ARM®-based 32-bit Cortex®-M4 MCU with 64 to 256 KB Flash, sLib, USBOTG, 11 timers, ADC, 2 COMPs, 12 communication interfaces

Feature

- **Core: ARM®32-bit Cortex®-M4F CPU**
  - 150 MHz maximum frequency, with a Memory Protection Unit (MPU), single-cycle multiplication and hardware division
  - DSP instructions
- **Memories**
  - 64 to 256 Kbytes of internal Flash memory
  - 18 Kbytes of boot code area used as a Bootloader or as a general instruction/data memory (one-time-configured)
  - sLib: configurable part of main Flash set as a library area with code executable but secured, non-readable
  - 32 Kbytes of SRAM
- **Clock, reset and power control**
  - 2.6 V ~ 3.6 V application supply
  - Power-on reset (POR)/ low-voltage reset (LVR), and power voltage monitor (PVM)
  - 4 to 25 MHz crystal (HEXT)
  - Internal 48 MHz factory-trimmed clock (HICK), offering 1% accuracy at T_A=25 °C, 2.5 % accuracy at T_A=-40 to +105 °C
  - Embedded low-speed RC oscillator
  - 32.768 kHz oscillator
- **Low power**
  - Sleep, Deepsleep, and Standby modes
  - VBAT supply for ERTC and 20 x 32-bit battery powered registers (BPR)
- **1 x 12-bit 0.5 µs A/D converter (up to 16 input channels)**
  - Conversion range: 0 V to 3.6 V
  - Sample and hold capability
  - Temperature sensor
- **2 x COMPs**
- **DMA: 12-channel DMA controller**
  - Peripherals supported: timers, ADC, SDIO, I²S, SPI, I²C and USART
- **Debug mode**
  - Serial wire debug (SWD) and JTAG
- **Up to 55 fast GPIOs**
  - All mappable to 16 external interrupt vectors
  - Almost 5 V-tolerant
  - All fast I/Os, registers accessible with f_{AHB} speed
- **Up to 11 Timers (TMR)**
  - 5 x 16-bit and 2 x 32-bit timers, each with 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
  - 1 x 16-bit 7-channel advanced timer, 6-channel PWM output with dead-time generator and emergency stop
  - 2 x Watchdog timers (WDT and WWDT)
  - SysTick timer: 24-bit downcounter
- **ERTC: enhanced RTC**
- **Up to 12 communication interfaces**
  - 2 x I²C interfaces (SMBus/PMBus support)
  - 5 x USARTs (ISO7816 interface, LIN, IrDA and modem control)
  - 2 x SPIs, both with I²S interface multiplexed
  - CAN interface (2.0B Active)
  - USB full speed interface/host/OTG controller
  - SDIO interface
- **CRC Calculation Unit**
- **96-bit ID (UID)**
- **Packaging**
  - LQFP64 10 x 10 mm
  - LQFP64 7 x 7 mm
  - LQFP48 7 x 7 mm
  - QFN48 6 x 6 mm
  - QFN32 4 x 4 mm
- **List of Models**

Internal Flash	Model
256 Kbytes	AT32F415RCT7, AT32F415RCT7-7, AT32F415CCT7, AT32F415CCU7, AT32F415KCU7-4
128 Kbytes	AT32F415RBT7, AT32F415RBT7-7, AT32F415CBT7, AT32F415CBU7, AT32F415KBU7-4
64 Kbytes	AT32F415R8T7, AT32F415R8T7-7, AT32F415C8T7, AT32F415K8U7-4

Contents

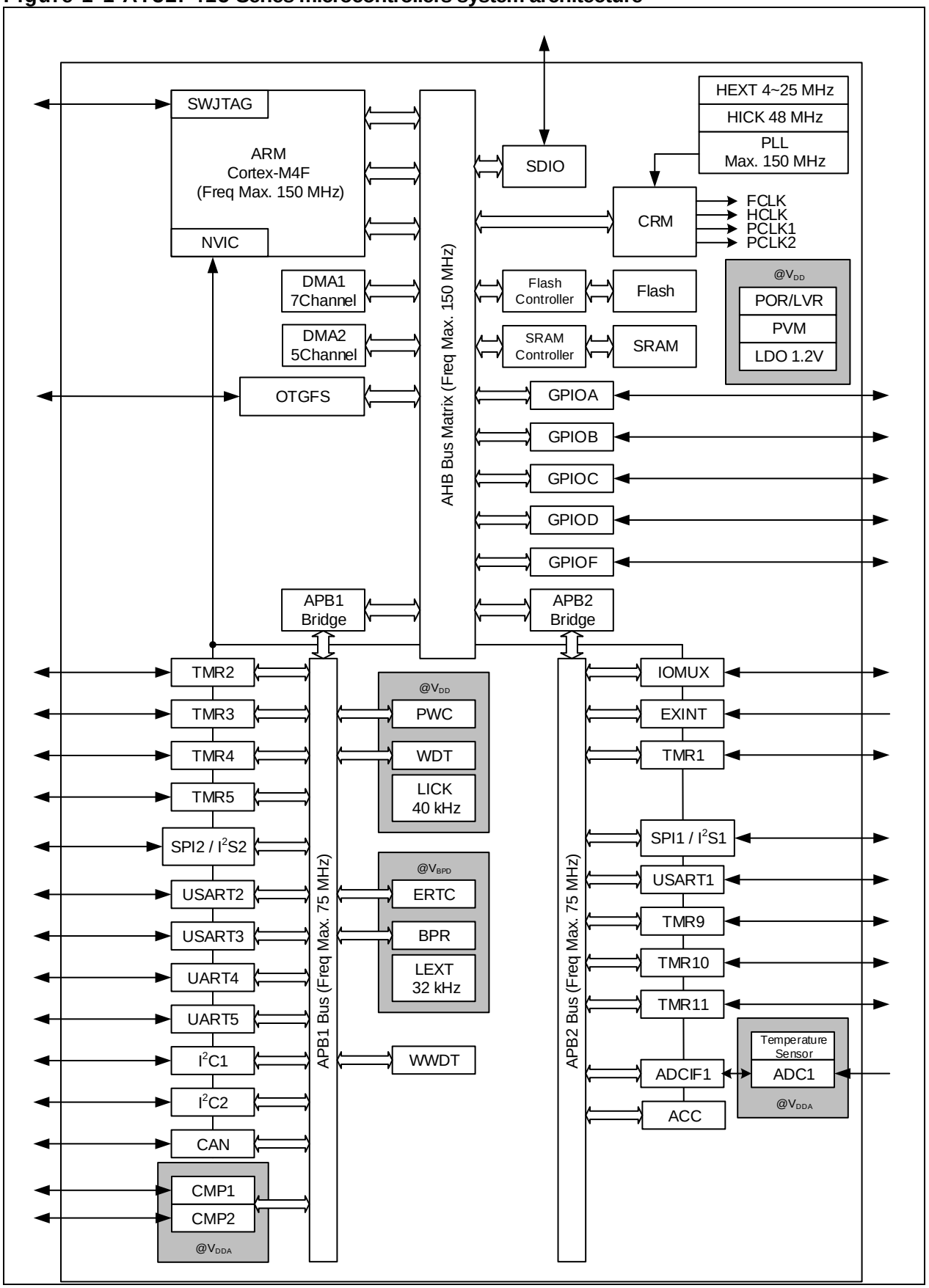
List of figures

List of tables

1 System architecture

AT32F415 series microcontrollers incorporates a 32-bit ARM® Cortex®-M4 processor core, multiple 16-bit and 32-bit timers, DMA controller, ERTC, communication interfaces such as SPI, I2C, USART/UART, SDIO, CAN bus controller, USB2.0 OTG full-speed interface, 12-bit ADC, programmable voltage monitor (PVM) and other peripherals. Cortex®-M4 processer supports enhanced high-performance DSP instruction set, including extended single-cycle 16-bit/32-bit multiply accumulator (MAC), dual 16-bit MAC instructions, optimized 8-bit/16-bit SIMD operation and saturation operation instructions, as shown in Figure 1-1:

Figure 1-1 AT32F415 Series microcontrollers system architecture



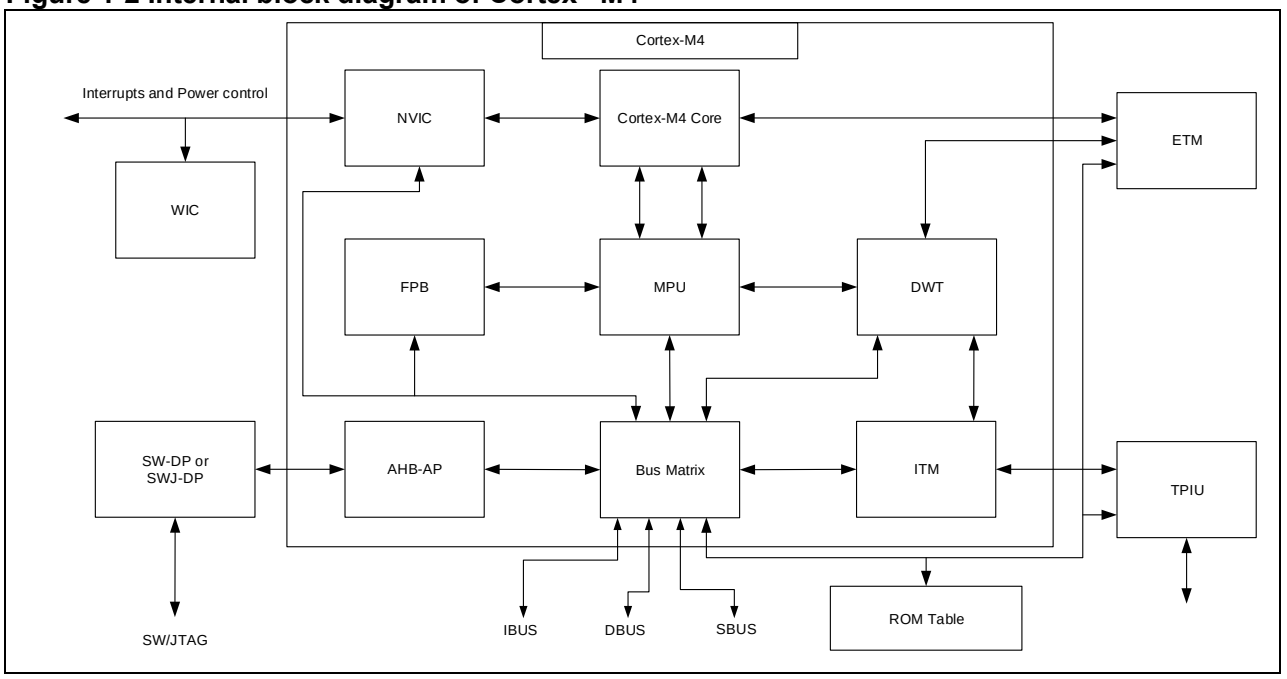
1.1 System overview

1.1.1 ARM Cortex®-M4 processor

Cortex®-M4 processor is a low-power consumption processor featuring low gate count, low interrupt latency, and low-cost debug. It supports DSP instruction set, and is suitable for deeply-embedded applications that require quicker response to interruption. Cortex®-M4 processor is based on ARMv7-M architecture, supporting both Thumb instruction set and DSP instruction set.

Figure 1-2 shows the internal block diagram of Cortex®-M4 processor. Please refer to *ARM Cortex® -M4 Technical Reference Manual* for more information.

Figure 1-2 Internal block diagram of Cortex®-M4



1.1.2 Bit band

With the help of bit-band, read and write access to a single bit can be performed using common load/store operations. The Cortex®-M4 memory includes two bit-band regions: the least significant 1M bytes of SRAM and the least significant 1Mbytes of peripherals. In addition to access to bit-band addresses, their respective bit-band alias region can be used to access to any bit in these two bit-band regions. The bit-band alias region transforms each bit into a 32-bit word. Thus, accessing to an address in an alias region has the same effect as read-modify-write operation on the targeted bit in a bit-band region.

Figure 1-3 Comparison between bit-band region and its alias region: image A

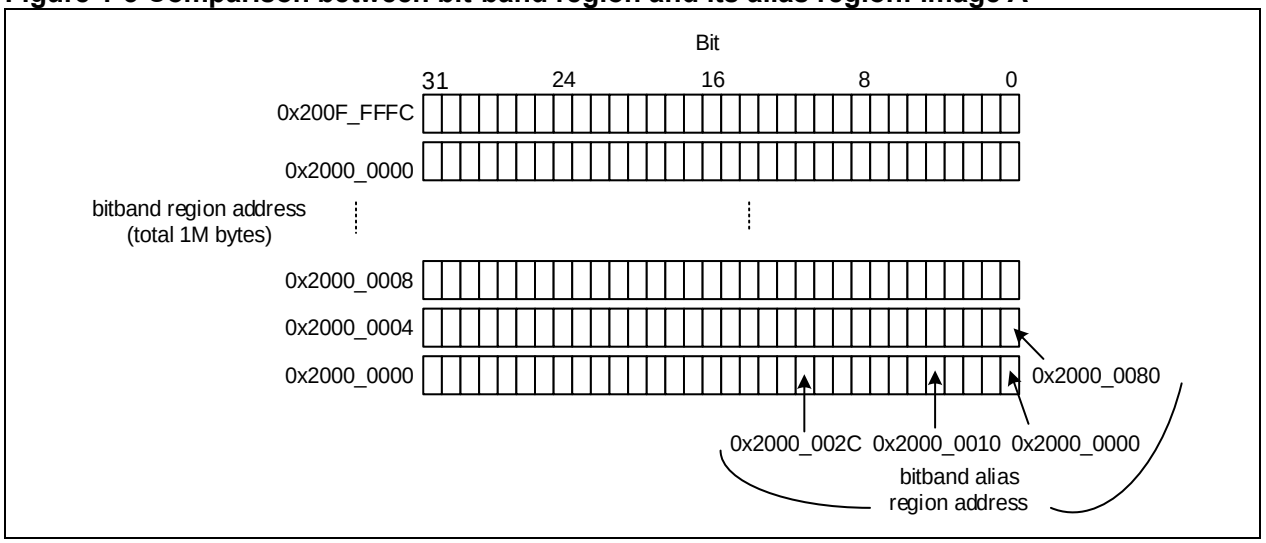
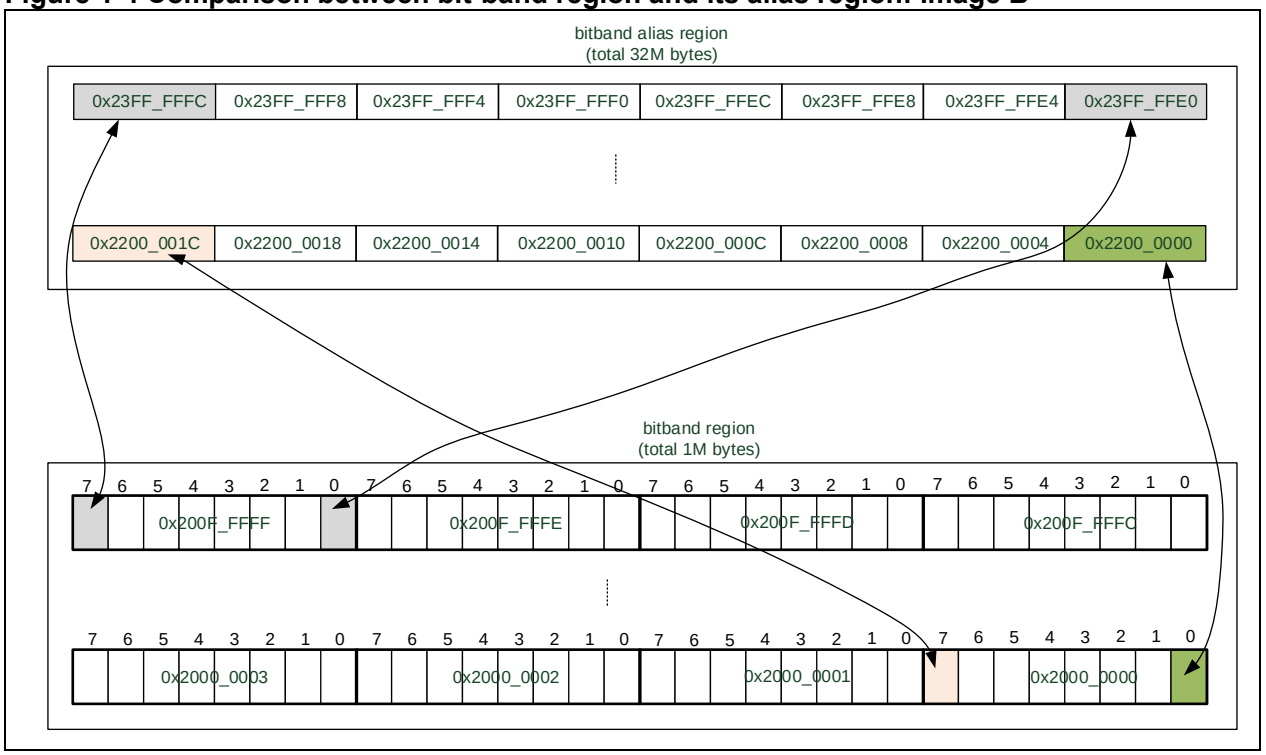


Figure 1-4 Comparison between bit-band region and its alias region: image B



Bit-band region: address region for bit-band operations

Bit-band alias region: access to the alias region has the same effect as read-modify-write operation on the bit-band region

Each bit in a bit-band region is mapped into a word (LSB) in an alias region. When accessing to the address in a bit-band alias region, such address is transformed into a bit-band address first. For a read operation, read one word in the bit-band region, and then move the targeted bit to the right to LSB before returning LSB. For a write operation, first move the targeted bit to the left to the corresponding bit number, then perform a read-modify-write operation on bit level.

The address ranges of two memories supporting bit-band operations:

The lowest 1 Byte of the SRAM: 0x2000_0000~0x200F_FFFF

The lowest 1 Mbyte of peripherals: 0x4000_0000~0x400F_FFFF

For a bit in the SRAM bit-band region, if the byte address is A, the bit number is n (0<=n<=7), then the alias address where the bit is :

AliasAddr = 0x2200_0000+ (A-0x2000_0000)*32+n*4

For a bit in the peripheral bit-band region, if the byte address is A, the bit number is n (0<=n<=7), then the alias address where the bit is:

AliasAddr = 0x4200_0000+ (A-0x4000_0000)*32+n*4

Table 1-1 shows the mapping between bit-band region and alias region in SRAM:

Table 1-1 Bit-band address mapping in SRAM

Bit-band region	Equivalent alias address
0x2000_0000.0	0x2200_0000.0
0x2000_0000.1	0x2200_0004.0
0x2000_0000.2	0x2200_0008.0
…	…
0x2000_0000.31	0x2200_007C.0
0x2000_0004.0	0x2200_0080.0

0x2000_0004.1	0x2200_0084.0
0x2000_0004.2	0x2200_0088.0
…	…
0x200F_FFFC.31	0x23FF_FFFC.0

Table 1-2 shows the mapping between bit-band region and alias region in the peripheral area:

Table 1-2 Bit-band address mapping in the peripheral area

Bit-band region	Equivalent alias address
0x4000_0000.0	0x4200_0000.0
0x4000_0000.1	0x4200_0004.0
0x4000_0000.2	0x4200_0008.0
…	…
0x4000_0000.31	0x4200_007C.0
0x4000_0004.0	0x4200_0080.0
0x4000_0004.1	0x4200_0084.0
0x4000_0004.2	0x4200_0088.0
…	…
0x400F_FFFC.31	0x43FF_FFFC.0

In addition, bit-band operations can also simplify jump process. When jump operation is based on a bit level, the previous steps are:

- Read the whole register
- Mask the undesired bits
- Compare and jump

For now, you just need do:

- Read the bit status from the bit-band alias region
- Compare and jump

Apart from making code more concise, its important function is also reflected in multi-task environment. When it comes to multiple tasks, it turns the read-modify-write operations into a hardware-supported atomic operation to avoid the scenario where the read-modify-write operation is disrupted, resulting in disorder.

