon detecting a protocol exception condition.
13	EFBI	R/W	Edge Filtering during Bus Integration 0: Disable edge filtering 1: Two consecutive dominant tq are required to detect a hard synchronization edge
14	TXP	R/W	Transmit Pause If this bit is set, the CAN FD pauses for two CAN bits before starting the next transmission after successfully transmitting a frame. 0: Disable 1: Enable
15	NISO	R/W	Non ISO Operation If this bit is set, the CAN FD uses the CAN FD frame format specification V1.0 defined by Bosch CAN FD. 0: CAN FD frame format, compliant with ISO 11898-1:2015 1: CAN FD frame format, compliant with Bosch CAN FD specification V1.0
31:16	Reserved		

39.8.7 Nominal Bit Timing and Prescaler Register (NBTP)

Offset address: 0x1C

Reset value: 0x0600 0A03

This register is writable only after setting CCCR[CCE] and CCCR[INIT]. The CAN bit time can be programmed in the range of 4 to 385 time quanta (tq). The CAN time quantum can be programmed in the range of 1 to 512 can_cclk cycles. Tq = (NBRP + 1) mtq.

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2. Therefore, the bit time length is (Programmed value): [NTSEG1 + NTSEG2 + 3] tq or (Functional value): [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.

An Information Processing Time (IPT) is zero, meaning that the next bit data is available at the first clock edge after the sampling point.

Field	Name	R/W	Description
6:0	NTSEG2	R/W	Nominal Time Period after Sampling Point

Field	Name	R/W	Description
			Value of 0x01-0x7F: 1 to 127. The hardware interprets this value as one more than the programmed value. Note: When the CAN clock (canfdx bit clk) is 8 MHz, the reset value 0x0600 0A03 configures the CAN FD bit rate to 500 kbit/s.
7	Reserved		
15:8	NTSEG1	R/W	Nominal Time Period before Sampling Point Value of 0x01-0xFF: 1 to 255. The hardware interprets this value as one more than the programmed value.
24:16	NBRP	R/W	Nominal Bit Rate Prescaler 0x00-0x1FF is the divider value used to generate the bit time quantum oscillator frequency. The bit time is composed of multiples of this quantum. The effective value of bit ranges from 0 to 511. The hardware interprets this value as one more than the programmed value.
31:25	NSJW	R/W	Nominal (Re)Synchronization Jump Width Range of 0x00-0x7F: 0 to 127. The hardware interprets this value as one more than the programmed value.

39.8.8 Timestamp Counter Configuration (TSCC)

Offset address: 0x20

Reset value: 0x0000 0000

The internal 16-bit timestamp counter is configured. It handles internal and external timestamps.

Field	Name	R/W	Description
1:0	TSS	R/W	Timestamp Select 00: The timestamp counter value is always 0x0000. 01: The timestamp counter value increments according to TCP. 10: Use external timestamp counter value. 11: Same as “00”.
15:2	Reserved		
19:16	TCP	R/W	Timestamp Counter Prescaler 0x0-0xF: Configures the timestamp and timeout counter in units which are multiples of the CAN bit time [1…16]. The hardware interprets this value as one more than the programmed value. Note: Using CAN FD requires an external counter to generate the timestamp (TSS=10).
31:20	Reserved		

39.8.9 Timestamp Counter (TSCV)

Offset address: 0x24

Reset value: 0x0000 0000

Field	Name	R/W	Description
15:0	TSC	RC	Time Stamp Counter At the start of a frame (including both reception and transmission), the value of the internal/external timestamp counter is captured.

Field	Name	R/W	Description
			When TSCC[TSS]=01, the timestamp counter increments in multiples of the CAN bit time [1…16] as configured by TSCC[TCP]. A counter overflow sets the interrupt flag IR[TSW]. A write access to it resets the counter to zero. When TSCC[TSS]=10, TSC reflects the value of the external timestamp counter, and a write access has no effect. Note: “Overflow” refers to the timestamp counter’s value changing from a non-zero value to zero, where this change is not caused by a write access to TSCV. A write operation to this register resets the timestamp counter.
31:16	Reserved		

39.8.10 Timeout Counter Configuration (TOCC)

Offset address: 0x28

Reset value: 0xFFFF 0000

Field	Name	R/W	Description
0	ETOC	R/W	Enable Timeout Counter 0: Disable timeout counter 1: Enable timeout counter Note: For the information on the timeout function in CAN FD, refer to the “Timeout Counter” section.
2:1	TOS	R/W	Timeout Select When operating in continuous mode, a write to TOCV presets the counter to the value configured by TOCC[TOP] and the countdown continues. When the timeout counter is controlled by a FIFO, an empty FIFO presets the counter to the value configured by TOCC[TOP]. Decrementing begins when the first FIFO element is stored. 00: Continuous operation 01: Timeout controlled by Tx Event FIFO 10: Timeout controlled by Rx FIFO 0 11: Timeout controlled by Rx FIFO 1
15:3	Reserved		
31:16	TOP	R/W	Timeout Period The start value for the timeout counter (a decrementing counter), used to configure the timeout period.

39.8.11 Timeout Counter Value (TOCV)

Offset address: 0x2C

Reset value: 0x0000 FFFF

Field	Name	R/W	Description
15:0	TOC	RC	Timeout Counter The timeout counter decrements in multiples of the CAN bit time [1…16] as configured by TSCC[TCP]. When it decrements to zero, it sets the interrupt flag IR[TOO], and the timeout counter stops. The start and reset/restart conditions are configured via TOCC[TOS]. Writing to this register will reset the timeout counter.

Field	Name	R/W	Description
31:16	Reserved		

39.8.12 Error Counter Register (ECR)

Offset address: 0x40

Reset value: 0x0000 0000

Field	Name	R/W	Description
7:0	TEC	R	Transmit Error Counter Indicates the current state of the transmit error counter. The value ranges from 0 to 255. Note: When CCCR[ASM] is set, the CAN protocol controller does not increment the TEC and REC upon detection of a CAN protocol error, but the CEL is still incremented.
14:8	REC	R	Receive Error Counter Indicates the current state of the receive error counter. The value ranges from 0 to 127.
15	RP	R	Receive Error Passive 0: The Receive Error Counter is below the error passive level of 128. 1: The Receive Error Counter has reached the error passive level of 128.
23:16	CEL	RC	CAN Error Logging This counter is incremented whenever a CAN protocol error causes the 8-bit Transmit Error Counter (TEC) or the 7-bit Receive Error Counter (REC) to be incremented. It is also incremented when the bus-off limit is reached. The counter is not incremented when only the RP bit is set without a change in the REC. The increment of the CEL occurs after the REC or TEC is incremented. A read access to the CEL resets this counter. The counter stops upon reaching 0xFF; the next increment of the TEC or REC sets the interrupt flag IR[ELO]. A read access to this register resets the CEL to zero.
31:24	Reserved		

39.8.13 Protocol Status Register (PSR)

Offset address: 0x44

Reset value: 0x0000 0707

Field	Name	R/W	Description
2:0	LEC	R/S	Last Error Code LED indicates the type of the last error on the CAN bus. This field is cleared to “0” when a message is transferred (received or transmitted) without error. 0 (No error): No error has occurred since the LEC was last reset by a successful message reception or transmission. 1 (Stuff Error): More than 5 consecutive bits of the same level were detected in a part of the received message, which is not allowed. 2 (Format Error): A fixed-format part of the received frame had the wrong format.

Field	Name	R/W	Description
			3 (AckError): CAN FD-transmitted message was not acknowledged by another node. 4 (Bit 1 Error): During message transmission (except in the arbitration field), the device attempted to send a recessive level (logic “1”), but the monitored bus value was dominant; 5 (Bit 0 Error): During message transmission (or in the Ack slot, active error flag, or overload flag), the device attempted to send a dominant level (logic “0” for data or identifier), but the monitored bus value was recessive. During bus-off recovery, this status is set each time a sequence of 11 consecutive recessive bits is monitored. This allows the CPU to monitor the progress of the bus-off recovery sequence (indicating the bus is not stuck dominant or persistently disturbed). 6 (CRC Error): The CRC checksum of the received message was incorrect. The CRC of the incoming message did not match the CRC calculated from the received data. 7 (No Change): Any read access to this register re-initializes the LEC to “7”. When the LEC shows a value of “7”, it means no new CAN bus event has been detected since the last read access to the Protocol Status Register by the CPU. Note: When a CAN FD format frame with the BRS flag set enters the data phase, the next CAN event (error or valid frame) is displayed in the DLEC other than in the LEC. Errors in the fixed bit-stuff bits of the CAN FD CRC sequence are indicated as format errors other than stuff errors. Note: The bus-off recovery sequence (see ISO 11898-1:2015) cannot be shortened by setting or resetting CCCR[INIT]. If a device enters the bus-off state, it sets CCCR[INIT] autonomously, stopping all bus activity. Once the CPU clears CCCR[INIT], the device waits for 129 occurrences of bus idle (129 sequences of 11 consecutive recessive bits) before returning to normal operation. The error management counters are reset at the end of the bus-off recovery sequence. During the waiting period after CCCR[INIT] is reset, the Bit 0 Error code is written to PSR[LEC] each time a sequence of 11 consecutive recessive bits is monitored. This allows the CPU to easily check if the CAN bus is stuck dominant or persistently disturbed, and to monitor the bus-off recovery sequence. The ECR[REC] is used to count these sequences. A read access to this register sets the LEC to “111”.
4:3	ACT	R	Activity Status Monitors the CAN communication status of the module. 00: Synchronizing on a CAN node 01: Idle node, neither transmitting nor receiving 10: Receiving node, operating as a receiver 11: Transmitting node, operating as a transmitter Note: A protocol exception event sets ACT to 00.
5	EP	R	Error Passive Flag 0: CAN FD is in Error_Active state. Normally participates in bus communication and transmits an Active Error Flag upon detecting an error; 1: CAN FD is in Error_Passive state.

Field	Name	R/W	Description
6	EW	R	Warning Status 0: Both error counters are below the Error_Warning limit of 96; 1: At least one error counter has reached or exceeded the Error_Warning limit of 96.
7	BO	R	Bus_Off Status 0: CAN FD is not in Bus_Off state 1: CAN FD has entered Bus_Off state
10:8	DLEC	S	Data Phase Last Error Code Indicates the type of the last error that occurred in the data phase of a CAN FD format frame with the BRS flag set. The encoding is identical to the LEC. This field is cleared when a CAN FD format frame with the BRS flag set is transferred (received or transmitted) without error. A read access to this register sets the DLEC to “111”.
11	RESI	X	ESI flag of last received CAN FD Message This bit is set together with RFDF, independent of acceptance filtering. 0: No ESI flag set for the last received CAN FD message 1: ESI flag set for the last received CAN FD message. A read access to this register resets RESI.
12	RBRS	X	BRS flag of last received CAN FD Message This bit is set together with RFDF, independent of acceptance filtering. 0: No BRS flag set for the last received CAN FD message 1: BRS flag set for the last received CAN FD message A read access to this register resets RBRS
13	RFDF	X	Received CAN FD Message This bit is set independent of acceptance filtering. 0: No CAN FD message received since the CPU reset this bit 1: A CAN FD format message with the FDF flag set received A read access to this register resets RFDF
14	PXE	X	Protocol Exception Event 0: No protocol exception event occurring since the last read access 1: A protocol exception event occurred A read access to this register resets PXE
15	Reserved		
22:16	TDCV	R	Transceiver Delay Compensation Value Range: 0x00 – 0x7F. Defines the position of the secondary sampling point (SSP). It is defined by the sum of the measured delay from canfdx_tx to canfdx_rx and TDCR[TDCO]. During the data phase, the SSP position is the number of minimal time quanta (tq) from the start of the transmit bit to the secondary sampling point. Valid values are 0 to 127 mtq.
31:23	Reserved		

39.8.14 Transmitter Delay Compensation Register (TDCR)

Offset address: 0x48

Reset value: 0x0000 0000

Field	Name	R/W	Description
6:0	TDCF	R	Transmitter Delay Compensation Filter Window Length Range: 0x00-0x7F; Defines the minimum value for the secondary sampling point position. Dominant edges on canfdx_rx that would cause the SSP position to occur earlier are ignored for transmitter delay measurement. This feature is enabled when TDCF is configured to a value greater than TDCO. Valid values are 0 to 127 mtq.
7	Reserved		
14:8	TDCO	R	Transmitter Delay Compensation SSP Offset Range: 0x00-0x7F. This offset value defines the distance between the measured delay from can_fd_tx to can_fd_rx and the secondary sampling point. Valid values are 0 to 127 mtq.
31:15	Reserved		

39.8.15 Interrupt Register (IR)

Offset address: 0x50

Reset value: 0x0000 0000

When one of the conditions listed below is detected (a valid edge), the corresponding flag bit is set. These flag bits remain set until they are cleared by the host. A flag bit is cleared by writing “1” to its bit position. Writing “0”’ cannot clear flag bits. A hardware reset clears this register. The configuration of IE (Interrupt Enable) controls whether an interrupt is generated. The configuration of ILS (Interrupt Line Select) controls on which interrupt line the signal is asserted.

Field	Name	R/W	Description
0	RF0N	R/W	Rx FIFO 0 New Message 0: No new message written to Rx FIFO 0 1: A new message written to Rx FIFO 0
1	RF0W	R/W	Rx FIFO 0 Watermark Reached 0: The fill level of Rx FIFO 0 is below the watermark 1: The fill level of Rx FIFO 0 has reached or exceeded the watermark
2	RF0F	R/W	Rx FIFO 0 Full 0: Rx FIFO 0 not full 1: Rx FIFO 0 full
3	RF0L	R/W	Rx FIFO 0 Message Lost 0: No Rx FIFO 0 message lost 1: An Rx FIFO 0 message lost. This bit is also set after attempting to write to Rx FIFO 0 when its size is zero.
4	RF1N	R/W	Rx FIFO 1 New Message 0: No new message written to Rx FIFO 1 1: A new message written to Rx FIFO 1.
5	RF1W	R/W	Rx FIFO 1 Watermark Reached 0: The fill level of Rx FIFO 1 is below the watermark 1: The fill level of Rx FIFO 1 has reached or exceeded the watermark.
6	RF1F	R/W	Rx FIFO 1 Full 0: Rx FIFO 1 not full 1: Rx FIFO 1 full

Field	Name	R/W	Description
7	RF1L	R/W	Rx FIFO 1 Message Lost 0: No Rx FIFO 1 message lost 1: An Rx FIFO 1 message lost. This bit is also set after attempting to write to Rx FIFO 1 when its size is zero.
8	HPM	R/W	High Priority Message 0: No high-priority message received 1: A high-priority message received
9	TC	R/W	Transmission Completed 0: No transmission completed 1: Transmission completed
10	TCF	R/W	Transmission cancellation Finished 0: No transmission cancellation completed 1: Transmission cancellation completed
11	TFE	R/W	Tx FIFO Empty 0: The Tx FIFO is not empty 1: The Tx FIFO is empty
12	TEFN	R/W	Tx Event FIFO New Entry 0: No change in the Tx Event FIFO 1: The Tx Handler has written a Tx Event FIFO element
13	TEFW	R/W	Tx Event FIFO Watermark Reached 0: The fill level of the Tx Event FIFO is below the watermark 1: The fill level of the Tx Event FIFO has reached or exceeded the watermark
14	TEFF	R/W	Tx Event FIFO Full 0: The Tx Event FIFO is not full 1: The Tx Event FIFO is full
15	TEFL	R/W	Tx Event FIFO Element Lost 0: No Tx Event FIFO element lost 1: A Tx Event FIFO element lost. This bit is also set after attempting to write to the Tx Event FIFO when its size is zero.
16	TSW	R/W	Timestamp Wraparound 0: The Timestamp Counter has not wrapped around 1: The Timestamp Counter has wrapped around
17	MRAF	R/W	Message RAM Access Failure This flag is set when the Receive Handler: Fails to complete the acceptance filtering or storage of a received message before the arbitration field of the next message is received. In this case, acceptance filtering or message storage is aborted, and the Rx Handler starts processing the next message. Is unable to write a message to the Message RAM. In this case, message storage is aborted. In both cases above, the FIFO write index is not updated, and the new data flag of a dedicated receive buffer is not set. The partially stored message will be overwritten when the next message is stored to this location. This flag bit is also set when the Tx Handler fails to read a message from the Message RAM in time. In this case, the message transmission is aborted. If a Tx Handler access failure occurs, the CAN FD module switches to Restricted Operation Mode. To exit Restricted Operation Mode, the host CPU must reset CCCR[ASM].

Field	Name	R/W	Description
			0: No Message RAM access failure 1: A Message RAM access failure
18	TOO	R/W	Timeout Occurred 0: No timeout 1: The timeout period has been reached
19	DRX	R/W	Message Stored to Dedicated Rx Buffer This flag is set whenever a received message is stored to a Dedicated Receive Buffer. 0: No receive buffer updated 1: At least one received message has been stored to a receive buffer.
21:20	Reserved		
22	ELO	R/W	Error Logging Overflow 0: The CAN error logging counter has not overflowed 1: The CAN error logging counter has overflowed
23	EP	R/W	Error Passive Status 0: No change in the error-passive status 1: Change in the error-passive status
24	EW	R/W	Warning Status 0: No change in the error-warning status 1: Change in error-warning status
25	BO	R/W	Bus_Off Status 0: No change in Bus-Off status 1: Change in the Bus-Off status
26	WDI	R/W	Watchdog Interrupt 0: No Message RAM Watchdog event 1: A Message RAM Watchdog event has occurred due to a missing READY signal
27	PEA	R/W	Protocol Error in Arbitration Phase (Nominal bit time is used) 0: No protocol error has occurred in the Arbitration Phase 1: A protocol error was detected in the Arbitration Phase (PSR[LEC]≠0, 7)
28	PED	R/W	Protocol Error in Data Phase (Data Bit Time is used) 0: No protocol error has occurred in the Data Phase 1: A protocol error was detected in the Data Phase (PSR[DLEC]≠0, 7)
29	ARA	R/W	Access to Reserved Address 0: No access to a reserved address 1: An access to a reserved address
31:30	Reserved		

39.8.16 Interrupt Enable (IE)

Offset address: 0x54

Reset value: 0x0000 0000

The settings in the Interrupt Enable Register determine which status changes in the Interrupt Register will signal an event via the interrupt lines.

0: Disable interrupt

1: Enable interrupt

Field	Name	R/W	Description
0	RF0NE	R/W	Rx FIFO 0 New Message Interrupt Enable

Field	Name	R/W	Description
1	RF0WE	R/W	Rx FIFO 0 Watermark Reached Interrupt Enable
2	RF0FE	R/W	RX FIFO 0 Full Interrupt Enable
3	RF0LE	R/W	Rx FIFO 0 Message Lost Interrupt Enable
4	RF1NE	R/W	Rx FIFO 1 New Message Interrupt Enable
5	RF1WE	R/W	Rx FIFO 1 Watermark Reached Interrupt Enable
6	RF1FE	R/W	RX FIFO 1 Full Interrupt Enable
7	RF1LE	R/W	Rx FIFO 1 Message Lost Interrupt Enable
8	HPME	R/W	High Priority Message Interrupt Enable
9	TCE	R/W	Transmission Completed Interrupt Enable
10	TCFE	R/W	Transmission cancellation Finished Interrupt Enable
11	TFEE	R/W	TX FIFO Empty Interrupt Enable
12	TEFNE	R/W	Tx Event FIFO New Entry Interrupt Enable
13	TEFWE	R/W	Tx FIFO Watermark Reached Interrupt Enable
14	TEFFE	R/W	TX Event FIFO Full Interrupt Enable
15	TEFLE	R/W	Tx Event FIFO Element Lost Interrupt Enable
16	TSWE	R/W	Timestamp Wraparound Interrupt Enable
17	MRAFE	R/W	Message RAM Access Failure Interrupt Enable
18	TOOE	R/W	Timeout Occurred Interrupt Enable
19	DRXE	R/W	Message Stored to Dedicated Rx Buffer Interrupt Enable
21:20	Reserved		
22	ELOE	R/W	Error Logging Overflow Interrupt Enable
23	EPE	R/W	Error Passive Interrupt Enable
24	EWE	R/W	Warning Status Interrupt Enable
25	BOE	R/W	Bus_Off Status Interrupt Enable
26	WDIE	R/W	Watchdog Interrupt Enable
27	PEAE	R/W	Protocol Error in Arbitration Phase Enable
28	PEDE	R/W	Protocol Error in Data Phase Enable
29	ARAE	R/W	Access to Reserved Address Enable
31:30	Reserved		

39.8.17 Interrupt Line Select

Offset address: 0x58

Reset value: 0xXXXX XXXX

The Interrupt Line Select register assigns interrupts generated by specific interrupt flags in the interrupt register to one of the two module interrupt lines. To

generate an interrupt, the corresponding interrupt line must be enabled via ILE[EINT0] and ILE[EINT1].

0: Interrupt applied to Interrupt Line CANFDA_INT0

1: Interrupt applied to Interrupt Line CANFDA_INT1

Field	Name	R/W	Description
0	RF0NL	R/W	Rx FIFO 0 New Message Interrupt Line Select
1	RF0WL	R/W	Rx FIFO 0 Watermark Reached Interrupt Line Select
2	RF0FL	R/W	RX FIFO 0 Full Interrupt Line Select
3	RF0LL	R/W	Rx FIFO 0 Message Lost Interrupt Line Select
4	RF1NL	R/W	Rx FIFO 1 New Message Interrupt Line Select
5	RF1WL	R/W	Rx FIFO 1 Watermark Reached Interrupt Line Select
6	RF1FL	R/W	RX FIFO 1 Full Interrupt Line Select
7	RF1LL	R/W	Rx FIFO 1 Message Lost Interrupt Line Select
8	HPML	R/W	High Priority Message Interrupt Line Select
9	TCL	R/W	Transmission Completed Interrupt Line Select
10	TCFL	R/W	Transmission cancellation Finished Interrupt Line Select
11	TFEL	R/W	TX FIFO Empty Interrupt Line Select
12	TEFNL	R/W	Tx Event FIFO New Entry Interrupt Line Select
13	TEFWL	R/W	Tx Event FIFO Watermark Reached Interrupt Line Select
14	TEFFL	R/W	Tx Event FIFO Full Interrupt Line Select
15	TEFLL	R/W	Tx Event FIFO Element Lost Interrupt Line Select
16	TSWL	R/W	Timestamp Wraparound Interrupt Line Select
17	MRAFL	R/W	Message RAM Access Failure Interrupt Line Select
18	TOOL	R/W	Timeout Occurred Interrupt Line Select
19	DRXL	R/W	Message Stored to Dedicated Rx Buffer Interrupt Line Select
21:20	Reserved		
22	ELOL	R/W	Error Logging Overflow Interrupt Line Select
23	EPL	R/W	Error Passive Status Interrupt Line Select
24	EWL	R/W	Warning Status Interrupt Line Select
25	BOL	R/W	Bus_Off Status Interrupt Line Select
26	WDIL	R/W	Watchdog Interrupt Line Select
27	PEAL	R/W	Protocol Error Interrupt Line Select in Arbitration Phase
28	PEDL	R/W	Protocol Error Interrupt Line Select in Data Phase
29	ARAL	R/W	Access to Reserved Address Interrupt Line Select

Field	Name	R/W	Description
31:30	Reserved		

39.8.18 Interrupt Line Enable

Offset address: 0x5C

Reset value: 0x0000 0000

The two interrupt lines connected to the CPU can be individually enabled/disabled by programming EINT0 and EINT1 bits, respectively.

Field	Name	R/W	Description
0	EINT0	R/W	Enable Interrupt Line 0 0: Disable Interrupt Line CANFDA_INT0 1: Enable Interrupt Line CANFDA_INT0
1	EINT1	R/W	Enable Interrupt Line 1 0: Disable Interrupt Line CANFDA_INT1 1: Enable Interrupt Line CANFDA_INT1
31:2	Reserved		

39.8.19 Global Filter Configuration (GFC)

Offset address: 0x80

Reset value: 0x0000 0000

This is the global setting for message ID filtering.

Field	Name	R/W	Description
0	RRFE	R	Reject Extended Remote Frames 0: Filters remote frames with 29-bit extended IDs 1: Rejects all remote frames with 29-bit extended IDs.
1	RRFS	R	Reject Standard Remote Frames 0: Filters remote frames with 11-bit standard IDs 1: Rejects all remote frames with 11-bit standard IDs.
3:2	ANFE	R	Accept Non-matching Frames Extended Defines how a received message with a 29-bit ID that does not match any element in the filter list is handled. 00: Accept in Rx FIFO 0 01: Accept in Rx FIFO 1 10: Reject 11: Reject
5:4	ANFS	R	Accept Non-matching Standard Frames Defines how a received message with a 11-bit ID that does not match any element in the filter list is handled. 00: Accept in Rx FIFO 0 01: Accept in Rx FIFO 1 10: Reject 11: Reject
31:6	Reserved		

39.8.20 Standard ID Filter Configuration (SIDFC)

Offset address: 0x84

Reset value: 0x0000 0000

This is the configuration for filtering 11-bit standard message IDs. The standard ID filter element configuration controls the filtering path for standard messages.

Field	Name	R/W	Description
1:0	Reserved		
15:2	FLSSA	R	Standard Filter List Start Address The start address (32-bit word address) of the standard message ID Filter List in the Message RAM.
23:16	LSS	R	Standard List Size 0: No Standard Message ID Filters 1 to 128: Number of Standard Message ID Filter Elements Values greater than 128 are interpreted as 128.
31:24	Reserved		

39.8.21 Extended ID Filter Configuration (XIDFC)

Offset address: 0x88

Reset value: 0x0000 0000

This is the configuration for filtering 29-bit extended message IDs. The extended ID filter element configuration controls the filtering path for standard messages.

Field	Name	R/W	Description
1:0	Reserved		
15:2	FLESA	R	Extended Filter List Start Address The start address (32-bit word address) of the extended message ID Filter List in the Message RAM.
22:16	LSE	R	Extended List Size 0: No Extended Message ID Filters 1 to 64: Number of Extended Message ID Filter Elements Values greater than 64 are interpreted as 64.
31:23	Reserved		

39.8.22 Extended ID and Mask (XIDAM)

Offset address: 0x90

Reset value: 0x1FFF FFFF

Field	Name	R/W	Description
28:0	EIDM	R	Extended ID Mask Acceptance filtering for extended frames: The AND 896peration will be implemented between the extended ID and the Message ID of the received frames. This is used to mask the 29-bit ID in protocols like SAE J1939. When the reset values of all bits are “1”, the mask is invalid.
31:29	Reserved		

39.8.23 High Priority Message Status (HPMS)

Offset address: 0x94

Reset value: 0x0000 0000

This register is updated whenever a Message ID Filter element, configured to generate a priority event, finds a match. This can be used to monitor the status of incoming high-priority messages and enables fast access to them.