1.1.3 Interrupt and exception vectors

Table 1-3 AT32F415 series vector table

Pos.	Priority	Priority Type	Name	Description	Address
	-	-	-	Reserved	0x0000_0000
	-3	Fixed	Reset	Reset	0x0000_0004
	-2	Fixed	NMI	Non maskable interrupt CRM clock fail detector (CFD) is linked to NMI vector	0x0000_0008
	-1	Fixed	HardFault	All class of fault	0x0000_000C
	0	Configurable	MemoryManage	Memory management	0x0000_0010
	1	Configurable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014

	2	Configurable	UsageFault	Undefined instruction or illegal state	0x0000_0018
	-	-	-	Reserved	0x0000_001C ~0x0000_002B
	3	Configurable	SVCall	System service call via SWI instruction	0x0000_002C
	4	Configurable	Debug Monitor	Debug monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	Configurable	PendSV	Pendable request for system service	0x0000_0038
	6	Configurable	SysTick	System tick timer	0x0000_003C
0	7	Configurable	WWDT	Window timer interrupt	0x0000_0040
1	8	Configurable	PVM	PVM interrupt linked to EXINT 16	0x0000_0044
2	9	Configurable	TAMPER	Tamper interrupt linked to EXINT21	0x0000_0048
3	10	Configurable	ERTC	RTC global interrupt linked to EXINT22	0x0000_004C
4	11	Configurable	FLASH	Flash global interrupt	0x0000_0050
5	12	Configurable	CRM	Clock and Reset manage (CRM) interrupt	0x0000_0054
6	13	Configurable	EXINT0	EXINT line0 interrupt	0x0000_0058
7	14	Configurable	EXINT1	EXINT line1 interrupt	0x0000_005C
8	15	Configurable	EXINT2	EXINT line2 interrupt	0x0000_0060
9	16	Configurable	EXINT3	EXINT line3 interrupt	0x0000_0064
10	17	Configurable	EXINT4	EXINT line4 interrupt	0x0000_0068
11	18	Configurable	DMA1 channel 1	DMA1 channel 1 global interrupt	0x0000_006C
12	19	Configurable	DMA1 channel 2	DMA1 channel 2 global interrupt	0x0000_0070
13	20	Configurable	DMA1 channel 3	DMA1 channel 3 global interrupt	0x0000_0074
14	21	Configurable	DMA1 channel 4	DMA1 channel 4 global interrupt	0x0000_0078
15	22	Configurable	DMA1 channel 5	DMA1 channel 5 global interrupt	0x0000_007C
16	23	Configurable	DMA1 channel 6	DMA1 channel 6 global interrupt	0x0000_0080
17	24	Configurable	DMA1 channel 7	DMA1 channel 7 global interrupt	0x0000_0084
18	25	Configurable	ADC1	ADC1 global interrupt	0x0000_0088
19	26	Configurable	CAN1_TX	CAN1 transmit interrupt	0x0000_008C
20	27	Configurable	CAN1_RX0	CAN1 receive0 interrupt	0x0000_0090
21	28	Configurable	CAN1_RX1	CAN1 receive1 interrupt	0x0000_0094
22	29	Configurable	CAN1_SE	CAN1 status error interrupt	0x0000_0098
23	30	Configurable	EXINT9_5	EXINT line[9:5] interrupt	0x0000_009C
24	31	Configurable	TMR1_BRK_TMR9	TMR1 break interrupt and TMR9 global interrupt	0x0000_00A0
25	32	Configurable	TMR1_OVF_TMR10	TMR1 overflow interrupt and TMR10 global interrupt	0x0000_00A4
26	33	Configurable	TMR1_TRG_HALL_TMR11	TMR1 trigger, HALL interrupt and TMR11 global interrupt	0x0000_00A8

27	34	Configurable	TMR1_CH	TMR1 channel interrupt	0x0000_00AC
28	35	Configurable	TMR2	TMR2 global interrupt	0x0000_00B0
29	36	Configurable	TMR3	TMR3 global interrupt	0x0000_00B4
30	37	Configurable	TMR4	TMR4 global interrupt	0x0000_00B8
31	38	Configurable	I2C1_EVT	I^2C1 event interrupt	0x0000_00BC
32	39	Configurable	I2C1_ETR	I^2C1 error interrupt	0x0000_00C0
33	40	Configurable	I2C2_EVT	I^2C2 event interrupt	0x0000_00C4
34	41	Configurable	I2C2_ERR	I^2C2 error interrupt	0x0000_00C8
35	42	Configurable	SPI1	SPI1 global interrupt	0x0000_00CC
36	43	Configurable	SPI2	SPI2 global interrupt	0x0000_00D0
37	44	Configurable	USART1	USART1 global interrupt	0x0000_00D4
38	45	Configurable	USART2	USART2 global interrupt	0x0000_00D8
39	46	Configurable	USART3	USART3 global interrupt	0x0000_00DC
40	47	Configurable	EXINT15_10	EXINT[15:10] global interrupt	0x0000_00E0
41	48	Configurable	ERTCAlarm	ERTC alarm interrupt linked to EXINT	0x0000_00E4
42	49	Configurable	OTGFS1	OTGFS1 wakeup interrupt	0x0000_00E8
43	50	-	-	Reserved	0x0000_00EC
44	51	-	-	Reserved	0x0000_00F0
45	52	-	-	Reserved	0x0000_00F4
46	53	-	-	Reserved	0x0000_00F8
47	54	-	-	Reserved	0x0000_00FC
48	55	-	-	Reserved	0x0000_0100
49	56	Configurable	SDIO	SDIO global interrupt	0x0000_0104
50	57	Configurable	TMR5	TMR5 global interrupt	0x0000_0108
51	58	-	-	Reserved	0x0000_010C
52	59	Configurable	UART4	UART4 global interrupt	0x0000_0110
53	60	Configurable	UART5	UART5 global interrupt	0x0000_0114
54	61	-	-	Reserved	0x0000_0118
55	62	-	-	Reserved	0x0000_011C
56	63	Configurable	DMA2 channel 1	DMA2 channel 1 global interrupt	0x0000_0120
57	64	Configurable	DMA2 channel 2	DMA2 channel 2 global interrupt	0x0000_0124
58	65	Configurable	DMA2 channel 3	DMA2 channel 3 global interrupt	0x0000_0128
59	66	Configurable	DMA2 channel 4_5	DMA2 channel 4 and DMA2 channel 5 global interrupt	0x0000_012C
60	67	-	-	Reserved	0x0000_0130

61	68	-	-	Reserved	0x0000_0134
62	69	-	-	Reserved	0x0000_0138
63	70	-	-	Reserved	0x0000_013C
64	71	-	-	Reserved	0x0000_0140
65	72	-	-	Reserved	0x0000_0144
66	73	-	-	Reserved	0x0000_0148
67	74	Configurable	OTGFS	OTGFS interrupt	0x0000_014C
68	75	-	-	Reserved	0x0000_0150
69	76	-	-	Reserved	0x0000_0154
70	77	Configurable	CMP1	CMP1 interrupt from EXINT19	0x0000_0158
71	78	Configurable	CMP2	CMP2 interrupt from EXINT20	0x0000_015C
72	79	-	-	Reserved	0x0000_0160
73	80	-	-	Reserved	0x0000_0164
74	81	-	-	Reserved	0x0000_0168
75	82	Configurable	DMA2 channel 6_7	DMA2 channel 6 and channel 7 global interrupt	0x0000_016C

1.1.4 System Tick (SysTick)

The System Tick is a 24-bit downcounter. It will be reloaded with the initial value automatically when it is decremented to zero. It can generate periodic interrupts, so it is often used as multi-task scheduling counter for embedded operating system, and also to call the periodic tasks for non-embedded system.

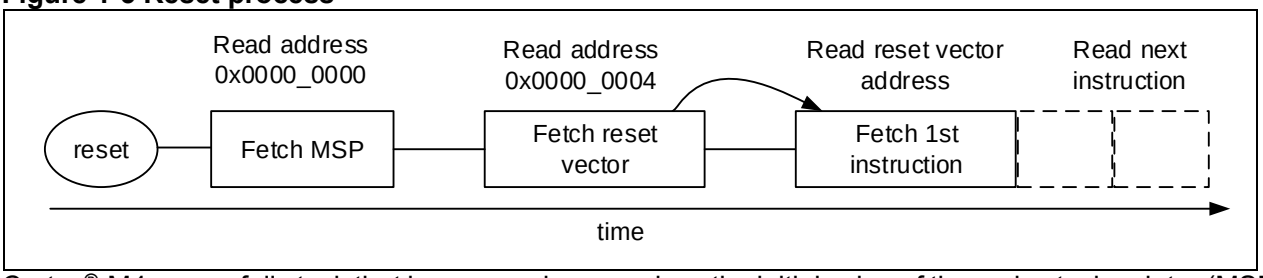
The System Tick calibration value is fixed to 9000, which gives a reference time base of 1 ms when the System Tick clock is set to 9 MHz.

1.1.5 Reset

The processor reads the first two words from the CODE memory after a system reset and before program execution.

- Get the initial value of the main stack pointer (MSP) from address 0x0000_0000
- Get the initial value of the program counter (PC) from address 0x0000_0004. This value is a reset vector and LSB must be 1. Then take the instructions from the address corresponding to this value.

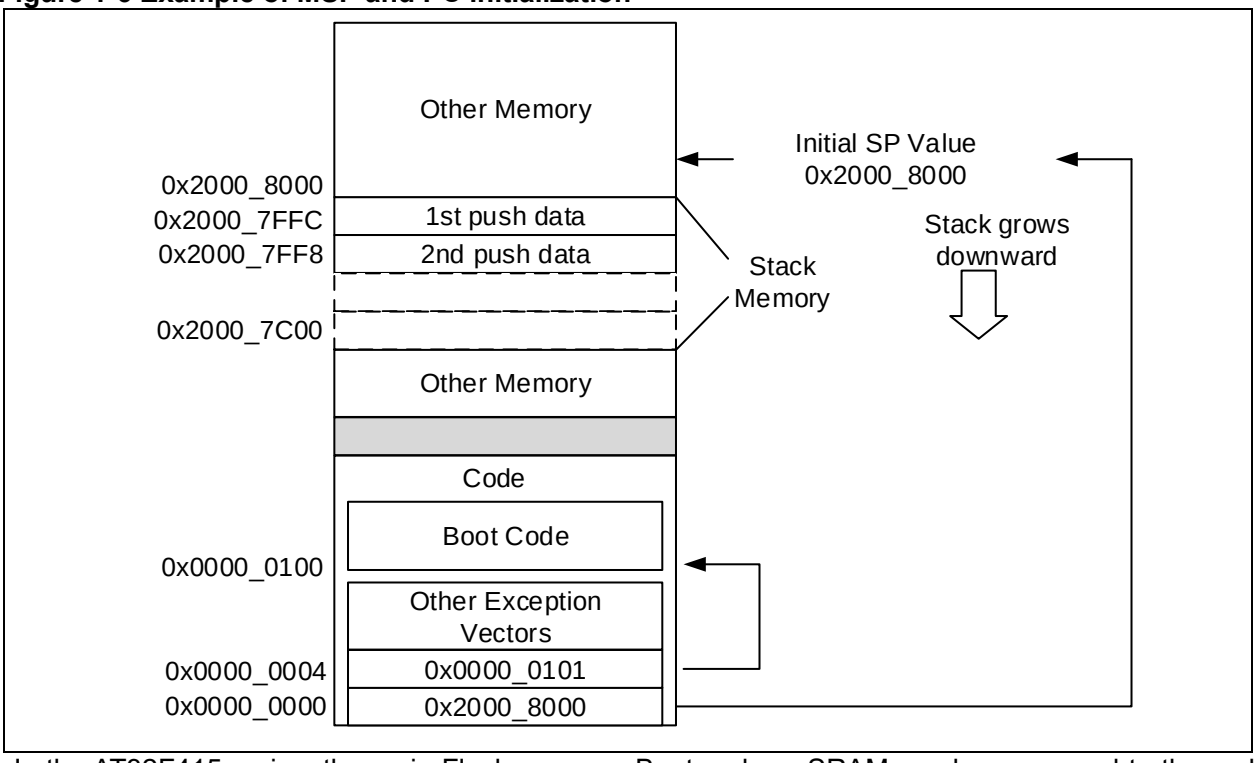
Figure 1-5 Reset process



Cortex®-M4 uses a full stack that increases downward, so the initial value of the main stack pointer (MSP) must be the end address of the stack memory plus 1. For example, if the stack area is set between 0x2000_7C00 and 0x2000_7FFF, then the initial value of MSP must be defined as 0x2000_8000.

The vector table follows the initial value of MSP. Cortex®-M4 operates in Thumb state, and thus each value in the vector table must set the LSB to 1. In *Figure 1-6*, 0x0000_0101 is used to represent 0x0000_0100. After the instruction at 0x0000_0100 is executed, the program starts running formally. Before that, it is a must for initializing MSP, because the first instruction may be interrupted by NMI or other faults before being executed. After the completion of MSP initialization, it is ready to prepare stack room for its service routines.

Figure 1-6 Example of MSP and PC initialization



In the AT32F415 series, the main Flash memory, Boot code or SRAM can be remapped to the code area between 0x0000_0000 and 0x07FF_FFFF. BOOT1 and BOOT0 are used to set the specific memory from which CODE starts.

{BOOT1, BOOT0}=00/10, CODE starts from the main Flash memory

{BOOT1, BOOT0}=01, CODE starts from Boot code

{BOOT1, BOOT0}=11, CODE starts from SRAM

After a system reset or when leaving from Standby mode, the pin values of both BOOT1 and BOOT0 will be relatched.

Boot code memory contains an embedded boot loader program that provides not only Flash programming function through USART1, USART2 or USB interface, but also provides extra firmware including communication protocol stacks that can be called for use by software developer through API.

1.2 List of abbreviations for registers

Table 1-4 List of abbreviations for registers

Register type	Description
rw	Software can read and write to this bit.
ro	Software can only read this bit.
wo	Software can only write to the bit. Reading it returns its reset value.
rrc	Software can read this bit. Reading this bit automatically clears it.
rw0c	Software can read this bit and clear it by writing 0. Writing 1 has no effect on this bit.
rw1c	Software can read this bit and clear it by writing 1. Writing 0 has no effect on this bit.
rw1s	Software can read this bit and set it by writing 1. Writing 0 has no effect on this bit.
tog	Software can read this bit and toggle it by writing 1. Writing 0 has no effect on this bit.
rwt	Software can read this bit. Writing any value will trigger an event.
resd	Reserved.

1.3 Device characteristics information

Table 1-5 List of abbreviations for registers

Register abbr.	Base address	Reset value
F_SIZE	0x1FFF F7E0	0xXXXX
UID[31: 0]	0x1FFF F7E8	0xXXXX XXXX
UID[63: 32]	0x1FFF F7EC	0xXXXX XXXX
UID[95: 64]	0x1FFF F7F0	0xXXXX XXXX

1.3.1 Flash memory size register

This register contains the information about Flash memory size.

Bit	Abbr.	Reset value	Type	Description
Bit 15: 0	F_SIZE	0xXXXX	ro	Flash size, in terms of Kbyte For example: 0x0080 = 128 Kbyte

1.3.2 Device electronic signature

The device electronic signature contains the memory size and the unique device ID (96 bits). It is stored in the information block of the Flash memory. The 96-bit ID is unique for any device, and cannot be altered by users. It can be used for the following:

- Serial number: such as USB string serial number
- Part of security keys

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[31: 0]	0xXXXX XXXX	ro	UID for bit 31 to bit 0

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[63: 32]	0xXXXX XXXX	ro	UID for bit 63 to bit 32

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UID[95: 64]	0xXXXX XXXX	ro	UID for bit 95 to bit 64

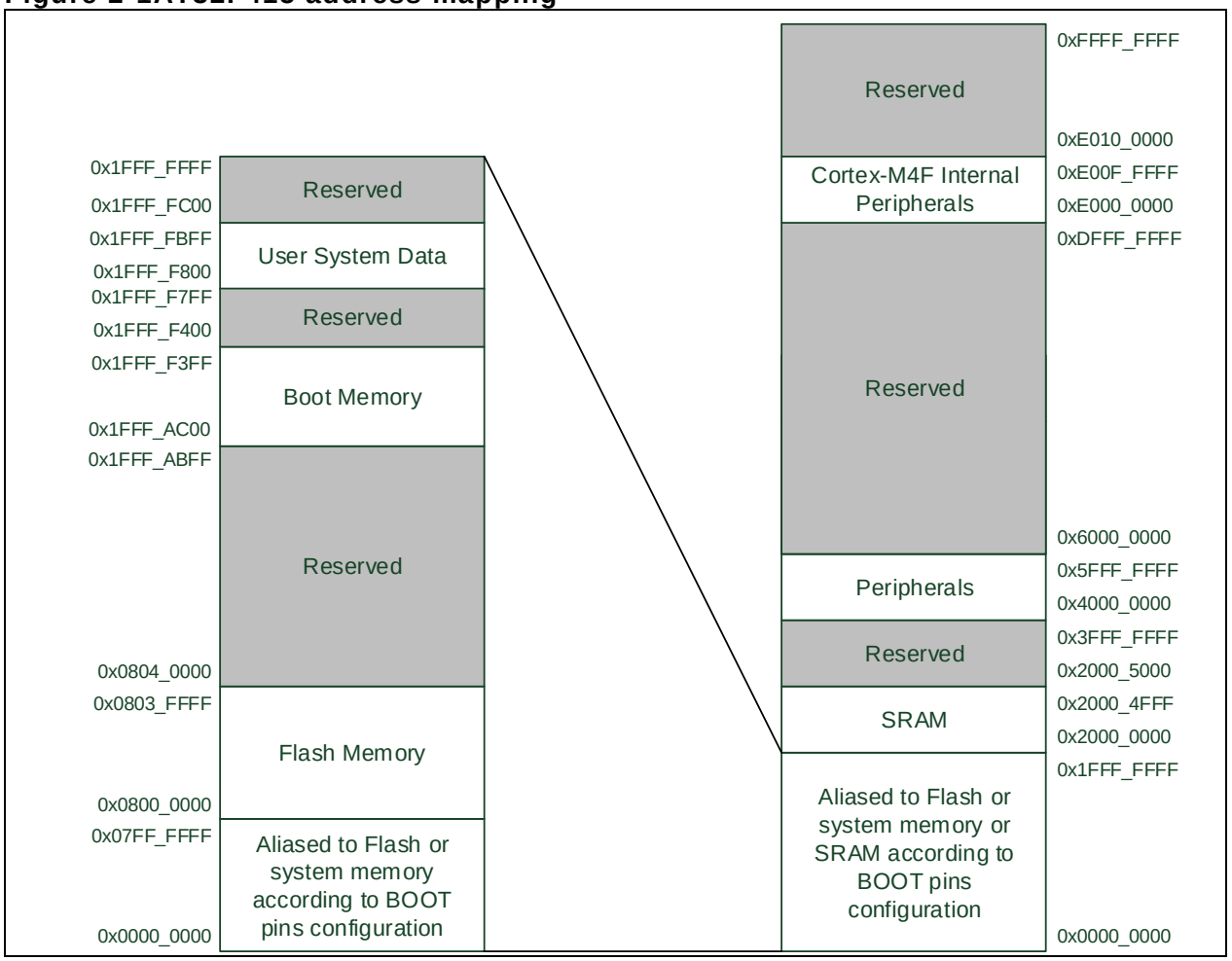
Note: UID[95:88] is series ID, which is 0x05 for AT32F415.

2 Memory resources

2.1 Internal memory address map

Internal memory contains program memory (Flash), data memory (SRAM), peripheral registers and core registers. Their respective address mapping are shown in *Figure 2-1*.

Figure 2-1AT32F415 address mapping



2.2 Flash memory

AT32F415 series provide up to 256 KB of on-chip Flash memory, supporting a single-cycle 32-bit read operation.

Refer to *Chapter 4.3.15* for more details about Flash memory controller and register configuration.

Flash memory organization (256 KB)

The main memory contains bank 1 (256 Kbytes), including 128 sectors, 2 Kbytes per sector.

Table 2-1 Flash memory organization (256 KB)

Block		Name	Address range
Main memory	Bank1 (256 KB)	Sector 0	0x0800 0000 – 0x0800 07FF
		Sector 1	0x0800 0800 – 0x0800 0FFF
		Sector 2	0x0800 1000 – 0x0800 17FF
		Sector 3	0x0800 1800 – 0x0800 1FFF
		Sector 4	0x0800 2000 – 0x0800 27FF
		.	.
		Sector 127	0x0803 F800 – 0x0803 FFFF
Information block		Boot loader	0x1FFF AC00 – 0x1FFF F3FF
		User system data	0x1FFF F800 – 0x1FFF FBFF

Flash memory organization (128 KB)

The main memory contains bank 1 (128 Kbytes), including 128 sectors, 1 Kbytes per sector.

Table 2-2 Flash memory organization (128 KB)

Block		Name	Address range
Main memory	Bank1 (128 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
		Sector 1	0x0800 0400 – 0x0800 07FF
		Sector 2	0x0800 0800 – 0x0800 0BFF
		Sector 3	0x0800 0C00 – 0x0800 0FFF
		Sector 4	0x0800 1000 – 0x0800 13FF
		.	.
		Sector 127	0x0801 FC00 – 0x0801 FFFF
Information block		Boot loader	0x1FFF AC00 – 0x1FFF F3FF
		User system data	0x1FFF F800 – 0x1FFF FBFF

Flash memory organization (64 KB)

The main memory contains bank 1 (64 Kbytes), including 64 sectors, 1 Kbytes per sector.

Table 2-3 Flash memory organization (64 KB)

Block		Name	Address range
Main memory	Bank1 (64 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
		Sector 1	0x0800 0400 – 0x0800 07FF
		Sector 2	0x0800 0800 – 0x0800 0BFF
		Sector 3	0x0800 0C00 – 0x0800 0FFF
		Sector 4	0x0800 1000 – 0x0800 13FF
		Sector 63	0x0800 FC00 – 0x0800 FFFF
Information block		Boot loader	0x1FFF AC00 – 0x1FFF F3FF
		User system data	0x1FFF F800 – 0x1FFF FBFF

2.3 SRAM memory

The AT32F415 series contain a 32-KB on-chip SRAM which starts at the address 0x2000_0000. It can be accessed by bytes, half-words (16 bit) or words (32 bit).

2.4 Peripheral address map

Table 2-4 Peripheral boundary address

Bus	Boundary address	Peripherals
AHB	0x6000 0000 - 0xFFFF FFFF	Reserved
	0x5004 0000 - 0x5FFF FFFF	Reserved
	0x5000 0000 - 0x5003 FFFF	OTGFS
	0x4002 8000 - 0x4FFF FFFF	Reserved
	0x4002 3400 - 0x4002 7FFF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2000 - 0x4002 23FF	Flash memory interface (FLASH)
	0x4002 1400 - 0x4002 1FFF	Reserved
	0x4002 1000 - 0x4002 13FF	Clock and reset manage (CRM)
	0x4002 0800 - 0x4002 0FFF	Reserved
	0x4002 0400 - 0x4002 07FF	DMA2
	0x4002 0000 - 0x4002 03FF	DMA1
	0x4001 8400 - 0x4001 7FFF	Reserved
	0x4001 8000 - 0x4001 83FF	SDIO1
APB2	0x4001 6000 - 0x4001 7FFF	Reserved
	0x4001 5800 - 0x4001 5BFF	Reserved
	0x4001 5400 - 0x4001 57FF	TMR11 timer
	0x4001 5000 - 0x4001 53FF	TMR10 timer
	0x4001 4C00 - 0x4001 4FFF	TMR9 timer
	0x4001 3C00 - 0x4001 4BFF	Reserved
	0x4001 3800 - 0x4001 3BFF	USART1
	0x4001 3400 - 0x4001 37FF	Reserved
	0x4001 3000 - 0x4001 33FF	SPI1/I^2S1
	0x4001 2C00 - 0x4001 2FFF	TMR1 timer
	0x4001 2800 - 0x4001 2BFF	Reserved
	0x4001 2400 - 0x4001 27FF	ADC1
	0x4001 2000 - 0x4001 23FF	Reserved
	0x4001 1C00 - 0x4001 1FFF	GPIO port F
	0x4001 1800 - 0x4001 1BFF	Reserved
	0x4001 1400 - 0x4001 17FF	GPIO port D
	0x4001 1000 - 0x4001 13FF	GPIO port C
	0X4001 0C00 - 0x4001 0FFF	GPIO port B
	0x4001 0800 - 0x4001 0BFF	GPIO port A
	0x4001 0400 - 0x4001 07FF	EXINT

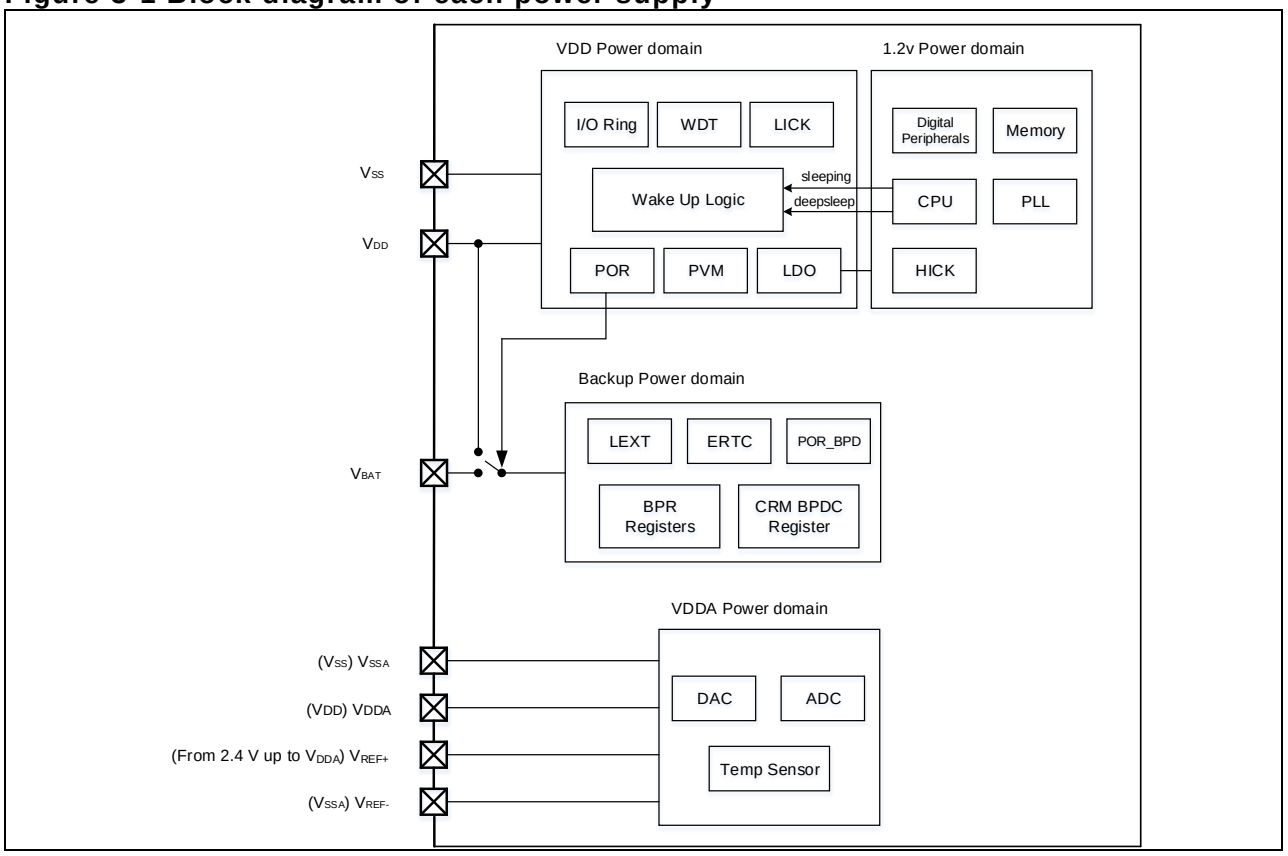
Bus	Boundary address	Peripherals
	0x4001 0000 - 0x4001 03FF	IOMUX
APB1	0x4000 8000 - 0x4000 FFFF	Reserved
	0x4000 7C00 - 0x4000 7FFF	Reserved
	0x4000 7800 - 0x4000 7BFF	Reserved
	0x4000 7400 - 0x4000 77FF	Reserved
	0x4000 7000 - 0x4000 73FF	Power control (PWC)
	0x4000 6800 - 0x4000 6FFF	Reserved
	0x4000 6400 - 0x4000 67FF	CAN1
	0x4000 6000 - 0x4000 63FF	Reserved
	0x4000 5C00 - 0x4000 5FFF	Reserved
	0x4000 5800 - 0x4000 5BFF	I^2C2
	0x4000 5400 - 0x4000 57FF	I^2C1
	0x4000 5000 - 0x4000 53FF	UART5
	0x4000 4C00 - 0x4000 4FFF	UART4
	0x4000 4800 - 0x4000 4BFF	USART3
	0x4000 4400 - 0x4000 47FF	USART2
	0x4000 4000 - 0x4000 43FF	Reserved
	0x4000 3C00 - 0x4000 3FFF	Reserved
	0x4000 3800 - 0x4000 3BFF	SPI2/I^2S2
	0x4000 3400 - 0x4000 37FF	Reserved
	0x4000 3000 - 0x4000 33FF	Watchdog timer (WDT)
	0x4000 2C00 - 0x4000 2FFF	Window watchdog timer (WWDT)
	0x4000 2800 - 0x4000 2BFF	ERTC
	0x4000 2400 - 0x4000 27FF	CMP controller
	0x4000 2000 - 0x4000 23FF	Reserved
	0x4000 1C00 - 0x4000 1FFF	Reserved
	0x4000 1800 - 0x4000 1BFF	Reserved
	0x4000 1400 - 0x4000 17FF	Reserved
	0x4000 1000 - 0x4000 13FF	Reserved
	0x4000 0C00 - 0x4000 0FFF	TMR5 timer
	0x4000 0800 - 0x4000 0BFF	TMR4 timer
	0x4000 0400 - 0x4000 07FF	TMR3 timer
	0x4000 0000 - 0x4000 03FF	TMR2 timer

3 Power control (PWC)

3.1 Introduction

For AT32F415 series, its operating voltage supply is 2.6 V ~ 3.6 V, with a temperature range of -40~+105 ℃. To reduce power consumption, this series provides three types of power saving modes, including Sleep, Deepsleep and Standby modes so as to achieve the best trade-off among the conflicting demands of CPU operating time, speed and power consumption. The AT32F415 series has three power domains —VDD/VDDA domain, 1.2 V domain and battery powered domain. The VDD/VDDA domain is supplied directly by external power, the 1.2 V domain is powered by an embedded LDO in the VDD/VDDA domain, and the battery powered domain is supplied through a V_{BAT} pin.

Figure 3-1 Block diagram of each power supply



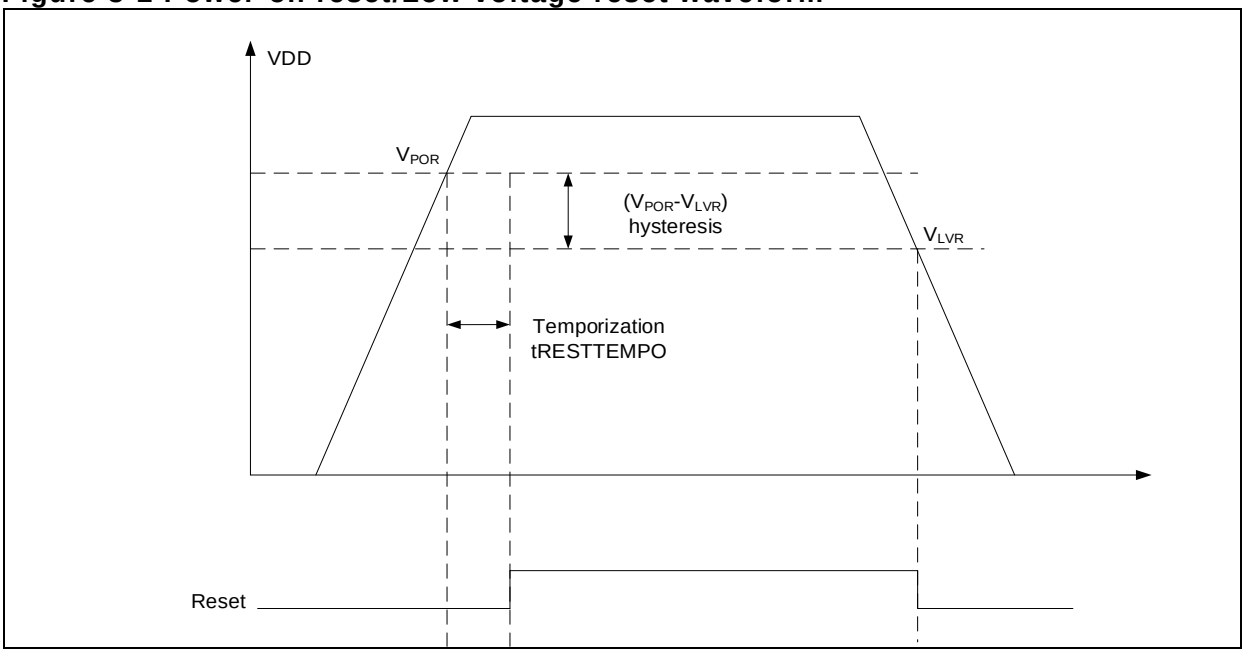
3.2 Main Features

- Three power domains: VDD/VDDA domain,1.2 V domain and battery powered domain
- Three types of power saving modes: Sleep mode, Deepsleep mode, and Standby mode
- Internal voltage regulator supplies 1.2 V voltage source for the core domain
- Power voltage detector is provided to issue an interrupt when the supply voltage is lower or higher than a programmed threshold
- The battery powered domain is powered by V_{BAT} when VDD is powered off
- VDD/VDDA applies separated digital and analog module to reduce noise on external power

3.3 POR/LVR

A POR analog module embedded in the VDD/VDDA domain is used to generate a power reset. The power reset signal is released at V_{POR} when the VDD is increased from 0 V to the operating voltage, or it is triggered at V_{LVR} when the VDD drops from the operating voltage to 0 V. During the power-on reset period, the reset signal has certain amount of time delay compared to VDD boost process. At the same time, hysteresis occurs in power-on reset (POR) and low voltage reset (LVR).

Figure 3-2 Power-on reset/Low voltage reset waveform

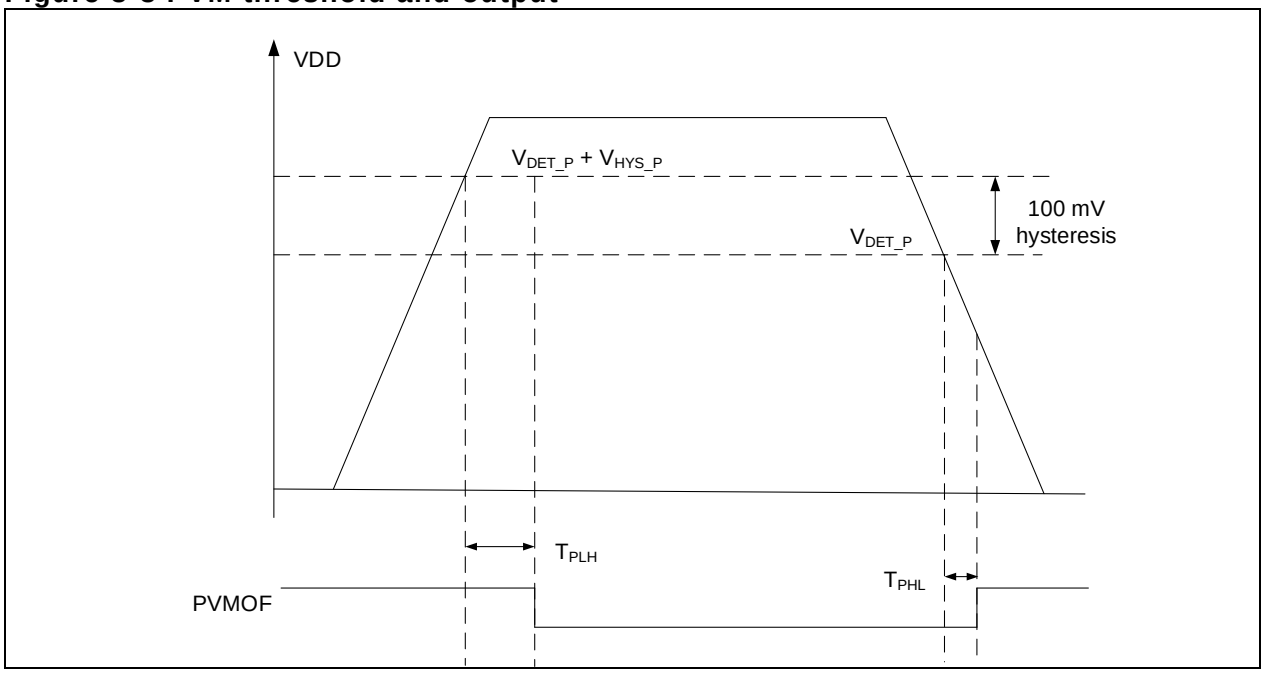


3.4 Power voltage monitor (PVM)

The PVM is used to monitor the power supply variations. It is enabled by setting the PVMEN bit in the power control register (PWC_CTRL), and the threshold value for voltage monitor is selected with the PVMSEL[2: 0].

After PVM is enabled, the comparison result between VDD and the programmed threshold is indicated by the PVMOF bit in the PWC_CTRLSTS register, with the hysteresis voltage VHYS_P being 100 mv. The PVM interrupt will be generated through the EXTI line 16 when VDD rises above the PVM threshold.

Figure 3-3 PVM threshold and output



3.5 Power domain

1.2 V domain

1.2 V core domain includes a CPU core, SRAM, embedded digital peripherals and Phase Locked Loop (PLL). Such power domain is supplied by LDO (voltage regulator). It can be configured as power-on, low-power or power-off in three power saving modes.

VDD/VDDA domain

VDD/VDDA domain includes VDD domain and VDDA domain. The VDD domain contains I/O circuit, power-saving mode wakeup circuit, watchdog timer (WDT), power-on reset/low voltage reset (POR/LVR), LDO, ERTC circuit, LEXT oscillator and all PAD circuits except PC13, PC14 and PC15. The VDDA domain contains an ADC (AD converter), temperature sensor and so on.

Typically, to ensure a better accuracy of ADC at a low voltage, the digital circuit is supplied by VDD while the analog circuit is powered by VDDA. The external reference voltage VREF+ and VREF- are connected to the VDDA pin and VSSA pin, respectively.

Battery powered domain

The battery powered domain contains ERTC circuit, LEXT oscillator, PC13, PC14 and PC15, which is powered by either VDD or VBAT pin. When the VDD is cut off, the battery powered domain is automatically switched to VBAT pin to ensure that RTC can work normally.

1) When the battery powered domain is powered by VDD, the PC13 can be used as a general-purpose I/O, tamper pin, ERTC calibration clock, ERTC alarm or second output, while the PC14 and PC15 can be used as a GPIO or LEXT pin. (As an I/O port, PC13, PC14 and PC15 must be limited below 2 MHz, and to the maximum load of 30 pF, and these I/O ports must not be used as current sources)
2) When the battery powered domain is powered by V_{BAT}, the PC13 can be used as a tamper pin, ERTC alarm or second output, while the PC14 and PC15 can only be used as a LEXT pin.

The switch of the battery powered domain will not be disconnected from V_{BAT} because of the VDD being at its rising phrase or due to VDD low voltage reset. If the power switch has not been switched to the VDD when the VDD is powered on quickly, it is recommended to add a low voltage drop diode between VDD and V_{BAT} in order to prevent the currents of VDD from being injected to VBAT. If there is no external battery in the application, it is better to connect the VBAT to a 100 nF ceramic filter capacitor that is externally connected to VDD.

3.6 Power saving modes

When the CPU does not need to be kept running, there are three types of low-power modes available (Sleep mode, Deepsleep mode and Standby mode) to save power. Users can select the mode that gives the best compromise according to the low-power consumption, short startup time, and available wakeup sources. In addition, the power consumption in Run mode can be reduced by slowing down the system clocks or gating the clocks to the APB and AHB peripherals when they are not used.

Sleep mode

The Sleep mode is entered by executing WFI or WFE instruction. There are two options to select the Sleep mode entry mechanism through the SLEEPONEXIT bit in the Cortex ® -M4 system control register.

SLEEP-NOW mode:

When SLEEPDEEP=0 and SLEEPONEXIT=0, the MCU enters Sleep mode as soon as WFI or WFE instruction is executed.

When SLEEPDEEP=0 and SLEEPONEXIT=1, the MCU enters Sleep mode as soon as the system exits the lowest-priority interrupt service routine by executing the WFI instruction.

In Sleep mode, all clocks and LDO work normally except CPU clocks (stopped), and all I/O pins keep the same state as in Run mode. The LDO provides a 1.2 V power (for CPU core, memory and embedded peripherals) as it is in normal power consumption mode. The LDO output voltage is configurable by the PWC_LDOOV register.

1) If the WFI is executed to enter Sleep mode, any peripheral interrupt can wake up the device from Sleep mode.
2) If the WFE is executed to enter Sleep mode, the MCU exits Sleep mode as soon as an event occurs.

The wakeup event can be generated by the following:

- Enabling a peripheral interrupt (it is not enabled in the NVIC) and enabling the SEVONPEND bit. When the MCU resumes, the peripheral interrupt pending bit and NVIC channel pending bit must be cleared.
- Configuring an internal EXINT line as an event mode to generate a wakeup event.

The wakeup time required by a WFE instruction is the shortest, since no time is wasted on interrupt entry/exit.

Deepsleep Mode

Deepsleep mode is entered by setting the SLEEPDEEP bit in the Cortex®-M4 system control register and clearing the LPSEL bit in the power control register before WFI or WFE instructions.

The LDO status is selected by setting the VRSEL bit in the power control register (PWC_CTRL). When VRSEL=0, the LDO works in normal mode. When VRSEL=1, the LDO is set in low-power consumption mode.