Field	Name	R/W	Description
5:0	BIDX	R	Buffer Index The index of the Rx FIFO element where the message is stored. This is only valid if MSI[1] = 1.
7:6	MSI	R	Message Storage Indicator 00: No FIFO selected 01: FIFO message lost 10: Message stored in FIFO 0 11: Message stored in FIFO 1
14:8	FIDX	R	Filter Index The index of the matching filter element. The range is from 0 to SIDFC[LSS]-1 or XIDFC[LSE]-1.
15	FLST	R	Filter List Indicates the list of filter matching filter elements. 0: List of Standard Filters 1: List of Extended Filters.
31:16	Reserved		

39.8.24 New Data 1 (NDAT1)

Offset address: 0x98

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	NDy	R/W	New Data (y=0…31) This register holds the new data flags for Rx Buffers 0 to 31. The flags are set when the corresponding Rx Buffer is updated by a received frame. These flag bits remain set until they are cleared by the host. A flag bit is cleared by writing “1” to its bit position. Writing “0”’ cannot clear flag bits. A hardware reset clears this register. 0: Rx Buffer not updated 1: Rx Buffer updated with a new message

39.8.25 New Data 2 (NDAT2)

Offset address: 0x9C

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	ND(y+32)	R/W	New Data (y=0…31) This register holds the new data flags for Rx Buffers 32 to 63. The flags are set when the corresponding Rx Buffer is updated by a received frame. These flag bits remain set until they are cleared by the host. A flag bit is cleared by writing “1” to its bit position. Writing “0”’ cannot clear flag bits. A hardware reset clears this register. 0: Rx Buffer not updated 1: Rx Buffer updated with a new message

39.8.26 Rx FIFO 0 Configuration (RXF0C)

Offset address: 0xA0

Reset value: 0x0000 0000

Field	Name	R/W	Description
1:0	Reserved		

Field	Name	R/W	Description
15:2	F0SA	R	Rx FIFO 0 Start Address The start address of Rx FIFO 0 in Message RAM (32-bit word address)
22:16	F0S	R	Rx FIFO 0 Size 0: No Rx FIFO 0 1 to 64: Number of Rx FIFO 0 elements Values greater than 64 are interpreted as 64 Index of Rx FIFO 0 elements ranges from 0 to F0S -1
23	Reserved		
30:24	F0WM	R	Rx FIFO 0 Watermark 0: Disable watermark interrupt 1~64: Level of Rx FIFO 0 watermark interrupt (IR[RF0W]) More than 64: Disable watermark interrupt
31	F0OM	R	FIFO 0 Operation Mode FIFO 0 CAN FD operate in Block Mode or Overwrite Mode. 0: FIFO 0 Block Mode 1: FIFO 0 Overwrite Mode

39.8.27 Rx FIFO 0 Status (RXF0S)

Offset address: 0xA4

Reset value: 0x0000 0000

Field	Name	R/W	Description
6:0	F0FL	R	Rx FIFO 0 Fill Level Number of elements stored in Rx FIFO 0, ranging from 0 to 64.
7	Reserved		
13:8	F0GI	R	Rx FIFO 0 Read Index Rx FIFO 0 read index pointer, ranging from 0 to 63.
15:14	Reserved		
21:16	F0PI	R	Rx FIFO 0 Write Index Rx FIFO 0 write index pointer, ranging from 0 to 63.
23:22	Reserved		
24	F0F	R	Rx FIFO 0 Full 0: Rx FIFO 0 not full 1: Rx FIFO 0 full
25	RF0L	R	Rx FIFO 0 Message Lost This bit is a copy of interrupt flag IR[RF0L]. It is reset when IR[RF0L] is reset. 0: No Rx FIFO 0 message lost 1: Rx FIFO 0 message lost. This bit is also set after attempting to write to Rx FIFO 0 when its fill level is zero. Note: When RXF0C[F0OM] = 1, overwriting the oldest message does not set this flag.
31:26	Reserved		

39.8.28 Rx FIFO 0 Acknowledge (RXF0A)

Offset address: 0xA8

Reset value: 0x0000 0000

Field	Name	R/W	Description
5:0	F0AI	R/W	Rx FIFO 0 Acknowledge Index After the host has read one message or a sequence of messages from Rx FIFO 0, it must write the Buffer Index of the last element read from Rx FIFO 0 to F0AI. This sets the Rx FIFO 0 read index RXF0S[F0GI] to F0AI + 1 and updates the FIFO 0 fill level RXF0S[F0FL].
31:6	Reserved		

39.8.29 Rx Buffer Configuration (RXBC)

Offset address: 0xAC

Reset value: 0x0000 0000

Field	Name	R/W	Description
1:0	Reserved		
15:2	RBSA	R	Rx Buffer Start Address Configures the start address (32-bit word address) of the Rx Buffer section in the Message RAM.
31:16	Reserved		

39.8.30 Rx FIFO 1 Configuration (RXF1C)

Offset address: 0xB0

Reset value: 0x0000 0000

Field	Name	R/W	Description
1:0	Reserved		
15:2	F1SA	R	Rx FIFO 1 Start Address The start address of Rx FIFO 1 in Message RAM (32-bit word address)
22:16	F1S	R	Rx FIFO 1 Size 0: No Rx FIFO 1 1 to 64: Number of Rx FIFO 1 elements Values greater than 64 are interpreted as 64 Index of Rx FIFO 1 elements ranges from 0 to F1S -1
23	Reserved		
30:24	F1WM	R	Rx FIFO 1 Watermark 0: Disable watermark interrupt 1~64: Level of Rx FIFO 1 watermark interrupt (IR[RF1W]) More than 64: Disable watermark interrupt
31	F1OM	R	FIFO 1 Operation Mode FIFO 1 CAN FD operate in Block Mode or Overwrite Mode 0: FIFO 1 in Block Mode 1: FIFO 1 in Overwrite Mode

39.8.31 Rx FIFO 1 Status (RXF1S)

Offset address: 0xB4

Reset value: 0x0000 0000

Field	Name	R/W	Description
6:0	F1FL	R	Rx FIFO 1 Fill Level Number of elements stored in Rx FIFO 1, ranging from 0 to 64.
7	Reserved		

Field	Name	R/W	Description
13:8	F1GI	R	Rx FIFO 1 Read Index Rx FIFO 1 read index pointer, ranging from 0 to 63.
15:14	Reserved		
21:16	F1PI	R	Rx FIFO 1 Write Index Rx FIFO 1 write index pointer, ranging from 0 to 63.
23:22	Reserved		
24	F1F	R	Rx FIFO 1 Full 0: Rx FIFO 1 not full 1: Rx FIFO 1 full Bit 21: 16 F1PI [5:0]: Rx FIFO 1 write index Rx FIFO 1 write index pointer, ranging from 0 to 63.
25	RF1L	R	Rx FIFO 1 Message Lost This bit is a copy of interrupt flag IR[RF1L]. It is reset when IR[RF1L] is reset. 0: No Rx FIFO 1 message lost 1: Rx FIFO 1 message lost. This bit is also set after attempting to write to Rx FIFO 1 when its fill level is zero. Note: When RXF1C[F1OM] = 1, overwriting the oldest message does not set this flag.
31:26	Reserved		

39.8.32 Rx FIFO 1 Acknowledge (RXF1A)

Offset address: 0xB8

Reset value: 0x0000 0000

Field	Name	R/W	Description
5:0	F1AI	R/W	Rx FIFO 1 Acknowledge Index After the host has read one message or a sequence of messages from Rx FIFO 1, it must write the Buffer Index of the last element read from Rx FIFO 1 to F1AI. This sets the Rx FIFO 1 read index RXF1S[F1GI] to F1AI + 1 and updates the FIFO 1 fill level RXF1S[F1FL].
31:6	Reserved		

39.8.33 Rx Buffer/ FIFO Element Size Configuration (RXESC)

Offset address: 0xBC

Reset value: 0x0000 0000

Configures the number of data bytes for elements belonging to an Rx Buffer / Rx FIFO. A data field size greater than 8 bytes is only used for CAN FD operation.

Field	Name	R/W	Description
2:0	F0DS	R	Rx FIFO 0 Data Field Size 000: 8-byte data field 001: 12-byte data field 0 10: 16-byte data field 011: 20-byte data field 100: 24-byte data field 101: 32-byte data field 110: 48-byte data field 111: 64-byte data field Note: If a received CAN frame has a data field size larger than the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of

Field	Name	R/W	Description
			bytes configured by RXESC is stored into the Rx Buffer or Rx FIFO element. The remaining data field of the frame is ignored.
3	Reserved		
6:4	F1DS	R	Rx FIFO 1 Data Field Size 000: 8-byte data field 001: 12-byte data field 010: 16-byte data field 011: 20-byte data field 100: 24-byte data field 101: 32-byte data field 110: 48-byte data field 111: 64-byte data field
7	Reserved		
10:8	RBDS	R	Rx Buffer Data Field Size 000: 8-byte data field 001: 12-byte data field 010: 16-byte data field 011: 20-byte data field 100: 24-byte data field 101: 32-byte data field 110: 48-byte data field 111: 64-byte data field
31:11	Reserved		

39.8.34 Tx Buffer Configuration (TXBC)

Offset address: 0xC0

Reset value: 0x0000 0000

Field	Name	R/W	Description
1:0	Reserved		
15:2	TBSA	R	Tx Buffers Start Address Start address of the Tx Buffer section in the Message RAM. Note: The sum of TFQS and NDTB may not be greater than 32. Erroneous configurations are not checked. The Tx Buffer section in the Message RAM starts with the dedicated Tx Buffers.
21:16	NDTB	R	Number of dedicated Tx buffers 0: No dedicated Tx buffers 1 to 32: Number of dedicated Tx buffers Values greater than 32 are interpreted as 32
23:22	Reserved		
29:24	TFQS	R	Transmit FIFO/Queue Size 0: No Tx FIFO/Queue 1 to 32: Number of Tx buffers for FIFO/Queue Values greater than 32 are interpreted as 32
30	TFQM	R	Tx FIFO/Queue Mode 0 : Tx FIFO mode 1 : Tx Queue Mode
31	Reserved		

39.8.35 Tx FIFO / Queue Status (TXFQS)

Offset address: 0xC4

Reset value: 0x0000 0000

The Tx FIFO/Queue status is related to the pending Tx requests listed in register TXBRP. Therefore, the effect of adding/canceling requests may be delayed due to an ongoing Tx scan (TXBRP has not been updated yet).

Field	Name	R/W	Description
5:0	TFFL	R	Tx FIFO Free Level The number of consecutive free Tx FIFO elements starting from TFGI, ranging from 0 to 32. Reads as 0 when configured to Tx queue operation mode (TXBC[TFQM]=1). Note: In configurations mixing dedicated Tx Buffers with a Tx FIFO or Tx Queue, the write and read indices indicate the Tx Buffer number starting from the first dedicated Tx Buffer. Example: For a configuration with 12 dedicated Tx Buffers and a Tx FIFO of 20 buffers, write index 15 points to the fourth buffer of the Tx FIFO.
7:6	Reserved		
12:8	TFGI	R	Tx FIFO Read Index Tx FIFO read index pointer, ranging from 0 to 31. Reads as 0 when configured to Tx queue operation mode (TXBC[TFQM]=1).
15:13	Reserved		
20:16	TFQPI	R	Tx FIFO/Queue Write Index Tx FIFO/Queue write index pointer, ranging from 0 to 31.
21	TFQF	R	Tx FIFO/Queue Full 0: Tx FIFO / Queue not full 1: Tx FIFO / Queue full
31:22	Reserved		

39.8.36 Tx Buffer Element Size Configuration (TXESC)

Offset address: 0xC8

Reset value: 0x0000 0000

Configures the number of data bytes in Tx buffer element. A data field size greater than 8 bytes is only used for CAN FD operation.

Field	Name	R/W	Description
2:0	TBDS	R	Tx Buffer Data Field Size 000: 8-byte data field 001: 12-byte data field 010: 16-byte data field 011: 20-byte data field 100: 24-byte data field 101: 32-byte data field 110: 48-byte data field 111: 64-byte data field Note: If the data length code (DLC) of a Tx Buffer element is configured with a value higher than the Tx Buffer data field size (TXESC[TBDS]), the undefined bytes of the Tx Buffer are transmitted as 0xCC (pad bytes).
31:3	Reserved		

39.8.37 Tx Buffer Request Pending (TXBRP)

Offset address: 0xCC

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	TRPy	R	Pending Request for Tx Buffer y (y=0…31) Each Tx Buffer has its own Transmission Request Pending bit. These bits are set via the TXBAR register. They are reset after the requested transmission is completed or canceled via the TXBCR register. TXBRP bits are only set for Tx Buffers configured via TXBC. After a TXBRP bit is set, a Tx scan is initiated to check for the pending Tx request with the highest priority (the Tx Buffer with the lowest Message ID). canceling a request resets the corresponding Transmission Request Pending bit in the TXBRP register. If the transmission has already started at the time of the cancellation request, the reset occurs at the end of the transmission, regardless of whether it was successful or not. The cancellation Request bit is reset immediately after the corresponding TXBRP bit is reset. The completion of the cancellation is signaled via TXBCF after the cancellation request: Upon successful transmission, along with the corresponding TXBTO bit When the transmission has not yet started at the cancellation point When the transmission is aborted due to arbitration loss When an error occurs during frame transmission In DAR mode, all transmissions are automatically canceled if they are unsuccessful. For all unsuccessful transmissions, the corresponding TXBCF bit is set. 0: No transmission request pending 1: Transmission request pending Note: TXBRP bits set during an ongoing Tx scan are not considered for that specific Tx scan. If a cancellation is requested for such a Tx Buffer, this add request is canceled immediately, and the corresponding TXBRP bit is reset.

39.8.38 Tx Buffer Add Request (TXBAR)

Offset address: 0xD0

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	ARy	R/W	Add Request for Tx Buffer y (y=0…31) Each Tx Buffer has its own Add Request bit. Writing 1 sets the corresponding Add Request bit; writing 0 has no effect. This allows the host to set transmission requests for multiple Tx Buffers with a single write to TXBAR. TXBAR bits are only set for Tx Buffers configured via TXBC. These bits are reset immediately when no Tx scan is running; otherwise, they remain set until the Tx scan procedure is completed. 0: No transmission request added 1: Transmission request added Note: If an Add Request is applied to a Tx Buffer that already has a pending transmission request (its corresponding TXBRP bit is already set), this Add Request is ignored.

39.8.39 Tx Buffer cancellation Request (TXBCR)

Offset address: 0xD4

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	CRy	R/W	cancellation Request for Tx Buffer y (y=0…31) Each Tx Buffer has its own cancellation Request bit. Writing 1 sets the corresponding cancellation Request bit; writing 0 has no effect. This allows the host to set cancellation requests for multiple Tx Buffers with a single write to TXBCR. TXBCR bits are only set for Tx Buffers configured via TXBC. These bits remain set until the corresponding bits in TXBRP are reset. 0: No cancellation pending 1: cancellation pending

39.8.40 Transmission Occurred for Tx Buffer (TXBTO)

Offset address: 0xD8

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	TOy	R/W	Transmission Occurred for Tx Buffer y (y=0…31) Each Tx Buffer has its corresponding “Transmission Occurred” bit. These bits are set when the corresponding TXBRP bit is cleared after a successful transmission. They are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit in the TXBAR register. 0: No transmission 1: Transmission occurred

39.8.41 Tx Buffer cancellation Finished (TXBCF)

Offset address: 0xDC

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	CFy	R/W	cancellation Finished for Tx Buffer y (y=0…31) Each Tx Buffer has its own cancellation Finished bit. These bits are set when the corresponding TXBRP bit is cleared after a cancellation was requested via TXBCR. If the corresponding TXBRP bit was not set at the time of cancellation, the CF bit is set immediately. They are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit in the TXBAR register. 0: Tx Buffer cancellation not finished 1: Tx Buffer cancellation finished

39.8.42 Tx Buffer Transmission Interrupt Enable (TXBTIE)

Offset address: 0xE0

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	TIEy	R/W	Transmission Interrupt Enable for Tx Buffer y (y=0…31) Each Tx Buffer has its corresponding Transmission Interrupt Enable bit. 0: Disable 1: Enable

39.8.43 Tx Buffer cancellation Finished Interrupt Enable (TXBCIE)

Offset address: 0xE4

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	CFIEy	R/W	cancellation Finished Interrupt Enable for Tx Buffer y (y=0…31) Each Tx Buffer has its corresponding cancellation Finished Interrupt Enable bit. 0: Disable 1: Enable

39.8.44 Tx Event FIFO Configuration (TXEFC)

Offset address: 0xF0

Reset value: 0x0000 0000

This register is writable only after CCCR[CCE] and CCCR[INIT] have been set.

Field	Name	R/W	Description
1:0	Reserved		
15:2	EFSA	R	Event FIFO Start Address The start address of Tx Event FIFO in Message RAM (32-bit word address)
21:16	EFS	R	Event FIFO Size 0: Disable Tx event FIFO 1 to 32: Number of Tx event FIFO elements Values greater than 32 are interpreted as 32 Index of Tx event FIFO elements ranges from 0 to EFS -1
23:22	Reserved		
29:24	EFWM	R	Event FIFO Watermark 0: Disable watermark interrupt 1~32: Level of Tx Even FIFO Watermark Interrupt (IR[TEFW]) More than 32: Disable watermark interrupt
31:30	Reserved		

39.8.45 Tx Event FIFO Status (TXEFS)

Offset address: 0xF4

Reset value: 0x0000 0000

Field	Name	R/W	Description
5:0	EFFL	R	Event FIFO Fill Level Number of elements stored in Tx Event FIFO, ranging from 0 to 32.
7:6	Reserved		
12:8	EFGI	R	Event FIFO Read Index Tx Event FIFO read index pointer, ranging from 0 to 31.
15:13	Reserved		
20:16	EFPI	R	Event FIFO Write Index Tx Event FIFO write index pointer, ranging from 0 to 31.
23:21	Reserved		
24	EFF	R	Event FIFO Full 0: The Tx Event FIFO is not full 1: The Tx Event FIFO is full
25	TEFL	R	Tx Event FIFO Element Lost This bit is a copy of interrupt flag IR[TEFL]. It is reset when IR[TEFL] is reset. 0: No Tx Event FIFO element lost 1: A Tx Event FIFO element lost. This bit is also set after attempting to write to the Tx Event FIFO when its size is zero.

Field	Name	R/W	Description
31:26	Reserved		

39.8.46 Tx Event FIFO Acknowledge (TXEFA)

Offset address: 0xF8

Reset value: 0x0000 0000

Field	Name	R/W	Description
4:0	EFAI	R/W	Event FIFO Acknowledge Index After the host has read one element or a sequence of elements from the Tx Event FIFO, it must write the index of the last element read from the Tx Event FIFO to EFAI. This sets the Tx Event FIFO read index TXEFS[EFGI] to EFAI + 1 and updates the Event FIFO fill level TXEFS[EFFL].
31:5	Reserved		

Note: The corresponding registers of CAN FD_ERM module are shown as follows.

39.8.47 ERM and Low-Power Control (ERM_CTRL)

Offset address: 0x100

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	ECC_EN	R/W	Enable SRAM ECC Check and Global Interrupt for SRAM ECC Check 0: Disable CAN FD SRAM ECC Check and Global ECC Interrupt 1: Enable CAN FD SRAM ECC Check and Global ECC Interrupt
1	BEC_INT_EN	R/W	Bit Error Correction Interrupt Enable 0: Disable 1: Enable Note: To ensure correct generation of interrupts, ensure ECC_EN = 1.
2	BEU_INT_EN	R/W	Bit Error Non-Correction Interrupt Enable 0: Disable 1: Enable Note: To ensure correct generation of interrupts, ensure ECC_EN = 1.
3	WAKE_INT_EN	R/W	CAN FD Wake-up Interrupt Enable 0: Disable 1: Enable
31:4	Reserved		

39.8.48 ERM and Low-Power Status (ERM_STS)

Offset address: 0x104

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	BEC_FLAG	R/W	Single-Bit Error Detected and Correctable 0: No error detected 1: Single-bit error detected

Field	Name	R/W	Description
1	BEU_FLAG	R/W	Multi-bit Error Detected and Uncorrectable 0: No error detected or only single-bit error detected 1: Multi-bit error detected
31:2	Reserved		

39.8.49 ERM Error Address Register (ERM_ECC_ADDR_LOG)

Offset address: 0x108

Reset value: 0x0000 0000

Field	Name	R/W	Description
8:0	ERR_ADDR	R/W	Error Location Address Note: This address is the corresponding CAN FD SRAM offset address shifted right by 2 bits.
31:9	Reserved		

39.8.50 ECC Error Data Register (ERM_ECC_DATA_LOG)

Offset address: 0x10C

Reset value: 0x0000 0000

Field	Name	R/W	Description
31:0	ERR_DATA	RO	Data in which an error occurred (Incorrect data occurred)

39.8.51 ECC RAM hardware initialisation (ERM_RAM_INIT)

Offset address: 0x110

Reset value: 0x0000 0000

Field	Name	R/W	Description
0	RAM_INIT	R/W	Enable the initialisation of the CAN FD mailbox RAM 0 Read: Initialisation complete or not initiated. Write: No operation 1 Read: Initialisation is in progress Write: Enable the initialisation of CAN FD mailbox RAM. Once initialisation is complete, this bit will automatically be cleared to 0.
1	RAM_INIT_DONE	RO	Message RAM initialisation status 0: Initialisation is in progress or has not yet started. 1: Initialisation complete
31:2	Reserved		

39.8.52 External Timestamp Control (ERM_EXTC_CTRL)

Offset address: 0x114

Reset value: 0x0000 0000

Field	Name	R/W	Description
9:0	ETCP	R/W	External timestamp pre-division value canfdx_pclk is divided by adding 1 to the pre-division value, and is used to provide a clock for the external timestamp counter.
30:10	Reserved		

Field	Name	R/W	Description
31	ETCE	R/W	External timestamp counter enabled 0: External timestamp counter disabled 1: External timestamp counter enabled

39.8.53 External timestamp value (ERM_EXTC_VAL)

Offset address: 0x118

Reset value: 0x0000 0000

Field	Name	R/W	Description
15:0	ETSCV	R/W	External timestamp counter A read operation returns the current value of the counter. Any write operation to this register will reset the counter to 0.
31:16	Reserved		

40 Local Interconnection Network (LIN)

40.1 Full Name and Abbreviation Description of Terms

Table 240 Full Name and Abbreviation Description of Terms

Full name in English	English abbreviation
Local Interconnect Network	LIN
Universal Asynchronous Transceiver	UART
Non-return to Zero	NRZ
Finite State Machine	FSM
UART TX Shift Register/ UART RX Shift Register	UARTTXSHF/ UARTRXSHF

40.2 Introduction

LIN can be used as a serial communication interface, and in this compatible mode, the LIN function is compatible with other independent serial port modules (UART), but the registers and codes are different. When the module is used as a serial port, it is in compatible mode.

Through configuration, this module can serve as UART or LIN, and the hardware characteristics are enhanced so that G32R502 can be compatible with the LIN function.

The module complies with the LIN2.1 protocol specified in the LIN specification package. The LIN standard is based on the UART serial data link format, and it is used for multicast transmission between network nodes.

40.3 Main characteristics of UART

(1) Supports full-duplex or half-duplex

(2) Standard UART communication, asynchronous communication mode (synchronous mode is not supported), standard NRZ format

(3) Multi-buffer receive and transmit unit, supporting double-buffer receive and transmit

(4) In the frame format, each character can be configured to contain 3-13 bits under the following conditions
- Stop bits are configurable to 1 or 2 bits
- Parity bits are configurable to zero or 1 bit (odd or even parity)
- Address-bit mode requires an additional address bit
- Data word length is configurable to 1 to 8 bits

(5) Two independently enabled interrupt lines, namely level 0 and level 1

(6) Support sleep mode, allowing the CPU to be freed, awakened, and receive messages during multiprocessor communication

(7) Two multiprocessor communication formats, allowing communication among multiple devices

(8) Support 2^{24} high-precision baud rates
- Maximum baud rate of 3.125 Mbps at a 100 MHz clock
- 24-bit programmable baud rate

(9) It can transmit and receive data through DMA

(10) Five error flags and seven status flags

(11) LINTX and LINRX are external pins; UART hardware control is not supported (but can be implemented via software)

40.4 Main characteristics of LIN

(1) Conform to LIN1.3, 2.0, and 2.1 protocols

(2) Multi-buffering receiving and transmitting units

(3) The main header can be automatically generated, including the following fields
- Synchronization field
- Programmable synchronization interrupt field
- Identifier field

(4) Identification mask, used to filter messages

(5) 2 priority encoded interrupt lines, used for receiving and transmitting ID, errors, and statuses

(6) It supports automatic wake-up, and can generate wake-up signals, and configure timeout time

(7) It supports automatic bus idle detection, and LIN2.0 checksum

(8) It supports detection of errors, including:
- Parity check error
- Synchronization field error
- Checksum error
- No-response error
- Bus error
- Bit error

(9) It supports 231 programmable transfer rate (7 decimal places)

(10) The highest baud rate is 20kpbs

(11) It can transmit and receive data through DMA

(12) LINTX and LINRX are used as external pins, and the LINRX master level is awakened by the transceiver

(13) Slave synchronization functions are as follows:
- Synchronous verification
- Synchronization interrupt detection
- Update baud rate (optional)

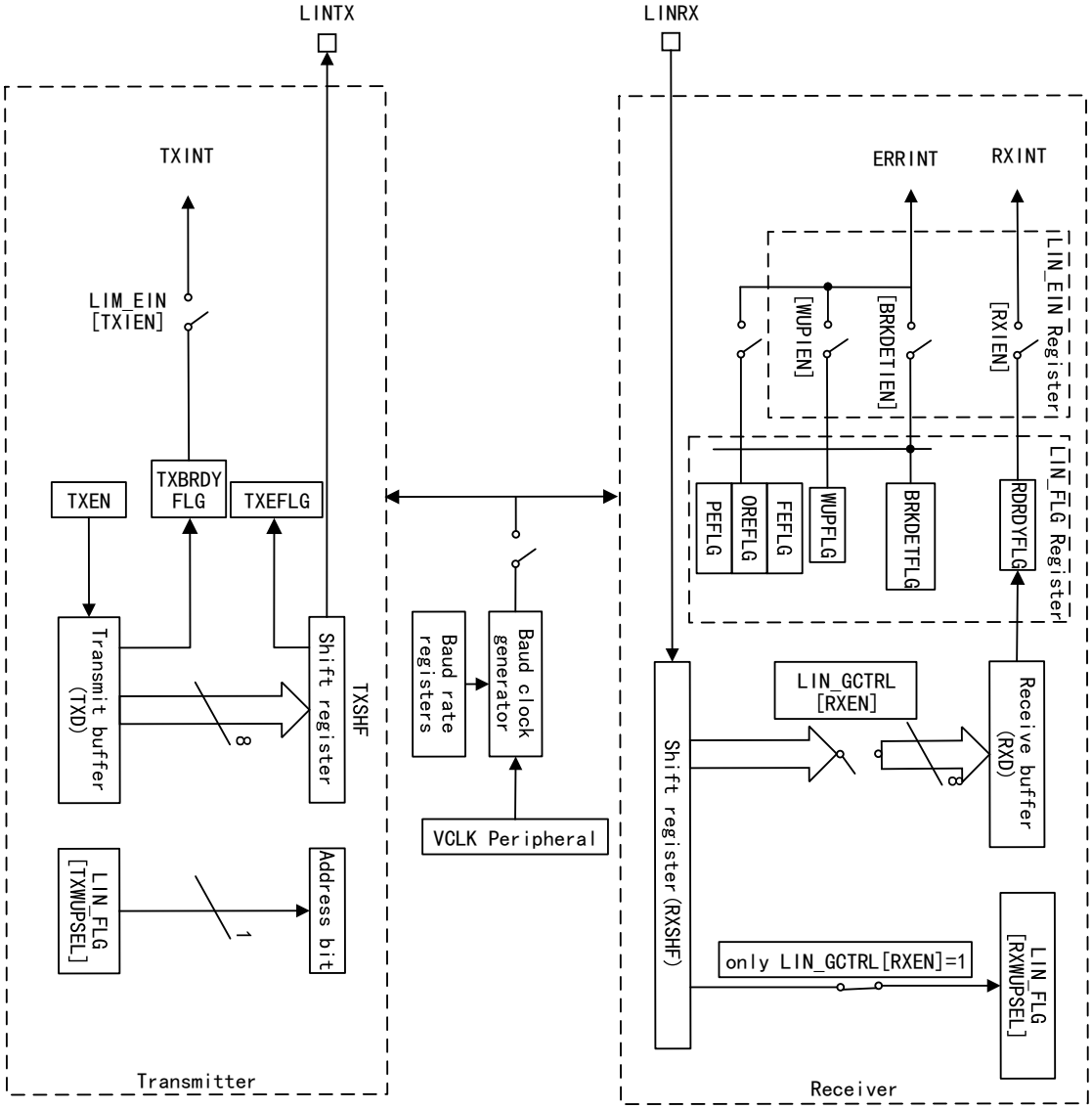
(14) Update wake-up/sleep

(15) Enhancement options:
- Synchronizer finite state machine (FSM) frames, supporting frame processing
- Processing extended frame
- Baud rate generator

40.5 Structure block diagram

The structure of the serial port is shown below.

Figure 256 Serial Port Structure Block Diagram



The structure includes three main modules: receiver, transmitter, and baud rate clock generator.

(1) Receiver (RX)

- The shift register receives data through the LINRX pin, one bit at a time
- The shift register shifts the received data and transmits the complete data to the receive data buffer
- Independent enable bits and interrupts
- It can operate independently or work together with the transmitter in full-duplex mode

(2) Transmitter (TX)

- Data from the CPU is loaded into the transmit data buffer, which then transmits the data to the shift register.