In Deepsleep mode, all clocks in 1.2 V domain are stopped, and both HICK and HEXT oscillators are disabled. The LDO supplies power to the 1.2 V domain in normal mode or low-power mode. All I/O pins keep the same state as in Run mode. SRAM and register contents are preserved.

1) When the Sleep mode is entered by executing a WFI instruction, the interrupt generated on any external interrupt line in Interrupt mode can wake up the system from Deepsleep mode.
2) When the Sleep mode is entered by executing a WFE instruction, the interrupt generated on any external interrupt line in Event mode can wake up the system from Deepsleep mode.

When the MCU exits the Deepsleep mode, the HICK RC oscillator is enabled and selected as a system clock after stabilization. When the LDO operates in low-power mode, an additional wakeup delay is incurred for the reason that the LDO must be stabilized before the system is waken from the Deepsleep mode.

Standby Mode

Standby mode can achieve the lowest power consumption for the device. In this mode, the LDO is disabled. The whole 1.2 V domain, PLL, HICK and HEXT oscillators are also powered off except VDD/VDDA domain. SRAM and register contents are lost.

The Standby mode is entered by the following procedures:

- Set the SLEEPDEE bit in the Cortex®-M4 system control register
- Set the LPSEL bit in the power control register (PWC_CTRL)
- Clear the SWEF bit in the power control/status register (PWC_CTRLSTS)
- Execute a WFI/WFE instruction

In Standby mode, all I/O pins remain in a high-impedance state except reset pins, TAMPER pins that are set as anti-tamper or calibration output, and the wakeup pins.

The MCU leaves the Standby mode when an external reset (NRST pin), a WDT reset, ERTC periodic wakeup, ERTC timestamp, ERTC tamper event and a rising edge on the WKUP pin or the rising edge of an ERTC alarm event occurs.

Debug mode

By default, the debug connection is lost if the MCU enters Deepsleep mode or Standby mode while debugging. The reason is that the Cortex®-M4 core is no longer clocked. However, the software can be debugged even in the low-power mode by setting some configuration bits in the DEBUG register (DEBUG_CTRL).

3.7 PWC registers

The peripheral registers must be accessed by words (32 bit)

Table 3-1 PW register map and reset values

Register abbr.	Offset	Reset value
PWC_CTRL	0x00	0x0000 0000
PWC_CTRLSTS	0x04	0x0000 0000

3.7.1 Power control register (PWC_CTRL)

Bit	Name	Reset value	Type	Description
Bit 31: 9	Reserved	0x0000 00	resd	Kept at its default value.
Bit 8	BPWEN	0	rw	Battery powered domain write enable 0: Disabled 1: Enabled Note: After reset, ERTC is write protected. To write the battery powered domain, this bit must be enabled.
Bit 7: 5	PVMSEL	0x0	rw	Power voltage monitoring boundary select 000: Unused, not configurable 001: 2.3 V 010: 2.4 V 011: 2.5 V 100: 2.6 V 101: 2.7 V 110: 2.8 V 111: 2.9 V
Bit 4	PVMEN	0	rw	Power voltage monitoring enable 0: Disabled 1: Enabled
Bit 3	CLSEF	0	wo	Clear SEF flag 0: No effect 1: Clear the SEF flag Note: This bit is cleared by hardware after clearing the SEF flag. Reading this bit at any time will return all zero.
Bit 2	CLSWEF	0	wo	Clear SWEF flag 0: No effect 1: Clear the SWEF flag Note: Clear the SWEF flag after two system clock cycles. This bit is cleared by hardware after clearing the SWEF flag. Reading this bit at any time will return all zero.
Bit 1	LPSEL	0	rw	Low power mode select when Cortex®-M4F sleepdeep 0: Enter DEEPSLEEP mode 1: Enter Standby mode
Bit 0	VRSEL	0	rw	Voltage regulator state select in Deepsleep mode 0: Enabled 1: Low-power consumption mode

3.7.2 Power control/status register (PWC_CTRLSTS)

Unlike a standard APB read, an additional APB cycles are needed to read this register.

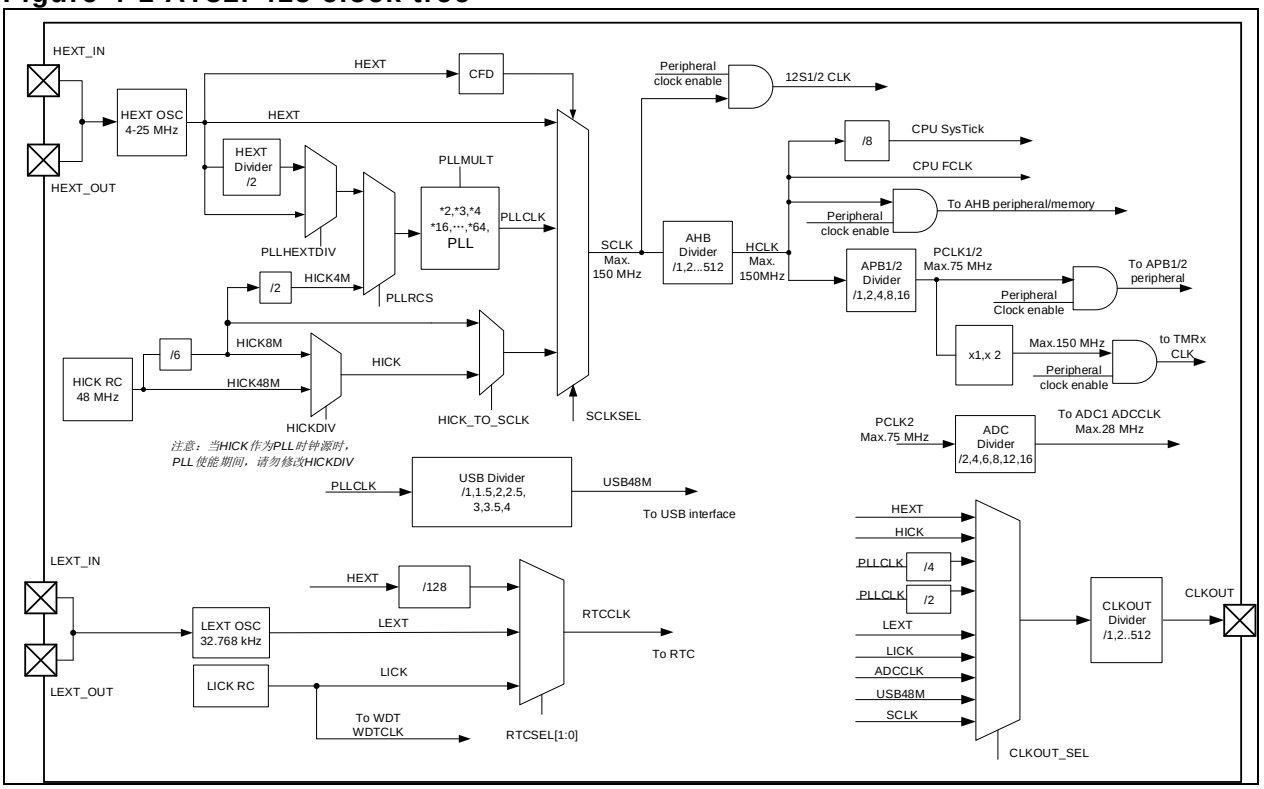
Bit	Name	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	SWPEN	0	rw	Standby wake-up pin enable 0: Disabled (this pin is used for general-purpose I/O) 1: Enabled (this pin is forced in input pull-down mode, and no longer used for general-purpose I/O) Note: This bit is cleared by hardware after system reset. In Standby mode, this bit is forced to input pull-down mode irrespective of whether this wake-up pin is enabled
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	PVMOF	0	ro	Power voltage monitoring output flag 0: Power voltage is higher than the threshold 1: Power voltage is lower than the threshold Note: The power voltage monitor is stopped in Standby mode.
Bit 1	SEF	0	ro	Standby mode entry flag 0: Device is not in Standby mode 1: Device is in Standby mode Note: This bit is set by hardware (enter Standby mode) and cleared by POR/LVR or by setting the CLSEF bit.
Bit 0	SWEF	0	ro	Standby wake-up event flag 0: No wakeup event occurred 1: A wakeup event occurred Note: This bit is set by hardware (on a wakeup event), and cleared by POR/LVR or by setting the CLSWEF bit. A wakeup event is generated by one of the following: When the rising edge on the Standby wakeup pin occurs; When the ERTC alarm event occurs; If the Standby wakeup pin is enabled when the Standby wakeup pin level is high.

4 Clock and reset manage (CRM)

4.1 Clock

AT32F415 series provide different clock sources: HEXT oscillator clock, HICK oscillator clock, PLL clock, LEXT oscillator and LICK oscillator.

Figure 4-1 AT32F415 clock tree



- HEXT (high speed external crystal)
- HICK (high speed internal clock)
- PLL (phased-locked loops)
- LEXT (low speed external crystal)
- LICK (low speed internal clock)

AHB, APB1 and APB2 all support multiple frequency division. The AHB domain has a maximum of 150 MHz, and both APB1 and APB2 are up to 75 MHz.

4.1.1 Clock sources

- High speed external oscillator (HEXT)

The HEXT includes two clock sources: crystal/ceramic resonator and bypass clock.

The HEXT crystal/ceramic resonator is connected externally to a 4~25 MHz HEXT crystal that produces a highly accurate clock for the system. The HEXT clock signal is not released until it becomes stable.

An external clock source can be provided by HEXT bypass. Its frequency can be up to 25 MHz. The external clock signal should be connected to the HEXT_IN pin while the HEXT_OUT pin should be left floating.

- High speed internal clock (HICK)

The HICK oscillator is clocked by a high-speed RC in the microcontroller. The internal frequency of the HICK clock is 48 MHz. Although it is less accurate, the HICK clock frequency of each device is calibrated to 1% accuracy (25°C) in factory. The factory calibration value is loaded in the HICKCAL[7: 0] bit of the clock control register. The RC oscillator speed may be affected by voltage or temperature variations. Thus the HICK frequency can be trimmed using the HICKTRIM[5: 0] bit in the clock control register.

The HICK clock signal is not released until it becomes stable.

- PLL clock

PLL configuration mode: regular integer frequency multiplication mode, and flexible configuration mode

1) Regular integer frequency multiplication mode (default mode)

   PLL clock calculation formula:

   PLL output clock = PLL input clock x PLL frequency multiplication factor

   Configuration procedures:

   a) Clear PLLCFGEN (RCC_PLL[31])
   b) Set PLL input clock frequency, refer to PLL_FREF (RCC_PLL[26:24]) for details
   c) Set PLL frequency multiplication factor, refer to PLLMUL (RCC_CFG[30,29],RCC_CFG[21:18])
   d) Enable PLL
   e) Wait for PLL to be stabilized

2) Flexible configuration mode

   PLL clock calculation formula:

   PLL output clock = PLL input clock x PLL frequency multiplication factor / (PLL pre-frequency division factor x PLL post-frequency division factor)

   500MHz <= PLL input clock x PLL frequency multiplication factor / PLL pre-frequency division factor <= 1000MHz

   2MHz <= PLL input clock / PLL pre-frequency division factor <= 16MHz

   Configuration procedures:

   a) Set the PLL_NS, PLL_MS and PLL_FR bits of the RCC_PLL register respectively using the calculated PLL frequency multiplication factor, PLL pre-frequency division factor and PLL post-frequency division factor
   b) Enable PLLCFGEN (RCC_PLL[31])
   c) Enable PLL
   d) Wait for PLL to be stabilized

Note: The PLL_FREF and PLLMUL registers are not available in flexible configuration mode.

Note: In flexible configuration mode, non-integer frequency multiplication is supported.

Example: when PLL input clock is 12.288 MHz, PLL output frequency is equal to 12.288 x 125 / (2 x 8) = 96 MHz

When PLL input clock is 5 MH, PLL output frequency is equal to 5 x 108 / (5 x 1) = 108 MHz

- Low speed external oscillator (LEXT)

The LEXT oscillator provides two clock sources: LEXT crystal/ceramic resonator and LEXT bypass.

LEXT crystal/ceramic resonator:

The LEXT crystal/ceramic resonator provides a 32.768 KHz low-speed clock source. The LEXT clock signal is not released before it becomes stable.

LEXT bypass clock:

In this mode, an external clock source with a frequency of 32.768 kHz can be provided. The external clock signal should be connected to the LEXT_IN pin while the LEXT_OUT must remain floating.

- Low speed internal RC oscillator (LICK)

The LICK oscillator is clocked by an internal low-speed RC oscillator. The clock frequency is between 30 kHz and 60 kHz. It acts as a low-power clock source that can be kept running in Deepsleep mode and Standby mode for watchdog and auto-wakeup unit.

The LICK clock signal is not released before it becomes stable.

4.1.2 System clock

After a system reset, the HICK oscillator is selected as system clock. The system clock can make flexible switch among HICK oscillator, HEXT oscillator and PLL clock. However, a switch from one clock source to another occurs only when the target clock source becomes stable. When the HICK oscillator is used directly or indirectly through the PLL as the system clock, it cannot be stopped.

4.1.3 Peripheral clock

Most peripherals use HCLK, PCLK1 or PCLK2 clock. The individual peripherals have their dedicated clocks.

System Tick timer (SysTick) is clocked by HCLK or HCLK/8.

ADC is clocked by APB2 divided by 2, 4, 6, 8, 12, 16.

The timers are clocked by APB1/2. In particular, if the APB prescaler is 1, the timer clock frequency is equal to that of APB1/2; otherwise, the timer clock frequency doubles that of the APB1/2 frequency.

A frequency-divided PLL clock can be used as the clock source of USB. If the PLL frequency divider is selected, the USB frequency divider provides a 48 MHz USBCLK, and thus the PLL must be set as 48*N*0.5 MHz (N=2,3,4,5…)

ERTC clock sources: HEXT/128 oscillator, LEXT oscillator and LICK oscillator. Once the clock source is selected, it cannot be altered without resetting the battery powered domain. If the LEXT is used as an ERTC clock, the ERTC is not affected when the VDD is powered off. If the HEXT or LICK is selected as an ERTC clock, the ERTC state is not guaranteed when both HEXT and LICK are powered off.

Watchdog is clocked by LICK oscillator. If the watchdog is enabled by either hardware option or software access, the LICK oscillator is forced ON. The clock is provided to the watchdog only after the LICK oscillator temporization.

4.1.4 Clock fail detector

The clock fail detector (CFD) is designed to respond to HEXT clock failure when the HEXT is used as a system clock, directly or indirectly. If a failure is detected on the HEXT clock, a clock failure event is sent to the break input of TMR1 and an interrupt is generated. This interrupt is directly linked to CPU NMI so that the software can perform rescue operations. The NMI interrupt keeps executing until the CFD interrupt pending bit is cleared. This is why the CFD interrupt has to be cleared in the NMI service routine. The HEXT clock failure will result in a switch of the system clock to the HICK clock, the CFD to be disabled , HEXT clock to be stopped, and even PLL to be disabled if the HEXT clock is selected as the system clock through PLL.

4.1.5 Auto step-by-step system clock switch

The automatic frequency switch is designed to ensure a smooth and stable switch of system frequency when the system clock source is switched from others to the PLL or when the AHB prescaler is changed from large to small. When the operational target is larger than 108 MHz, it is recommended to enable the automatic frequency switch. Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain working, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes.

4.1.6 Internal clock output

The microcontroller allows the internal clock signal to be output to external CLKOUT pins. That is, ADCCLK, USB48M, SCLK, LICK, LEXT, HICK, HEXT, PLLCLK/2 and PLLCLK/4 can be used as CLKOUT clocks.

4.1.7 Interrupts

The microcontroller specifies a stable flag for each clock source. As a result, when a clock source is enabled, it is possible to determine if the clock is stable by checking the flag pertaining to the clock source. An interrupt request is generated when the interrupt corresponding to the clock source is enabled.

If a failure is detected on the HEXT clock, the CFD interrupt is generated. Such interrupt is directly linked to CPU NMI.

4.2 Reset

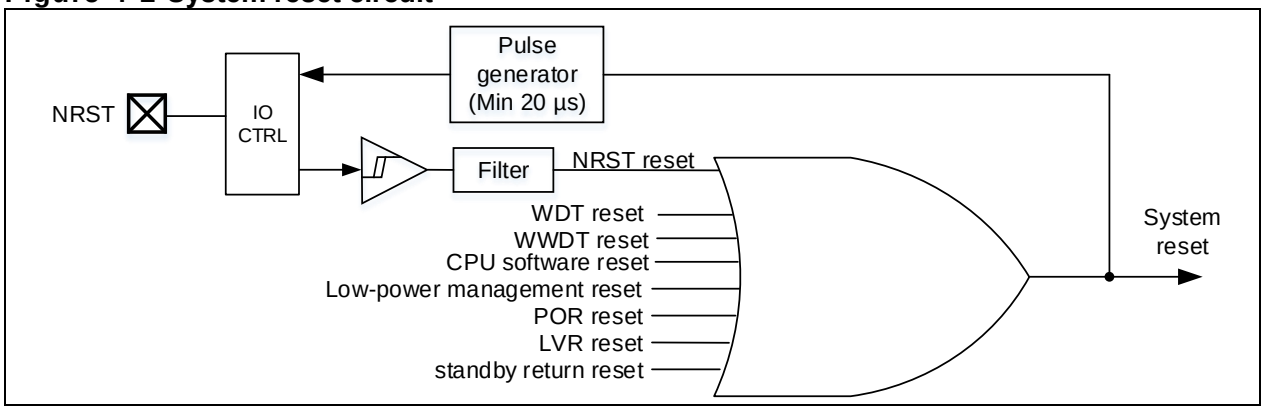
4.2.1 System reset

AT32F415 series provide the following system reset sources:

- NRST reset: on the external NRST pin
- WDT reset: watchdog overflow
- WWDT reset: window watchdog overflow
- CPU software reset: Cortex®-M4 software reset
- Low-power management reset: This type of reset is enabled when entering Standby mode (by clearing the nSTDBY_RST bit in the user system data area); this type of reset is also enabled when entering Deepsleep mode (by clearing the nDEPSLP_RST in the user system data area).
- When exiting Standby mode

NRST reset, WDT reset, WWDT reset, software reset and low-power management reset sets all registers to their reset values except the clock control/status register (CRM_CTRLSTS) and the battery powered domain; the power-on reset, low-voltage reset or reset generated when exiting Standby mode sets all registers to their reset values except the battery powered domain registers.

Figure 4-2 System reset circuit



4.2.2 Battery powered domain reset

Battery powered domain has two specific reset sources:

- Software reset: triggered by setting the BPDRST bit in the battery powered domain control register (CRM_BPDC)
- VDD or VBAT power on, if both supplies have been powered off.

Software reset affects only the battery powered domain.

4.3 CRM registers

These peripheral registers have to be accessed by bytes (8 bits), half words (16 bits) or words (32 bits).

Table 4-1 CRM register map and reset values

Register	Offset	Reset value
CRM_CTRL	0x000	0x0000 XX83
CRM_CFG	0x004	0x0000 0000
CRM_CLKINT	0x008	0x0000 0000
CRM_APB2RST	0x00C	0x0000 0000
CRM_APB1RST	0x010	0x0000 0000
CRM_AHBEN	0x014	0x0000 0014
CRM_APB2EN	0x018	0x0000 0000
CRM_APB1EN	0x01C	0x0000 0000
CRM_BPDC	0x020	0x0000 0000
CRM_CTRLSTS	0x024	0x0C00 0000
CRM_AHBRST	0x028	0x0000 0000
CRM_PLL	0x02C	0x0000 1F10
CRM_MISC1	0x030	0x0000 0000
CRM_OTG_EXTCTRL	0x044	0x0000 0000
CRM_MISC2	0x054	0x0000 000D

4.3.1 Clock control register (CRM_CTRL)

No-wait states, accessible by bytes, half-words or words.

Bit	Name	Reset value	Type	Description
Bit 31: 26	Reserved	0x00	resd	Kept at its default value.
Bit 25	PLLSTBL	0x0	ro	PLL clock stable This bit is set by hardware after PLL is ready. 0: PLL clock is not ready. 1: PLL clock is ready.
Bit 24	PLLEN	0x0	rw	PLL enable This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the PLL clock is used as the system clock, this bit cannot be cleared. 0: PLL is OFF 1: PLL is ON.
Bit 23: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	CFDEN	0x0	rw	Clock failure detector enable 0: OFF 1: ON
Bit 18	HEXTBYPS	0x0	rw	High speed external crystal bypass This bit can be written only if the HEXT is disabled. 0: OFF 1: ON
Bit 17	HEXTSTBL	0x0	ro	High speed external crystal stable

				This bit is set by hardware after HEXT becomes stable. 0: HEXT is not ready. 1: HEXT is ready.
Bit 16	HEXTEN	0x0	rw	High speed external crystal enable This bit is set and cleared by software. It can also be cleared by hardware when entering Standby or Deepsleep mode. When the HEXT clock is used as the system clock, this bit cannot be cleared 0: OFF. 1: ON
Bit 15: 8	HICKCAL	0xXX	rw	High speed internal clock calibration The default value of this field is the initial factory calibration value. When the HICK output frequency is 48 MHz, it needs adjust 240 kHz (design value) based on this frequency for each HICKCAL value change; when HICK output frequency is 8 MHz (design value), it needs adjust 40 kHz based on this frequency for each HICKCAL value change. Note: This bit can be written only if the HICKCAL_KEY[7: 0] is set as 0x5A.
Bit 7: 2	HICKTRIM	0x20	rw	High speed internal clock trimming These bits work with the HICKCAL[7: 0] to determine the HICK oscillator frequency. The default value is 32, which can trim the HICK to be ±1%.
Bit 1	HICKSTBL	0x1	ro	High speed internal clock stable This bit is set by hardware after the HICK is ready. 0: Not ready 1: Ready
Bit 0	HICKEN	0x1	rw	High speed internal clock enable This bit is set and cleared by software. It can also be set by hardware when exiting Standby or Deepsleep mode. When a HEXT clock failure occurs. This bit can also be set. When the HICK is used as the system clock, this bit cannot be cleared. 0: Disabled 1: Enabled

4.3.2 Clock configuration register (CRM_CFG)

Access: 0 to 2 wait states, accessible by bytes, half-words or words. 1 or 2 wait states are inserted only when the access occurs during a clock source switch.

Bit	Name	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 26:24	CLKOUT_SEL	0x0	rw	Clock output selection CLKOUT_SEL[3] is the bit 16 of the CRM_MISC1 register. 0000: None 0001: Reserved 0010: LICK 0011: LEXT 0100: SCLK 0101: HICK 0110: HEXT 0111: PLL/2 1100: PLL/4 1101: USB 1110: ADC
Bit 27 Bit 23: 22	USBDIV	0x0	rw	USB division The PLL clock after division is used as USB clock. 000: PLL/1.5 001: Forbidden 010: PLL/2.5 011: PLL/2 100: PLL/3.5 101: PLL/3 110: PLL/4

				111: PLL/4
Bit 30: 29 Bit 21: 18	PLLMULT	0x00	rw	PLL multiplication factor 000000: PLL x 2 000001: PLL x 3 000010: PLL x 4 000011: PLL x 5 …… 001100: PLL x 14 001101: PLL x 15 001110: PLL x 16 001111: PLL x 16 010000: PLL x 17 010001: PLL x 18 010010: PLL x 19 010011: PLL x 20 …… 111110: PLL x 63 111111: PLL x 64 Note: The PLLRANGE bit must be programmed based on the PLL multiplication value.
Bit 17	PLLHEXTDIV	0x0	rw	HEXT division selection for PLL entry clock 0: No division 1: HEXT/2
Bit 16	PLLRCS	0x0	rw	PLL reference clock select 0: HICK-divided clock (4MHz) 1: HEXT clock
Bit 28 Bit 15: 14	ADCDIV	0x0	rw	ADC division The PCLK that is divided by the following factors serves the ADC. 000: PCLK/2 001: PCLK/4 010: PCLK/6 011: PCLK/8 100: PCLK/2 101: PCLK/12 110: PCLK/8 111: PCLK/16
Bit 13: 11	APB2DIV	0x0	rw	APB2 division The divided HCLK is used as APB2 clock. 0xx: not divided 100: divided by 2 101: divided by 4 110: divided by 8 111: divided by 16 Note: The software must set these bits correctly to ensure that the APB2 clock frequency does not exceed 75 MHz.
Bit 10: 8	APB1DIV	0x0	rw	APB1 division The divided HCLK is used as APB1 clock. 0xx: not divided 100: divided by 2 101: divided by 4 110: divided by 8 111: divided by 16 Note: The software must set these bits correctly to ensure that the APB1 clock frequency does not exceed 75 MHz
Bit 7: 4	AHBDIV	0x0	rw	AHB division The divided SCLK is used as AHB clock. 0xxx: SCLK not divided 1000: SCLK divided by 2 1100: SCLK divided by 64 1001: SCLK divided by 4 1101: SCLK divided by 128 1010: SCLK divided by 8 1110: SCLK divided by 256 1011: SCLK divided by 16 1111: SCLK divided by 512
Bit 3: 2	SCLKSTS	0x0	R0	System clock select status 00: HICK 01: HEXT 10: PLL 11: Reserved. Kept at its default value.
Bit 1: 0	SCLKSEL	0x0	rw	System clock select 00: HICK

				01: HEXT 10: PLL 11: Reserved. Kept at its default value.

4.3.3 Clock interrupt register (CRM_CLKINT)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CFDFC	0x0	wo	Clock failure detection flag clear Writing 1 by software to clear CFDF. 0: No effect 1: Clear
Bit 22: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	PLLSTBLFC	0x0	wo	PLL stable flag clear Writing 1 by software to clear PLLSTBLF. 0: No effect 1: Clear
Bit 19	HEXTSTBLFC	0x0	wo	HEXT stable flag clear Writing 1 by software to clear HEXTSTBLF. 0: No effect 1: Clear
Bit 18	HICKSTBLFC	0x0	wo	HICK stable flag clear Writing 1 by software to clear HICKSTBLF. 0: No effect 1: Clear
Bit 17	LEXTSTBLFC	0x0	wo	LEXT stable flag clear Writing 1 by software to clear LEXTSTBLF. 0: No effect 1: Clear
Bit 16	LICKSTBLFC	0x0	wo	LICK stable flag clear Writing 1 by software to clear LICKSTBLF. 0: No effect 1: Clear
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	PLLSTBLIEN	0x0	rw	PLL stable interrupt enable 0: Disabled 1: Enabled
Bit 11	HEXTSTBLIEN	0x0	rw	HEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 10	HICKSTBLIEN	0x0	rw	HICK stable interrupt enable 0: Disabled 1: Enabled
Bit 9	LEXTSTBLIEN	0x0	rw	LEXT stable interrupt enable 0: Disabled 1: Enabled
Bit 8	LICKSTBLIEN	0x0	rw	LICK stable interrupt enable 0: Disabled 1: Enabled
Bit 7	CFDF	0x0	ro	Clock Failure Detection flag This bit is set by hardware when the HEXT clock failure occurs. 0: No clock failure 1: Clock failure
Bit 6: 5	Reserved	0x0	resd	Keep at its default value.
Bit 4	PLLSTBLF	0x0	ro	PLL stable flag Set by hardware. 0: PLL is not ready. 1: PLL is ready.
Bit 3	HEXTSTBLF	0x0	ro	HEXT stable flag Set by hardware. 0: HEXT is not ready. 1: HEXT is ready.
Bit 2	HICKSTBLF	0x0	ro	HICK stable flag

				Set by hardware. 0: HICK is not ready. 1: HICK is ready.
Bit 1	LEXTSTBLF	0x0	ro	LEXT stable flag Set by hardware. 0: LEXT is not ready. 1: LEXT is ready.
Bit 0	LICKSTBLF	0x0	ro	LICK stable interrupt flag Set by hardware. 0: LICK is not ready. 1: LICK is ready.

4.3.4 APB2 peripheral reset register (CRM_APB2RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	TMR11RST	0x0	rw	TMR17 reset 0: Does not reset TMR17 1: Reset TMR17
Bit 20	TMR10RST	0x0	rw	TMR10 set 0: Does not reset TMR10 1: Reset TMR10
Bit 19	TMR9ST	0x0	rw	TMR9 reset 0: Does not reset TMR9 1: Reset TMR9
Bit 18: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	USART1RST	0x0	rw	USART1 reset 0: Does not reset USART1 1: Reset USART1
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	SPI1RST	0x0	rw	SPI1 reset 0: Does not reset SPI1 1: Reset SPI1
Bit 11	TMR1RST	0x0	rw	TMR1 reset 0: Does not reset TMR1 1: Reset TMR1
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	ADC1RST	0x0	rw	ADC1 reset 0: Does not reset ADC1 1: Reset ADC1
Bit 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GPIOFRST	0x0	rw	GPIOF reset 0: Does not reset GPIOF 1: Reset GPIOF
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	GPIODRST	0x0	rw	GPIOD reset 0: Does not reset GPIOD 1: Reset GPIOD
Bit 4	GPIOCRST	0x0	rw	GPIOC reset 0: Does not reset GPIOC 1: Reset GPIOC
Bit 3	GPIOBRST	0x0	rw	GPIOB reset 0: Does not reset GPIOB 1: Reset GPIOB
Bit 2	GPIOARST	0x0	rw	GPIOA reset 0: Does not reset GPIOA 1: Reset GPIOA
Bit 1	EXINTRST	0x0	rw	EXINT reset 0: Does not reset EXINT 1: Reset EXINT. Kept at default value.
Bit 0	IOMUXRST	0	rw	IOMUX reset 0: Does not reset IOMUX 1: Reset IOMUX

4.3.5 APB1 peripheral reset register1 (CRM_APB1RST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	PWCRST	0x0	rw	PWC reset 0: Does not reset PWC 1: Reset PWC
Bit 27: 26	Reserved	0x0	resd	Kept at its default value.
Bit 25	CANRST	0x0	rw	CAN reset 0: Does not reset CAN 1: Reset CAN
Bit 24: 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	I2C2RST	0x0	rw	I2C2 reset 0: Does not reset I2C2 1: Reset I2C2
Bit 21	I2C1RST	0x0	rw	I2C1 reset 0: Does not reset I2C1 1: Reset I2C1
Bit 20	USART5RST	0x0	rw	USART5 reset 0: Does not reset USART5 1: Reset USART5
Bit 19	USART4RST	0x0	rw	USART4 reset 0: Does not reset USART4 1: Reset USART4
Bit 18	USART3RST	0x0	rw	USART3 reset 0: Does not reset USART3 1: Reset USART3
Bit 17	USART2RST	0x0	rw	USART2 reset 0: Does not reset USART2 1: Reset USART2
Bit 16: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SPI2RST	0x0	rw	SPI2 reset 0: Does not reset SPI2 1: Reset SPI2
Bit 13:12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTRST	0x0	rw	WWDT reset 0: Does not reset WWDT 1: Reset WWDT
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	CMPRST	0x0	rw	CMP reset 0: Does not reset CMP 1: Reset CMP
Bit 8:4	Reserved	0x00	resd	Kept at its default value.
Bit 3	TMR5RST	0x0	rw	TMR5 reset 0: Does not reset TMR5 1: Reset TMR5
Bit 2	TMR4RST	0x0	rw	TMR4 reset 0: Does not reset TMR4 1: Reset TMR4
Bit 1	TMR3RST	0x0	rw	TMR3 reset 0: Does not reset TMR3 1: Reset TMR3
Bit 0	TMR2RST	0x0	rw	TMR2 reset 0: Does not reset TMR2 1: Reset TMR2

4.3.6 AHB peripheral clock enable register (CRM_AHBEN)

Access: by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31: 13	Reserved	0x00000	resd	Kept at its default value.
Bit 12	OTGFS1EN	0x0	rw	OTGFS1 clock enable 0: Disabled 1: Enabled
Bit 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	SDIO1EN	0x0	rw	SDIO1 clock enable 0: Disabled 1: Enabled
Bit 9:7	Reserved	0x0	resd	Kept at its default value.
Bit 6	CRCEN	0x0	rw	CRC clock enable 0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	FLASHEN	0x1	rw	FLASH clock enable This bit is used to enable Flash clock in Sleep or Deepsleep mode. 0: Disabled 1: Enabled
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	SRAMEN	0x1	rw	SRAM clock enable This bit is used to enable SRRM clock in Sleep or Deepsleep mode. 0: Disabled 1: Enabled
Bit 1	DMA2EN	0x0	resd	DMA2 clock enable 0: Disabled 1: Enabled
Bit 0	DMA1EN	0x0	rw	DMA1 clock enable 0: Disabled 1: Enabled

4.3.7 APB2 peripheral clock enable register (CRM_APB2EN)

Bit	Name	Reset value	Type	Description
Bit 31: 22	Reserved	0x00	resd	Kept at its default value.
Bit 21	TMR11EN	0x0	rw	TMR11 clock enable 0: Disabled 1: Enabled
Bit 20	TMR10EN	0x0	rw	TMR10 clock enable 0: Disabled 1: Enabled
Bit 19	TMR9EN	0x0	rw	TMR9 clock enable 0: Disabled 1: Enabled
Bit 18: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	USART1EN	0x0	rw	USART1 clock enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	SPI1EN	0x0	rw	SPI1 clock enable 0: Disabled 1: Enabled
Bit 11	TMR1EN	0x0	rw	TMR1 clock enable 0: Disabled 1: Enabled
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	ADC1EN	0x0	rw	ADC1 clock enable

				0: Disabled 1: Enabled
Bit 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GPIOFEN	0x0	rw	GPIOF clock enable 0: Disabled 1: Enabled
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	GPIODEN	0x0	rw	GPIOD clock enable 0: Disabled 1: Enabled
Bit 4	GPIOCEN	0x0	rw	GPIOC clock enable 0: Disabled 1: Enabled
Bit 3	GPIOBEN	0x0	rw	GPIOB clock enable 0: Disabled 1: Enabled
Bit 2	GPIOAEN	0x0	rw	GPIOF clock enable 0: Disabled 1: Enabled
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	IOMUXEN	0x0	rw	IOMUX clock enable 0: Disabled 1: Enabled

4.3.8 APB1 peripheral clock enable register (CRM_APB1EN)

Access: 0 wait state, accessible by words, half-words and bytes.

No-wait states in most cases. However, when accessing to peripherals on APB1, wait-states are inserted until the end of peripheral access on the APB1 bus.

Bit	Name	Reset value	Type	Description
Bit 31: 29	Reserved	0x0	resd	Kept at its default value.
Bit 28	PWCEN	0x0	rw	PWC clock enable 0: Disabled 1: Enabled
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	Reserved	0x0	resd	Kept at its default value.
Bit 25	CANEN	0x0	rw	CAN clock enable 0: Disabled 1: Enabled
Bit 24: 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	I2C2EN	0	rw	I2C2 clock enable 0: Disabled 1: Enabled
Bit 21	I2C1EN	0	rw	I2C1 clock enable 0: Disabled 1: Enabled
Bit 20	USART5EN	0x0	rw	USART5 clock enable 0: Disabled 1: Enabled
Bit 19	USART4EN	0x0	rw	USART4 clock enable 0: Disabled 1: Enabled
Bit 18	USART3EN	0x0	rw	USART3 clock enable 0: Disabled 1: Enabled
Bit 17	USART2EN	0x0	rw	USART2 clock enable 0: Disabled 1: Enabled
Bit 16: 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SPI2EN	0x0	rw	SPI2 clock enable 0: Disabled 1: Enabled
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.

Bit 11	WWDTEN	0	rw	WWDT clock enable 0: Disabled 1: Enabled
Bit 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	CMPEN	0x0	rw	CMP clock enable 0: Disabled 1: Enabled
Bit 8: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	TMR5EN	0x0	rw	TMR5 clock enable 0: Disabled 1: Enabled
Bit 2	TMR4EN	0x0	rw	TMR4 clock enable 0: Disabled 1: Enabled
Bit 1	TMR3EN	0x0	rw	TMR3 clock enable 0: Disabled 1: Enabled
Bit 0	TMR2EN	0x0	rw	TMR2 clock enable 0: Disabled 1: Enabled

4.3.9 Battery powered domain control register (CRM_BPDC)

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

Note: LEXTEN, LEXTBYPS, ERTCSEL, and ERTCEN bits of the battery powered domain control register (CRM_BPDC) are in the battery powered domain. As a result, these bits are write protected after reset, and can only be modified by setting the BPWEN bit in the power control register (PWR_CTRL). These bits could be reset only by battery powered domain reset. Any internal or external reset does not affect these bits.

Bit	Name	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	BPDRST	0x0	rw	Battery powered domain software reset 0: Do not reset battery powered domain software 1: Reset battery powered domain software
Bit 15	ERTCEN	0x0	rw	ERTC clock enable Set and cleared by software. 0: Disabled 1: Enabled
Bit 14: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	ERTCSEL	0x0	rw	ERTC clock selection Once the ERTC clock source is selected, it cannot be changed until the BPDRST bit is reset. 00: No clock 01: LEXT 10: LICK 11: HEXT/128
Bit 7: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2	LEXTBYPS	0x0	rw	Low speed external crystal bypass 0: Disabled 1: Enabled
Bit 1	LEXTSTBL	0x0	ro	Low speed external oscillator stable Set by hardware after the LEXT is ready. 0: LEXT is not ready. 1: LEXT is ready.
Bit 0	LEXTEN	0x0	rw	External low-speed oscillator enable 0: Disabled 1: Enabled

4.3.10 Control/status register (CRM_CTRLSTS)

Reset flag can only be cleared by power reset or by writing the RSTFC bit, while others are cleared by system reset.

Access: 0 to 3 wait states, accessible by words, half-words or bytes. Wait states are inserted in the case of consecutive accesses to this register.

Bit	Name	Reset value	Type	Description
Bit 31	LPRSTF	0x0	ro	Low-power reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No low-power reset occurs 1: Low-power reset occurs
Bit 30	WWDTRSTF	0x0	ro	Window watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No window watchdog timer reset occurs 1: Window watchdog timer reset occurs
Bit 29	WDTRSTF	0x0	ro	Watchdog timer reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No watchdog timer reset occurs 1: Watchdog timer reset occurs.
Bit 28	SWRSTF	0x0	ro	Software reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No software reset occurs 1: Software reset occurs.
Bit 27	PORRSTF	0x1	ro	POR/LVR reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No POR/LVR reset occurs 1: POR/LVR reset occurs.
Bit 26	NRSTF	0x1	ro	NRST pin reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No NRST pin reset occurs 1: NRST pin reset occurs
Bit 25	Reserved	0x0	resd	Kept at its default value.
Bit 24	RSTFC	0x0	rw	Reset flag clear Cleared by writing 1 through software. 0: No effect 1: Clear the reset flag.
Bit 23: 2	Reserved	0x000000	resd	Kept at its default value.
Bit 1	LICKSTBL	0x0	ro	LICK stable 0: LICK is not ready. 1: LICK is ready.
Bit 0	LICKEN	0x0	rw	LICK enable 0: Disabled 1: Enabled

4.3.11 APB peripheral reset register (CRM_AHBRST)

Access: 0 wait state, accessible by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31:13	Reserved	0x00000	resd	Kept at its default value.
Bit 12	OTGFS1RST	0x0	rw	USB reset 0: Does not reset USB 1: Reset USB
Bit 11: 0	Reserved	0x000	resd	Kept at its default value.

4.3.12 PLL configuration register (CRM_PLL)

Access: 0 wait state, by words, half-words and bytes.

Bit	Name	Reset value	Type	Description
Bit 31	PLLCFGEN	0x0	rw	PLL configuration enable 0: Common integer multiplication mode, which is done by PLL_FREF and PLLMULT registers. 1: Flexible configuration mode, which is done by PLL_MS/PLL_NS/PLL_FR registers.
Bit 30: 27	Reserved	0x0	resd	Kept at its default value.
Bit 26: 24	PLL_FREF	0x0	rw	PLL input clock selection This field is valid only if PLLCFGEN=0. 000: 3.9 ~ 5 MHz 001: 5.2 ~ 6.25 MHz 010: 7.8125 ~ 8.33 MHz 011: 8.33 ~ 12.5 MHz 100: 15.625 ~ 20.83 MHz 101: 20.83 ~ 31.255 MHz 110: Reserved 111: Reserved
Bit 23: 17	Reserved	0x00	resd	Kept at its default value.
Bit 16: 8	PLL_NS	0x1F	rw	PLL multiplication factor PLL_NS range (31~500)
Bit 7: 4	PLL_MS	0x1	rw	PLL pre-division PLL_MS range (1~15)
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2: 0	PLL_FR	0x0	rw	PLL post-division factor PLL_FR range (0~5) 000: PLL post-division=1, divided by 1 001: PLL post-division=2, divided by 2 010: PLL post-division=4, divided by 4 011: PLL post-division=8, divided by 8 100: PLL post- division=16, divided by 16 101: PLL post- division=32, divided by 32 Others: Reserved It should be noted the relationship between the PLL-FR values and post-division factors.