- The shift register shifts the data and transmits it through the LINTX pin to the outside, one bit at a time
- Independent enable bits and interrupts
- It can operate independently or work together with the receiver in full-duplex mode

(3) Baud rate clock generator

- Input clock VCLK provides the clock source for the baud rate clock generator, with $f_{VCLK} = f_{SYSCLK}$
- The editable baud rate clock generator scales the clock signal and then provides the baud rate clock to the module

The serial port ensures data integrity by checking for interrupts, parity errors, overflows, and framing errors.

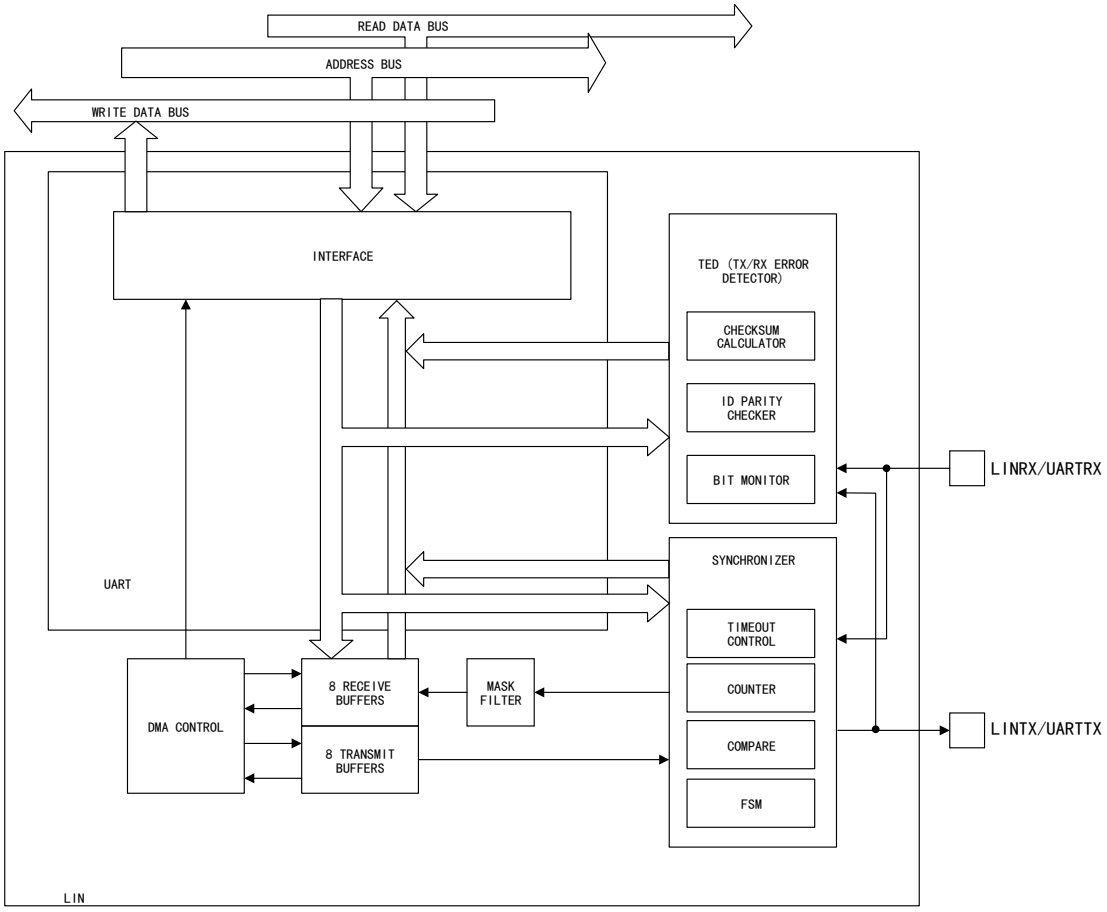
The bit rate can be configured via a 24-bit baud rate select register, supporting over 16 million rates.

The difference between the UART/LIN module and the UART module is that the former adds the following features:

- Mask filter
- Error checker, including a parity calculator, checksum calculator, and bit monitor
- Synchronizer
- Multi-buffer receiver
- Multi-buffer transmitter

To achieve LIN compatibility, hardware enhancements have been made, including the UART interface, DMA control sub-module, and baud rate generator. The block diagram of the UART/LIN module is shown below.

Figure 257 Structure Block Diagram of UART/LIN Block



40.6 LIN Functional Description

40.6.1 LIN standard

The LIN module incorporates CPU performance consumption features defined in hardware LIN specifications versions 1.3 and 2.0. The primary mode of this module is compatible with the LIN2.1 standard.

To achieve LIN2.0 compatibility on top of LIN1.3, the following functions are implemented:

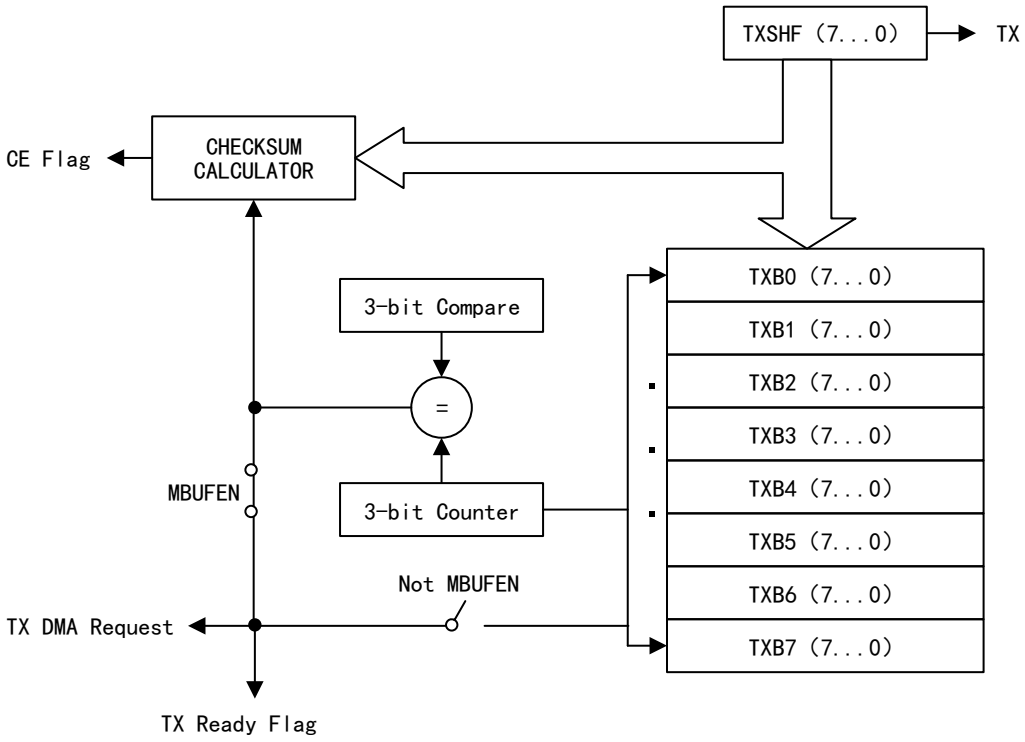
- Update wake-up/enter sleep mode
- Enhanced synchronizer FSM support for frame processing
- Enhanced processing for extended frames
- Enhanced baud rate generator
- Support for LIN2.0 checksums

40.6.2 Transmit Data Buffer

To reduce CPU burden, the UART/LIN module provides eight transmit buffers (LINTXB0 and LINTXB1 registers) that can be used to transmit LIN n-byte (N = 1-8) responses. The entire LIN response field can be preloaded into the transmit buffer. In addition, if you do not choose to use multi-buffer mode (MBUFEN bit

clear 0), you can also perform DMA transfers by byte.

Figure 258 Transmit Buffer



In Multi-buffer mode, the multi-buffer 3-bit counter is used to count the number of bytes of data transferred from the transmit buffer register; Otherwise, the counter counts the number of bytes of data transferred from TXB0 to TXSHF. The comparison register contains the number of bytes of data expected to be transferred. The ID field can be used to pass the message length (LIN protocol uses parity bits in identifiers, control length bits are programmable), if not the ID pass length, then the FLCFG value is used to indicate the desired length and is used to load the comparison register. Choose to use the length control field or FLCFG value by configuring the COMCFG bit.

After transmitting the response, with the corresponding function bits enabled, transmit interrupts, transmit ready flags, and DMA requests may be generated. Depending on whether multi-buffer mode (MBUFEN bit) is enabled, a DMA request can be generated for each transmitted byte or for the entire response.

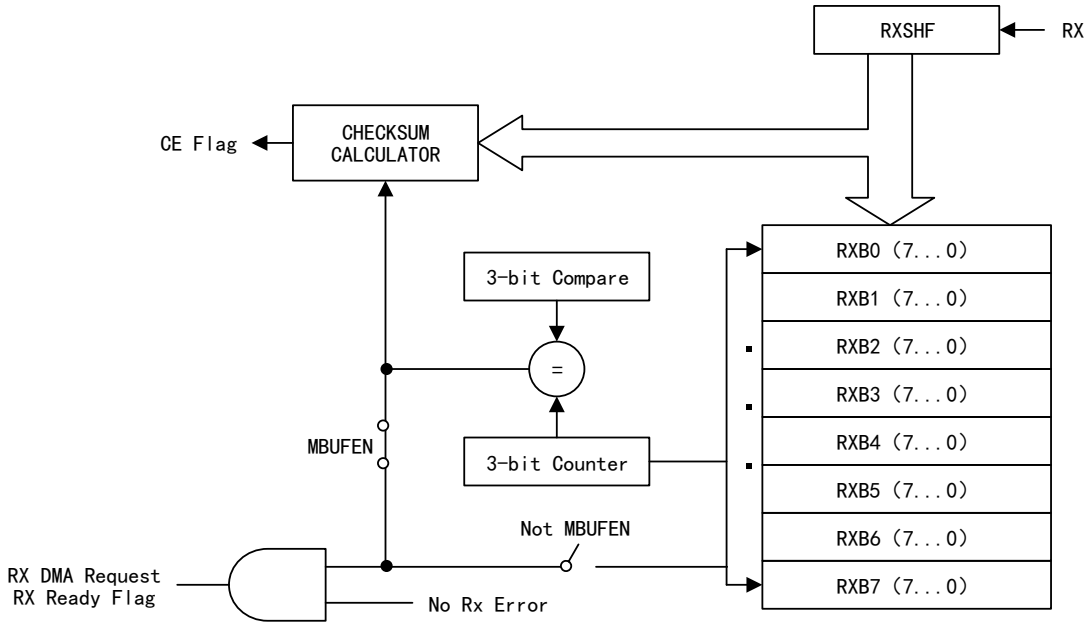
The checksum byte is automatically generated by the checksum calculator and transmitted after the data field transmission is completed. The 3-bit multi-buffer counter counts the data bytes transferred from the TXBy buffer to the TXSHF register.

Note: Transmit interrupt requests can be suppressed until the next batch of data is written to the transmit buffers LINTXB0 and LINTXB1 by disabling the corresponding interrupt via the LINICLR register or disabling the transmitter using the TXEN bit.

40.6.3 Receive Data Buffers

To reduce CPU load during LIN N-byte (N=1–8) response reception in interrupt or DMA mode, the UART/LIN module has 8 receive buffers. These buffers can store an entire LIN response in the RXBy receive buffering. The receive buffer is shown in the figure below.

Figure 259 Receive Buffer



The checksum byte is checked by the internal calculator, and if an error is detected, the checksum error flag bit is set, and if the corresponding interrupt is enabled, the device generates a checksum error interrupt.

In contrast to transmit mode, in Multi-buffer mode, the 3-bit counter of multi-buffer is used to count the number of data bytes transferred from the receive shift register to the buffer; Otherwise, the counter counts the number of bytes of data transferred to the TXB.

But like the transmit mode, the comparison register contains the number of bytes of data expected to be received. The ID field can be used to pass the message length (LIN protocol uses parity bits in identifiers, control length bits are programmable), if not the ID pass length, then the FLCFG value is used to indicate the desired length and is used to load the comparison register. Choose to use the length control field or FLCFG value by configuring the COMCFG bit.

After receiving the response, a receive interrupt and receive ready flag as well as a DMA request can occur if the bits of the corresponding function are enabled. Depending on whether multi-buffer mode (MBUFEN bits) is enabled, DMA requests can be generated for each received byte or for the entire response. The checksum calculator detects checksum bytes before confirming receipt.

Note: In multi-buffer mode, RXRDYFLG is not cleared by the LININTVECT0/1 register, but by the LINRXBx register, when FLCFG is greater than 4, by the LINRXB1 register, otherwise by the LINRXB0 register.

40.6.4 Communication format

The UART/LIN module can operate in both LIN mode and UART mode. Enhanced baud rate generation, DMA control, and the additional receive/transmit buffers required for LIN mode operations are also part of the enhanced buffered UART module. LIN mode is selected by enabling the LINEN bit.

UART/LIN is built around the UART platform and uses a similar sampling scheme: 16 samples per bit, with majority voting on samples 8, 9, and 10. For the START bit, the first three samples are used.

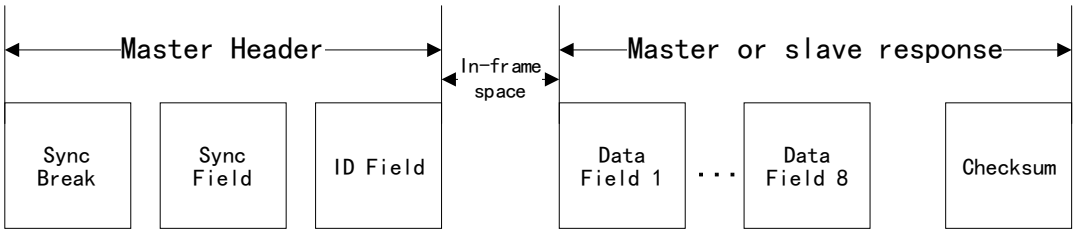
40.6.5 Frame format

The message frame in the LIN protocol is shown below. Each frame contains: a main header, an intra-frame response space (which can be 0), and a response and interbyte space (which can be 0).

Since there is no arbitration mechanism in the LIN protocol, multiple slave nodes responding to the header may be considered an error.

LIN bus is a single-channel wire and gate, the dominant level is represented by binary 0; The invisible level is represented by a binary 1.

Figure 260 Frame Format: Master Header and Response

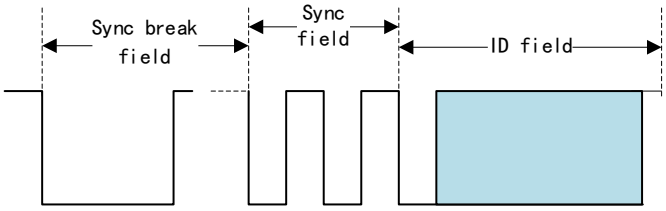


40.6.5.1 Header

The message header is initiated by the master (as shown below) and consists of three sequential fields:

- The synchronization interrupt field indicates the start of a message
- The synchronization field conveying the LIN bus baud rate information
- The identifier field indicates the content of the message

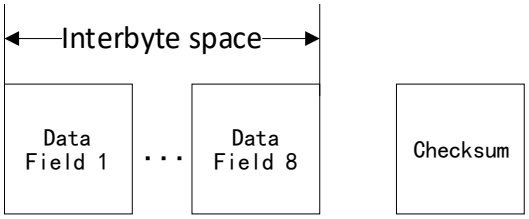
Figure 261Header Fields - Synchronization interrupt, synchronization and ID



40.6.5.2 **Response**

The format of the response is shown below. The response contains two types of fields: data and checksum. The data field consists of a data byte, a start bit, and a stop bit, totaling 10 bits. The least significant bit (LSB) is transmitted first. The checksum field consists of a checksum byte, a start bit, and a stop bit. The checksum byte is the modulo-256 sum of all data bytes in the response data field.

Figure 262 Response Format of LIN Message Frame



The response format is the stream consisting of N data fields and a checksum field. Typically, N ranges from 1 to 8, except for extended command frames. The response length N can be indicated by optional length control bits in the ID field (used in LIN standards prior to 1.x), as shown in Table 27-8, or by the FLCFG value in the LINLCFG[18:16] register, as shown in the table below. The UART/LIN module supports response lengths from 1 to 8 bytes, in compliance with LIN2.0.

Table 241 Response Length Information for LIN Standards Prior to V1.3 Using IDBYTE Field Bits [5:4]

ID5	ID4	Number of data bytes
0	0	2
0	1	2
1	0	4
1	1	8

Table 242 Response Length Information Configured via LINLCFG[18:16]

LINLCFG[18:16]	Byte Count
000	1

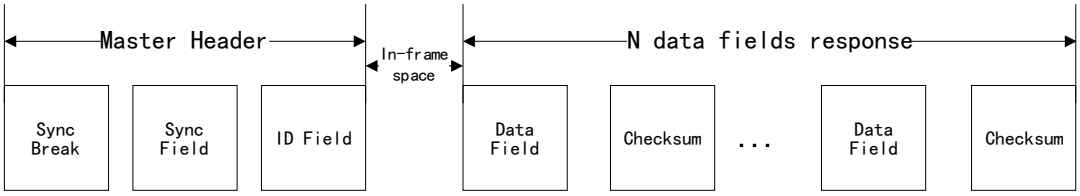
001	2
010	3
011	4
100	5
101	6
110	7
111	8

40.6.6 Extended Frame Processing

In LIN2.0 and earlier versions, the message frame also includes two extension frames with identifiers 62 (user programmable) and 63 (reserved extension). The response data of user-programmable frames (identifiers 62 or 3Eh) is unrestricted, the length is set at network configuration time, and shared with the LIN bus node.

Extended frame communication is triggered when a reserved extended frame (identifier 63 or 3Fh) is received, and the frame format is shown below. Setting the STOEXTCOM bit stops extended frame communication, requiring a bit before issuing another header.

Figure 263 Optional Embedded Checksum in Extended Frame Response



Receiving ID 62 (0x3E) will generate an ID interrupt (if enabled and matched). This interrupt allows the CPU to use a software counter to track the transmitted bytes and determine when to calculate and insert the checksum byte (recommended at a periodic rate). To handle this process, set the TXCHAEN bit, which will initiate the automatic transmission of the checksum byte. The last data field shall always be the checksum in accordance with the LIN protocol.

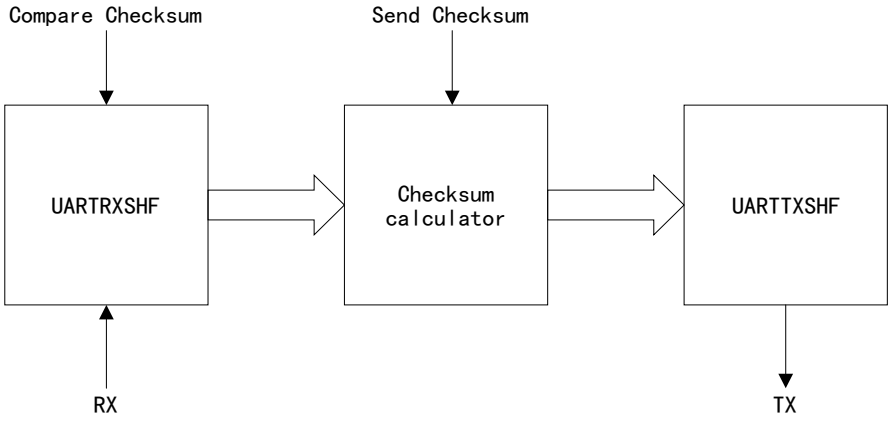
The receiving node uses the periodic checksum insertion defined during network configuration to verify the checksum of the ongoing message and enhance reliability.

For the transmitting node, each time the transmit checksum bit TXCHAEN is set, the checksum is automatically embedded.

For the receiving node, each time the compare checksum bit COMPCHASUM is set, the checksum is verified, as shown in the figure.

Note: The enhanced LIN 2.0 checksum does not apply to reserved identifiers. Reserved identifiers always use the classic checksum.

Figure 264 Checksum Comparison and Transmission for Extended Frames



40.6.7 Synchronizer

In message exchanges between the master and slave nodes, the synchronizer serves three purposes:

(1) Generating the master header data stream

(2) Synchronizing with the LIN bus for the response

(3) Detecting timeouts locally.

To match the LIN_speed value indicated in the LIN descriptor file, the bitrate is programmed using the prescaler in the LINBR register.

The LIN synchronizer can detect incoming breaks at any time and initialize communication, with its main functions including:

(1) Generating master header signals

(2) Slave detection

(3) Synchronizing to the message header with optional baud rate adjustment, response transmission timing, and timeout control

40.6.8 Baud rate

During initial configuration, the CPU sets the transmission baud rate for all nodes and defines the bit time T_{bit}.

Bit time can be configured using the PSCSELL/H bits (P) and the FDIVSEL (M) bits in the baud rate register (LINBR). The SFDIVSEL (U) bits allow an additional 3-bit fractional frequency division value, further fine-tuning the data field baud rate.

The prescaler values in the LINBR register are fully programmable, with ranges

for the three values shown in the following table:

Table 243 Range of Prescaler Values

Field Name	Frequency Division Value Presented	Range
PSCSELL/H	P	0, 1, 2, 3, 4, …, 2^{24}-1
FDIVSEL	M	0, 1, 2, 3, 4, …, 15
SFDIVSEL	U	0, 1, 2, 3, 4, …, 7

PSCSELL/H and FDIVSEL frequency dividers can be used in UART mode and LIN mode to select the baud rate. The SFDIVSEL value determines "aTVCLK" (a = 0, 1) that is added to each T_{bit}; see the "Subfractional Frequency Divider" section for details.

If the LIN slave is in adaptive baud rate mode and the ABEN bit is set, all frequency divider values are automatically obtained during header reception while measuring the synchronization field.

The LIN protocol defines the baud rate range as 1 kHz ≤ FLINCLK ≤ 20 kHz

All transmitted bits are shifted in and out within the T_{bit} period.

40.6.8.1 Fractional Frequency Divider

The increment of M is 1/16 of P. If you want to fine-tune the baud rate, the M bits can be configured to adjust the integer prescaler P.

Bit time T_{bit} is expressed in VCLK cycles (T_{VCLK}) as follows:

(1) If P ≠ 0, M = any value:

$$T_{bit} = 16\left(P + 1 + \frac{M}{16}\right)T_{VCLK}$$

(2) P = 0:

$$T_{bit} = 32T_{VCLK}$$

Thus, the frequency of LINCLK is:

(3) P ≠ 0:

$$F_{LINCLK} = \frac{F_{VCLK}}{16\left(P + 1 + \frac{M}{16}\right)}$$

(4) P = 0:

$$F_{CLK} = \frac{F_{VCLK}}{32}$$

40.6.8.2 Ultra-fractional Frequency Divider

The ultra-fractional frequency divider can be configured in both LIN master mode and slave mode.

- Master mode fields: Synchronization field + Identifier field + Response field + Checksum field
- Slave mode fields: Response field + Checksum field

Based on a 4-bit fractional frequency divider M, the ultra-fractional frequency divider uses an additional 3-bit modulation value, as shown in the table below.

The Data field can be considered as the Sync field (0x55), the Identifier field, and the Response field. A bit set to 1 indicates that an additional VCLK cycle is added to its T_{bit}.

Table 244 Ultra-fractional Modulation

LINBR [30:28]	Start bit	Data[0:7]								Stop bit
		[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

In LIN master mode, the synchronization field, identifier field, and response field are bit-modulated.

In LIN slave mode, the identifier field and response field are bit-modulated.

The baud rate within the LIN data field varies based on the d fraction (0 < d < 1) of the LINBR[30:28] value, which is derived from the average of the internal peripheral clock.

The average bit time, represented in T_{VCLK}, is as follows:

(1) P ≠ 0, with any M and d (0 < d < 1):

$$T_{bit}^{a} = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right] \mathrm{T}_{VCLK}$$

(2) P = 0:

$$T_{bit} = 32T_{VCLK}$$

The instantaneous bit time, represented in T_{VCLK}, is as follows:

(3) P ≠ 0, with any M and d (d = 0, 1):

$$T_{bit}^{i} = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right] \mathrm{T}_{VCLK}$$

(4) P = 0:

$$T_{bit} = 32T_{VCLK}$$

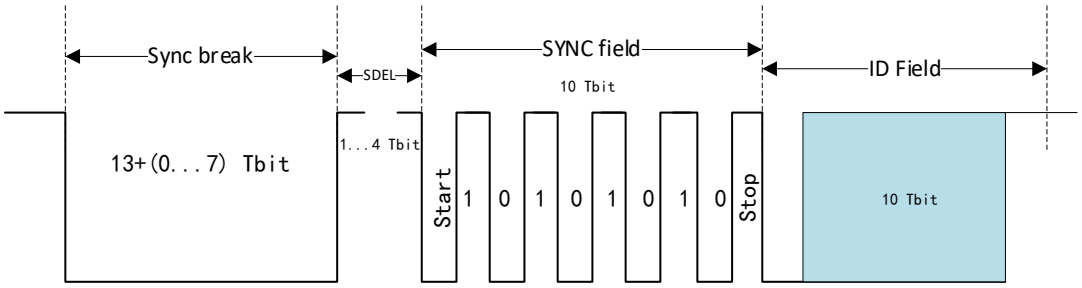
40.6.9 Message Header Generation

Supports automatic generation of LIN protocol header data streams, does not require CPU interaction, only needs to be triggered by CPU and DMA generation, LIN can carry out its own generation process. Triggering can be achieved by CPU or DMA writing to LINID[IDBYTE].

The header, transmitted by the master, initiates LIN communication, consisting of a synchronization interrupt field, a synchronization field, and an identifier field, as shown below.

The LIN protocol uses parity bits in the identifier, and the length bits are programmable.

Figure 265 Message Header in T_{bit} Units



The BREAK field is composed of two parts: the SYNCH BREAK and the synchronization separator (SDEL).

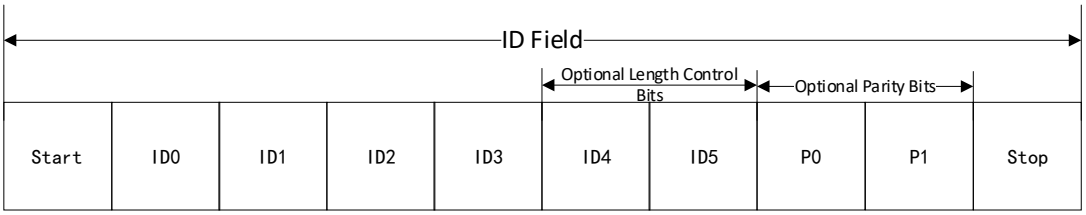
As shown in the figure, the SYNCH BREAK can have 13-20 dominant bits. The length of the SYNCH BREAK can be extended from the minimum value to a value determined by the 3-bit SBEXTSEL field in the LINCOMP register.

SDEL consists of at least one recessive bit and at most four recessive bits. The separator marks the end of the SYNCH BREAK field. The length of the synchronization separator is determined by the 2-bit SDCOMPSEL field in the LINCOMP register.

The SYNCH FIELD is used to transmit Tbit information and resynchronize LIN bus nodes. It consists of a start bit, byte 0x55, and a stop bit. Identifier field ID bytes 6 bits used as identifier, and the length of the support optional control and two optional parity identifier. If the parity bit is enabled, then the parity identifier is used and checked, and if the length control bit is not used, then the identifier will have 64, followed by the parity bit; If neither of these bits is used, there can

be up to 256 identifiers.

Figure 266 ID Field



Note: Length control bits apply only to LIN standards prior to version 1.3. If compliant with standards prior to LIN 1.3, the IDBYTE field conveys response length information. The LINLCFG register stores the response length for later versions of the LIN protocol.

If a slave LIN module in multi-buffer mode is transmitting data and a new header arrives, the module might end up responding with data from a previous interrupt response (instead of data corresponding to the new ID). To prevent this, follow these steps:

(1) Check for a bit error (BE) during the response transmission. If the BE flag is set, it indicates a collision occurred on the LIN bus (due to a new sync interrupt).

(2) In the bit error ISR, configure the TXB0 and TXB1 registers with the next set of data to be transmitted on a TX match for the incoming ID. To prevent TXB0/TXB1 from being written twice for a single ID, ensure that no updates occur due to bit errors before writing to TXB0/TXB1.

(3) Once the complete ID is received and matched, the node will transmit the newly configured data.

40.6.9.1 Event-triggered Frame Processing

The LIN 2.0 protocol uses event-triggered frames, which can lead to conflicts. Event-triggered frames must be processed in software.