4.3.13 Additional register (CRM_MISC1)

Bit	Name	Reset value	Type	Description
Bit 31: 28	CLKOUTDIV	0x0	rw	Clock output division 0xxx: Clock output 1000: Clock output divided by 2 1001: Clock output divided by 4 1010: Clock output divided by 8 1011: Clock output divided by 16 1100: Clock output divided by 64 1101: Clock output divided by 128 1110: Clock output divided by 256 1111: Clock output divided by 512
Bit 27: 26	Reserved	0x0	resd	Kept its default value.
Bit 25	HICKDIV	0x0	rw	HICK 6 divider selection This bit is used to select HICK or HICK /6. If the HICK/6 is selected, the clock frequency is 8 MHz. Otherwise, the clock frequency is 48 MHz. 0: HICK/6 1: HICK Note: 1.When the HICK is used as PLL clock source, the HICKDIV must not change during PLL enable. 2. In any case, HICK always input 4 MHz to PLL.
Bit 24: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	CLKFMC_SRC	0x0	rw	FMC clock source 0: 8 M HICK 1: HICK

Bit 19: 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	CLKOUT_SEL[3]	0x0	rw	Clock output selection This bit works with the bit [26:24] of the CRM_CFG register.
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 0	HICKCAL_KEY	0x00	rw	HICK calibration key The HICKCAL [7:0] can be written only when this field is set 0x5A.

4.3.14 OTG_FS extended control register (CRM_OTG_EXTCTRL)

The application must program this register before enabling OTG_FS.

Bit	Name	Reset value	Type	Description
Bit 31	EP3_RMPEN	0x0	rw	Endpoint3 remap enable 0: OTG_FS endpoint 3 remap is disabled, meaning that it is only used as an endpoint 3 to communicate with host 1: OTG_FS endpoint 3 remap is enabled, meaning that it can be used as endpoint 3 and endpoint 4 to communicate with host simultaneously.
Bit 30	USBDIV_RST	0x0	rw	USB divider reset 0: Does not reset USB divider 1: Reset USB divider
Bit 29: 0	Reserved	0x0000 0000	resd	Kept at its default value.

Note: This control register is a new feature for revision C.

4.3.15 Additional register (CRM_MISC2)

Bit	Name	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	HICK_TO_SCLK	0x0	rw	HICK as system clock frequency select When the HICK is selected as the clock source SCLKSEL, the frequency of SCLK is: 0: Fixed 8 MHz, that is, HICK/6 1: 48 MHz or 8 MHz, depending on the HICKDIV
Bit 8: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	AUTO_STEP_EN	0x0	rw	Auto step-by-step system clock switch enable When the system clock source is switched from others to the PLL or when the AHB prescaler is changed from large to small (system frequency is from small to large), it is recommended to enable the auto step-by-step system clock switch if the operational target is larger than 108 MHz,. Once it is enabled, the AHB bus is halted by hardware till the completion of the switch. During this switch period, the DMA remain working, and the interrupt events are recorded and then handled by NVIC when the AHB bus resumes. 00: Disabled 01: Reserved 10: Reserved 11: Enabled. When AHBDIV or SCLKSEL is modified, the auto step-by-step
Bit 3: 0	Reserved	0x0	resd	Kept at its default value.

5 Flash memory controller (FLASH)

5.1 FLASH introduction

Flash memory is divided into three parts: main Flash memory, information block and Flash memory registers.

- Main Flash memory is up to 256 KB
- Information block consists of 18 KB boot loader and the user system data area. The boot loader uses USART1, USART2 or OTGFS device mode for ISP programming.

Main Flash memory contains bank 1 only (256 KB), including 128 sectors, 2K per sector.

Table 5-1 Flash memory architecture(256 K)

Bank		Name	Address range
Main memory	Bank1 (256 KB)	Sector 0	0x0800 0000 – 0x0800 07FF
		Sector 1	0x0800 0800 – 0x0800 0FFF
		Sector 2	0x0800 1000 – 0x0800 17FF
		…	…
		Sector 127	0x0803 F800 – 0x0803 FFFF
Information block		18 KB boot loader	0x1FFF AC00 – 0x1FFF F3FF
		1 KB user system data	0x1FFF F800 – 0x1FFF FBFF

Main Flash memory contains bank 1 only (128 KB), including 128 sectors, 1K per sector.

Table 5-2 Flash memory architecture(128 K)

Bank		Name	Address range
Main memory	Bank1 (128 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
		Sector 1	0x0800 0400 – 0x0800 07FF
		Sector 2	0x0800 0800 – 0x0800 0BFF
		…	…
		Sector 127	0x0801 FC00 – 0x0801 FFFF
Information block		18KB boot loader	0x1FFF AC00 – 0x1FFF F3FF
		1KB user system data	0x1FFF F800 – 0x1FFF FBFF

Main Flash memory contains bank 1 only (64 KB), including 64 sectors, 1K per sector.

Table 5-3 Flash memory architecture(64 K)

Bank		Name	Address range
Main memory	Bank1 (64 KB)	Sector 0	0x0800 0000 – 0x0800 03FF
		Sector 1	0x0800 0400 – 0x0800 07FF
		Sector 2	0x0800 0800 – 0x0800 0BFF
		…	…
		Sector 63	0x0800 FC00 – 0x0800 FFFF
Information block		18KB boot loader	0x1FFF AC00 – 0x1FFF F3FF
		1KB user system data	0x1FFF F800 – 0x1FFF FBFF

User system data area

The system data will be read from the information block of Flash memory whenever a system reset occurs, and is saved in the user system data register (FLASH_USD)and erase programming protection status register (FLASH_EPPS).

Each system data occupies two bytes, where the low bytes corresponds to the contents in the system data area, and the high bytes represent the inverse code that is used to verify the correctness of the selected bit. When the high byte is not equal to the inverse code of the low byte (except when both high

and low byte are all 0xFF), the system data loader will issue a system data error flag (USDERR) and the corresponding system data and their inverse codes are forced 0xFF.

Note: The update of the contents in the user system data area becomes effective only after a system reset.

Table 5-4 User system data area

Address	Bit	Description	
0x1FFF_F800	[7: 0]	FAP[7: 0]: Flash memory access protection (Access protection enable/disable result is stored in the FLASH_USD[1] register and bit [26] 0xA5: Flash access protection disabled 0XCC: High-level Flash access protection enabled Others; Low-level Flash access protection enabled	
	[15: 8]	nFAP[7: 0]: Inverse code of FAP[7: 0]	
	[23: 16]	SSB[7:0]: System configuration byte (it is stored in the FLASH_USD[9: 2] register)	
		Bit 7: 3	Reserved
		Bit 2 (nSTDBY_RST)	0: Reset occurs when entering Standby mode 1: No reset occurs when entering Standby mode
		Bit 1 (nDEPSLP_RST)	0: Reset occurs when entering Deepsleep mode 1: No reset occurs when entering Deepsleep mode
		Bit 0 (nWDT_ATO_EN)	0: Watchdog is enabled 1: Watchdog is disabled
	[31: 24]	nSSB[7: 0]: Inverse code of SSB[7: 0]	
0x1FFF_F804	[7: 0]	Data0[7: 0]: User data 0 (It is stored in the FLASH_USD[17:10] register)	
	[15: 8]	nData0[7: 0]: Inverse code of Data0[7: 0]	
	[23: 16]	Data1[7: 0]: User data 1 (It is stored in the FLASH_USD[25: 18] register)	
	[31: 24]	nData1[7: 0]: Inverse code of Data1[7: 0]	
0x1FFF_F808	[7: 0]	EPP0[7:0]:Flash erase/write protection byte 0 (in the FLASH_EPPS[7: 0]) This field is used to protect sector0~ sector15 of main Flash memory. Each bit takes care of 2 KB pages 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[15: 8]	nEPP0[7: 0]: Inverse code of EPP0[7: 0]	
	[23: 16]	EPP1[7: 0]: Flash erase/write protection byte 1 (stored in the FLASH_EPPS[15: 8]) This field is used to protect page16~page31 of main Flash memory. Each bit takes care of 2 KB sectors 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[31: 24]	nEPP1[7: 0]: Inverse code of EPP1[7: 0]	
0x1FFF_F80C	[7: 0]	EPP2[7: 0]: Flash erase/write protection byte 2 (stored in the FLASH_EPPS[23: 16]) This field is used to protect page32~page47 of main Flash memory. Each bit takes care of 2 KB pages 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[15: 8]	Inverse code of nEPP2[7: 0]: EPP2[7: 0]	
	[23: 16]	EPP3[7:0]: Flash erase/write protection byte 3 (stored in the FLASH_EPPS[31: 24]) This field is used to protect page48~page61 of main Flash memory. Each bit takes care of 2 KB pages Bit [7] is used to protect the page 62 and the remaining pages, as well as main Flash memory extension area. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	
	[31: 24]	nEPP3[7: 0]: Inverse code of EPP3[7: 0]	
0x1FFF_F810	[7: 0]	Data2[7: 0]: User system data 2	
	[15: 8]	nData2[7: 0]: Inverse code of Data2[7: 0]	
	[23: 16]	Data3[7: 0]: User system data 3	

	[31: 24]	nData3[7: 0]: Inverse code of Data3[7: 0]
0x1FFF_F814	[7: 0]	Data4[7: 0]: User system data 4
	[15: 8]	nData4[7: 0]: Inverse code of Data4[7: 0]
	[23: 16]	Data5[7: 0]: User system data 5
	[31: 24]	nData5[7: 0]: Inverse code of Data5[7: 0]
…	…	
…	…	
0x1FFF_FBFC	[7: 0]	Data504[7: 0]: User system data 504
	[15: 8]	nData504[7: 0]: Inverse code of Data504[7: 0]
	[23: 16]	Data505[7: 0]: User system data 505
	[31: 24]	nData505[7: 0]: Inverse code of Data505[7: 0]

5.2 Flash memory operation

5.2.1 Unlock/lock

After reset, Flash memory is protected, by default. FLASH_CTRL cannot be written. Write and erase operation can be performed only when the Flash memory is unlocked.

Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH_UNLOCK register.

Note: Writing an incorrect key sequence leads to a bus error and the Flash memory is also locked until the next reset.

Lock procedure:

Flash memory block can be locked by setting the OPLK bit in the FLASH_CTRL register.

5.2.2 Erase operation

Erase operation must be done before programming. Flash memory erase includes page erase and mass erase.

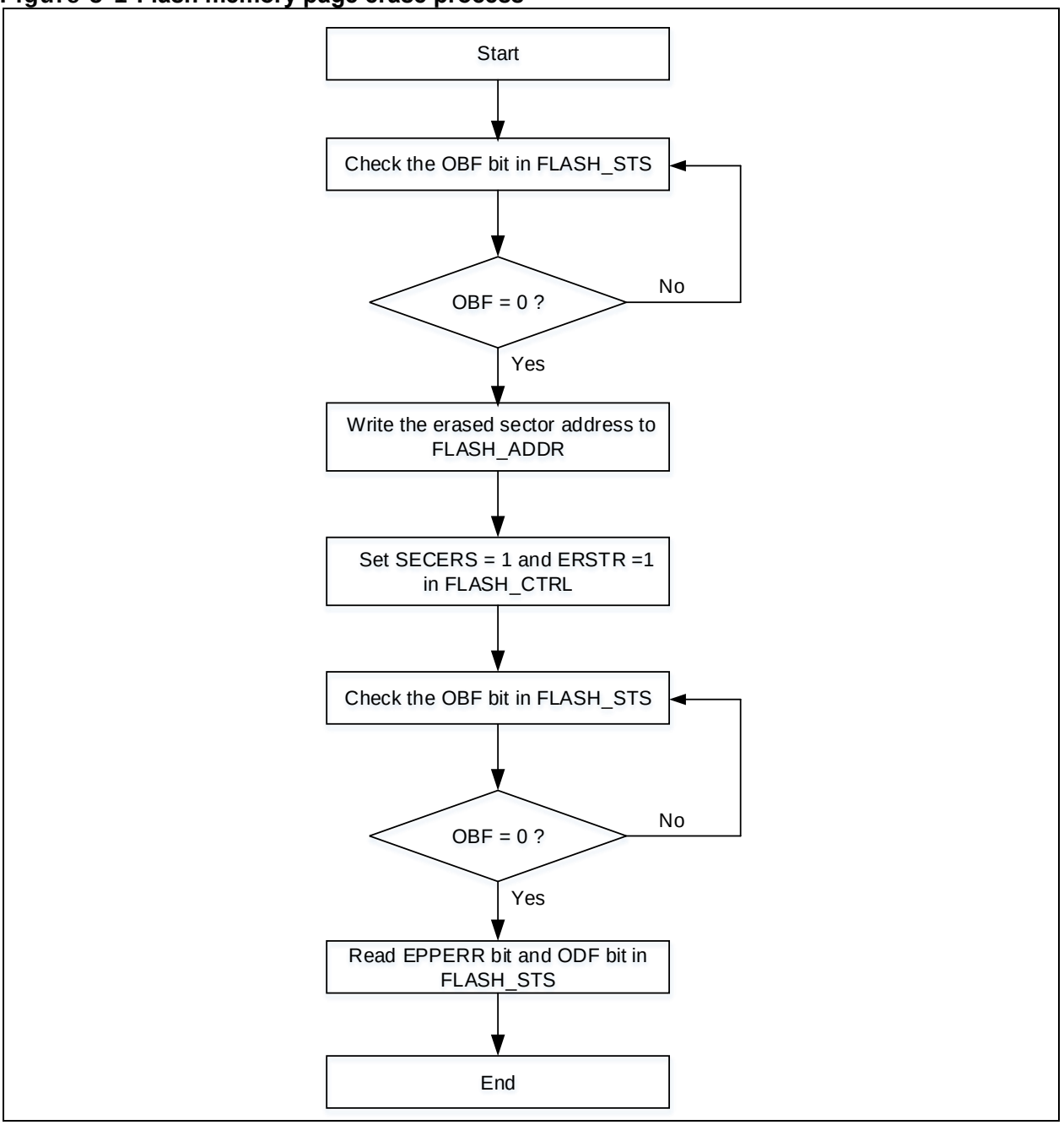
Page erase

Any page in the Flash memory and its extension area can be erased with page erase function independently. Below should be followed during page erase:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Write the page to be erased in the FLASH_ADDR register
- Set the SECERS and ERSTR bit in the FLASH_CTRL register to enable page erase
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the EPPERR bit and ODF bit in the FLASH_STS register to verify the erased pages.

Note: When the boot loader code area is configured as the Flash memory extension area, performing page-erase operation erases the entire Flash memory extension area.

Figure 5-1 Flash memory page erase process



Mass erase

Mass erase function can erase the whole Flash memory.

The following process is recommended:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the BANKERS and ERSTR bit in the FLASH_CTRL register to enable mass erase;
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the EPPERR bit and ODF bit in the FLASH_STS register to verify the erased pages.

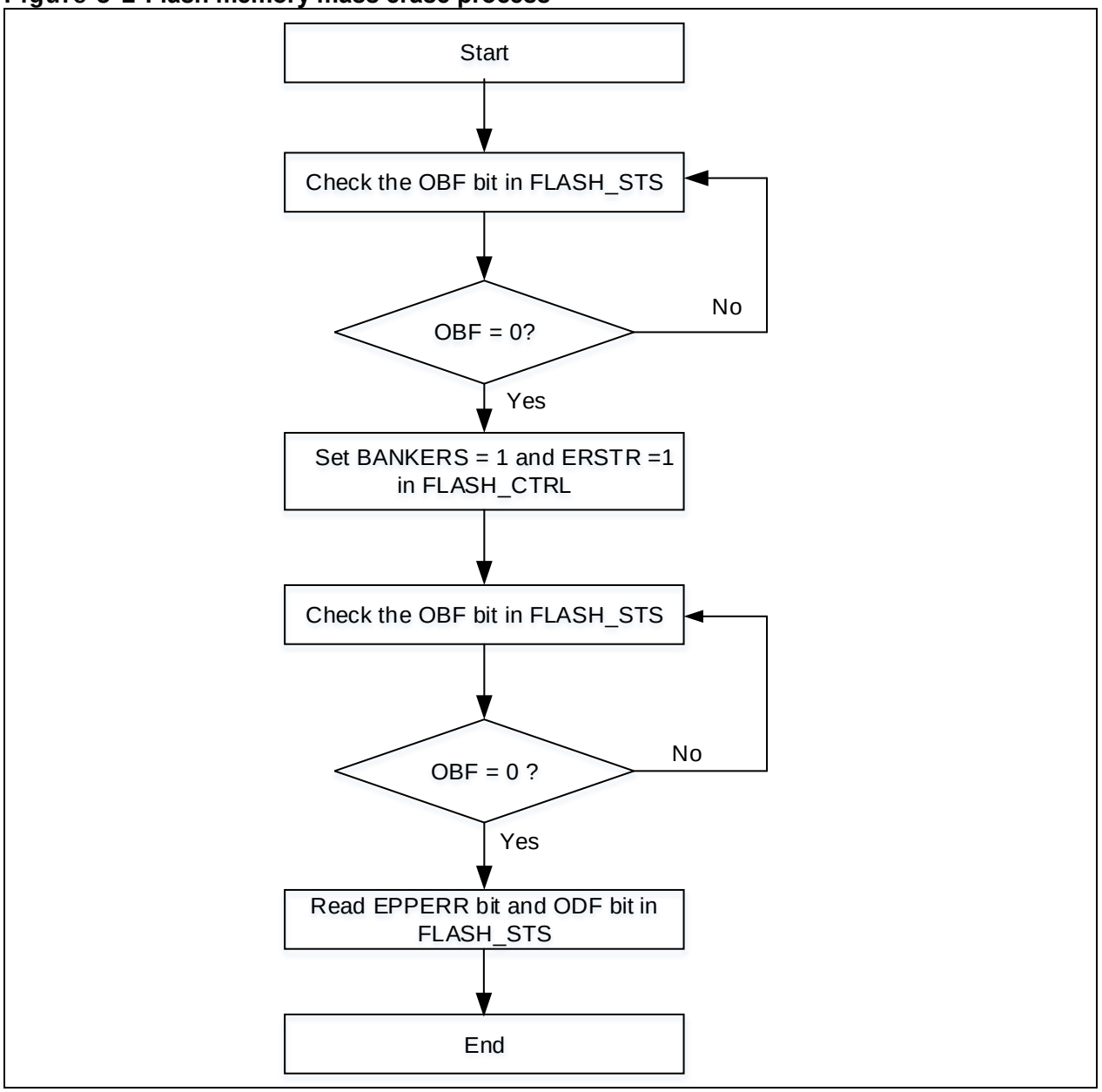
Note:

1) When the boot loader code area is configured as the Flash memory extension area, performing mass-erase operation erases automatically the entire the entire Flash memory and its extension area.

2) Read access during erase operation halts the CPU and waits until the completion of erase.

3) Internal HICK must be enabled prior to erase operation.

Figure 5-2 Flash memory mass erase process



5.2.3 Programming operation

The Flash memory can be programmed with 32 bits, 16 bits or 8 bits at a time.

The following process is recommended:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the FPRGM bit in the FLASH_CTRL register, so that the Flash memory programming instructions can be received;
- Write the data (word/half-word/byte) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH_STS register becomes “0”, read the EPPERR, PRGMERR and ODF bit to verify the programming result.

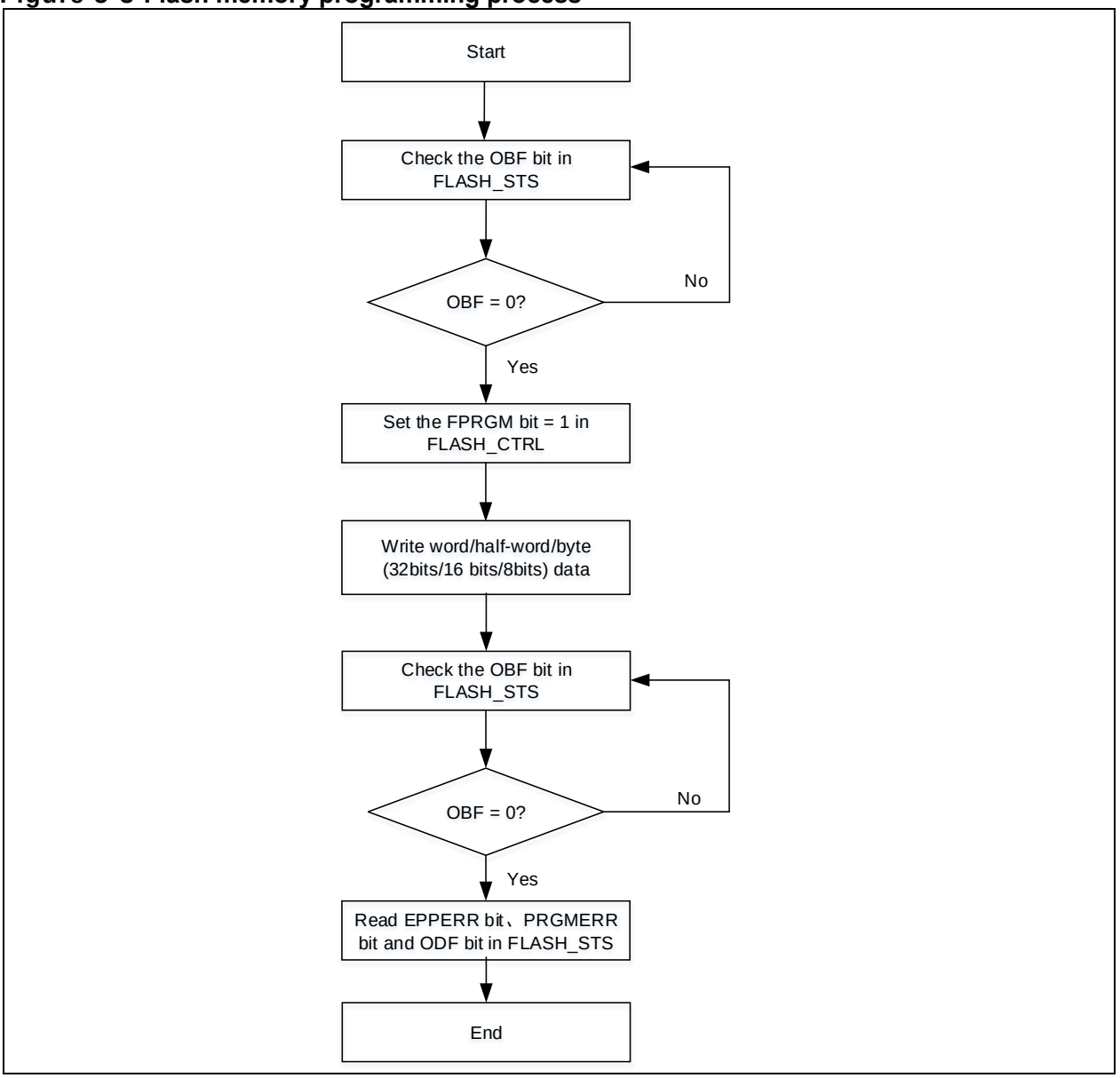
Note:

1. When the address to be written is not erased in advance, the programming operation is not executed unless the data to be written is all 0. In this case, a programming error is reported by the PRGMERR bit in the FLASH_STS register.