NRE indicates no response error. It indicates that the slave device does not respond to the event triggering frame header. In this case, the NREFLG position is set to 1 and the NRE is interrupted (if interrupt is enabled). If a conflict occurs, a frame error and a checksum error may be generated first, and the corresponding interrupt may occur.

Frame errors (FE) and checksum errors (CSE) depend on the behavior and synchronization of the responding slave. If the slave is fully synchronized and stops transmitting after a collision, only the NRE error might be flagged despite the collision. To detect whether a byte was received before the NRE error is flagged, the bus busy flag (BUSYFLG) can be used as an indicator.

The BUSYFLG is set upon receiving the first bit of a header and remains set until header reception is complete, and then set again when receiving the first

bit of a response. If a collision occurs, this flag is cleared within the same cycle that sets the NREFLG flag.

Software can implement the following sequence:

- Once the header receive is complete (poll the RXID flag), wait for the BUSYFLG to be set or the NREFLG flag to be set.
- If the BUSYFLG is not set before the NREFLG flag, this indicates a genuine no-response situation (no data was transmitted on the bus).
- If the BUSYFLG is set, wait for either the NREFLG flag to be set or for successful receive. If the NREFLG flag is set, then a collision occurs on the bus.

Even in the event of a collision, the received (corrupted) data can still be accessed in the RX buffers.

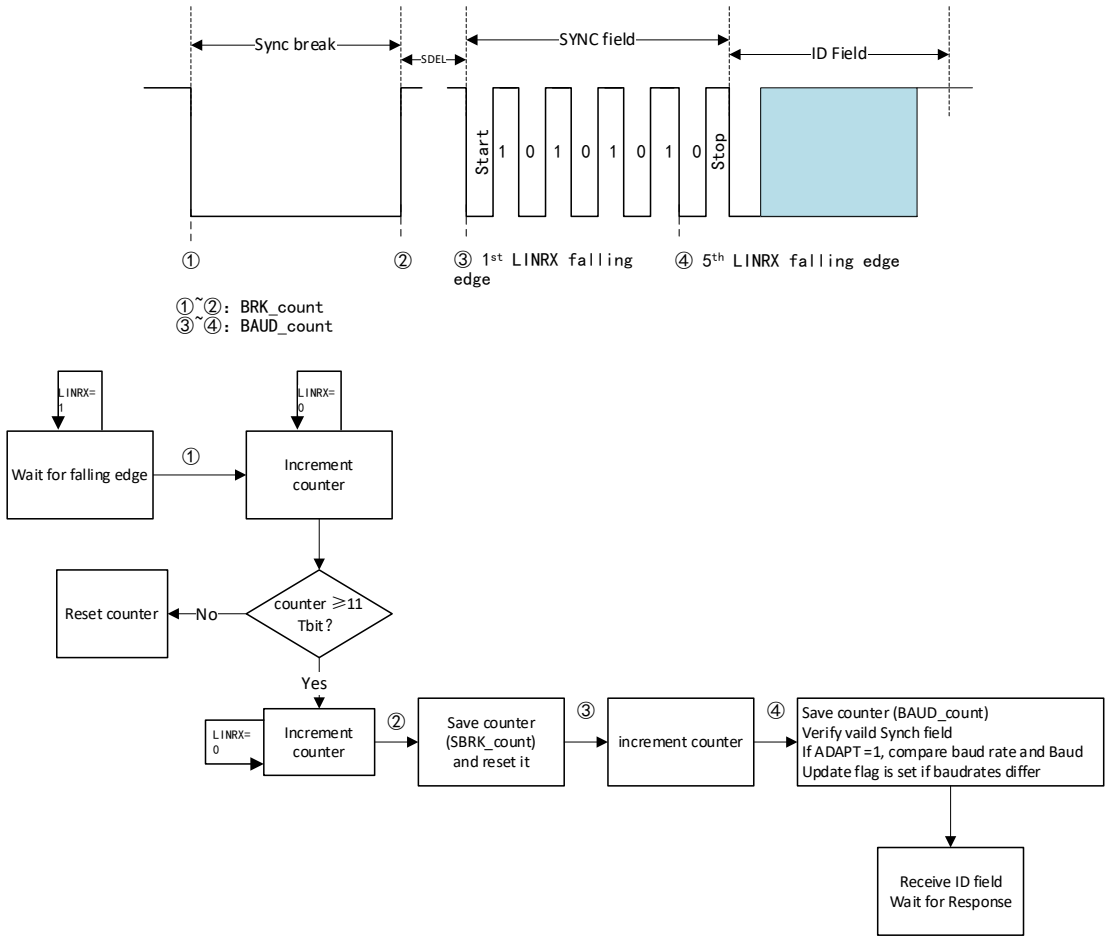
40.6.9.2 Header Receive and Adaptive Baud Rate

The slave node baud rate can optionally be adjusted to match the detected bit rate.

Enable the adaptive baud rate option by setting the ABEN bit. During header reception, the slave measures the baud rate during the synchronization field detection. If the ABEN bit is set, the measured baud rate is compared with the programmed baud rate of the slave node, and adjusted to the LIN bus baud rate if necessary.

The LIN synchronizer determines two measurements: BRK_count and BAUD_count. These values are calculated during the header reception synchronization field verification.

Figure 267 Synchronization Measurement, synchronous verification process and baud rate adjustment



The synchronizer counter uses VCLK as a time base to measure synchronization interruptions relative to the Tbit of the detection node. As shown in the figure above, with two counters, BRK_count and BAUD_count, a valid sequence of synchronous interrupts can be detected, starting at the falling edge and ending at the rising edge.

According to the LIN protocol, the slave node needs to use a threshold of 11Tbit when receiving synchronous interrupts. The falling edge of the LINRX pin triggers a synchronizer counter to detect the dominant data stream for synchronization interruptions. If a synchronization separator is detected, LIN saves the value of the counter before resetting it.

As shown in the waveform diagram, after four consecutive falling edges are detected, BAUD_count is measured at the fifth falling edge immediately following. The consistency of the bit timing calculation and the required accuracy is achieved according to the recommendations of LIN 2.0.

The single Tbit time from a device node can be calculated using BAUD_count/8.

BAUD_count The counter moves 3 bits to the right and rounds off the first

irrelevant bit to obtain a Tbit unit. If the ABEN bit is set, the detected baud rate is compared to the programmed baud rate.

The following formula is provided to evaluate the consistency of the detected edges, BAUD_count1 means that BAUD_count is shifted 2 bits to the right and BAUD_count1 means that BAUD_count is shifted 3 bits to the right:

$$BAUD_count + BAUD_count1 + BAUD_count2 \leqslant BRK_count$$

In the process of receiving the header, the value of BRK_count is determined to be less than 11Tbit. If the result is "yes", then the synchronization interrupt is invalid. If the ABEN bit is set, the LINMBRPSC register measures the value of the two synchronizer counters and automatically adjusts the LIN bus rate.

Note: In adaptive mode, the baud rate set by LINMBRPSC must be less than 10% of the baud rate expected by LIN, otherwise a numeric byte of 0x00 May be mistaken for a synchronization interrupt.

Relative to the node from the device disconnect threshold for 11 T_{bit}, in LIN V1.3 is specified in the breakpoint for 13 T_{bit}. This means that when designing and adjusting LIN, it is necessary to pay attention to the baud rate setting to prevent false detection and ensure the reliability of communication.

If the synchronization field is not detected within a given tolerance range, an inconsistent synchronization field error flag is set. If the corresponding bit in the LINIEN register is set, an ISFE interrupt is generated. The ID byte shall be received after the synchronization field verification succeeds. If a valid interrupt (greater than 11 T_{bit}) is detected at any time, the receiver state machine shall be reset to receive this new frame. This reset condition is only valid during the response state and not valid if an additional synchronization interrupt occurs during header reception.

Note: When an inconsistent synchronization field error occurs, the recommended action is for the application to clear the RDY bit, and then set the RDY bit to ensure the internal state machine returns to normal.

40.6.10 Timeout Control

Any LIN node that listens on the bus and expects a response initiated by the master can flag a no-response error timeout event.

LIN protocol timeout events are processed by hardware, and there are four types:

- No-response timeout error
- Idle bus detection
- Timeout after a wake-up signal
- Timeout after three wake-up signals

40.6.10.1 No-response error

A no-response error occurs when any node waiting for a response exceeds the TFRAME_MAX duration, and the message frame is not fully completed within

the allowed maximum length TFRAME_MAX. At this point, the no-response error (NRE) is flagged in the NREFLG bit of the LINFLG register. If enabled, this triggers an interrupt.

The LIN 1.3 standard specifies the minimum time to transmit a frame (where N = number of data fields):

$$T_{FRAME_MIN} = T_{HEADER_MIN} + T_{DATA_FIELD} + T_{CHECKSUM_FIELD} = 44 + 10N$$

The maximum time range is:

$$T_{FRAME}_MAX = T_{FRAME}_MIN * 1.4 = (44 + 10n) * 1.4$$

The timeout value T_{FRAME_MAX} is derived from the number of data fields, as shown in the following table.

Table 245 Timeout Values in T_{bit} Units

N	T_{DATA_FIELD}	T_{FRAME_MIN}	T_{FRAME_MAX}
1	10	54	76
2	20	64	90
3	30	74	104
4	40	84	118
5	50	94	132
6	60	104	146
7	70	114	160
8	80	124	174

The N value is either embedded in the message header's ID field or part of the descriptor file. In the latter case, the CLCFG value in the LINLCFG register represents the N value.

Note: The length encoding in the ID field does not apply to the two extended frame identifiers, 0x3E (62) and 0x3F (63). In these cases, the ID field may be followed by an arbitrary number of data byte fields. Additionally, the LIN 2.0 protocol specification states that ID field 0x3F (63) can not be used. For these scenarios, NRE will not be processed by LIN hardware.

40.6.10.2 Idle bus detection

Bus idle detection timeout occurs when a node detects an inactive LIN bus: no transitions are detected between recessive and dominant values on the bus.

This happens after at least 4 seconds (80000 F_{LINCLK} cycles at the fastest bus rate of 20 kbps). If a node detects no activity on the bus after setting the TIMEOUT bit, it can assume that the LIN bus is in sleep mode. The software can use the Timeout flag to determine when the LIN bus is inactive and place LIN into sleep mode by writing to the LPREQ bit.

Note: After flagging a timeout, RDY shall be asserted before entering low power consumption mode. If there is an incomplete frame on the bus before idle time, the receiver must be reset.

40.6.10.3 Timeouts after a Wake-up Signal and after Three Wake-up Signals

These two timeout scenarios are related to wake-up signals. The node that initiates a wake-up shall receive a header from the master within the defined wake-up signal timeout period. For more details, see the “Wake-up Timeout” section.

40.6.11 TX/RX Error Detector (TED)

The TED logic consists of a bit monitor, ID parity checker, and checksum error. The TED can detect the following error sources:

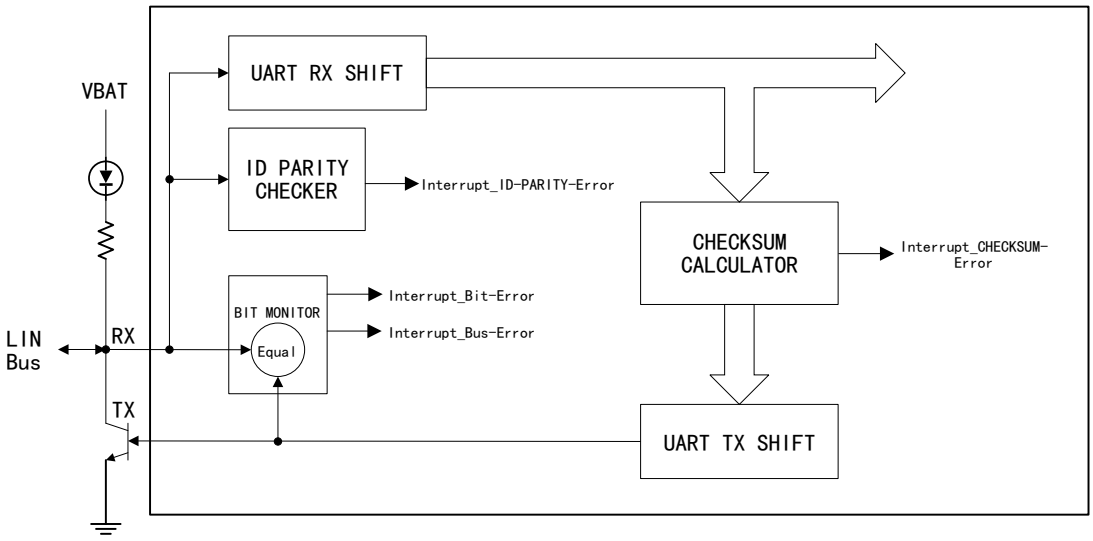
- Physical bus error (PBEFLG)
- Identifier parity error (PEFLG)
- Bit error (BEFLG)
- Checksum error (CSEFLG)

Interrupts can only be generated when the corresponding interrupt enable bits are set. If these errors are not detected before the end of the frame, both transmitted and received data are considered valid.

40.6.11.1 Bit Error (BE)

A bit error is detected during the bit time when the monitored bit value does not match the transmitted bit value. After signaling a bit error (BE), transmission is terminated no later than the next byte. The bit monitor verifies that the bits transmitted on LINTX are correct on the LIN bus by reading back the LINRX pin, as shown in the figure below:

Figure 268 TX/RX Error Detector



If a bit error occurs during the slave’s response due to receiving a header, the NREFLG/TIMEOUT flag will not be set for the new frame.

40.6.11.2 Physical Bus Error

If no valid messages can be generated on the bus (e.g., due to the bus being shorted to GND or VBAT), the physical bus error must be detected by the master. The bit monitor detects a PBE during header transmission if a synchronization interrupt cannot be generated (for example, because the bus is shorted to VBAT) or if a synchronization interrupt separator cannot be generated (for example, because the bus is shorted to GND). Once the synchronization separator is verified, any further deviations between the monitored and transmitted bit values are flagged as bit errors for that frame.

40.6.11.3 ID Parity Error

If parity checking is enabled, an ID parity error (PEFLG) is detected when one of the two parity bits in the transmitted ID byte does not match the parity calculated by the receiving node. These two parity bits are generated using a mixed parity algorithm:

$$\mathrm{P0} = \mathrm{ID0} \oplus \mathrm{ID1} \oplus \mathrm{ID2} \oplus \mathrm{ID4}\ (\text{even Parity})$$

$$\mathrm{P1} = \mathrm{ID1} \oplus \mathrm{ID3} \oplus \mathrm{ID4} \oplus \mathrm{ID5}\ (\text{odd Parity})$$

If a PEFLG is detected, the PEFLG flag is set, and the received ID is considered invalid. For more details, refer to the Message Filtering and Validation section.

40.6.11.4 Checksum Error

If the modulo-256 sum of all received data bytes (including the ID byte for enhanced checksum types) and the checksum byte does not equal 0xFF, a checksum error is detected and flagged at the receiver. The modulo-256 sum is calculated by adding each byte using carry addition, where the carry from each addition is added to the least significant bit (LSB) of the resulting sum.

For transmitting nodes, the checksum byte transmitted at the end of the message is the inverted sum of all data bytes (see the “Classic Verification and Generation for Transmitting Nodes” figure), which corresponds to the classic checksum implementation. The checksum byte is the inverted sum of the identifier byte and all data bytes (see the “LIN2.0 Compatible Checksum Generation for Transmitting Nodes” figure) and is used in the LIN 2.0 compatible enhanced checksum implementation. The classic checksum implementation shall always be used for reserved identifiers 60 to 63. In such cases, the CHASUMCFG bit is overridden. For signal-carrying frame identifiers (0 to 59), the type of checksum used depends on the CHASUMCFG bit.

Figure 269 Classic Verification and Generation for Transmitting Nodes

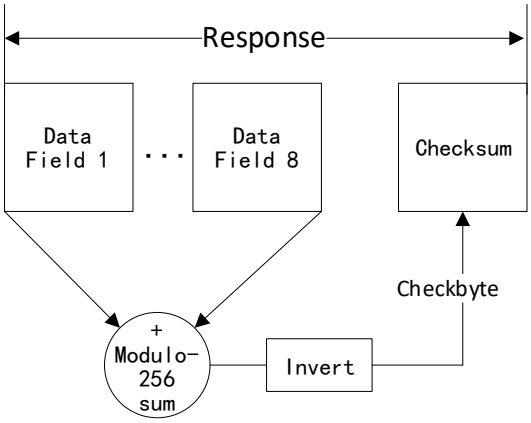
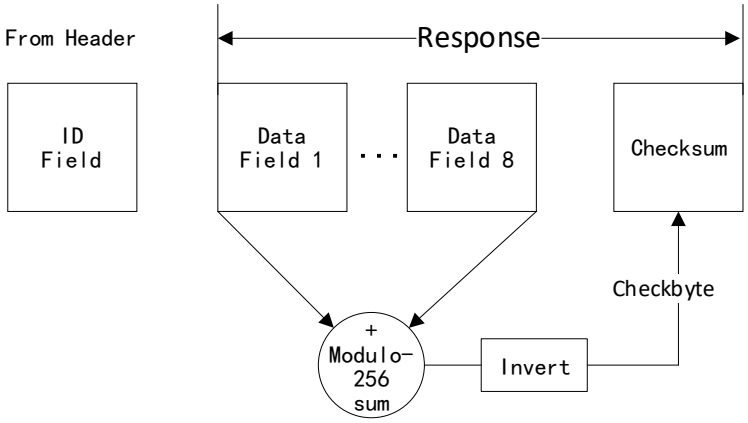


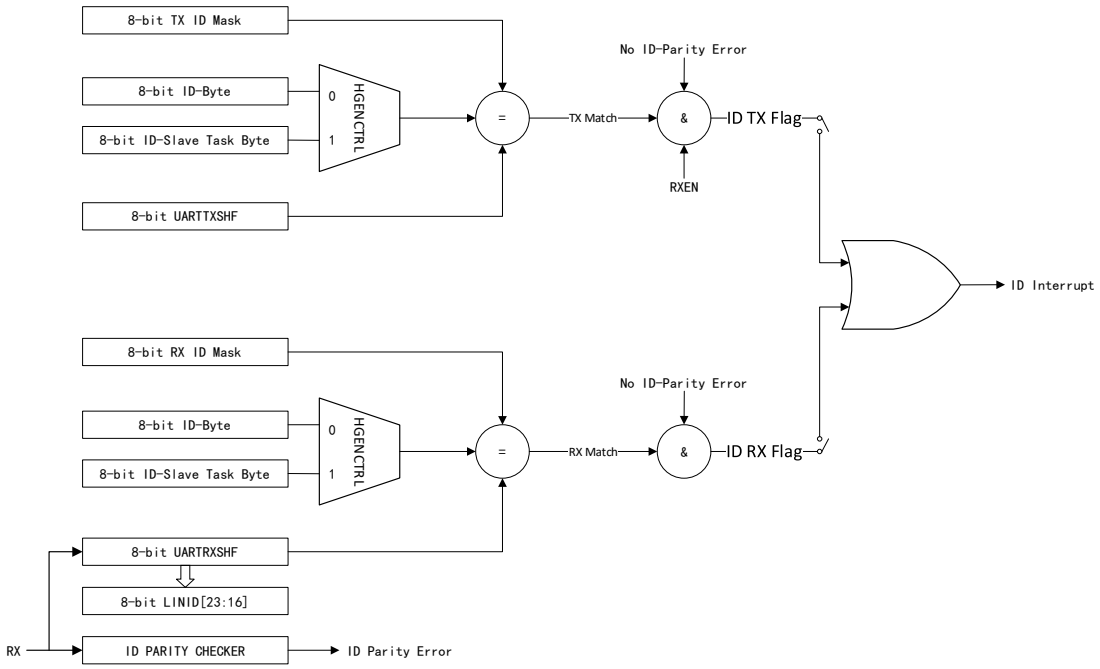
Figure 270 LIN 2.0 Compatible Checksum Generation for Transmitting Nodes



40.6.12 Message Filtering and Validation

Message filtering uses the entire identifier to determine which node participates in the response, whether to receive or transmit a response. Two receive masks are used for this purpose, as shown in the figure below.

Figure 271 ID Reception Filtering, and Validation



During receiving the header, all nodes filter the ID field. The node needs to determine whether the type of response is sent or received. For this, the message ID filtering has a receive mask and a transmit mask respectively. After receiving the ID field, all nodes are compared with the identifiers of the ID-Slavetask stored in the LINID register, and invalid identifiers are filtered through the transmit or receive mask bits of the LINIDMASK.

If an RX match occurs without a parity error and the RXEN bit is set, an ID RX flag is triggered, and an interrupt will occur if enabled. If a TX match occurs without a parity error and the TXENA bit is set, an ID TX flag is triggered, and an interrupt will occur if enabled in the LINIEN register.

Bits that are masked are ignored in the comparison. To construct a mask for a group of identifiers, the XOR function can be used.

For example, using LINID[7:0] = 0x20, a mask that accepts ID 0x26 and 0x25 can be created by comparing the 5 most significant bits and ignoring the 3 least significant bits. The acceptance mask could be:

$$(0x26 + 0x25) \oplus 0x20 = 0x07$$

he value of the HGENCTRL bit causes different masks to treat the received identifiers and idbytes differently. When HGENCTRL=0, LIN node compares the two and uses the mask of LINMASK register to filter out identifiers that do not need to be compared. When HGENCTRL=1, the all-1 mask (0XFF) filters out all receive identifiers and will have a matching ID regardless of the value of the IDBYTE.

Note:If an RX match occurs without a parity error and the RXEN bit is set, an ID RX flag is triggered, and an interrupt will occur if enabled. A zero mask will compare all bits of the identifier received in the shift register to the ID-BYTE field in LINID[7:0]. A mask of all 1s will filter out all bits of the received identifier, resulting in no match.

- If HGENCTRL = 1
  - The received ID is compared with the ID-Slave-Task byte using the RXID and TXID masks.
  - A mask of all 1s always results in a match
  - A mask of all zeros means that all bits must match exactly to yield a match
  - If some bits of the mask are set to 1, those bits will not be used in the filtering criteria.
- If HGENCTRL = 0
  - The received ID is compared with the ID byte using the RXID and TXID masks.
  - A mask of all 1s results in no match
  - A mask of all zeros means that all bits must match exactly to yield a match
  - If some bits of the mask are set to 1, those bits will not be used in the filtering criteria.

When receiving a header, the received identifier is copied to the receive ID field LINID[23:16]. If there is no parity error and either a TX or RX match occurs, the corresponding TX or RX ID flag is set. If the ID interrupt is enabled, an ID interrupt is generated.

After the ID interrupt is generated, the CPU can read the receive ID field LINID[23:16] and determine what response to load into the transmit buffer.

Note: When byte 0 is written to TXB0 (LINTXB0[31:24]), a response transmission is automatically triggered.

In multi-buffer mode:

- When the TXBRDYFLG flag is set, it indicates that the transmission is ready, which means that all response data bytes and checksum bytes have been loaded into the shift register.
- When the TXEFLG flag is set, means that transmit buffer and shift register is cleared, the checksum byte has been sent.

In non-multi-buffer mode:

- The TXBRDYFLG flag is set each time a byte is copied to the shift register, and the last byte of the frame is set only after the checksum byte is copied to the UARTTXSHF register.

- Each time the TXD0 and UARTTXSHF registers are emptied, the TXEFLG flag is set, except for the last byte of the frame in which the checksum byte must be transmitted.

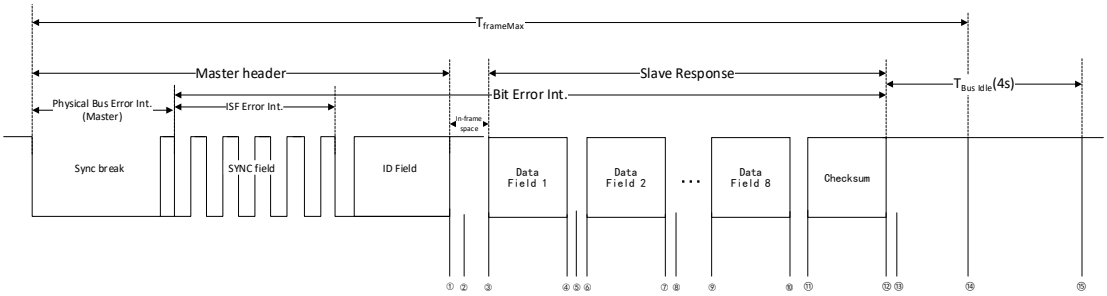
All 8 bits of the receiving node will use the ID byte to verify the identifier (if parity is enabled). If an error is detected, UART/LIN will flag the identifier of the error.

40.6.13 LIN interrupt

LIN and UART modes share a common interrupt block, as described in section 40.5. The UART/LIN module includes 16 interrupt sources, 8 of which are LIN-mode only, as shown in Table 244.

The following figure illustrates a message frame that indicates the timing and sequence of LIN interrupts.

Figure 272 LIN Message Frame Illustrating the Timing and Sequence of LIN Interrupts



①Parity Error Int. ID interrupt
②FE
③TX Int. (Single buffer)
④RX Int. (Single buffer) Overrun Error Int.
⑤FE
⑥TX Int. (Single buffer)
⑦RX Int. (Single buffer)
⑧FE
⑨TX Int. (Single buffer)
⑩Frame Error Int. (FE)
⑪TX Int. (Multi buffer/last byte single buffer)
⑫Checksum Error Int. RX Int. (Multi-buffer/last byte single buffer)
⑬FE
⑭No Response Error Int.
⑮Timeout-Bus Idle(4s) Int.

40.6.13.1 Process LIN Interrupts

When processing interrupts, clear the corresponding flags in the flag register (LINFLG) before clearing the global interrupt flag (LIN_GLB_INT_CLR). ISR shall follow the following guidelines. This will prevent any false or repeated interrupts.

- Clear the LIN interrupt flags in the LINFLG register.
- Read the LIN interrupt status register to ensure the flag is cleared.
- Clear the global interrupt flag bits in LIN_GLB_INT_CLR.

40.6.14 LIN DMA interface

The LIN DMA interface uses the UART DMA interface logic. DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the UART/LIN module. Depending on whether the multi-buffer enable control bit (MBUFEN) is set, multi-buffer mode is enabled, and DMA transfers operate in one of two modes.

Note: Do not use DMA to transmit data to multiple slave IDs. Writing to the LINID register initiates a new transmission. If DMA writes to the LINID register before the LIN state machine is ready to accept a new ID. This will cause LIN to miss the transmission.

40.6.14.1 Transmit DMA request

In LIN mode with the multi-buffer option enabled, after one transmission (with up to 8 data bytes stored in the transmit buffer in the LINTXBx registers), a DMA request is generated to reload the transmit buffer for the next transmission. If the multi-buffer option is disabled, DMA requests are generated on a per-byte basis until all bytes are transmitted. This DMA function is enabled and disabled using the TXDREQEN and TXDREQCLR bits, respectively.

40.6.14.2 Receive DMA request

When the multi-buffer option is enabled, a DMA request is generated if the received response (up to 8 data bytes) is transmitted to the receive buffer (RXBy). If the multi-buffer option is disabled, DMA requests are generated on a per-byte basis until all expected response data fields are received. This DMA function is enabled and disabled using the RXDREQEN and RXDREQCLR bits, respectively.

40.6.15 Transmit/Receive Data Configuration

Before transmitting or receiving data in LIN mode, software shall perform the following configuration steps: The order of register programming can vary as long as the RDY bit in the LINGCTRL1 register remains cleared to 0 throughout the entire LIN configuration process.

(1) Enable LIN by setting the RST bit.

(2) Clear the SWnRST bit to 0 before configuring LIN.

(3) Enable the LINRX and LINTX pins by setting the RXPEN and TXPEN bits.

(4) Select the LIN mode by setting the LIN mode bit.

(5) Select the master or slave mode by setting the clock bit.

(6) Select the desired frame format (checksum, parity, and length control) by setting LINGCTRL1.

(7) Select multi-buffer mode by setting the MBUFEN bit.

(8) Select the desired communication baud rate by configuring the LINBR register.

(9) Configure the maximum baud rate for communication by setting the LINMBRPSC register.

(10) Set the CONTSUS bit to prevent LIN from stopping emulated break points until the current reception or transmission is complete (this bit is only used in simulation environments).

(11) If needed, set the LBEN bit to internally loop back the transmitter to the receiver (this function is for self-inspection purposes).

(12) To receive data, select receive enable RXEN bit.

(13) To transmit data, select transmit enable TXEN bit.

(14) Select the RXIDMASK and TXIDMASK fields in the LINIDMASK register.

(15) After configuring LIN, set SWnRST to 1.

(16) Perform data transmission or configuration (refer to the "Message Filtering and Validation" section, "Transmit Data Configuration" section, and "Configure Data Configuration" section).

After the transmit interrupt is enabled and the software reset is released, LIN can only generate a DMA request, the hardware will not actively trigger the transmission interrupt, and the data can only be written to the relevant register by the software and then transferred to LIN TX for the first transfer.

40.6.15.1 Transmit Data Configuration

If both TXPEN and TXEN bits are set to 1, the LIN transmitter is enabled. If the TXPEN bit is not set, the LINTX pin functions as a general-purpose I/O pin rather than a LIN function pin. No value written to TXB0 will be transmitted until the TXEN bit is set to 1. These two control bits allow the LIN transmitter to remain inactive independently of the receiver.

The TXIDFLG is set after a valid LIN ID is received with a TX match. If enabled, an ID interrupt is generated.

Transmit Data in Single Buffer Mode

When the MBUFEN bit is set to 0, single-buffer mode is selected. After waiting for the sent data to be written to TXB0 (TXBRDYFLG is set), it is transferred to the shift register, and the data is transferred by LIN. If the buffer and shift register are empty, the TXEFLG bit is set.

Transmit data can be processed in the following ways:

(1) Polling the transmit ready flag

(2) Transmit interrupt

(3) Direct memory access (DMA)

In the polling method, the software can poll the TXBRDYFLG bit higher before

writing data to TXB0. This method can overrun the CPU unnecessarily. To avoid this, consider using either interrupt or DMA methods. To use the interrupt method, set the TXIEN bit. To use the DMA method, set the TXDREQEN bit. When TXBRDYFLG is set, an interrupt or a DMA request is generated. When LIN completes transmission of all pending frames, the UARTTXSHF register and TXB0 become empty, and the TXBRDYFLG bit is set, generating an interrupt or DMA request if enabled. Once all data has been transmitted, interrupts or DMA requests shall stop. This can be achieved by disabling the transmit interrupt (TXICLR), disabling the DMA request (TXDREQCLR bit), or disabling the transmitter (clearing the TXEN bit). If the CONTSUS (Transmit checksum) bit is set to 1 to enable the checksum scheme, the checksum byte will be sent after the current byte is transmitted. The SC bit is cleared once the checksum byte is transmitted.

The TXBRDYFLG flag cannot be cleared by reading the corresponding interrupt offset in the LINIVOx register.

Transmit Data in Multi-buffer Mode

When the MBUFEN bit is set to 1, multi-buffer mode is selected. Similar to single-buffer mode, you can use polling, DMA, or interrupt methods to write the data to be transmitted. Transmitted data must be written to the LINTXB0 and LINTXB1 registers according to the number of bytes. LIN waits for the byte 0 to be written to LINTXB0 register (TXB0) and transmits the programmed number of bytes to TXSHF for automatic transmission one by one. If the checksum scheme is enabled by setting the CONTSUS bit to 1, the checksum will be sent after the last programmed data byte, as indicated by the FLCFG field, is transmitted. The CONTSUS bit is cleared once the checksum byte is transmitted.

40.6.15.2 Receive Data Configuration

If both the RXPEN and RXEN bits are set to 1, the LIN receiver can receive messages. If the RXPEN bit is not set, the LINRX pin functions as a general-purpose I/O pin rather than a LIN function pin.

The RXIDFLG is set after a valid LIN ID is received using RX Match. If enabled, an ID interrupt is generated.

Receive Data in Single Buffer Mode

When the MBUFEN bit is cleared to 0, single-buffer mode is selected. In this mode, LIN sets the RXRDYFLG bit when newly received data in RXSHF is transmitted to RXB0. After the new data in RXRDYFLG is read, the UART clears the RXRDYFLG bit. Additionally, if any of these error conditions are detected in the received data as it is transmitted from RXSHF to RD0, LIN will set the FEFLG, OEFLG, or PEFLG flags. Configurable interrupt capabilities support

these error conditions.

Receive data can be processed in the following ways:

(1) Polling the receive ready flag

(2) Receive interrupt

(3) Direct Memory Access

In the polling method, software can poll the RXRDYFLG bit and read data from the RXB0 byte of the LINRXB0 register as soon as RXRDYFLG is set high. This method can overrun the CPU unnecessarily. To avoid this, consider using either interrupt or DMA methods. To use the interrupt method, set the RXIEN bit. To use the DMA method, set the RXDREQEN bit. When the RXRDYFLG bit is set, it generates either an interrupt or a DMA request. If the checksum scheme is enabled by setting the compare checksum (COMPCHASUM) bit to 1, the checksum is compared for the current received byte, which is expected to be the checksum byte. Once the checksum is received, the COMPCHASUM bit is cleared. If a checksum error occurs, the CSEFLG flag is immediately set.

Receive Data in Multi-buffer Mode

When the MBUFEN bit is set to 1, multi-buffer mode is selected. In this mode, LIN sets the RXRDYFLG bit after receiving the programmed data and checksum fields, signifying a complete frame. Error condition detection logic is similar to single-buffer mode, except it monitors the entire frame. Similar to single-buffer mode, you can use polling, DMA, or interrupt methods to read the data. Received data must be read from the LINRD0 and LINRXB1 registers according to the number of bytes. For FLCFG values less than or equal to 4, reading from the LINRD0 register clears the RXRDYFLG flag. For FLCFG values greater than 4, reading from the LINRXB1 register clears the RXRDYFLG flag. If the checksum scheme is enabled by setting the COMPCHASUM bit to 1 during data reception, the byte received after the programmed number of data bytes indicated by the FLCFG field is considered the checksum byte. Once the checksum is completed, the COMPCHASUM bit is cleared.

40.7 Low-power mode

The UART/LIN module can be placed in local or global low power consumption mode. Global low power consumption mode is enabled by the system and is not controlled by the UART/LIN module. In global low power consumption mode, all clocks to the UART/LIN module are disabled, resulting in the module completely inactive. If the receiver requests global low power consumption mode while receiving data, the UART/LIN module completes the current reception before entering low power consumption mode, which occurs only after the Busy bit

(LINFLG.3) is cleared.

The LIN module can enter low power consumption mode if there is no activity on the LINRX pin for more than 4 seconds (which can be a constant recessive or dominant level) or if a sleep command frame is received. Once the Timeout flag (LINFLG.4) is set or a Sleep command is received, the application software must set the LPREQ bit to put the module into local low power consumption mode. A wake-up signal will terminate the sleep mode of the LIN bus.

When local low power consumption mode is enabled upon receive and transmit, if the wake-up interrupt is enabled and low power consumption mode is requested while the receiver is receiving data, the UART/LIN immediately generates a wake-up interrupt to clear the potential. As a result, the UART/LIN is prevented from entering low power consumption mode and completes the current reception. Otherwise, if the wake-up interrupt is disabled, the UART/LIN finishes the current reception and then enters low power consumption mode.

40.7.1 Sleep Mode

In the LIN protocol, use the Sleep command to put the bus in sleep mode. The Sleep command's diagnostic main request frame has the identifier 0x3C and the first data segment 0x00.

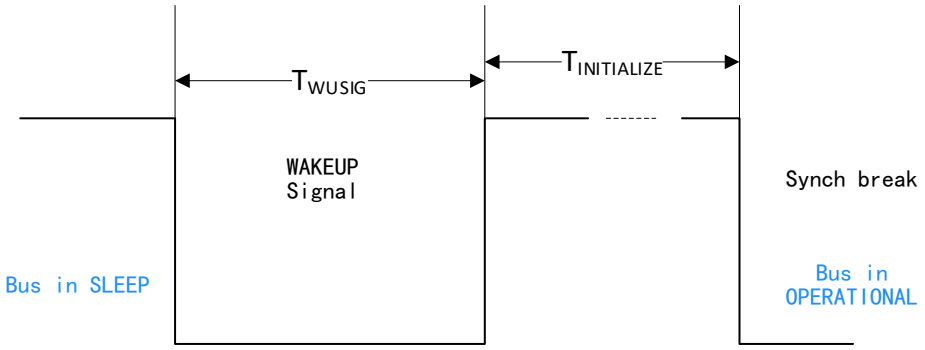
By setting the LINCTRL2 [LPREQ], UART/LIN module can oneself into local low power mode, the principle is to make the clock source to the module. However, in the case of a local power failure, the clock can enter the UART in a specific way, so all registers are accessible.

40.7.2 Wake-up

Enabling or disabling the wake interrupt determines whether the low level on the receiving pin will cause the module to exit low power mode automatically. If the device generates a request to enter low power mode and the wake interrupt is enabled, the wake can be triggered and the LPREQ bit cleared when the receiving pin detects low power.

In LIN mode, any node can terminate sleep mode by transmitting a wake-up signal, as shown in the figure below. A slave node detects that the bus is in sleep mode and that a wake-up request is pending, then transmits the wake-up signal. The wake-up signal is a dominant value (T_{WUSIG}) on the LIN bus, lasting at least 5 T_{bits} of the LIN bus baud rate. The wake-up signal is generated by transmitting a 0xF0 byte, consisting of 5 dominant T_{bits} and 5 recessive T_{bits}.

Figure 273 Generation of Wake-up Signal



Ideally (without noise and load), writing 0xF0 to the buffer would load to the transmitter, meeting TWUSIG's wake-up signal timing requirements. After placing WUPGEN at position 1, the device reads the value of the buffer and transmits a wake up signal.

Note that the WUPGEN bit can be set or reset only when the software is reset and the node is in power off mode. The condition of clearing the Wupgen bit is that a valid synchronization interruption is detected. The host transmits a wake up request, then exits power down mode after detecting a wake up pulse, and clears the WUPGEN bit so that the host does not transmit a wake up request again.

When the LIN transceiver receives a wake-up signal, it converts it into a dominant level or signal at the RX pin to the microcontroller, enabling a wake-up to the voltage regulator. When the LPREQ bit is set, if the LIN module detects a recessive-to-dominant transition (falling edge) on the RX pin and the LINIEN register is enabled, it will generate a wake-up interrupt.

A wake up request is defined as a level over 150ms detected by the LIN transceiver on the bus. The LIN slave is ready to listen to the bus within 100ms of the end of the wake signal.

40.7.3 Wake-up timeout

After the wake up signal does not wait for the response, the behavior of suspending the wake up request within 1.5s is defined by the LIN protocol as a timeout. The specific rules are as follows: After the wake up signal is sent, all nodes wait for the master node to transmit the header. If no synchronization signal is detected within 150ms of the wake up signal, the node that initiated the wake up request will re-transmit the wake up signal, and if three wake up attempts are unsuccessful, the request will not be renewed within 1.5s.

In order for LIN2.0 to be compatible with LIN1.3's timeout condition, the node triggering the wake signal can adjust the bit time base by setting the LINMBRPSC register to meet the target time: 128Tbits× programmed pre-divider.

The wake timeout handling of LIN protocol is implemented by hardware.

40.8 Simulation Mode

In emulation mode, when the program is suspended, the operation of the UART/LIN module is determined by the CONTSUS bit.

In debug mode, the CONTSUS bit is set or cleared, determining whether the counter continues to work or stops.

During emulation, the flag bit of the LINFLG register is not affected by the read operation.

40.9 UART Functional Description

40.9.1 Transmit and Receive Format

Since the communication format varies depending on the specific application, many properties of the UART/LIN are user-configurable. UART configuration options include:

- UART frame format
- UART timing mode
- UART baud rate
- UART multi-processor mode

40.9.2 Frame Format

UART uses a programmable frame format. All frames consist of the following:

- One start bit
- 1 to 8 data bits
- Zero or one address bit
- Zero or one parity bit
- One or two stop bits

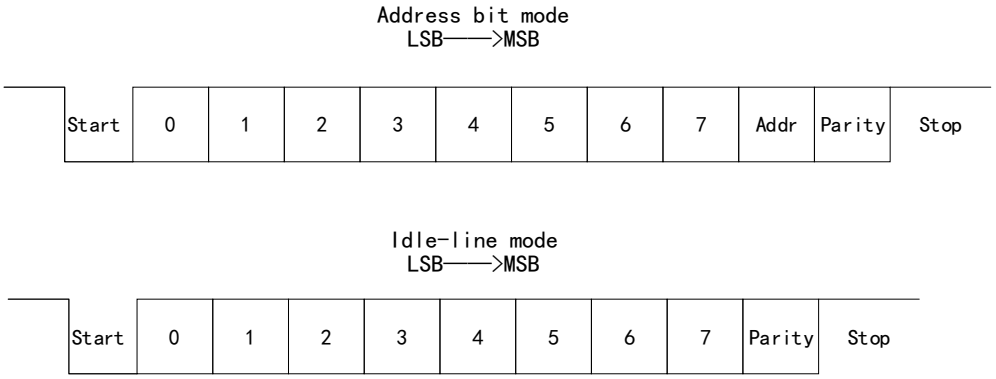
The sent and received data frames can be programmed with bits in the LINGCR1 register. The data is transmitted in a non-return-to-zero (NRZ) format, which means that in the idle state, the transmitting and receiving lines are logically high. The transmission of each frame begins with a start bit, at which point the transmitter pulls the UART line down (logically low). After the start bit, the least significant bit (LSB) of the data frame is sent and received first.

When the UART is set to address a pattern, each data frame contains an address; In idle line mode, the data frame has no address bits. The format of the frame with and without address bits is shown below.

If can make the parity bit, each data frame will have a parity bit. The value of this parity bit depends on the number of 1s in the frame, as well as the type of parity selected. The two examples shown below both set parity.

All data frame contains a stop bit, and at the end of the stop bit is always high level. A high level at the end of the frame is used to mark the end of the frame to ensure synchronization between communication devices. If a stop bit is configured in the LINGCR1 register, two stop bits are sent. Each frame in the example below uses a stop bit.

Figure 274 Typical UART Data Frame Format



40.9.3 Asynchronous Timing Format

The UART can be set to asynchronous timing mode by setting LINGCTRL1[1].

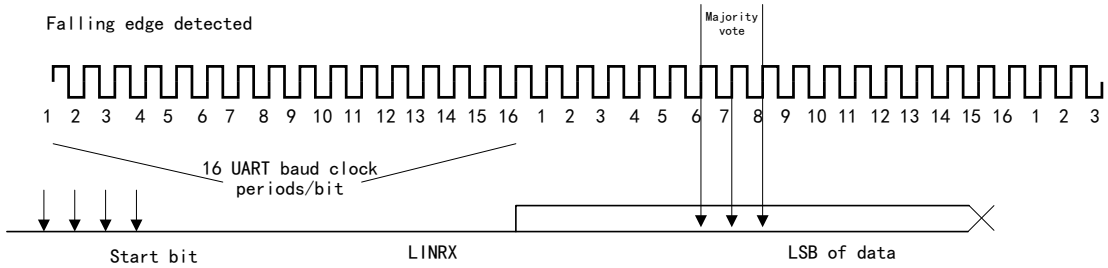
In asynchronous timing mode, the UART only needs to connect the receiving data line and the transmitting data line. Each clock cycle is sampled once, and each bit occupies 16 UART baud clock cycles in the frame, so each bit consists of 16 samples. If the baud rates of the devices involved in communication do not match, the receiving error is easy.

When the receiver receives the falling edge signal on the LINRX pin, the UART recognizes it as a valid starting bit if all four subsequent samples are at the logic level 0. Once a falling edge is detected, the UART assumes that the process of receiving the frame has begun and synchronizes itself with the bus.

LINRX is considered an effective start bit only if it remains low for at least four consecutive UART baud clock cycles, otherwise the bus is idle. The UART then samples the LINRX on cycles 7,8, and 9 to get the value of each bit, ultimately determining the value stored in the receive shift register. In this mode, UART effectively reduces errors due to transmission delays, rise and fall times, and noise through in-place intermediate sampling.

The transmitter also has 16 UART baud clock cycles on the duration of each bit. On the first clock cycle of a bit, the transmitter transmits the value of that bit to the LINTX pin and holds the value on the LINTX for the next 16 UART baud clock cycles.

Figure 275 Asynchronous Communication Bit Timing



40.9.4 Baud rate

UART/LIN will generate a serial clock inside, is decided by VALK and preassigned frequency coefficient P, M, in asynchronous timing mode, the UART according to the following formula to produce a potter clock:

$$\text{UARTCLK Frequency} = \frac{\text{VCLK Frequency}}{P + 1 + \frac{M}{16}}$$

$$\text{Asynchronous baud value} = \frac{\text{UARTCLK Frequency}}{16}$$

When P=0,

$$\text{Asynchronous baud value} = \frac{\text{VCLK Frequency}}{32}$$

Ultra-Fractional Frequency Division

In asynchronous mode (idle line and address bit mode), the UART can modulate finer baud rates using ultra-decimal dividers. A bit with a 1 in the table (normal configuration) adds an additional VCLK cycle to its Tbit. For characters longer than 10, then the tuning table is the original flipped version, see the (Maximum Configuration) table. In the field, baud rate will change according to the LINBR register values, d, U, TVCLK, P, M, etc factor decision. Calculation is as follows: 0 < d < 1 for P indicates a 0, and all the M and d (0 or 1), instantaneous TVCLK said a time:

$$T^{i}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right] T_{VCLK}$$

For P=0, T_{VCLK} represents the instantaneous bit time:

$$T_{bit} = 32\ T_{VCLK}$$

For P≠0, and for all M and d (0<d<1), TVCLK represents the average bit time:

$$T^{a}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right] T_{VCLK}$$

For P=0, T_{VCLK} represents the average bit time:

$T_{bit} = 32\ T_{VCLK}$

Table 246 Superfractional Bit Modulation in UART Mode (Standard Configuration)

Normal configuration = Start bit + 8 data bits + Stop bit										
LINBR[30:28]	Start bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Stop bit
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

Table 247 Superfractional Bit Modulation in UART Mode (Maximum Configuration)

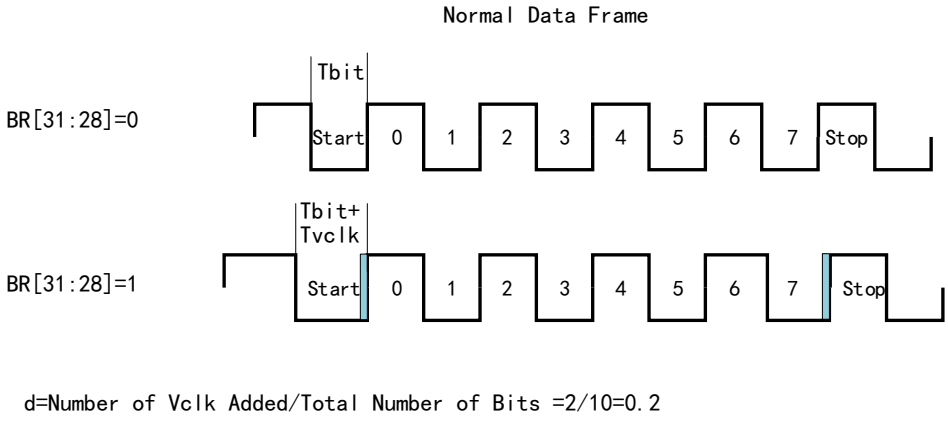
Maximum configuration = Start bit + 8 data bits + Address bit + Parity bit + Stop bit 0 + Stop bit 1													
LINBR [30:28]	Start bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Address Bit	Parity check bit	Stop bit 0	Stop bit 1
0h	0	0	0	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0	0	0	0
2h	1	0	0	0	1	0	0	0	1	0	0	0	1
3h	1	0	1	0	1	0	0	0	1	0	1	0	1
4h	1	0	1	0	1	0	1	0	1	0	1	0	1
5h	1	1	1	0	1	0	1	0	1	1	1	0	1
6h	1	1	1	0	1	1	1	0	1	1	1	0	1
7h	1	1	1	1	1	1	1	0	1	1	1	1	1

Table 248 Superfractional Bit Modulation in UART Mode (Minimum Configuration)

Minimum configuration = Start bit + 1 data bit + Stop bit			
LINBR[30:28]	Start bit	D[0]	Stop bit
0h	0	0	0
1h	1	0	0
2h	1	0	0
3h	1	0	1
4h	1	0	1

Minimum configuration = Start bit + 1 data bit + Stop bit			
5h	1	1	1
6h	1	1	1
7h	1	1	1

Figure 276 Example of Superfractional Division



d=Number of Vclk Added/Total Number of Bits =2/10=0.2

Table 249 Comparison of Baud Values for Different P Values, Asynchronous Mode

VCLK = 50 MHz				
24-Bit Register Value		Baud Rate Selection		Percentage Error
Decimal	Hexadecimal	Ideal Value	Actual Value	
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

40.9.5 Multiprocessor communication mode

UART can connect multiple serial communication devices for data transmission. In a multiprocessor environment, the UART can select idle line mode or address bit mode through registers, and several data frames can be sent to a single or all devices. When transmitting data to a single device, the receiving device needs to recognize when to address it. When the received message is not targeted at a device, the device can choose to ignore subsequent data. When

there are only two devices in the UART network, there is no need for addressing and therefore no need for complex multi-processor communication schemes.

If not in a multiprocessor environment, all frames can be considered data frames by software. At this point, there is a difference between the free line mode and the address bit mode, and the address bit protocol allows each frame to have an additional address bit.

Although the UART supports full-duplex communication, it does not arbitrate if the protocol used by the UART is transmitted by only one device at a time.

40.9.5.1 Idle-line Multiprocessor Mode

The difference between the address frame and the data frame is the number of idle cycles in front of the frame, the address frame has more than or equal to 10 idle bits, and the data frame has less than 10. The following figure shows the formats of the two frames.

There are two ways to transmit an address frame in idle line mode:

- Artificially, when programmed, an idle cycle is inserted between the last data frame of the previous block and the address frame of the new block. This method can only be implemented through a delay loop.
- Configure the UART to automatically transmit an idle cycle between the last data frame of the previous block and the address frame of the new block. This method can be implemented by using the transmit buffer and the TXWUPSEL bit. The steps are as follows:

(4) Write the TXWUPSEL bit to 1

(5) The virtual data values are written to the LINTXD register and the UART is triggered to start the idle cycle when the transmitter shift register is empty

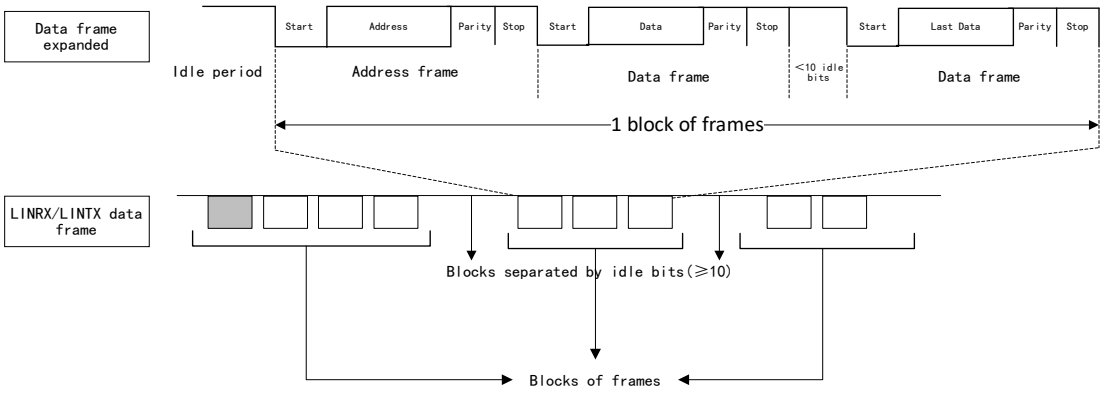
(6) UART Clears the TXWUPSEL flag

(7) Write the address value to LINTXD

In step 3, when the UART clears the TXWUPSEL bit, if TX interrupt is enabled after TXRDTFLG is set, the transmission from LINTXD to UARTTXSHF will be interrupted. Therefore, do not use software to poll the TXWUPSEL bit waiting for UART to clear.

When using idle line multiprocessor communication, there must be more than 10 idle cycles before the address frame, and in order to avoid a delay of 10 cycles between frames, the data frame needs to be written to the transmitter quickly. Otherwise, an error may occur on the device.

Figure 277 Idle-line Multiprocessor Communication Format

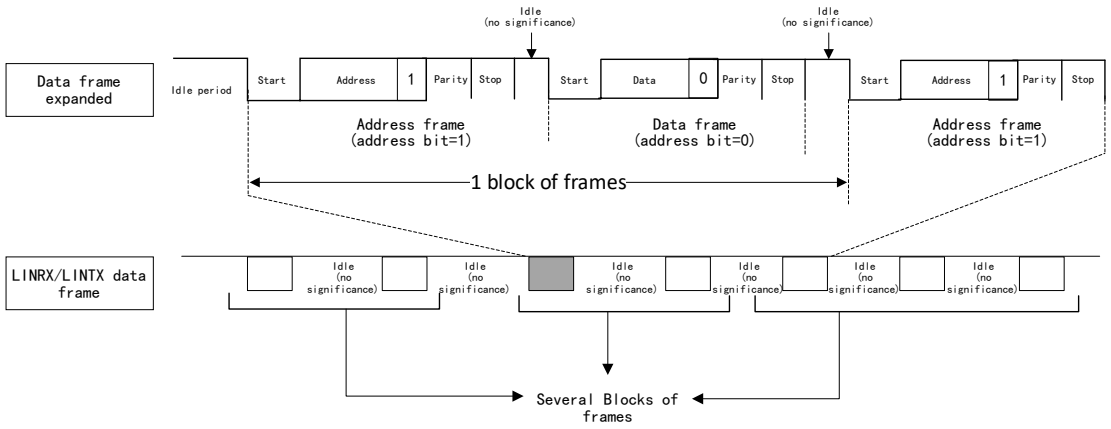


40.9.5.2 Address Bit Multiprocessor Mode

In the address bit protocol, each frame has an address bit after the data segment, the value of the address bit is used to determine whether the frame is an address frame or a data frame, if it is 1, then the frame is called an address frame, if it is 0, then it is called a data frame. In this mode, the timing of idle time becomes unimportant.

Address bits mode frame format as shown in the figure below:

Figure 278 Address Bit Multiprocessor Communication Format



In address bit mode, there is no need to perform virtual write operations on LINTXD in advance. After the TXWUPSEL is written to 1 by the software, it is sent as an address bit. When the contents of LINTXD are shifted from the TXWUPSEL register, this bit is cleared, so any frame other than the address bit (TXWUPSEL) can be used as a data frame.

40.9.6 Transmit data

There are two modes for UART data transmission: single-buffer mode and multi-buffer mode. The mode is selected by configuring the LIN_GCTRL1[MBUFEN] bit.

Transmission is controlled by the LIN_PCTRL0[TXPEN] and LIN_GCTRL1[TXEN] bits. The TXPEN bit enables the transmit pin; the LINTX pin can be used as a UART pin only when TXPEN is set to 1. TXEN enables transmission; the buffered value is not transmitted until this bit is set to 1. When both bits are 0, the transmitter operates independently of the receiver.

Transmit Data in Single Buffer Mode

When the transmitter is ready to write data to TXD, the TXBRDYFLG bit is set. After writing, the data is transmitted to TXSHF and then sent. When both TXD and TXSHF are empty, the TXEFLG bit is set.

There are several methods to transmit data:

(1) Polling: The CPU continuously accesses the TXBRDYFLG bit before writing data to TXD, causing operating overrun.

(2) Receive Interrupt: Set the TXIEN bit to enable interrupts. When all pending frames are transmitted, both TXD and TXSHF are empty, and TXBRDYFLG is set, generating an interrupt. After transmission is complete, disable the interrupt by setting the TXICLR bit.

(3) DMA: Set the TXDREQEN bit to enable DMA requests. When all pending frames are transmitted, both TXD and TXSHF are empty, and TXBRDYFLG is set, generating a DMA request. After transmission is complete, disable the transmitter by clearing the TXEN bit.

Note: The TXBRDYFLG bit cannot be cleared through the LINIVOx register.

Transmit Data in Multi-buffer Mode

Similar to single-buffer mode, data transmission in multi-buffer mode follows the same process. However, data must be written to the TXD register. While the UART waits for data to be written, the number of bytes is transmitted to TXSHF one by one.