2. Read operation to the Flash memory during programming halts CPU and waits until the completion of programming.

3. Internal HICK must be enabled prior to programming.

Figure 5-3 Flash memory programming process



5.2.4 Read operation

Flash memory can be accessed through AHB bus of the CPU.

5.3 Main Flash memory extension area

Bootloader code area can also be programmed as the extension area of the main Flash memory to store user-application code. When used as main Flash memory extension area, it behaves like the main Flash memory, including read, unlock, erase and programming operations.

5.4 User system data area

5.4.1 Unlock/lock

After reset, user system data area is protected, by default. Write and erase operations can be performed only after the Flash memory is unlocked before the unlock operation for the user system data area.

Unlock procedure:

Flash memory block can be unlocked by writing KEY1 (0x45670123) and KEY2 (0xCDEF89AB) to the FLASH_UNLOCK register;

When KEY1 (0x45670123) and KEY2 (0xCDEF89AB) is written to the FLASH_USD_UNLOCK register, the USDULKS bit in the FLASH_CTRL register will be automatically set by hardware, indicating that it

supports write/erase operation to the user system data area.

Note: Writing an incorrect key sequence leads to bus error and the Flash memory is also locked until the next reset.

Lock procedure:

User system data area is locked by clearing the USDULKS bit in the FLASH_CTRL register by software.

5.4.2 Erase operation

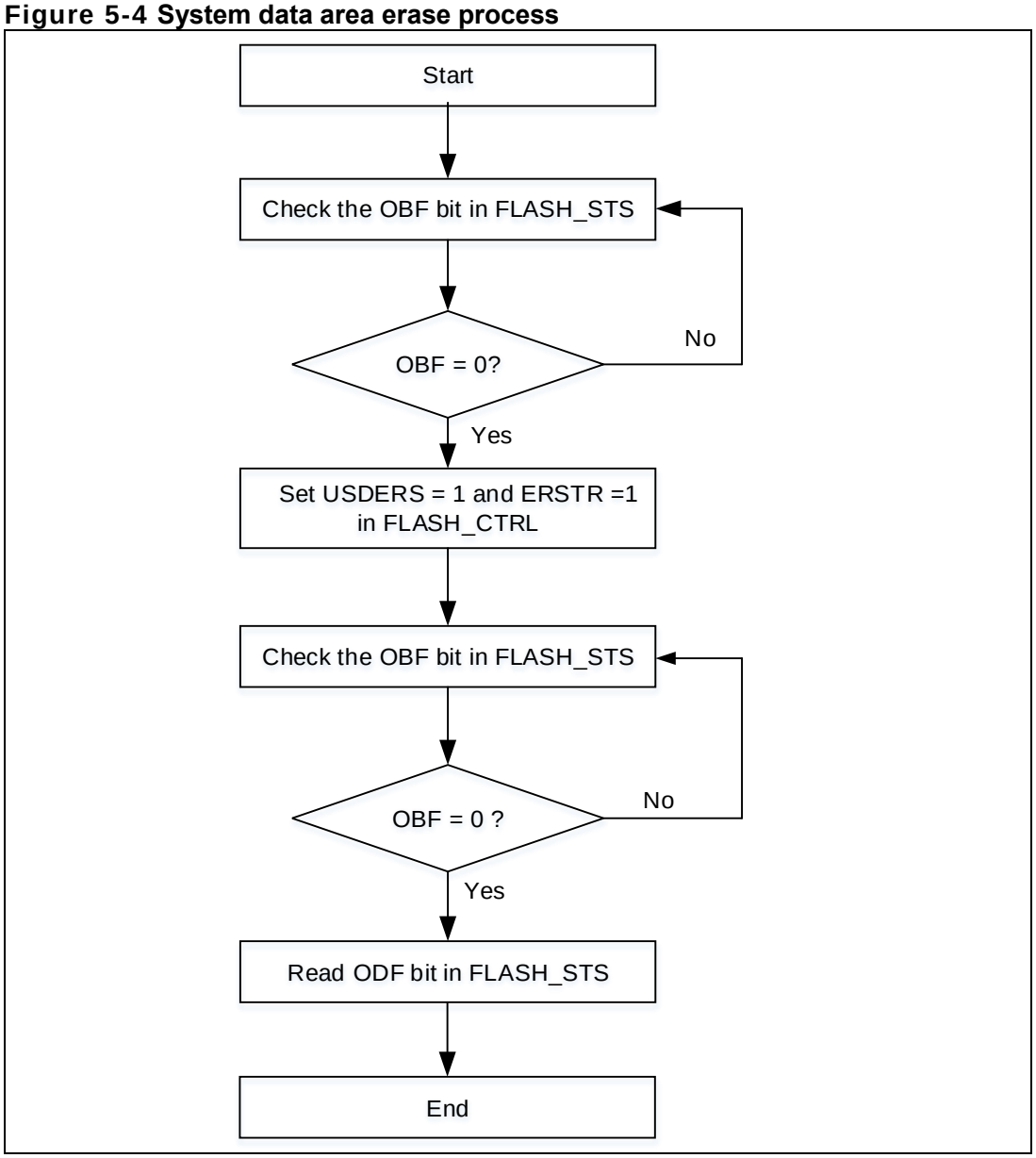
Erase operation must be done before programming. User system data area can perform erase operation independently.

Below should be followed during page erase:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the USDERS and ERSTR bit in the FLASH_CTRL register to enable erase operation;
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the ODF bit in the FLASH_STSx register to verify the erase result.

Note: Read operation to the Flash memory during programming halts CPU and waits until the completion of erase. The internal HICK must be enabled prior to erase operation.

Figure 5-4 System data area erase process



5.4.3 Programming operation

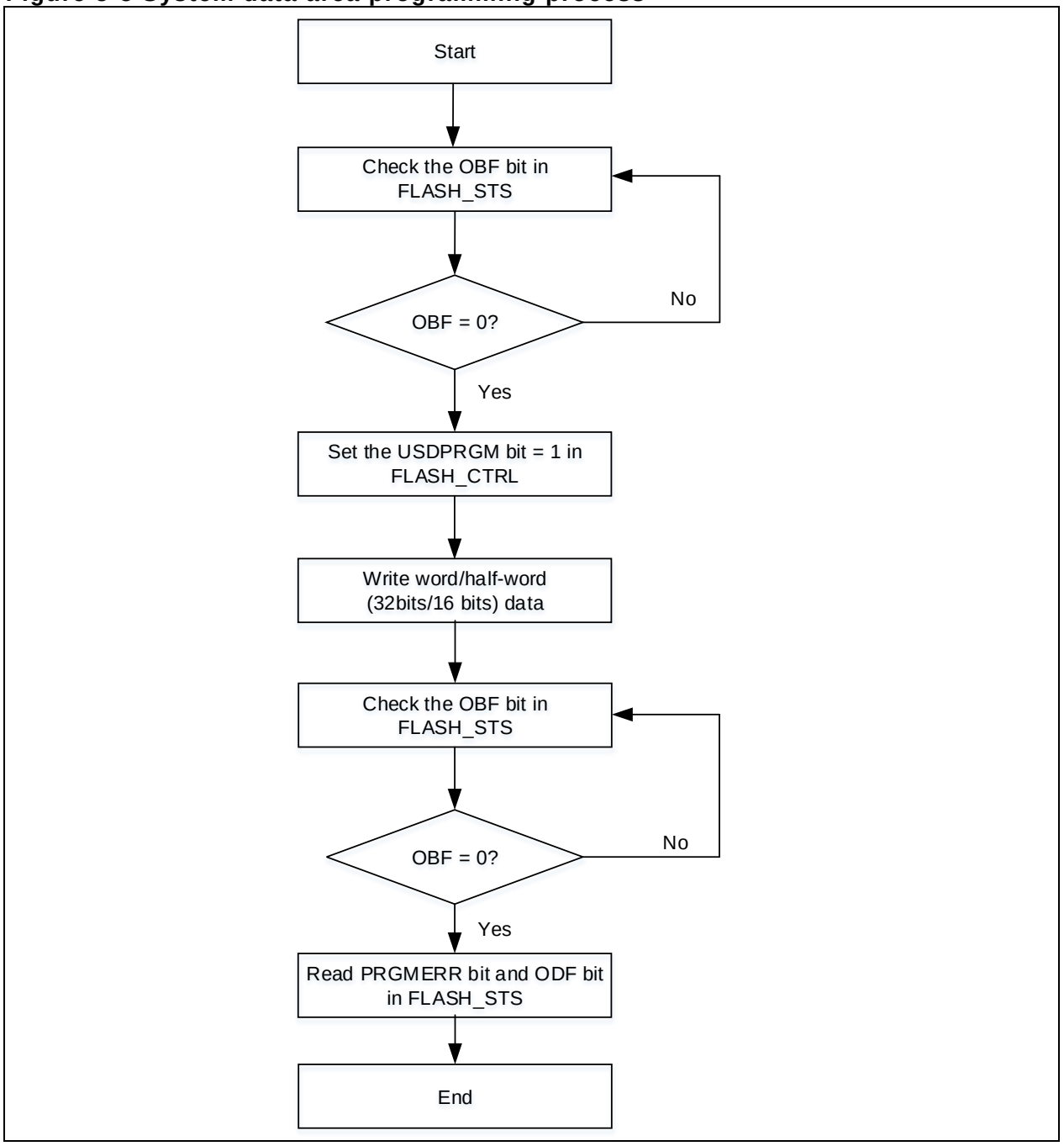
The User system data area can be programmed with 16 bits or 32 bits at a time.

The following process is recommended:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Set the USDPRGM bit in the FLASH_CTRL register, so that the programming instructions for the user system data area can be received;
- Write the data (half-word/word) to be programmed to the designated address;
- Wait until the OBF bit in the FLASH_STS register becomes “0”, read the PRGMERR and ODF bit to verify the programming result.

Note: Read operation to the Flash memory during programming halts CPU and waits until the completion of programming. The internal HICK must be enabled prior to programming operation.

Figure 5-5 System data area programming process



5.4.4 Read operation

User system data area can be accessed through AHB bus of the CPU.

5.5 Flash memory protection

Flash memory includes access and erase/program protection.

5.5.1 Access protection

Flash memory access protection is divided into two parts: high-level and low-level.

Once enabled, only the Flash program is allowed to read Flash memory data. This read operation is not permitted in debug mode or by booting from non-Flash memory.

Low-level access protection

When the contents in the nFAP and FAP bytes are different from 0x5A and 0xA5, and 0x33 and 0xCC, the low-level Flash memory access protection is enabled after a system reset.

When the Flash access is protected, the user can re-erase the system data area, and unlock Flash access protection (switching from protected to unprotected state will trigger mass erase on the Flash memory automatically) by writing 0xA5 to FAP byte, and then perform a system reset. Subsequently, the system data loader will be reloaded with system data and updated with Flash memory access protection disable state (FAP byte)

High-level access protection

When the content in the nFAP is different from 0x33 , and the content in the FAP byte is not equal to 0xCC, the high-level Flash memory access protection is enabled after a system reset.

Once enabled, it cannot be unlocked, and it is not permissible for users to re-erase and write the system data area.

Note:

1) The main memory extension area can also be protected.

2) If the access protection bit is set in debug mode, then the debug mode has to be cleared by POR instead of system reset in order to resume access to Flash memory data

Table 5-3 shows Flash memory access limits when Flash access protection is enabled.

Table 5-5 Flash memory access limit

Block	Protection level	Access limits					
		In debug mode or boot from SRAM and boot loader code area			Boot from main Flash memory		
		Read	Write	Erase	Read	Write	Erase
Main memory	Low-level protection	Not allowed		Not allowed (1) (2)	Accessible		
	High-level protection(4)	Not allowed			Accessible		
User system data area	Low-level protection	Not allowed	Accessible		Accessible		
	High-level protection(4)	Not allowed		Not allowed(3)	Not allowed		Not allowed(3)

(1) Main Flash memory is cleared automatically by hardware only when the access protection is disabled;

(2) Only page erase is forbidden, and mass erase is not affected;

(3) The user system data area can only be cleared by hardware through the FAP_HL_DIS bit;

(4) High-level Flash access protection is used to protect Flash memory against access and protect user system data area against erroneous erase operation.

5.5.2 Erase/program protection

Erase/program protection is performed on the basis of 2 pages.

This is used to protect the contents in the Flash memory against inadvertent operation when the program crash occurs.

Erase/program operation is not permitted under one of the following events, and the EPPERR bit is set accordingly:

- The pages (Flash memory and its extended area) with erase/program protection enabled
- The sectors (Flash memory and its extended area) with erase/program protection enabled
- When the Flash access protection is enabled, the page 0~1 in the main Flash memory will be protected against erase/program automatically,
- When the Flash access protection is enabled, the main Flash memory and its extended area are protected when it is in debug mode or when it is started from non-main Flash memory.

Note: For 256K Flash memory, page size is in terms of 2K bytes; For Flash memory smaller than 256K, its page size is in terms of 1K bytes.

5.6 Special functions

5.6.1 Security library settings

Security library is a defined area protected by a code in the main memory. This area is only executable but cannot be read (Except for I-Code and D-code buses), written, or deleted, unless a correct code is keyed in. Security library includes instruction security library and data security library.

Advantages of security library:

Security library is protected by codes so that solution providers can program core algorithm into this area;

Security library cannot be read or deleted (including ISP/IAP/SWD) but only executed unless code defined by the solution provider is keyed in;

The rest of the area can be used for secondary development by solution providers;

Solution providers can sell core algorithm with security library function and do not have to develop full solutions for every customer.

Security library helps prevent from deliberate damage or changing terminal application codes.

Note: Security library can only be located in the main Flash memory;

Security library code must be programmed by page, with its start address aligned with the main memory address;

Only I-Code bus is allowed to read instruction security library;

Only I-Code and D-Code bus are allowed to read the read-only area;

In an attempt of writing or deleting security library code, a warning message will be issued by WRPRTFLR =1 in the FLASH_STS register;

Executing mass erase in the main memory will not erase the security library.

By default, security library setting register is unreadable and write protected. To enable write access to this register, security library should be unlocked first, by writing 0xA35F6D24 to the SLIB_UNLOCK register, and checking the SLIB_ULKF bit in the SLIB_MISC_STS register to verify if it is unlocked successfully and then writing the programmed value into the security library setting register.

Optional CRC check for security library code is based on a page level.

The steps to enable security library are as follows:

- Check the OBF bit in the FLASH_STS register to ensure that there is no other ongoing programming operation;
- Write 0xA35F6D24 to the SLIB_UNLOCK register to unlock security library.
- Check the SLIB_ULKF bit of SLIB_MISC_STS register to verify that it is unlocked successfully.
- If the security library is located in Flash memory, then set the pages to be protected (including the addresses of instruction and data areas) in the SLIB_SET_RANGE register; if the security library is located in the Flash extension area, then set the EM_SLIB_SET register
- Wait until the OBF bit becomes “0”
- Set a security library password in the SLIB_SET_PWD register
- Wait until the OBF bit becomes “0”

- Program the code to be saved in security library
- Perform system reset, and then reload security library setting words
- Read the SLIB_STS0/STS1 register to verify the security library settings

Note:

It is not permissible to program the main Flash and its extended area as a security library simultaneously;

Security library must be enabled before the Flash access protection is activated.

Steps to unlock security library:

- Write the previously set security library password to the SLIB_PWD_CLR register
- Wait until the OBF bit becomes “0”
- Perform system reset, and then reload security library setting word
- Read the SLIB_STS0 register to check the security library settings

Note: Disabling the security library will automatically perform mass erase for the main memory and for security library setting block.

5.6.2 Bootloader code area used as Flash memory extension

There is only one chance for users to program the bootloader code area as the Flash memory extension area, which will have the same features as those of Flash memory after successful configuration as follows:

- Read the bit 0 in the SLIB_STS0 register to obtain the current mode of the bootloader code area
- Write the value 0xA35F6D24 to the SLIB_UNLOCK register to unlock the current mode of bootloader code area
- Write non-0xFF to the bit [7: 0] in the BTM_MODE_SET register
- Wait until the OBF bit becomes 0
- Perform a system reset, and reload setting words
- Read the SLIB_STS0 register to verify

Note: The main Flash extended are must be set before the Flash access protection is activated.

5.6.3 CRC verify

The sLib code or user code perform optional CRC check on a page level.

CRC verify procedure as follows:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress;
- Program the start address of the code to be verified in the FLASH_CRC_ADDR register
- Program the code count (in terms of pages) to be verified through bit [15:0] in the FLASH_CRC_CTR register
- Enable CRC verify by setting the bit 16 of the FLASH_CRC_CTR register
- Wait until the OBF bit becomes 0
- Read the FLASH_CRC_CHKR register to verify

Note: The values of the FLASH_CRC_ADDR register must be aligned with the start address of the page; CRC verify must not cross the main Flash memory and its extension area.

5.7 Flash memory registers

These peripheral registers must be accessed by words (32 bits).

Table 5-6 Flash memory interface—Register map and reset value

Register	Offset	Reset value
FLASH_PSR	0x00	0x0000 0030
FLASH_UNLOCK	0x04	0xXXXX XXXX
FLASH_USD_UNLOCK	0x08	0xXXXX XXXX
FLASH_STS	0x0C	0x0000 0000
FLASH_CTRL	0x10	0x0000 0080
FLASH_ADDR	0x14	0x0000 0000
FLASH_USD	0x1C	0x03FF FFFC
FLASH_EPPS	0x20	0xFFFF FFFF
SLIB_STS0	0x74	0x0000 0000
SLIB_STS1	0x78	0x0000 0000
SLIB_PWD_CLR	0x7C	0xFFFF FFFF
SLIB_MISC_STS	0x80	0x0000 0000
FLASH_CRC_ADDR	0x84	0x0000 0000
FLASH_CRC_CTRL	0x88	0x0000 0000
FLASH_CRC_CHKR	0x8C	0x0000 0000
SLIB_SET_PWD	0x160	0x0000 0000
SLIB_SET_RANGE	0x164	0x0000 0000
EM_SLIB_SET	0x168	0x0000 0000
BTM_MODE_SET	0x16C	0x0000 0000
SLIB_UNLOCK	0x170	0x0000 0000

5.7.1 Flash performance select register (FLASH_PSR)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 6	Reserved	0x00000	resd	Kept at its default value.
Bit 5	PFT_ENF	0x1	ro	Prefetch enable flag When this bit is set, it indicates that the Flash prefetch is enabled
Bit 4	PFT_EN	0x1	rw	Prefetch enable 0: Prefetch is disabled 1: Prefetch is enabled.
Bit 3	HFCYC_EN	0x0	rw	Half cycle acceleration access enable 0: Disabled 1: Enabled This bit is used to speed up access to Flash memory when WTCYC=0.
Bit 2: 0	WTCYC	0x0	rw	Wait states The wait states depends on the size of the system clock, and they are in terms of system clocks. 0: Zero wait state when 0MHz<system clock≤32MHz 1: One wait state when 32MHz<system clock≤64MHz 2: Two wait states when 64MHz<system clock≤96MHz 3: Three wait states when 96MHz<system clock≤128MHz 4: Four wait states when 128MHz<system clock≤150MHz

5.7.2 Flash unlock register (FLASH_UNLOCK)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	UKVAL	0xXXXX XXXX	wo	Unlock key value This is used to unlock Flash memory bank and its extension area.