40.9.7 Receive data

There are two modes for UART data receive: single-buffer mode and multi-buffer mode. The mode is selected by configuring the LIN_GCTRL1[MBUFEN] bit.

Receive is controlled by the LIN_PCTRL0[RXPEN] and LIN_GCTRL1[RXEN] bits. The RXPEN bit enables the transmit pin; the LINRX pin can be used as a UART pin only when RXPEN is set to 1. RXEN enables receive; the buffered value is not transmitted until this bit is set to 1.

Unlike the transmit pin, the receive pin can automatically receive data when it detects a valid idle time and data arrives at the pin.

Receive Data in Single Buffer Mode

When RXD writes received data into RXD, the RXBRDYFLG bit is set. During the transmission process, if there are frame errors, overflow errors, or parity errors, the corresponding flag bits in the LIN_FLG register will be set. The flag bits are cleared after the data in RXD is read.

When frame errors, overflow errors, or parity errors occur, the WUPFLG and BRKDETFLG flags are also set, although these do not necessarily occur while data is being written to RXD.

There are several methods to transmit data:

(1) Polling: The CPU continuously accesses the RXBRDYFLG bit before writing data to RXD, causing operating overrun.

(2) Receive Interrupt: Set the RXIEN bit to enable interrupts. After transmission is complete, disable the interrupt by setting the RXICLR bit.

(3) DMA: Set the RXDREQEN bit to enable DMA requests.

Receive Data in Multi-buffer Mode

Similar to single-buffer mode, data transmission in multi-buffer mode follows the same method. However, the error detection logic does not include a complete frame.

40.9.8 Multi-buffer Mode

UART reduces CPU load with multiple buffers (configure MBUFEN). The UART has 8 separate buffers for both transmitting and receiving.

Have a 3 - bit counter calculation to the buffer or buffer to registers bytes of data through LINLCFG [FLCFG] loading comparator contains the expected number of bytes sent or received.

After receiving an error-free response, receive interrupts can be enabled through LINIVOx registers, RXRDYFLG can be setting by sign the register, receive DMA requests can be generated.

After transmit a response to transmit interrupts can be enabled, RXRDYFLG can be setting by sign the register, transmitting DMA requests can be generated.

The function block diagrams for receive and transmit multi-buffer operations are shown in the figure below.

Figure 279 Receive Buffer

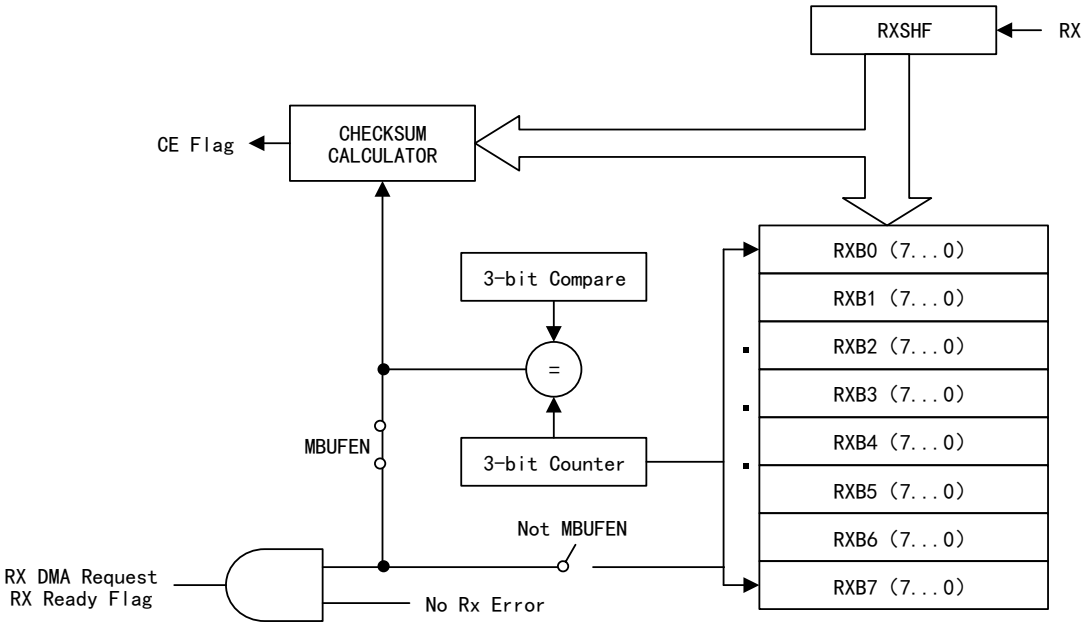
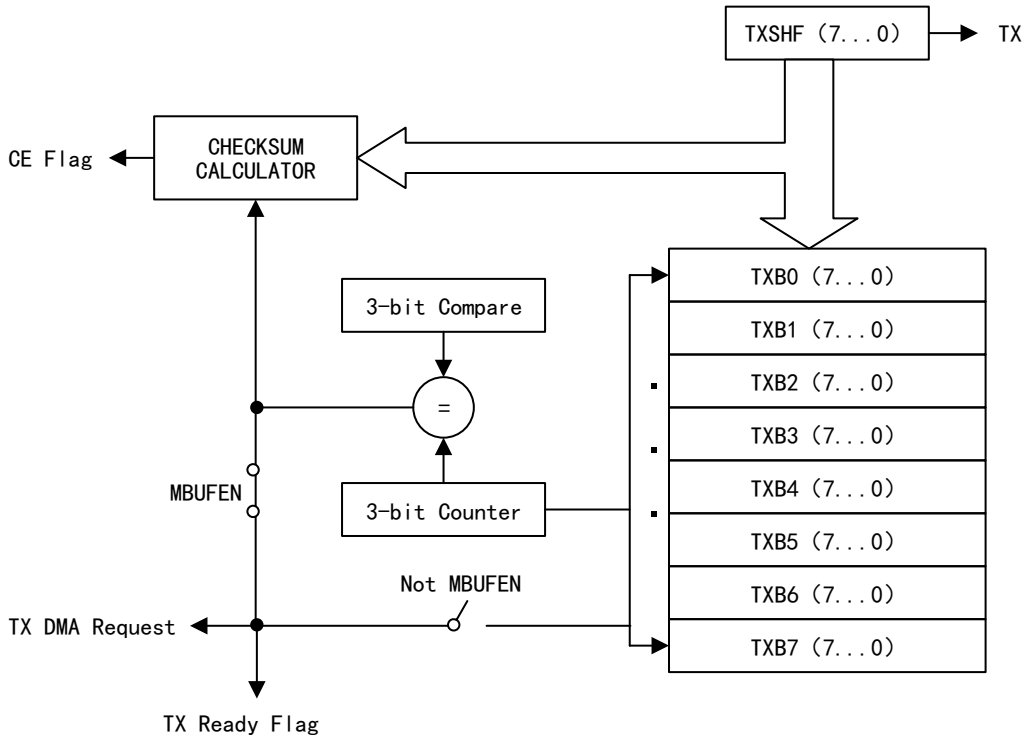


Figure 280 Transmit Buffer



40.9.9 Interrupt

UART has two interrupt lines, INT0 and INT1, which can generate interrupts with configurable priority. They are managed through the vector interrupt manager. The management structure block diagram is shown in the figure below.

Figure 281 Interrupt Management Structure Block Diagram

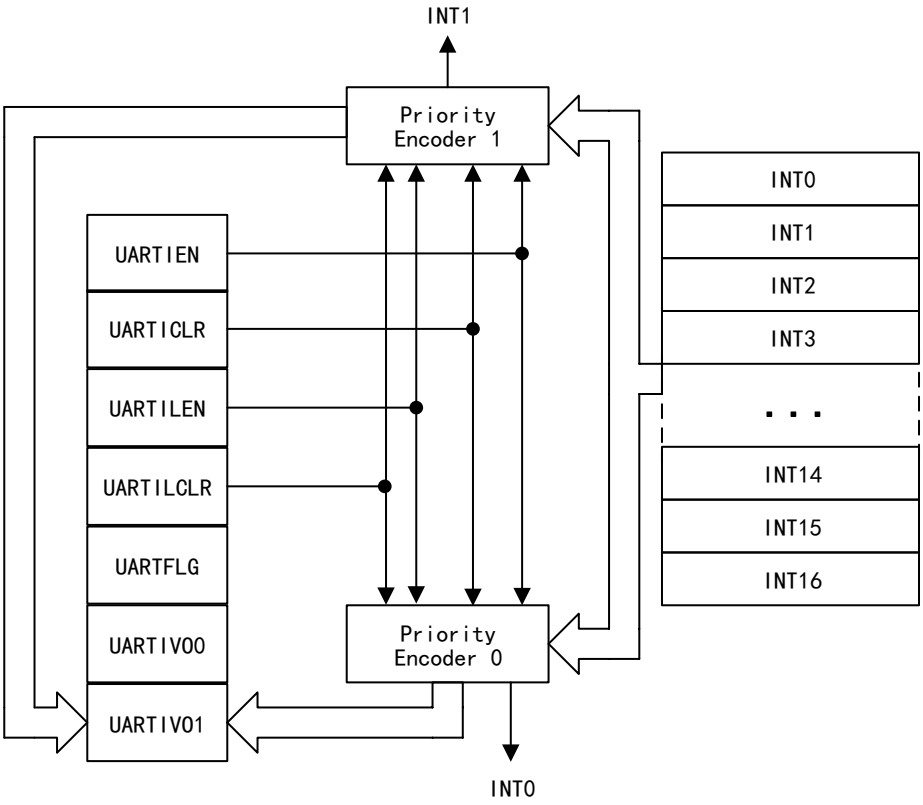
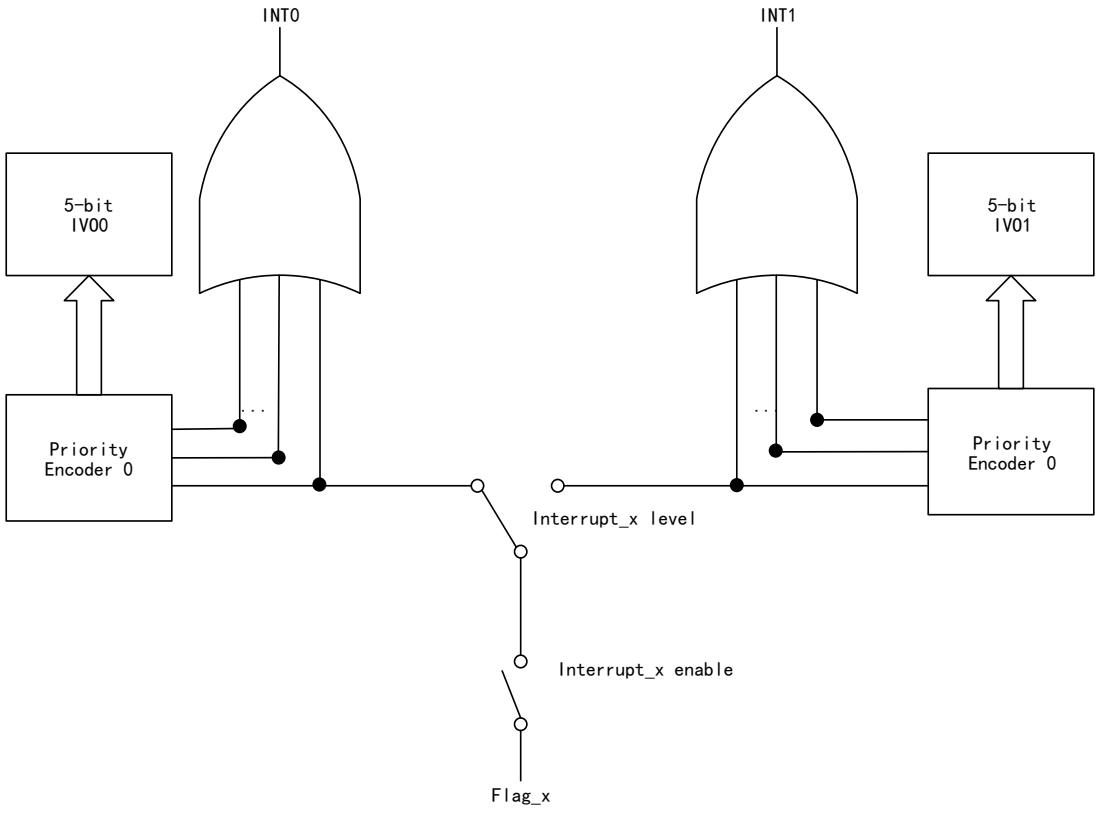


Figure 282 Generating Interrupts for Specific Flags



The LINIVOx registers determine the triggered interrupt source based on their respective priority encoders. The interrupt vector register returns the vector for the pending interrupt line. If multiple interrupts are pending, the register only retains the interrupt with the highest priority, ensuring the system processes interrupts in descending priority order.

Interrupts can be enabled and disabled by configuring the LINIEN and LINICLR registers.

Each interrupt level defaults to 0. The LINILEN register can be used to set it to level 1, or the LINILCLR register can be used to set it back to level 0.

Transmit interrupt

The transmission interrupt function can only be used when TXIEN=1 and TXDREQEN=0.

As described in the “Transmit Data” section, a transmission interrupt is only generated after the first transfer from TXD to TXSHF, once TXIEN and TXBRDYFLG are set. Thus, before generating any interrupts, the first data must be written to TXD. In the transmission interrupt service routine, writing data to TXD can transmit more data.

When using interrupts for data transmission, writing data to TXD clears the TXBRDYFLG bit, and writing data to TXSHF will set the TXBRDYFLG bit again. During this process, interrupt requests can be disabled by setting the TXICLR bit. However, when TXIEN is re-enabled, an interrupt is generated again, and the TXBRDYFLG bit is set.

To clear the interrupt request, the following method can be used:

- Clear TXEN bit
- Software Reset
- Hardware reset

When the next data is written to TXD, the interrupt will be triggered again.

Receive interrupt

When data is transferred from RXSHF to RXD, the RXBRDYFLG bit is set, allowing the UART to read the received data. Setting the RXIEN bit enables receive interrupts; if RXBRDYFLG is also set, a receive interrupt is generated.

The received data can be read within the ISR.

When UART and DMA coexist in a single device, SETRXDMA must be cleared in order to use interrupts.

Wake-up interrupt

If bus activity on the RX line prevents entering power-down mode or causes exit

from power-down mode, the UART sets the WUPFLG flag. If the WUPIEN is enabled, a wake-up interrupt is triggered when the WUPFLG flag is set.

Error interrupt

The following error detections are supported by the UART module interrupt:

- Parity error (PEFLG)
- Bit error (BEFLG)
- Frame error (FEFLG)
- Overflow error (OEFLG)
- Interrupt detection error (BRKDETFLG)

If no error is detected, the message is considered valid by both the sending and receiving devices.

In the UART/LIN module of the communist party of China has 16 interrupt source, and 8 in UART interrupt, shown in the following table.

Table 250 UART/LIN Interrupts

Offset/Priority	Interrupt	Applicable to UART	Applicable to LIN
0	No interrupt	-	-
1	Wake-up	Yes	Yes
2	Out of sync field error	No	Yes
3	Parity check error	Yes	Yes
4	ID	No	Yes
5	Physical bus error	No	Yes
6	Frame error	Yes	Yes
7	Interrupt signal detection	Yes	No
8	Checksum error	No	Yes
9	Overspeed error	Yes	Yes
10	Byte error (BEFLG)	Yes	Yes
11	Receive	Yes	Yes
12	Transmit	Yes	Yes
13	No-response error	No	Yes
14	Timeout after wake-up signal (150ms)	No	Yes
15	Timeout after 3 wake-up signals (1.5s)	No	Yes
16	Timeout (bus idle, 4s)	No	Yes

Table 251 UART Receive Flag Bits

UART flag	Register	Bit	Reset value
CSEFLG	LINFLG	29	0
ISFEFLG	LINFLG	28	0
NREFLG	LINFLG	27	0
FEFLG	LINFLG	26	0
OEFLG	LINFLG	25	0
PEFLG	LINFLG	24	0
RXWUPSEL	LINFLG	12	0
RXRDYFLG	LINFLG	9	0
BUSYFLG	LINFLG	3	0
RXIDLEFLG	LINFLG	2	1
WUPFLG	LINFLG	1	0
BRKDETFLG	LINFLG	0	0

When RDY = 0, these flag bits are frozen, and their reset values are all 0.

Table 252 UART Transmit Flag Bits

UART flag	Register	Bit	Reset value
BEFLG	LINFLG	31	0
PBEFLG	LINFLG	30	0
TXWUPSEL	LINFLG	10	0
TXEFLG	LINFLG	11	1
TXBRDYFLG	LINFLG	8	1

When RDY = 0, these flag bits are frozen, and their reset values are all 0.

40.9.10 DMA

Both the UART and LIN modules can use DMA requests (transmit and receive), and the DMA transfer mode depends on whether multiple buffering mode is used. The DMA request is determined by the CPU, by setting the TXDREQEN/TXDREQCLR can make or disable transmits the request; Enable or disable receiving requests by setting the RXDREQEN/RXDREQCLR bit.

Transmit DMA request

In multi-buffered mode, a TX DMA request is generated after sending the data that is ready or set to be transferred, and the device loads the data for the next transfer into the buffer. If multi-buffering mode is disabled, DMA requests are generated in bytes.

Receive DMA request

In multi-buffered mode, the receiver loads the received characters into the buffer. When the last character set is received and loaded into the buffer, an RX DMA request is generated. If multi-buffering mode is disabled, DMA requests are generated in bytes.

By setting the ADRXDREQEN bit in multiprocessor mode with the SLEEP bit disabled, UART can generate DMA requests for address and data frames, or generate receive interrupts for address frames and DMA requests for data frames. If the SLEEP bit is set, the UART does not generate a DMA request for the received data frame.

40.9.11 Low-power mode of UART

In global low power mode, when the entire system is in low power mode, the clock of the UART/LIN module is turned off, so the two modules will not operate.

By setting the LINCTRL2[LPREQ] bit, the UART/LIN module can enter the local low power mode by itself, by making the clock source not act on the module. However, in the case of a local power failure, the clock can enter the UART in a specific way, so all registers are accessible.

The low power mode can be exited by clearing the LPREQ bit. When LINRX is detected to be at low power level, the wake up interrupt enables the UART to automatically clear the LPREQ bit. If the wake up interrupt is disabled, UART/LIN immediately goes into low power mode when the request is generated, and no activity on the LINRX pin will cause the module to exit low power mode.

In the process of receiving data, if the device generates a request to enter the low power mode:

(1) If the wake up interrupt is enabled, the UART will immediately generate an interrupt, clear the LPREQ bit, will not enter low power mode, and will complete receiving

(2) If the wake up interrupt is disabled, the UART will enter low power mode after receiving

Multiprocessor mode in sleep mode

When the UART receives data, the data is loaded from the shift register into LINRXD, and the receive ready flag bit is set to generate a receive interrupt (if the interrupt is enabled), at which time the CPU reads a new frame before the receive is complete. In multiprocessor communication, this behavior becomes efficient to provide a selective indication of new data.

Sleep mode is suitable for idle line mode and address bit multiprocessor mode. The sleep mode allows the UART to ignore the data after the unmatched

address frame and not receive data until the correct address frame is matched.

In sleep mode, since the UART has no hardware support, the software must read the buffer and compare the address frame with the address already stored in memory to determine whether the UART is addressed.

If not addressed:

(1) The UART has been configured to enable sleep mode and receive

(2) The receiver receives an address frame and generates a receive interrupt

(3) The software compares the received address frame with the address already stored in memory to determine that the UART is not addressed and that the value of the SLEEP bit is not changed

(4) Data frames are loaded into UARTRXSHF for assembly, are not transferred to LINRXD, and no receive interruption is generated

If confirmed to be addressed:

(1) The UART has been configured to enable sleep mode and receive

(2) A new address frame is received and a receive interrupt is generated

(3) The software compares the received address frame with the address already stored in memory, determines that the UART is being addressed, and clears the SLEEP bit

(4) Data transferred to UARTRXSHF is loaded into LINRXD, which generates a receive interrupt after each data frame is received

(5) In each interrupt routine, the software checks the RXWUPSEL to determine if the current frame is an address frame

(6) Receiving another address frame, set RXWUPSEL, the software determines that UART is not addressed, and sets the SLEEP bit to 1. Data frames after this address frame do not generate receive interrupts

As can be seen from the above example, using sleep mode to ignore invalid data frames can reduce the generation of interrupts, which can free up more CPU resources.

When the UART is in hibernation, all the flags except the ready flag are kept updated so that the program can detect errors in time and take corrective action. The UART uses polling to query the receive readiness flag when needed. Once set, the software will receive a new address and compare it. If the UART is not addressed, the software will continue polling until the UART is addressed, the software changes the SLEEP bit, and then receives the data.

40.10 Register bank address

Table 253 LIN Register Bank Address

Device register	Register bank	Start address	End address
LINA	LIN_REGS	0x4000_0400	0x4000_07FF

40.11 Register address mapping

Table 254 LIN_REGS Register Address Mapping

Register name	Register description	Offset address	WRPRT
LINGCTRL0	Global control register 0	0x00	-
LINGCTRL1	Global control register 1	0x04	-
LINGCTRL2	Global control register 2	0x08	-
LINIEN	Enable interrupt register	0x0C	-
LINICLR	Clear interrupt register	0x10	-
LINILEN	Enable interrupt level register	0x14	-
LINILCLR	Clear interrupt level register	0x18	-
LINFLG	Flag register	0x1C	-
LINIVO0	Interrupt vector offset register 0	0x20	-
LINIVO1	Interrupt vector offset register 1	0x24	-
LINLCFG	Configure length register	0x28	-
LINBR	Baud rate register	0x2C	-
LINRXED	Receiver simulation data register	0x30	-
LINRXD	Receive data buffer register	0x34	-
LINTXD	Transmit data buffer register	0x38	-
LINPCTRL0	Control pin register 0	0x3C	-
LINPCTRL2	Control pin register 2	0x44	-
LINCOMP	Compare register	0x60	-
LINRXB0	Receive buffer register 0	0x64	-
LINRXB1	Receive buffer register 1	0x68	-
LINIDMASK	Mask ID register	0x6C	-
LINID	ID register	0x70	-
LINTXB0	Transmit buffer register 0	0x74	-
LINTXB1	Transmit buffer register 1	0x78	-
LINMBRPSC	Maximum baud rate prescaler register	0x7C	-

Register name	Register description	Offset address	WRPRT
LINEET	Analog Error and test register	0x90	-
LINGIEN	Global interrupt register	0xE0	-
LINGIFLG	Global interrupt flag register	0xE4	-
LINGICLR	Clear global interrupt flag register	0xE8	-

40.12 Register functional description

40.12.1 Global Control Register 1 (LINGCTRL0)

Offset address: 0x00

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	RST	R/W	USART/LIN Reset 0: Disable 1: Enable	0h
31:1	Reserved			0h

40.12.2 Global control register 1 (LINGCTRL1)

Offset address: 0x04

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	COMCFG	R/W	USART/LIN Communication Mode Configure In compatibility mode, it selects the UART communication mode. In LIN mode, it selects the length control option for ID field bits ID4 and ID5. 0: UART compatibility mode: Use idle line mode. LIN mode: ID4 and ID5 are not used for length control. 1: UART compatibility mode: Use address bit mode. LIN mode: ID4 and ID5 are used for length control.	0h
1	TIM	R/W	USART Timing Mode This bit is valid only in UART compatibility mode. When using UART mode, this bit must be set to 1. This bit configures the UART for asynchronous operation. 0: Reserved. 1: When the module is configured for UART operation, this must be set to 1.	0h
2	PARITYEN	R/W	USART/LIN Parity Enable Enable or disable the parity function. 0: UART compatibility mode: parity bit is disabled; no parity bit is generated during transmission, nor are expected during reception. LIN mode: ID parity bit verification is disabled. 1: UART-compatible mode: Parity bits are enabled; a parity bit is generated during transmission and expected during reception.	0h

Field	Name	R/W	Description	Reset value
			LIN mode: ID parity bit verification is enabled.	
3	PARITY	R/W	USART Odd/Even Parity Select This bit is valid only in UART compatibility mode (write). If the PARITY ENA bit (UARTGCR1.2) is set, PARITY specifies odd or even parity. The parity bit is calculated based on the data bits and address bits (in address bit mode) in each frame. The start bit and stop bits in the frame are not included in the parity calculation. 0: Use odd parity. The UART transmits and expects to receive a parity bit value such that the total number of bits with a value of 1 in the frame is odd. 1: Use even parity. The UART transmits and expects to receive a parity bit value such that the total number of bits with a value of 1 in the frame is even.	0h
4	STOPCFG	R/W	USART STOP Bit Configure Valid only in UART compatibility mode (write). Note: The receiver checks only one stop bit. However, in idle line mode, if STOP = 1, the receiver waits until the end of the second stop bit before starting to check idle time. 0: Use one stop bit. 1: Use two stop bits.	0h
5	CLK_MSCFG	R/W	USART Internal Clock Enable/LIN Master Slave Configure In UART mode, this bit enables the clock for the UART module. In LIN mode, this bit determines whether the LIN node is a slave or a master. 0: UART compatibility mode: Reserved. LIN mode: The module is in slave mode. 1: UART compatibility mode: Enable the clock for the UART module. LIN mode: The node is in master mode.	0h
6	LINEN	R/W	LIN Mode Enable If LIN mode is enabled, UART compatibility mode is disabled; conversely, if UART compatibility mode is enabled, LIN mode is disabled. 0: Enable LIN mode 1: Disable LIN mode	0h
7	RDY	R/W	USART/LIN Ready This bit is valid in both LIN and UART compatibility modes. UART/LIN can only be configured when RDY = 0. Only the following configuration bits can be modified at runtime (i.e., when RDY = 1): Stop extended frame (UARTGCR1[13]) COMPCHASUM bit SC bit 0: UART/LIN is in reset state, and no data is transmitted or received. Writing this bit to 0 initializes the UART/LIN	0h

Field	Name	R/W	Description	Reset value
			state machine and operation flags. All affected logic remains in the reset state until this bit is written to 1. 1: UART/LIN is in ready state and can transmit and receive data. Once this bit is set to 1, the configuration of the module shall not be changed.	
8	SLEEPEN	R/W	USART Sleep Mode Enable Valid only in UART compatibility mode (write). This bit controls the receive sleep function in a multiprocessor configuration. Clearing this bit causes the UART to exit sleep mode. When the SLEEP bit is set, the receiver remains operational, but RXRDYFLG will only update and LINRXD will only load new data upon detecting an address frame. Other receiver status flags will be updated, and if the corresponding interrupt enable bit is set, an error interrupt will be requested regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the UART is in sleep mode, the software can handle the error situation promptly. The SLEEP bit is not automatically cleared when an address byte is detected. 0: Disable 1: Enable	0h
9	ABEN	R/W	LIN Automatic Baud Rate Adjustment Enable This mode is valid only in LIN mode. This bit functions during the detection of the synchronization field. According to the node capability file definition, this bit enables two LIN protocol bit rate modes: automatic or selective. The software and network configuration will determine which of the two modes is used. When this bit is cleared, the fixed bit rate of the LIN 2.0 protocol shall be used. If the ADAPT bit is set, the LIN slave node detecting the baud rate will compare it with the frequency divider in the BRSR register and update it if they differ. The BRSR register will be updated with the new value. If this bit is not set, the BRSR register will not be adjusted. This field is writable only in LIN mode 0: Disable 1: Enable	0h
10	MBUFEN	R/W	USART/LIN Multi-buffer Mode Enable This bit is valid in both LIN and UART compatibility modes. This bit controls the use of the receive/transmit buffer, determining whether to use the RX/TX multi-buffer or a single RD0/TXB0 register. 0: Disable 1: Enable	0h
11	CHASUMCFG	R/W	LIN Checksum Configure This bit is valid only in LIN mode. This bit controls the type of checksum to be used: classic or enhanced. 0: Use classic checksum.	0h

Field	Name	R/W	Description	Reset value
			This checksum is compatible with LIN 1.3 slave nodes. The classic checksum is calculated as the modulo-256 sum of all data bytes. Frames transmitted with identifiers 60 (0x3C) to 63 (0x3F) must always use the classic checksum. 1: Use enhanced checksum. The enhanced checksum is compatible with LIN 2.0 and newer slave nodes. The enhanced checksum includes the modulo-256 sum of all data bytes and the protected identifier.	
12	HGENCTRL	R/W	LIN HGEN Control This bit is valid only in LIN mode. This bit controls the type of mask filtering comparison. 0: Use ID-Byte for ID filtering. The RECEIVEDID and IDBYTE fields in the LINID register are used to detect matches (using TX/RXMASK values). A mask of 0xFF in the LINMASK register will result in no match. 1h (read/write) = Use ID-SLAVETask byte for ID filtering (recommended). The RECEIVEDID and IDSLAVETASKBYTE fields in the LINID register are used to detect matches (using TX/RXMASK values). A mask of 0xFF in the LINMASK register will result in always matching.	0h
13	STOEXTCOM	R/W	Stop LIN Extended Frame Communication This bit is valid only in LIN mode. This bit can only be written during extended frame communication. When extended frame communication stops, this bit is automatically cleared. 0: No effect 1: Extended frame communication will stop once the transmit/receive of the current frame is complete.	0h
15:14	Reserved			0h
16	LBEN	R/W	USART/LIN Loopback Mode Enable This bit is valid in both LIN and UART compatibility modes. The self-inspection option for UART/LIN can be selected using this bit. If the LINTX and LINRX pins are configured for UART/LIN function, the LINTX pin is internally connected to the LINRX pin. Externally, during loopback operation, the LINTX pin outputs a high level, and the LINRX pin is in a high-impedance state. Changing the value of this bit while UART/LIN is transmitting or receiving data may cause errors. 0: Disable 1: Enable	0h
17	CONTSUS	R/W	USART/LIN Continue Work on Suspend This bit takes effect only when using a emulator debugging program and determines how UART/LIN operates when the program is paused. This bit affects the LIN counter. When this bit is set, the counter does	0h

Field	Name	R/W	Description	Reset value
			not stop during debugging. When this bit is cleared, the counter stops during debugging. 0: When entering debug mode, the UART/LIN state machine is frozen. Transmission and the LIN counter are stopped and resume upon exiting debug mode. 1: When entering debug mode, the UART/LIN continues operating until the current transmit and receive functions are complete.	
23:18	Reserved			0h
24	RXEN	R/W	USART/LIN Receive Enable This bit controls whether the receiver is allowed to transmit data from the shift buffer to the receive buffer or multi-buffer. It is valid in both LIN and UART compatibility modes. Note: (1) Clearing RXENA prevents received characters from being transmitted to the receive buffer or multi-buffer, prevents received data from updating RX status flags (see Table 7), and disables receive and error interrupts. However, the shift register continues to assemble data, regardless of the RXENA state. (2) If RXENA is cleared before the frame receive is complete, the data in the frame will not be transmitted to the receive buffer. (3) If RXENA is set before the frame receive is complete, the data in the frame will be transmitted to the receive buffer. If RXENA is set while UARTRXSHF is assembling a frame, the status flags for that frame cannot be guaranteed to be accurate. To ensure that the status flags accurately reflect what is detected on the bus during a specific frame, RXENA shall be set before the frame is detected. 0: Not allowed 1: Allowed	0h
25	TXEN	R/W	USART/LIN Transmit Enable This field is valid in LIN and UART modes. Data is transmitted from the UARTTXB or TXBy (where y = 0, 1, ..., 7) buffer to the UART_TXSHF shift register in LIN mode only when the TXENA bit is set. Note: Data written before TXENA is set will not be transmitted to UARTTXB or the transmit multi-buffer. If TXENA is cleared during transmission, the data previously written to UARTTXB will be transmitted (including the checksum byte in LIN mode). 0: Disable data transmission from UARTTXB or TXBy to UARTTXSHF 1: Enable data transmission from UARTTXB or TXBy to UARTTXSHF	0h
31:26	Reserved			0h

40.12.3 Global control register 2 (LINGCTRL2)

Offset address: 0x08

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	LPREQ	R/W	Low Power Consumption Mode Request This bit is valid in both LIN and UART compatibility modes. When the power-down bit is set, the UART/LIN module attempts to enter local low power consumption mode. If the LPREQ bit is set while the receiver is actively receiving data and the wake-up interrupt is disabled, the UART/LIN will delay entering low power consumption mode until reception is complete. In LIN mode, the user can set the LPREQ bit upon receiving a sleep command or detecting a bus idle condition (exceeding 4 seconds, equivalent to 80,000 cycles at 20 kHz). 0: Normal operation 1: Request local low power consumption mode.	0h
7:1	Reserved			0h
8	WUPGEN	R/W	Generate Wakeup Signal This bit controls the generation of a wakeup signal by transmitting the value in the TXB0 buffer. This bit will be cleared upon receiving a valid synchronization interrupt. 0: Invalid 1: Wake up for Transmit TXB0. This bit will be cleared when RDY is set.	0h
15:9	Reserved			0h
16	TXCHAEN	R/W	Transmit Checksum Byte Enable This mode is valid only in LIN mode (write). This bit is used by the transmitter to transmit a checksum byte along with an extended frame. In non-multi-buffer mode, the checksum byte is transmitted after the current byte is transmitted. In multi-buffer mode, the checksum byte is transmitted after the last byte count (indicated by LINLCFG[18:16]). 0: Do not transmit a checksum byte. 1: A checksum byte will be transmitted. After transmitting the checksum byte, this bit will be automatically cleared. If this bit is set before the first byte is transmitted (i.e., during the inter-frame gap), the checksum will not be transmitted.	0h
17	COMPCHASUM	R/W	Compare Checksum This mode is valid only in LIN mode. This bit is used by the receiver to trigger a checksum comparison in an extended frame. The user initiates this operation by writing 1 to this bit. In non-multi-buffer mode, once the COMPCHASUM bit is set, the checksum is compared on the byte currently being received, which is expected to be the checksum byte. In multi-buffer mode, the following applies to the COMPCHASUM bit:	0h

Field	Name	R/W	Description	Reset value
			If the COMPCHASUM bit is set while receiving data, the byte received after the programmed number of data bytes, indicated by LINLCFG[18:16], will be treated as the checksum byte. If the COMPCHASUM bit is set during idle (i.e., during the inter-frame gap), the next immediate byte will be treated as the checksum byte. If a checksum error occurs, it will be immediately marked as CSEFLG. Once the checksum is successfully compared, this bit will be automatically cleared. 0: Invalid 1: Compare the checksum on the expected checksum byte	
31:18	Reserved			0h

40.12.4 Enable interrupt register (LINIEN)

Offset address: 0x0C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	BRKDETIEN	R/W	Break Detect Interrupt Enable This bit is valid only in UART compatibility mode (write). Setting this bit causes the UART/LIN to generate an interrupt when a interrupt condition is detected on the LINRX pin. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
1	WUPIEN	R/W	Wake up Interrupt Enable This bit is valid in both LIN and UART compatibility modes. Setting this bit causes the UART/LIN to generate a wakeup interrupt, exiting low power consumption mode. The wakeup interrupt is asserted on the falling edge of the wakeup pulse. When enabled, a wake-up interrupt is asserted if a local low power consumption mode request occurs while the receiver is busy or if a low level is detected on the UARTRX pin during power consumption mode. If the module is not in power-down mode, a wakeup interrupt will not be asserted during the wakeup pulse. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
3:2	Reserved			0h
4	TOIEN	R/W	Timeout Interrupt Enable This bit is valid only in LIN mode (write). Setting this bit causes the UART/LIN to generate an interrupt when there is no LIN bus activity (bus idle) for at least 4 seconds. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
5	Reserved			0h

Field	Name	R/W	Description	Reset value
6	TOWUPSIEN	R/W	Timeout after Wakeup Signal Interrupt Enable This bit is valid only in LIN mode (write). Setting this bit causes the UART/LIN to generate an interrupt upon a timeout after transmitting a wakeup signal. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
7	TO3WUPSIEN	R/W	Timeout after 3 Wakeup Signal Interrupt Enable This bit is valid only in LIN mode (write). Setting this bit causes the UART/LIN to generate an interrupt upon a timeout after transmitting a wakeup signal. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
8	TXIEN	R/W	Transmitter Interrupt Enable Setting this bit allows the UART/LIN to generate a transmit interrupt when data is transmitted from UARTTXB to UARTTXSHF and the TXBRDYFLG bit is set. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
9	RXIEN	R/W	Receiver Interrupt Enable Setting this bit allows the UART/LIN to generate a receive interrupt when a frame is fully received and data is transmitted from UARTRXSHF to LINRXD. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
12:10	Reserved			0h
13	IDIEN	R/W	Identification Interrupt Enable This bit is valid only in LIN mode. When this bit is set, an interrupt is generated upon receiving a valid matching identifier. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
15:14	Reserved			0h
16	TXDREQEN	R/W	Transmit DMA Request Enable This bit is valid in both LIN and UART compatibility modes. To enable the transmitter's DMA request, this bit must be set. If this bit is cleared, an interrupt request will be generated based on the SETTXINT setting. 0: Transmitter DMA request is disabled. Writing 0 to this bit has no effect. 1: Enable transmitter DMA request.	0h
17	RXDREQEN	R/W	Receive DMA Request Enable This bit is valid in both LIN and UART compatibility modes. To enable the receiver's DMA request, this bit must be set. If this bit is cleared, an interrupt request is generated based on the SETTXINT setting. 0: Transmitter DMA request is disabled. Writing 0 to this bit has no effect.	0h

Field	Name	R/W	Description	Reset value
			1: Enable transmitter DMA request.	
18	ADRXDREQEN	R/W	Address & Data Frames Receive DMA Request Enable Set the receiver DMA for address and data frames. This bit is valid in both LIN and UART compatibility modes. To enable RX DMA requests for address frames and data frames, this bit must be set. If this bit is cleared, an RX interrupt request is generated for address frames, while DMA requests are generated for data frames. 0: Receiver DMA requests are disabled for address frames (RX interrupt requests are enabled for address frames). Writing 0 to this bit has no effect. 1: Receiver DMA requests are enabled for both address and data frames	0h
23:19	Reserved			0h
24	PIEN	R/W	Parity Interrupt Enable This bit is valid in LIN or UART mode. When enabled, the UART/LIN module generates an interrupt if a parity error occurs. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
25	OREIEN	R/W	Overrun Error interrupt Enable This bit is valid in LIN or UART mode. When enabled, the UART/LIN module generates an interrupt if an overrun error occurs. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
26	FEIEN	R/W	Framing Error Interrupt Enable This bit is valid in LIN or UART mode. When enabled, the UART/LIN module generates an interrupt if a frame error occurs. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
27	NREIEN	R/W	No Response Error Interrupt Enable This bit is valid only in LIN mode (write). When enabled, the UART/LIN module generates an interrupt if a no-response error occurs. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
28	ISFEIEN	R/W	Inconsistent Sync Field Error Interrupt Enable) This bit is valid only in LIN mode (write). When enabled, the UART/LIN module generates an interrupt upon an inconsistent sync field error. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
29	CSEIEN	R/W	Checksum-error Interrupt Enable This bit is valid only in LIN mode (write).	0h

Field	Name	R/W	Description	Reset value
			When enabled, the UART/LIN module generates an interrupt upon a checksum error. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	
30	PBEIEN	R/W	Physical Bus Error Interrupt Enable This bit is valid only in LIN mode (write). When enabled, the UART/LIN module generates an interrupt upon a physical bus error. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h
31	BEIEN	R/W	Bit Error Interrupt Enable This bit is valid only in LIN mode (write). When enabled, the UART/LIN module generates an interrupt upon a bit error. 0: Disable. Writing 0 to this bit has no effect. 1: Enable	0h

40.12.5 Interrupt Clear Register (LINICLR)

Offset address: 0x10

Reset type: SYSRSn

This register is used to clear the corresponding interrupts or requests mentioned in the LINIEN register (except "Enable Receiver DMA Request for Address and Data Frames") by setting the respective bits to 1.

The valid modes and write conditions of the bits (valid only in LIN mode or LIN/UART mode) are consistent with the LINIEN register.

Writing 0 to all bits has no effect; writing 1 sets the bit.

Field	Name	R/W	Description	Reset value
0	BRKDETICLR	RC_W1	Break Detect Interrupt Clear	0h
1	WUPICLR	RC_W1	Wake up Interrupt Clear	0h
3:2	Reserved			0h
4	TOICLR	RC_W1	Timeout Interrupt Clear	0h
5	Reserved			0h
6	TOWUPSICLR	RC_W1	Timeout after Wakeup Signal Interrupt Clear	0h
7	TO3WUPSICLR	RC_W1	Timeout after 3 Wakeup Signal Interrupt Clear	0h
8	TXICLR	RC_W1	Transmitter Interrupt Clear	0h
9	RXICLR	RC_W1	Receiver Interrupt Clear	0h
12:10	Reserved			0h
13	IDICLR	RC_W1	Identification Interrupt Clear	0h
15:14	Reserved			0h
16	TXDREQCLR	RC_W1	Transmit DMA Request Clear	0h

Field	Name	R/W	Description	Reset value
17	RXDREQCLR	RC_W1	Receive DMA Request Clear	0h
23:18	Reserved			0h
24	PICLR	RC_W1	Parity Interrupt Clear	0h
25	OREICLR	RC_W1	Overrun Error Interrupt Clear	0h
26	FEICLR	RC_W1	Framing Error Interrupt Clear	0h
27	NREICLR	RC_W1	No Response Error Interrupt Clear	0h
28	ISFEICLR	RC_W1	Inconsistent Sync Field Error Interrupt Clear	0h
29	CSEICLR	RC_W1	Checksum-error Interrupt Clear	0h
30	PBEICLR	RC_W1	Physical Bus Error Interrupt Clear	0h
31	BEICLR	RC_W1	Bit Error Interrupt Clear	0h

40.12.6 Enable Interrupt Level Register (LINILEN)

Offset address: 0x14

Reset type: SYSRSn

This register is used to set the level of the corresponding interrupts or requests mentioned in the LINIEN register. Writing 0 sets the level to INT0, and writing 1 sets the level to INT1.

The valid modes and write conditions of the bits (valid only in LIN mode or LIN/UART mode) are consistent with the LINIEN register.

Field	Name	R/W	Description	Reset value
0	BRKDETILEN	R/W	Break Detect Interrupt Level 1 Enable	0h
1	WUPILEN	R/W	Wake up Interrupt Level 1 Enable	0h
3:2	Reserved			0h
4	TOILEN	R/W	Timeout Interrupt Level 1 Enable	0h
5	Reserved			0h
6	TOWUPSILEN	R/W	Timeout after Wakeup Signal Interrupt Level 1 Enable)	0h
7	TO3WUPSILEN	R/W	Timeout After 3 Wakeup Signal Interrupt Level 1 Enable)	0h
8	TXILEN	R/W	Transmitter Interrupt Level 1 Enable	0h
9	RXILEN	R/W	Receiver Interrupt Level 1 Enable	0h
12:10	Reserved			0h
13	IDILEN	R/W	Identification Interrupt Level 1 Enable	0h
23:14	Reserved			0h
24	PILEN	R/W	Parity Interrupt Level 1 Enable	0h
25	OREILEN	R/W	Overrun Error Interrupt Level 1 Enable	0h

Field	Name	R/W	Description	Reset value
26	FEILEN	R/W	Framing Error Interrupt Level 1 Enable	0h
27	NREILEN	R/W	No Response Error Interrupt Level 1 Enable)	0h
28	ISFEILEN	R/W	Inconsistent Sync Field Error Interrupt Level 1 Enable)	0h
29	CSEILEN	R/W	Checksum-error Interrupt Level 1 Enable	0h
30	PBEILEN	R/W	Physical Bus Error Interrupt Level 1 Enable	0h
31	BEILEN	R/W	Bit Error Interrupt Level 1 Enable	0h

40.12.7 Clear Interrupt Level Register (LINILCLR)

Offset address: 0x18

Reset type: SYSRSn

This register is used to clear the level of the corresponding interrupts or requests mentioned in the LINIEN register. Writing 0 sets the level to INT0, and writing 1 sets the level to INT1 and clear this bit.

The valid modes and write conditions of the bits (valid only in LIN mode or LIN/UART mode) are consistent with the LINILEN register.

Field	Name	R/W	Description	Reset value
0	BRKDETILCLR	RC_W1	Break Detect Interrupt Level 1 Clear	0h
1	WUPILCLR	RC_W1	Wake Up Interrupt Level 1 Clear	0h
3:2	Reserved			0h
4	TOILCLR	RC_W1	Timeout Interrupt Level 1 Clear	0h
5	Reserved			0h
6	TOWUPSILCLR	RC_W1	Timeout after Wakeup Signal Interrupt Level 1 Clear	0h
7	TO3WUPSILCLR	RC_W1	Timeout after 3 Wakeup Signal Interrupt Level 1 Clear	0h
8	TXILCLR	RC_W1	Transmitter Interrupt Level 1 Clear	0h
9	RXILCLR	RC_W1	Receiver Interrupt Level 1 Clear	0h
12:10	Reserved			0h
13	IDILCLR	RC_W1	Identification Interrupt Level 1 Clear	0h
23:14	Reserved			0h
24	PILCLR	RC_W1	Parity Interrupt Level 1 Clear	0h
25	OREILCLR	RC_W1	Overrun Error Interrupt Level 1 Clear	0h
26	FEILCLR	RC_W1	Framing Error Interrupt Level 1 Clear	0h
27	NREILCLR	RC_W1	No Response Error Interrupt Level 1 Clear	0h
28	ISFEILCLR	RC_W1	Inconsistent Sync Field Error Interrupt Level 1 Clear	0h
29	CSEILCLR	RC_W1	Checksum-error Interrupt Level 1 Clear	0h

Field	Name	R/W	Description	Reset value
30	PBEILCLR	RC_W1	Physical Bus Error Interrupt Level 1 Clear	0h
31	BEILCLR	RC_W1	Bit Error Interrupt Level 1 Clear	0h

40.12.8 Flag Register (LINFLG)

Offset address: 0x1C

Reset type: SYSRSn

Note: Different fields are valid in different modes and writable only in specific modes.

Field	Name	R/W	Description	Reset value
0	BRKDETFLG	RC_W1	Break Detect Flag This bit is valid only in UART compatibility mode. This bit is set when the UART detects a break condition on the LINRX pin. A break condition occurs when the LINRX pin remains low for at least 10 bits continuously after the first stop bit is missing (i.e., after a framing error). If the BRKDETIEN bit is set, detecting a break condition causes the UART to generate an error interrupt. The BRKDETFLG bit can be cleared using the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: No break condition detected 1: Break condition detected	0h
1	WUPFLG	RC_W1	Wake Up Flag This bit is valid only in LIN mode (write). This bit is set by the UART/LIN when receiver or transmitter activity causes the module to exit low power consumption mode. If the WUPIEN bit is set, an interrupt is generated. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: Do not wake up from lower power consumption mode. 1: Wake up from lower power consumption mode.	0h
2	RXIDLEFLG	R	Receiver Idle Flag This bit is valid only in UART compatibility mode (write). When this bit is set, the UART looks for an idle period to resynchronize itself with the bitstream. When this	1h

Field	Name	R/W	Description	Reset value
			bit is set, the receiver will not receive any data. The bus must remain idle for 11 bit cycles to clear this bit. The UART enters this state under the following conditions: ● After system reset ● Set RDY bit ● Recover from lower power consumption mode. 0: Idle period detected, UART is ready to receive 1: No idle period detected, UART will not receive any data	
3	BUSYFLG	R	Bus Busy Flag This bit is valid in both LIN mode and UART compatibility mode (write). This bit indicates whether the receiver is receiving a frame. When the receiver detects the start of a start bit, this bit is set to 1. When frame receive is complete, the BUSYFLG bit is cleared. If WUPIEN is set and lower power consumption mode is requested while this bit is set, the UART/LIN will automatically prevent entering lower power consumption mode and generate a wakeup interrupt. The BUSYFLG bit is directly controlled by the UART receiver but can be cleared as follows: ● Set RDY bit ● RST bit ● System reset 0: The receiver is not currently receiving a frame 1: The receiver is currently receiving a frame	0h
4	LINIDLEFLG	RC_W1	LIN Bus Idle Flag This bit is valid only in LIN mode (write). This bit is set if there is no LIN bus activity for at least 4 seconds. LIN bus activity refers to the transition from recessive to dominant. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: No bus idle detected 1: LIN bus idle detected.	0h
5	Reserved			0h
6	TOWUPSFLG	RC_W1	Timeout after Wakeup Signal Flag This bit is valid only in LIN mode (write). This bit is set if no synchronization interrupt is received after a wakeup signal is transmitted. Use a timeout period of at least 150 milliseconds before transmitting another wakeup signal.	0h

Field	Name	R/W	Description	Reset value
			This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: No timeout after one wakeup signal (150 ms) 1: Timeout occurred after one wakeup signal	
7	TO3WUPSFLG	RC_W1	Timeout after 3 Wakeup Signal Flag This bit is valid only in LIN mode (write). This flag is set if no synchronization interrupt is received after transmitting 3 wakeup signals and 1.5 seconds have elapsed. Such a timeout period is used before initiating another round of wakeup signals. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: No timeout after 3 wakeup signals 1: Timeout occurred after 3 wakeup signals and 1.5 seconds	0h
8	TXBRDYFLG	R	Transmitter Buffer Ready Flag When this bit is set, it indicates that the transmit buffer register (LINTXD in compatibility mode or LINTXB0 and LINTXB1 in MBUF mode) is ready for another character to be written by the CPU. In UART compatibility mode, writing data to UARTTXB automatically clears this bit. In LIN mode, this bit is cleared once byte 0 (TXB0) is written to LINTXB0. This bit is set when the data in the TX buffer is shifted into the UARTTXSHF register. If the DMA enable bit is set, this event can trigger a transmit DMA event. This bit is set to 1 by the following methods: ● RST bit ● Set RDY ● System reset Note: Reading the corresponding interrupt offset in the LINIVO0/1 register can not clear the TXRDY flag. Note: Transmit interrupt requests can be suppressed by disabling the corresponding interrupt in the LINICLR register or by disabling the transmitter via the TXENA bit until the next batch of data is written to the transmit buffer LINTXB0 and LINTXB1. 0: Compatibility mode: LINTXB is full.	1h

Field	Name	R/W	Description	Reset value
			LIN mode: Multi-buffer is full. 1: Compatibility mode: UARTTXB is ready to receive the next character. LIN mode: Multi-buffer is ready to receive the next character.	
9	RXRDYFLG	RC_W1	Receiver Ready Flag In UART compatibility mode, the receiver sets this bit to indicate that LINRXD contains new data and is ready to be read by the CPU. In LIN mode, RXRDYFLG is set once a valid frame is received in multi-buffer mode. A valid frame is a message frame without errors. In non-multi-buffer mode, RXRDYFLG is set for each received byte. If there are no errors, this bit is set for the last byte of the frame. If the interrupt enable bit is set, UART/LIN generates a receive interrupt when the RXRDYFLG flag is set. RXRDYFLG can be cleared by the following methods: ● RST bit ● Set RDY ● System reset ● Writing 1 to this bit ● Reading LINRXD in UART compatibility mode ● Reading the last data byte RXBy of the response in LIN mode Note: Reading the corresponding interrupt offset in the LINIVO0/1 register can not clear the RXRDYFLG flag. 0: No new data in LINRXD/LINRXBy. 1: New data is ready to be read from LINRXD.	0h
10	TXWUPSEL	R/W	Transmitter Wakeup Mode Select This bit is valid only in UART compatibility mode. The TXWUPSEL bit controls whether the data in UARTTXB shall be transmitted as an address or data frame using the multiprocessor communication format. This bit is set to 1 or 0 by the software before a byte is written to UARTTXB. It is cleared by the UART when the data is transmitted from UARTTXB to UARTTXSHF or upon a system reset. TXWUPSEL is not cleared by the RDY bit. 0: Address bit mode: The frame to be transmitted will be data (address bit = 0). Idle line mode: The frame to be transmitted will be data. 1: Address bit mode: The frame to be transmitted will be an address (address bit = 1). Idle line mode: The next frame will be an address (writing this bit to 1 and then writing dummy data to LINTXD will result in an 11-bit idle period before the next frame is transmitted).	0h
11	TXEFLG	R	Transmit Empty Flag	1h

Field	Name	R/W	Description	Reset value
			The value of this flag indicates the contents of the transmitter's buffer register (LINTXD/LINTXBy) and shift register (UARTTXSHF). In multi-buffer mode, the flag indicates the value of the TXBx register and the shift register (UARTTXSHF). In non-multi-buffer mode, the flag indicates the value of LINTXB0 (byte) and the shift register (UARTTXSHF). This bit can be set by the following methods: ● RST bit ● Set RDY bit ● System reset Note: This bit does not trigger an interrupt request. 0: Compatibility mode or LIN mode without multi-buffer: The transmit buffer or shift register (or both) has been loaded with data. In LIN mode with multi-buffer: The multi-buffer or shift register (or both) has been loaded with data. 1: Compatibility mode or LIN mode without multi-buffer: The transmit buffer and shift register are both empty. In LIN mode with multi-buffer: The multi-buffer and shift register are both empty.	
12	RXWUPSEL	R	Receiver Wakeup Mode Select This bit is valid only in UART compatibility mode (write). The UART sets this bit to indicate that the current data in LINRXD is an address. This bit can be cleared by the following methods: ● RST bit ● Set RDY bit ● System reset ● Receive data frame 0: The data in LINRXD is not an address 1: The data in LINRXD is an address For more information on using the RXWUPSEL bit in sleep mode, refer to Sleep Mode in Multiprocessor Communication.	0h
13	TXIDFLG	RC_W1	Transmit Identifier Flag This bit is valid only in LIN mode (write). This flag is set when an identifier with a TX match and no ID parity error is received. For more details, refer to the Message Filtering and Validation section. When this flag is set, it indicates that a new valid identifier has been received with a TX match. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register	0h

Field	Name	R/W	Description	Reset value
			● RST bit ● Set RDY ● System reset ● Reading the LINID register ● Writing 1 to this bit 0: No valid ID received. 1: A valid ID is contained in LINID[23:16] received with a TX match.	
14	RXIDFLG	RC_W1	Receive Identifier Flag This bit is valid only in LIN mode (write). This flag is set when an identifier with a RX match and no ID parity error is received. For more details, refer to the Message Filtering and Validation section. When this flag is set, it indicates that a new valid identifier has been received with a RX match. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Reading the LINID register ● Writing 1 to this bit 0: No valid ID received. 1: A valid ID is contained in LINID[23:16] received with a RX match.	0h
23:15	Reserved			0h
24	PEFLG	RC_W1	Parity Error Flag This bit is valid in both LIN and UART compatibility modes. When a parity error is detected in received data, this bit is set. In UART address bit mode, parity is calculated based on the data and address fields of the received frame. In idle line mode, only data is used to calculate parity. An error occurs when the number of 1s in the received character does not match its parity bit. For more information on parity, refer to the description of UART Global Control Register 1. If the parity function is disabled, the PEFLG flag is disabled and reads as 0. Detecting a parity error causes LIN to generate an error interrupt (if PIEN bit = 1). This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset	0h

Field	Name	R/W	Description	Reset value
			● Receive a new character (UART compatibility mode) or frame (LIN mode) ● Writing 1 to this bit 0: No parity error detected, or parity is disabled. 1: Parity error detected.	
25	OREFLG	RC_W1	Overrun Error Flag This bit is valid in LIN or UART compatibility mode (write). This bit is set when data transmitted from UARTRXSHF to UARTRXB overwrites unread data in the UARTRXB or RXBy buffer. If the OREIEN bit is set to 1, detecting an overflow error causes LIN to generate an error interrupt. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit 0: Overrun error is not detected. 1: Overrun error is detected.	0h
26	FEFLG	RC_W1	Framing Error Flag This bit is valid in LIN or UART compatibility mode (write). This bit is set when the expected stop bit is not found. In UART compatibility mode, only the first stop bit is checked. A missing stop bit indicates loss of synchronization with the start bit and results in a character frame error. If the FEIEN bit is set, detecting a framing error causes UART to generate an error interrupt. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit ● Receive a new character (UART compatibility mode) or new frame (LIN mode) In multi-buffer mode, the frame is defined by the UARTFORMAT register. 0: Frame error is not detected. 1: Frame error is detected.	0h
27	NREFLG	RC_W1	No Response Error Flag This bit is valid only in LIN mode (write). This bit is set when the master's header is not completed within TFRAME_MAX. This timeout applies to message frames of unknown length (identifiers 0 to 61). This error is detected by the module's synchronizer. This bit can be cleared by the following methods:	0h

Field	Name	R/W	Description	Reset value
			● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit ● Receive a new synchronization interrupt 0: No response error is not detected. 1: No response error is detected.	
28	ISFEFLG	RC_W1	Inconsistent Sync Field Error Flag This bit is valid only in LIN mode (write). This bit is set when the synchronizer detects an inconsistent sync field error during the header receive process. For more information, refer to the Header Receive and Adaptive Baud Rate section. This bit can be cleared by the following methods: ● Read the corresponding interrupt offset from the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit ● Receive a new synchronization interrupt 0: No inconsistent synchronization field error detected. 1: Inconsistent sync field error detected.	0h
29	CSEFLG	RC_W1	Checksum Error Flag This bit is valid only in LIN mode (write). This bit is set when the receiving node detects a checksum error. The type of checksum to be used depends on the LINGCTRL1.CHASUMCFG bit. This bit can be cleared in the following ways: ● Reading the corresponding interrupt offset in the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit ● Receiving a new synchronization interrupt 0: No checksum error is detected 1: Checksum error is detected.	0h
30	PBEFLG	RC_W1	Physical Bus Error Flag This bit is valid only in LIN mode (write). This bit is set when a physical bus error occurs. This is detected by the bit monitor in TED. This bit can be cleared in the following ways: ● Reading the corresponding interrupt offset in the LINIVO0/1 register ● Set RDY bit	0h

Field	Name	R/W	Description	Reset value
			● RST bit ● System reset ● Writing 1 to this bit ● Receiving a new synchronization interrupt Note: PBEFLG is only flagged when a synchronization interrupt cannot be generated (due to a bus short to VBAT) or a synchronization separator cannot be generated (due to a bus short to GND). 0: No physical bus error detected. 1: Physical bus error detected.	
31	BEFLG	RC_W1	Bit Error Flag This bit is valid only in LIN mode. This bit is set when a bit error occurs. This is detected by the bit monitor in the internal bit monitor. This bit can be cleared in the following ways: ● Reading the corresponding interrupt offset in the LINIVO0/1 register ● Set RDY bit ● RST bit ● System reset ● Writing 1 to this bit ● Receiving a new synchronization interrupt This field can only be written in LIN mode. 0: Bit error is not detected 1: Bit error is detected	0h

40.12.9 Interrupt Vector Offset Register 0 (LINIVO0)

Offset address: 0x20

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	IVO0	R	INT0 Interrupt Vector Offset This register represents the offset of interrupt line INT0. Reading this register updates its value to the next pending interrupt with the highest priority in LINFLG and clears the flag corresponding to the read offset. Note: The flags for receive (LINFLG[9]) and transmit (LINFLG[8]) interrupts cannot be cleared by reading the corresponding offset vector in this register (refer to the detailed description of the LINFLG register).	0h
31:5	Reserved			0h

40.12.10 Interrupt Vector Offset Register 1 (LINIVO1)

Offset address: 0x24

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
4:0	IVO1	R	INT1 Interrupt Vector Offset	0h

Field	Name	R/W	Description	Reset value
			This register represents the offset of interrupt line INT1. Reading this register updates its value to the next pending interrupt with the highest priority in LINFLG and clears the flag corresponding to the read offset. Note: The flags for receive (LINFLG[9]) and transmit (LINFLG[8]) interrupts cannot be cleared by reading the corresponding offset vector in this register (refer to the detailed description of the LINFLG register).	
31:5	Reserved			0h

40.12.11 Configure Length Register (LINLCFG)

Offset address: 0x28

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	CLCFG	R/W	Character Length Configure These bits are valid only in UART compatibility mode. These bits set the UART character length from 1 to 8 bits. Note: In compatibility mode or buffer UART mode, when the received data length is less than eight bits, it is left-aligned in LINRXD/RXBy and padded with trailing zeros. Data read from LINRXD shall be right-shifted by the software to align the received data to the right. Note: Data written to LINTXD shall be right-aligned, but does not require leading zero padding. These bits are writable only in UART mode. 000: Character length is 1 bit. 001: Character length is 2 bits. 010: Character length is 3 bits. 011: Character length is 4 bits. 100: Character length is 5 bits. 101: Character length is 6 bits. 110: Character length is 7 bits. 111: Character length is 8 bits.	0h
15:3	Reserved			0h
18:16	FLCFG	R/W	Frame Length Configure In LIN mode, these bits represent the number of bytes in the response field, ranging from 1 to 8 bytes. In buffer UART mode, these bits represent the number of characters. When these bits are used to represent the LIN response length (LINGCTRL1[0] = 1), the value shall be updated to the expected length of the response when there is an ID RX match. In buffer UART mode, these bits indicate the number of characters, each with the bit length specified by LINLCFG[2:0]. That is, these bits represent the number of characters in the transmitter/receiver format: 1 to 8. Up to eight characters are allowed, with each character being eight bits. 000: Response field contains 1 byte/character. 001: Response field contains 2 bytes/character.	0h

Field	Name	R/W	Description	Reset value
			010: Response field contains 3 bytes/character. 011: Response field contains 4 bytes/character. 100: Response field contains 5 bytes/character. 101: Response field contains 6 bytes/character. 110: Response field contains 7 bytes/character. 111: Response field contains 8 bytes/character.	
31:19	Reserved			0h

40.12.12 Baud Rate Register (LINBR)

Offset address: 0x2C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
15:0	PSCSELL	R/W	Prescaler Select Low UART/LIN 24-bit integer prescaler selection. These bits are used to select the baud rate for the UART/LIN module. These bits are valid in both LIN mode and UART compatibility mode. UART/LIN has an internally generated serial clock determined by the LIN module input clock and the prescalers P and M in this register. UART/LIN uses a 24-bit integer prescaler P value to select one of over 16,700,000 available baud rates. An additional 4-bit fractional prescaler M refines the baud rate selection.	0h
23:16	PSCSELH	R/W	Prescaler Select High UART/LIN 24-bit integer prescaler selection. These bits are used to select the baud rate for the UART/LIN module. These bits are valid in both LIN mode and UART compatibility mode. UART/LIN has an internally generated serial clock determined by the LIN module input clock and the prescalers P and M in this register. UART/LIN uses a 24-bit integer prescaler P value to select one of over 16,700,000 available baud rates. An additional 4-bit fractional prescaler M refines the baud rate selection.	0h
27:24	FDIVSEL	R/W	4 Bit Fractional Divider Select These bits are valid in LIN or UART asynchronous mode. They are used to select the baud rate of the UART/LIN module and represent the fractional part of the baud rate specification. The M frequency divider allows fine-tuning of the P integer prescaler, providing 15 additional intermediate values for each P integer value.	0h
30:28	SFDIVSEL	R/W	Superfractional Divider Select These bits represent an additional fractional part of the baud rate specification. These bits allow ultra-fine tuning of the fractional baud rate, providing 7 more intermediate values for each M fractional divider value. For details, refer to the Superfractional Divider section.	0h
31	Reserved			0h

40.12.13 Receiver Emulation Data Register (LINRXED)

Offset address: 0x30

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	RXED	R	Receiver Emulation Data This bit is valid only in UART compatibility mode. Reading this bit does not clear the RXRDYFLG flag. This register shall only be used by emulators that need to continuously read the data buffer without affecting the RXRDYFLG flag.	0h
31:8	Reserved			0h

40.12.14 Receive Data Buffer Register (LINRXD)

Offset address: 0x34

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	RXD	R	Received Data This bit is valid only in UART compatibility mode. When a frame is fully received, the data in the frame is transferred from the receive shift register UARTRXSHF to this register. When this transfer occurs, the RXRDYFLG flag is set, and if the RX interrupt enable is set, a receive interrupt is generated. Reading data from LINRXD automatically clears the RXRDYFLG flag. When the UART receives data with a length of fewer than eight bits, it loads the data into this register in a left-aligned format and pads the trailing bits with zeros. Therefore, the software shall perform a logical shift to right-align the data to the correct position.	0h
31:8	Reserved			0h

40.12.15 Transmit Data Buffer Register (LINTXD)

Offset address: 0x38

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXD	R/W	Transmit Data This bit is valid only in UART compatibility mode. The data to be transmitted is written to this register. The shift from this register to the transmit shift register UARTTXSHF sets the TXBRDYFLG flag (LINFLG[23]), indicating that LINTXD is ready to load another byte of data. Note: If TXIEN (LINIEN[8]) is set, this data transmission will also trigger an interrupt. Note: When writing data with a length of fewer than eight bits to the LINRXD register, it must be right-aligned but does not require leading zero padding.	0h
31:8	Reserved			0h

40.12.16 Control Pin Register 0 (LINPCTRL0)

Offset address: 0x3C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	RXPEN	R/W	Receive Pin Enable This bit is valid in both LIN and UART modes. This bit defines the function of the LINRX pin. 0: Disable the LINRX pin. 1: Enable the LINRX pin.	0h
2	TXPEN	R/W	Transmit Pin Enable This bit is valid in both LIN and UART modes. This bit defines the function of the LINTX pin. 0: Disable the LINTX pin. 1: Enable the LINTX pin.	0h
31:3	Reserved			0h

40.12.17 Control pin register 2 (LINPCTRL2)

Offset address: 0x44

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	Reserved			0h
1	RXPVAL	R	Receive Pin Current Value This bit is valid in both LIN and UART compatibility modes. This bit contains the current value on the LINRX pin.	0h
2	TXPVAL	R	Transmit Pin Current Value This bit is valid in both LIN and UART compatibility modes. This bit contains the current value on the LINTX pin.	0h
31:3	Reserved			0h

40.12.18 Compare Register (LINCOMP)

Offset address: 0x60

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
2:0	SBEXTSEL	R/W	Sync Break Extend Select This is applicable only in LIN mode. These bits are used to configure the number of Tbits in the synchronization interrupt, extending the minimum 13 Tbits in the sync field to a maximum of 20 Tbits. The formula for calculating the time delay of the synchronization interrupt is: TSYNBRK = 13Tbit + SBREAK x Tbit These bits are writable only in LIN mode.	0h

Field	Name	R/W	Description	Reset value
			000: No additional Tbit in the synchronization interrupt. 001: Synchronization interrupt has 1 additional Tbit. 010: Synchronization interrupt has 2 additional Tbits. 011: Synchronization interrupt has 3 additional Tbits. 100: Synchronization interrupt has 4 additional Tbits. 101: Synchronization interrupt has 5 additional Tbits. 110: Synchronization interrupt has 6 additional Tbits. 111: Synchronization interrupt has 7 additional Tbits.	
7:3	Reserved			0h
9:8	SDCOMPSEL	R/W	Sync Delimiter Compare Select These bits are valid only in LIN mode. These bits are used to configure the number of Tbits in the sync delimiter within the synchronization field. The formula for calculating the time delay of the sync delimiter is: TSDEL = (SDEL + 1) Tbit These bits are writable only in LIN mode. 00: Sync delimiter has 1 Tbit. 01: Sync delimiter has 2 Tbits. 10 : Sync delimiter has 3 Tbits. 11: Sync delimiter has 4 Tbits.	0h
31:10	Reserved			0h

40.12.19 Receive Buffer Register 0 (LINRXB0)

Offset address: 0x64

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	RXB3	R	Receive Buffer 3 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
15:8	RXB2	R	Receive Buffer 2 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
23:16	RXB1	R	Receive Buffer 1 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
31:24	RXB0	R	Receive Buffer 0 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes. Reading this byte clears the RXDY byte. Note: RXB<x-1> corresponds to the LIN frame's data byte <x>.	0h

40.12.20 Receive Buffer Register 1 (LINRXB1)

Offset address: 0x68

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	RXB7	R	Receive Buffer 7 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
15:8	RXB6	R	Receive Buffer 6 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
23:16	RXB5	R	Receive Buffer 5 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h
31:24	RXB4	R	Receive Buffer 4 Each response data byte received in the UARTRXSHF register is transferred to the corresponding RXBy register based on the number of received bytes.	0h

40.12.21 Mask ID Register (LINIDMASK)

Offset address: 0x6C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXIDMASK	R/W	Transmit ID Mask This field is valid only in LIN mode. This 8-bit mask is used to filter incoming ID messages and compare them with the ID byte. When the received ID matches the TX ID mask, the TXIDFLG is set, and if enabled, triggers an ID interrupt. 0 in the mask indicates that the bit is compared with the ID byte. 1 in the mask indicates that the bit is filtered and not used for comparison. When HGENCTRL is set to 1, this field must be set to 0xFF if a full ID comparison is required.	0h
15:8	Reserved			0h
23:16	RXIDMASK	R/W	Receive ID Mask This field is valid only in LIN mode. This 8-bit mask is used to filter incoming ID messages and compare them with the ID byte. When the received ID matches the RX ID mask, the ID RX flag is set, and if enabled, triggers an ID interrupt. 0 in the mask indicates that the bit is compared with the ID byte. 1 in the mask indicates that the bit is filtered and not used for comparison. When HGENCTRL is set to 1, this field must be set to 0xFF if a full ID comparison is required.	0h
31:24	Reserved			0h

40.12.22 ID Register (LINID)

Offset address: 0x70

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	IDBYTE	R/W	ID Byte Set This field is valid only in LIN mode. This byte is the LIN mode message ID. On the master node, writing to this register by the CPU initiates header transmission. For slave tasks, when HGENCTRL (LINGCTRL1[12]) is '0', this byte is used for message filtering. These bits are writable only in LIN mode.	0h
15:8	IDSTBYTE	R/W	ID Slave Task Byte This field is valid only in LIN mode. This byte contains an identifier used to compare the received header ID to determine whether the LIN node needs to perform RX receive response, TX transmit response, or take no action. These bits are writable only in LIN mode.	0h
23:16	RXID	R	Received ID This bit is effective only in LIN mode. This byte contains the current message identifier. During header reception, if there is no ID parity error and an RX/TX match exists, the received ID is copied from the UARTRXSHF register to this byte. Note: If a framing error (FEFLG) is detected during ID reception, the received ID will not be copied to the LINID register.	0h
31:24	Reserved			0h

40.12.23 Transmit Buffer Register 0 (LINTXB0)

Offset address: 0x74

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXB3	R/W	Transmit Buffer 3 (LINTXB3) The 3rd byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
15:8	TXB2	R/W	Transmit Buffer 2 The 2nd byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
23:16	TXB1	R/W	Transmit Buffer 1 The 1st byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
31:24	TXB0	R/W	Transmit Buffer 0 The 0th byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission. Transmission begins when the 0th byte is written to the TXBO buffer.	0h

40.12.24 Transmit Buffer Register 1 (LINTXB1)

Offset address: 0x78

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
7:0	TXB7	R/W	Transmit Buffer 7 The 7th byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
15:8	TXB6	R/W	Transmit Buffer 6 The 6th byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
23:16	TXB5	R/W	Transmit Buffer 5 The 5th byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h
31:24	TXB4	R/W	Transmit Buffer 4 The 4th byte to be transmitted is written to this register and then copied to UARTTXSHF for transmission.	0h

40.12.25 Maximum Baud Rate Prescaler Register (LINMBRPSC)

Offset address: 0x7C

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
12:0	MBRPSC	R/W	Maximum Baud Rate Prescaler This field is valid only in LIN mode. If the ADAPT bit is set, this 13-bit prescaler is used during the synchronization phase of the slave module (refer to the Header Receive and Adaptive Baud Rate section). In this way, UART/LIN slave devices using automatic or selectable bitrate modes can automatically detect the valid rate of any LIN bus. The MBR value shall be programmed to ensure that the maximum baud rate does not exceed 10% of the expected operational baud rate in the LIN network. Otherwise, a 0x00 data byte might be incorrectly detected as a synchronization interrupt. The default value is based on a 70MHz LINCLK (0xDAC). This MBR prescaler is used by the wakeup and idle time counters to maintain constant expiration times relative to a 20kHz rate.	DACh
31:13	Reserved			0h

40.12.26 Analog Error and Test Register (LINEET)

Offset address: 0x90

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	ALBCFG	R/W	Analog Loopback Configure This bit defines whether the I/O buffers of the transmit or receive pins are included in the communication path during analog loopback mode.	0h

Field	Name	R/W	Description	Reset value
			0: Enable analog loopback through the transmit pin. 1: Enable analog loopback through the receive pin.	
1	LBCFG	R/W	Loopback Mode Configure In analog loopback mode, the complete communication path through I/O can be tested, while in digital loopback mode, the I/O buffers are excluded from this path. 0: Enable digital loopback. 1: Enable analog loopback in the module's I/O DFT mode (when IODFTENA = 1010).	0h
7:2	Reserved			0h
11:8	IODFTEN	R/W	IO DFT Enable This field is used to enable the IODFT mode of the UART/LIN module for testing. 0000: Disable 0001: Disable 0010: Disable 0011: Disable 0100: Disable 0101: Disable 0110: Disable 0111: Disable 1000: Disable 1001: Disable 1010: Enable 1011: Disable 1100: Disable 1101: Disable 1110: Disable 1111: Disable	5h
15:12	Reserved			0h
18:16	TXDLYCFG	R/W	Transmit Delay Configure These bits define the time delay for the value on LINTX before the value on LINRX becomes asynchronous. (Not applicable for the start bit) 000: No delay 001: Delay by 1 SCLK 010: Delay by 2 SCLKs 011: Delay by 3 SCLKs 100: Delay by 4 SCLKs 101: Delay by 5 SCLKs 110 Delay by 6 SCLKs 111: Delay by 7 SCLKs	0h
20:19	PSMASK	R/W	Pin Sample Mask These bits define the sampling numbers at which the value of the TX pin will be negated during transmission to verify that the received pin is correctly sampled using the majority detection circuits.	0h

Field	Name	R/W	Description	Reset value
			Note: During IODFT mode testing with the pin sampling mask, the prescaler P must be programmed to a value greater than 2. 00: No mask 01: Negate the value of the TX pin at TBIT_CENTE 10: Negate the value of the TX pin at TBIT_CENTER + 1 SCLK 11: Negate the value of the TX pin at TBIT_CENTER + 2 SCLKs	
23:21	Reserved			0h
24	BRKEEN	R/W	BRKDT Error Enable This bit is valid only in UART compatibility mode. This bit is used to generate a BRKDT error (UART mode only). When this bit is set, the stop bit of a frame is AND operation with '0' and passed to RSM, causing a framing error. The RX pin is then forced to a continuous low level for 10 Tbits to generate a BRKDT error.	0h
25	PEEN	R/W	Parity Error Enable This bit is valid only in UART compatibility mode. This bit is used to generate a parity error. When this bit is set, in compatibility mode, the received parity bit is inverted to generate a parity error.	0h
26	FEEN	R/W	Frame Error Enable This bit is used to generate a frame error. This bit is valid only in UART compatibility mode. When this bit is set, the received stop bit is AND operation with '0' and passed to the stop bit check circuit.	0h
27	Reserved			0h
28	ISFEEN	R/W	Inconsistent Sync Field Error Enable Inconsistent sync field error enable bit This bit is valid only in LIN mode. This bit is used to generate an ISF error. When this bit is set, the bit width in the sync field is altered, causing the ISF check to fail and the error flag to be set.	0h
29	CSEEN	R/W	Checksum Error Enable This bit is valid only in LIN mode. This bit is used to generate a checksum error. When this bit is set, the polarity of the CTYPE (checksum type) in the received checksum calculator is changed, resulting in a checksum error.	0h
30	PBEEN	R/W	Physical Bus Error Enable This bit is valid only in LIN mode. This bit is used to generate a physical bus error. When this bit is set, the received bits during the sync interrupt field transmit are OR operation with 1 and passed to the bit monitoring circuit.	0h
31	BEEN	R/W	Bit Error Enable This bit is valid only in LIN mode. This bit is used to generate a bit error. When this bit is set, the received bits	0h

Field	Name	R/W	Description	Reset value
			are OR operation with 1 and passed to the bit monitoring circuit.	

40.12.27 Global Interrupt Register (LINGIEN)

Offset address: 0xE0

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GIEN0	R/W	Global Interrupt Enable for LIN INT0 This bit determines whether the INT0 interrupt line will generate an interrupt to the PIE. 0: LIN INT0 line will not generate an interrupt to the PIE. 1: LIN INT0 line will generate an interrupt to the PIE if an enabled interrupt condition occurs.	0h
1	GIEN1	R/W	Global Interrupt Enable for LIN INT1 This bit determines whether the INT1 interrupt line will generate an interrupt to the PIE. 0: LIN INT1 line will not generate an interrupt to the PIE. 1: LIN INT1 line will generate an interrupt to the PIE if an enabled interrupt condition occurs.	0h
31:2	Reserved			0h

40.12.28 Global Interrupt Flag Register (LINGIFLG)

Offset address: 0xE4

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GIFLG0	R	Global Interrupt Flag for LIN INT0 This bit indicates whether an interrupt has been generated to the PIE due to an enabled interrupt on the INT0 interrupt line. For conditions that generate an interrupt, refer to the LIN Interrupt Status Register. This bit can be cleared by writing 1 to the corresponding bit in the LINGICLR register. 0: No active interrupt on the INT0 line. 1: An interrupt has been generated due to an enabled interrupt on the INT0 line.	0h
1	GIFLG1	R	Global Interrupt Flag for LIN INT1 Global Interrupt Flag for LIN INT1 This bit indicates whether an interrupt has been generated to the PIE due to an enabled interrupt on the INT1 interrupt line. For conditions that generate an interrupt, refer to the LIN Interrupt Status Register. This bit can be cleared by writing 1 to the corresponding bit in the LINGICLR register. 0: No active interrupt on the INT1 line. 1: An interrupt has been generated due to an enabled interrupt on the INT1 line.	0h
31:2	Reserved			0h

40.12.29 Clear Global Interrupt Flag Register (LINGICLR)

Offset address: 0xE8

Reset type: SYSRSn

Field	Name	R/W	Description	Reset value
0	GICLR0	R/W1C	Global Interrupt Flag Clear for LIN INT0 This bit is used to clear the corresponding bit in the LINGIFLG register. Write 1 to clear the GIFLG0 bit. Writing 0 has no effect.	0h
1	GICLR1	R/W1C	Global Interrupt Flag Clear for LIN INT1 This bit is used to clear the corresponding bit in the LINGIFLG register. Write 1 to clear the GIFLG1 bit. Writing 0 has no effect.	0h
31:2	Reserved			0h

41 Flexible logic block (FLB)

For the description of this module, refer to the *G32R5xx FLB User Manual V1.0*.

42 Revision history

Table 255 Document Revision History

Date	Version	Revision History
September 2026	1.0	New

Statement

This document is formulated and published by Geehy Semiconductor Co., Ltd. (hereinafter referred to as “Geehy”). The contents in this document are protected by laws and regulations of trademark, copyright and software copyright. Geehy reserves the right to make corrections and modifications to this document at any time. Read this document carefully before using Geehy products. Once you use the Geehy product, it means that you (hereinafter referred to as the “users”) have known and accepted all the contents of this document. Users shall use the Geehy product in accordance with relevant laws and regulations and the requirements of this document.

1. Ownership

This document can only be used in connection with the corresponding chip products or software products provided by Geehy. Without the prior permission of Geehy, no unit or individual may copy, transcribe, modify, edit or disseminate all or part of the contents of this document for any reason or in any form.

The “极海” or “Geehy” words or graphics with “®” or “™” in this document are trademarks of Geehy. Other product or service names displayed on Geehy products are the property of their respective owners.

2. No Intellectual Property License

Geehy owns all rights, ownership and intellectual property rights involved in this document.

Geehy shall not be deemed to grant the license or right of any intellectual property to users explicitly or implicitly due to the sale or distribution of Geehy products or this document.

If any third party’s products, services or intellectual property are involved in this document, it shall not be deemed that Geehy authorizes users to use the aforesaid third party’s products, services or intellectual property. For any information regarding the application of the product, Geehy hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party, unless otherwise agreed in the sales order or the sales contract.

3. Version Update

Users can obtain the latest document of the corresponding models when ordering Geehy products.

If the contents in this document are inconsistent with Geehy products, the agreement in the sales order or the sales contract shall prevail.

4. Information Reliability

The relevant data in this document are obtained from batch test by Geehy Laboratory or cooperative third-party testing organization. However, clerical errors or errors caused by differences in testing environment may occur inevitably. Therefore, users should understand that Geehy does not bear any responsibility for such errors that may occur in this document. The relevant data in this document are only used to guide users as performance parameter reference and do not constitute Geehy's guarantee for any product performance.

Users shall select appropriate Geehy products according to their own needs, and effectively verify and test the applicability of Geehy products to confirm that Geehy products meet their own needs, corresponding standards, safety or other reliability requirements. If losses are caused to users due to user's failure to fully verify and test Geehy products, Geehy will not bear any responsibility.

5. Legality

USERS SHALL ABIDE BY ALL APPLICABLE LOCAL LAWS AND REGULATIONS WHEN USING THIS DOCUMENT AND THE MATCHING GEEHY PRODUCTS. USERS SHALL UNDERSTAND THAT THE PRODUCTS MAY BE RESTRICTED BY THE EXPORT, RE-EXPORT OR OTHER LAWS OF THE COUNTRIES OF THE PRODUCTS SUPPLIERS, GEEHY, GEEHY DISTRIBUTORS AND USERS. USERS (ON BEHALF OF THEMSELVES, THEIR SUBSIDIARIES AND AFFILIATED ENTERPRISES) SHALL AGREE AND PROMISE TO ABIDE BY ALL APPLICABLE LAWS AND REGULATIONS ON THE EXPORT AND RE-EXPORT OF GEEHY PRODUCTS AND/OR TECHNOLOGIES AND DIRECT PRODUCTS.

6. Disclaimer of Warranty

THIS DOCUMENT IS PROVIDED BY GEEHY "AS IS" AND THERE IS NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE WARRANTIES OF MERCHANTABILITY AND

FITNESS FOR A PARTICULAR PURPOSE, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

GEEHY'S PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED FOR USE AS CRITICAL COMPONENTS IN MILITARY, LIFE-SUPPORT, POLLUTION CONTROL, OR HAZARDOUS SUBSTANCES MANAGEMENT SYSTEMS, NOR WHERE FAILURE COULD RESULT IN INJURY, DEATH, PROPERTY OR ENVIRONMENTAL DAMAGE.

IF THE PRODUCT IS NOT LABELED AS "AUTOMOTIVE GRADE," IT SHOULD NOT BE CONSIDERED SUITABLE FOR AUTOMOTIVE APPLICATIONS. GEEHY ASSUMES NO LIABILITY FOR THE USE BEYOND ITS SPECIFICATIONS OR GUIDELINES.

THE USER SHOULD ENSURE THAT THE APPLICATION OF THE PRODUCTS COMPLIES WITH ALL RELEVANT STANDARDS, INCLUDING BUT NOT LIMITED TO SAFETY, INFORMATION SECURITY, AND ENVIRONMENTAL REQUIREMENTS. THE USER ASSUMES FULL RESPONSIBILITY FOR THE SELECTION AND USE OF GEEHY PRODUCTS. GEEHY WILL BEAR NO RESPONSIBILITY FOR ANY DISPUTES ARISING FROM THE SUBSEQUENT DESIGN OR USE BY USERS.

7. Limitation of Liability

IN NO EVENT, UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING, WILL GEEHY OR ANY OTHER PARTY WHO PROVIDES THE DOCUMENT AND PRODUCTS "AS IS", BE LIABLE FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, DIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE DOCUMENT AND PRODUCTS (INCLUDING BUT NOT LIMITED TO LOSSES OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY USERS OR THIRD PARTIES). THIS INCLUDES POTENTIAL PERSONAL INJURY, PROPERTY, OR ENVIRONMENTAL DAMAGE, FOR WHICH GEEHY WILL NOT BE RESPONSIBLE.

8. Scope of Application

The information in this document replaces the information provided in all previous versions of the document.

9. Language Versions and Artificial Intelligence (AI) Disclaimer

This document is provided in multiple languages, with certain content translated via AI services. In the event of any inconsistency or conflict between the Chinese version and any translated version, the Chinese version shall prevail.

GEEHY ASSUMES NO LIABILITY FOR ANY ERRORS, MISUNDERSTANDINGS, OR MISINTERPRETATIONS ARISING FROM THE USE OF SUCH TRANSLATIONS, OR FROM THE RETRIEVAL, INTERPRETATION, AND PROCESSING OF THIS DOCUMENT'S CONTENT BY THIRD-PARTY AI SYSTEMS.

© 2026 Geehy Semiconductor Co., Ltd. - All Rights Reserved