Note: All these bits are write-only, and return 0 when being read.

5.7.3 Flash user system data unlock register (FLASH_USD_UNLOCK)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 0	USD_UKVAL	0xXXXX XXXX	wo	User system data Unlock key value

Note: All these bits are write-only, and return 0 when being read.

5.7.4 Flash status register (FLASH_STS)

Bit	Abbr.	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value
Bit 5	ODF	0	rw1c	Operation done flag This bit is set by hardware when Flash memory operations (program/erase) is completed. It is cleared by writing “1”.
Bit 4	EPPERR	0	rw1c	Erase/program protection error This bit is set by hardware when programming the erase/program- protected Flash memory address. It is cleared by writing “1”.
Bit 3	Reserved	0	resd	Kept at its default value.
Bit 2	PRGMERR	0	rw1c	Programming error When the programming address is not “0xFFFF”, this bit is set by hardware. It is cleared by writing “1”.
Bit 1	Reserved	0	resd	Kept at its default value.
Bit 0	OBF	0	ro	Operation busy flag When this bit is set, it indicates that Flash memory operation is in progress. It is cleared when operation is completed.

5.7.5 Flash control register (FLASH_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value
Bit 16	FAP_HL_DIS	0x0	rw	High level Flash access protection disable When this bit is set, the user system data area is automatically cleared by hardware; After a reset, it is unlocked, and low-level access protection is still present. .This bit is automatically cleared by hardware by writing 1 to it
Bit 15: 13	Reserved	0x0	resd	Kept its default value
Bit 12	ODIFE	0	rw	Operation done flag interrupt enable 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 11,8,3	Reserved	0	resd	Kept its default value
Bit 10	ERRIE	0	rw	Error interrupt enable This bit enables EPPERR or PROGERR interrupt. 0: Interrupt is disabled; 1: Interrupt is enabled.
Bit 9	USDULKS	0	rw	User system data unlock success This bit is set by hardware when the user system data is

				unlocked properly, indicating that erase/program operation to the user system data is allowed. This bit is cleared by writing “0”, which will re-lock the user system data area.
Bit 7	OPLK	1	rw	Operation lock This bit is set by default, indicating that Flash memory is protected against operations. This bit is cleared by hardware after unlock, indicating that erase/program operation to Flash memory is allowed. Writing “1” can re-lock Flash memory operations.
Bit 6	ERSTR	0	rw	Erase start An erase operation is triggered when this bit is set. This bit is cleared by hardware after the completion of the erase operation.
Bit 5	USDERS	0	rw	User system data erase It indicates the user system data erase.
Bit 4	USDPRGM	0	rw	User system data program It indicates the user system data program.
Bit 3	BLKERS	0	rw	Bank erase It indicates bank erase operation.
Bit 2	BANKERS	0	rw	Sector erase It indicates bank erase operation.
Bit 1	SECERS	0	rw	Page erase It indicates sector erase operation.
Bit 0	FPRGM	0	rw	Flash program It indicates Flash program operation.

5.7.6 Flash address register (FLASH_ADDR)

Bit	Register	Reset value	Type	Description
Bit 31: 0	FA	0x0000 0000	wo	Flash address Select the address of the banks/pages to be erased.

5.7.7 User system data register (FLASH_USD)

Bit	Register	Reset value	Type	Description
Bit 31: 27	Reserved	0x00	resd	Kept at its default value
Bit 26	FAP_HL	0	ro	Flash access protection high level The status of the Flash access protection is determined by bit 26 and bit 1. 00: Flash access protection disabled, and FAP=0xA5 01: Low-level Flash access protection enabled, and FAP=non-oxCC and 0xA5. 10: Reserved 11: High-level Flash access protection, and FAP=0xCC
Bit 25: 18	USER_D1	0xFF	ro	User data 1
Bit 17: 10	USER_D0	0xFF	ro	User data 0
Bit 9: 2	SSB	0xFF	ro	System setting byte Includes the system setting bytes in the loaded user system data area Bit 9: 5: Unused Bit 4: nSTDBY_RST Bit 3: nDEPSLP_RST Bit 2: nWDT_ATO_EN
Bit 1	FAP	0	ro	Flash access protection Access to Flash memory is not allowed when this bit is set.

Bit 0	USDERR	0	ro	User system data error When this bit is set, it indicates that certain byte does not match its inverse code in the user system data area. At this point, this byte and its inverse code will be forced to 0xFF when being read.

5.7.8 Erase/program protection status register (FLASH_EPPS)

Bit	Register	Reset value	Type	Description
Bit 31: 0	EPPS	0xFFFF FFFF	ro	Erase/Program protection status This register reflects the erase/program protection byte status in the loaded user system data.

5.7.9 Flash security library status register0 (SLIB_STS0)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
Bit 23: 16	EM_SLIB_DAT_SS	0x00	ro	Extension memory sLib data start page 00000000: Invalid sector 00000001: Page 1 00000010: Page 2 … 00001000: Page 8 (the last page of 256KB main Flash memory) … 00010001: Page 17 (the last page of 64KB and 128KB main Flash memory) 11111111: No data sLib
Bit 15: 4	Reserved	0x000	resd	Kept at its default value
Bit 3	SLIB_ENF	0	ro	SLIB_ENF: sLib enable flag When this bit is set, it indicates that the main Flash memory is partially or completely (depending on the setting of SLIB_STS1) used as security library code.
Bit 2	EM_SLIB_ENF	0	ro	Extension memory sLib enable flag When this bit is set, it indicates that the bootloader code area is used as the Flash extension area (BTM_AP_ENF is set), and stores security library code.
Bit 1	Reserved	0	resd	Kept at its default value
Bit 0	BTM_AP_ENF	0	ro	Boot memory store application code enabled flag When this bit is set, it indicates that the bootloader memory can be used as main Flash extension area to store user application code; otherwise, it is only used for system boot code.

5.7.10 Flash security library status register1 (SLIB_STS1)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31: 22	SLIB_ES	0x3FF	ro	Security library end page 0000000000: Page 0 0000000001: Page 1 0000000010: Page 2 … 0000111111: Page 63 (the last page of 64KB main Flash memory) … 0001111111: Page 127 (the last page of 256KB and 128KB main Flash memory)
Bit 21: 11	SLIB_DAT_SS	0x000	ro	Security library instruction start page 00000000000: Invalid page 00000000001: Page 1 00000000010: Page 2

				... 00000111111: Page 63 (the last page of 64KB main Flash memory) ... 00001111111: Page 127 (the last page of 256KB and 128KB main Flash memory) 11111111111: No data sLib
Bit 10: 0	SLIB_SS	0x000	ro	Security library start page 00000000000: Page 0 00000000001: Page 1 00000000010: Page 2 … 00000111111: Page 63 (the last page of 64KB main Flash memory) … 00001111111: Page 127 (the last page of 256KB and 128KB main Flash memory)

5.7.11 Security library password clear register (SLIB_PWD_CLR)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_PCLR_VAL	0x0000 0000	wo	Security library password clear value This register is used to key in a correct sLib password in order to unlock sLib function. The write status of this register is indicated by bit 0 and bit 1 of the SLIB_MISC_STS register.

5.7.12 Security library additional status register (SLIB_MISC_STS)

For Flash memory security library only.

Bit	Register	Reset value	Type	Description
Bit 31:3	Reserved	0x0000000	resd	Kept at its default value
Bit 2	SLIB_ULKF	0	ro	Security library unlock flag When this bit is set, it indicates that sLib-related setting registers can be configured.
Bit 1	SLIB_PWD_OK	0	ro	Security library password ok This bit is set by hardware when the password is correct.
Bit 0	SLIB_PWD_ERR	0	ro	Security library password error This bit is set by hardware when the password is incorrect and the setting value of the password clear register is different from 0xFFFF FFFF. Note: When this bit is set, the hardware will no longer agree to re-program the password clear register until the next reset.

5.7.13 Flash CRC address register (FLASH_CRC_ARR)

For main Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31:0	CRC_ADDR	0x0000 0000	wo	CRC address This register is used to select a start address of a page to be CRC checked..

Note: All these bits are write-only, and return no response when being read.

5.7.14 Flash CRC control register (FLASH_CRC_CTRL)

For main Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31:17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CRC_STRT	0x0	W0	CRC start This bit is used to enable CRC check for user code or sLib code. It is automatically cleared after enabling CRC by hardware. Note: CRC data ranges from CRC_ADDR to CRC_ADDR+CRC_SN*1
Bit 15: 0	CRC_SN	0x0000	wo	CRC page number This bit defines the pages to be CRC checked.

5.7.15 Flash CRC check result register (FLASH_CRC_CHKR)

For Flash memory and its extension area only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	CRC_CHKR	0x0000 0000	ro	CRC check result

Note: All these bits are write-only, and return no response when being read.

5.7.16 Security library password setting register (SLIB_SET_PWD)

For Flash security library password setting only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_PSET_VAL	0x0000 0000	ro	sLib password setting value Note: This register can be written only after sLib is unlocked. It is used to set a password of sLib. Writing 0xFFFF_FFFF or 0x0000_0000 has no effect.

Note: All these bits are write-only, and return 0 when being read.

5.7.17 Security library address setting register (SLIB_SET_RANGE)

For Flash security library address setting only.

Bit	Register	Reset value	Type	Description
Bit 31: 22	SLIB_ES_SET	0x000	wo	Security library end page setting Theses bits are used to set the security library end page. 0000000000: Page 0 0000000001: Page 1 0000000010: Page 2 … 0000111111: Page 63 (the last page of 64KB main Flash memory) … 0001111111: Page 127 (the last page of 256KB and 128KB main Flash memory)
Bit 21: 11	SLIB_ISS_SET	0x000	wo	Security library instruction start page setting These bits are used to set the security library instruction start page. 00000000000: Invalid page. Setting this bit will cause security library to fail to be enabled 00000000001: Page 1 00000000010: Page 2 … 00000111111: Page 63 (the last page of 64KB main Flash memory)

				… 00001111111: Page 127 (the last page of 256KB and 128KB main Flash memory) 11111111111: No data sLib
Bit 10: 0	SLIB_SS_SET	0x000	wo	Security library start page setting These bits are used to set the security library start page. 00000000000: Page 0 00000000001: Page 1 00000000010: Page 2 … 00000111111: Page 63 (the last page of 64KB main Flash memory) … 00001111111: Page 127 (the last page of 256KB and 128KB main Flash memory)

Note:

All these bits are write-only, and return 0 when being read.

This register can be written only after unlocking security library lock.

Being out of the Flash address range is an invalid setting.

5.7.18 Flash extension memory security library setting register (EM_SLIB_SET)

For Flash extension area only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
Bit 23: 16	EM_SLIB_ISS_SET	0x000	wo	Extension memory sLib instruction start page 00000000: Invalid page. Setting this bit will cause security library to fail to be enabled 00000001: Page 1 00000010: Page 2 … 00001000: Page 8 (the last page of 256KB main Flash memory) … 00010001: Page 17 (the last page of 64KB and 128KB main Flash memory) 11111111: No data sLib Others: Invalid Note: When it is set to 0xFF, it indicates that the Flash memory extension area from page 0 to page 17 is the security library, and the entire security library is used as instruction area.
Bit 15: 0	EM_SLIB_SET	0x000	wo	Extension memory sLib setting Extension memory is configured as security library by writing 0x5AA5.

Note: All these bits are write-only, and return no response when being read.

5.7.19 Boot mode setting register (BTM_MODE_SET)

For boot loader code area only.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	BTM_MODE_SET	0x00	wo	Boot memory mode setting 0xFF: Bootloader code area serves as a system area that stores system boot code Others: Bootloader code area serves a Flash extension area that stores application code Note: This register can be set only when Flash access protection is disabled.

Note: All these bits are write-only, and return no response when being read.

5.7.20 Security library unlock register (FLASH_UNLOCK)

For security library register unlock only.

Bit	Register	Reset value	Type	Description
Bit 31: 0	SLIB_UKVAL	0x0000 0000	wo	Security library unlock key value Fixed key value is 0xA35F_6D24, used for security library setting register unlock

Note: All these bits are write-only, and return 0 when being read.

6 General-purpose I/Os (GPIOs)

6.1 Introduction

AT32F415 series supports up to 55 bidirectional I/O pins, which are grouped as five categories, namely PA, PB, PC, PD and PF. Each of the GPIO group provides up to 16 I/O pins that feature communication, control and data collection. In addition, their main features also include:

Supports general-purpose I/O (GPIO) or multiplexed function I/O (IOMUX)

- Each pin can be configured by software as floating input, pull-up/pull-down input, analog input/output, push-pull/open-drain output, multiplexed push-pull/open-drain output
- Each pin's output drive capability is configurable by software
- Each pin can be configured as external interrupt input
- Each pin can be locked

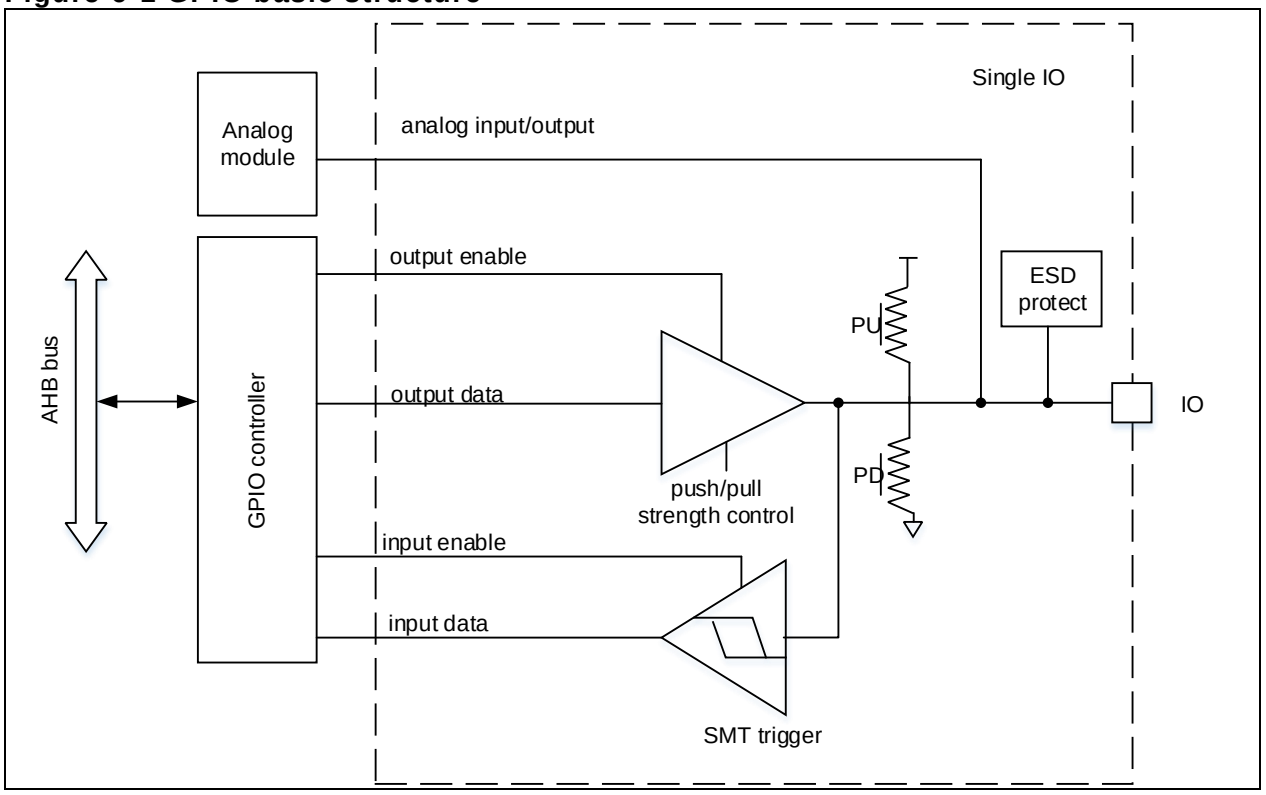
6.2 Functional overview

6.2.1 GPIO structure

Each of the GPIO pins can be configured by software as four input modes (floating, pull-up/pull-down and analog input) and four output modes (open-drain, push-pull, alternate function push-pull/open-drain output)

Each I/O port bit can be programmed freely. However, I/O port registers must be accessed by words (32 bits).

Figure 6-1 GPIO basic structure



6.2.2 GPIO reset status

After power-on or system reset, all pins are configured as floating input mode except JATG-related pins. JTAG pin configuration are as follows:

- PA15/JTDI, PA13/JTMS and PB4/JNTRST in pull-up input mode;
- PA14/JTCK in pull-down input mode;
- PB3/TDO in floating input mode.

6.2.3 General-purpose input configuration

Mode	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT register
Floating input	01				Unused
Pull-down input	10		000		0
Pull-up input					1

When I/O port is configured as input:

- Get I/O states by reading the input data register.
- Floating input, pull-up/pull-down input is configurable
- Schmitt-trigger input is activated.
- Output is disabled.

Note: In floating input mode, it is recommended to set the unused pins as analog input mode in order to avoid leakage caused by interference from unused pins in a complex environment.

6.2.4 Analog input/output configuration

Mode	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT register
Analog input/output	00		000		Unused

When I/O port is configured as analog input:

- Schmitt-trigger input is disabled.
- Digital input/output is disabled.
- Without any pull-up/pull-down resistor.

6.2.5 General-purpose output configuration

Mode	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT register
Push-Pull	00	001: Output mode, large sourcing/sinking strength 010: Output mode, normal sourcing/sinking strength 011: Output mode, normal sourcing/sinking strength 1xx: Output mode, Maximum sourcing/sinking strength			0 or 1
Open-Drain	01				0 or 1

When I/O port is configured as output:

- Schmitt-trigger input is enabled
- Output through output register
- Pull-up/pull-down resistors are disabled
- In open-drain mode, forced output 0, and use external pull-up resistor to output 1
- In push-pull mode, output register is used to output 0/1
- Multiplexed output is enabled when CONF=10 or 11, see section IOMUX for details.

6.2.6 GPIO port protection

Locking mechanism can freeze the I/O configuration for the purpose of protection. When LOCK is applied to a port bit, its configuration cannot be modified until the next reset or power on.

6.3 GPIO registers

Table 6-1 lists GPIO register map and their reset values. These peripheral registers must be accessed by words (32 bits).

Table 6-1 GPIO register map and reset values

Register	Offset	Reset value
GPIOx_CFGLR	0x00	0x4444 4444
GPIOx_CFGHR	0x04	0x4444 4444
GPIOx_IDT	0x08	0x0000 XXXX
GPIOx_ODT	0x0C	0x0000 0000
GPIOx_SCR	0x10	0x0000 0000
GPIOx_CLR	0x14	0x0000 0000
GPIOx_WPR	0x18	0x0000 0000

6.3.1 GPIO configuration register low (GPIOx_CFGLR) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 30 Bit 27: 26 Bit 23: 22 Bit 19: 18 Bit 15: 14 Bit 11: 10 Bit 7: 6 Bit 3: 2	IOFCy	0x1	rw	GPIOx function configuration (y=0~7) As input (IOMCy[1: 0]=00): 00: Analog 01: Floating (after reset) 10: Pull-down or pull-up 11: Reserved As output (IOMCy[1: 0]!=00): 00: General-purpose push-pull 01: General-purpose open drain 10: Multiplexed push-pull 11: Multiplexed open drain
Bit 29: 28 Bit 25: 24 Bit 21: 20 Bit 17: 16 Bit 13: 12 Bit 9: 8 Bit 5: 4 Bit 1: 0	IOMCy	0x0	rw	GPIOx mode configuration (y=0~7) 00: Input mode (after reset) 01: Output mode, large sourcing/sinking strength 10: Output mode, normal sourcing/sinking strength 11: Output mode, maximum sourcing/sinking strength

Note: Some port registers have different reset values.

6.3.2 GPIO configuration register high (GPIOx_CFGHR) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 30 Bit 27: 26 Bit 23: 22 Bit 19: 18 Bit 15: 14 Bit 11: 10 Bit 7: 6 Bit 3: 2	IOFCy	0x1	rw	GPIOx function configuration (y=8~15) As input (IOMCy[1: 0]=00): 00: Analog 01: Floating (after reset) 10: Pull-down or pull-up 11: Reserved As output (IOMCy[1: 0]!=00): 00: General-purpose push-pull 01: General-purpose open drain 10: Multiplexed push-pull 11: Multiplexed open drain
Bit 29: 28 Bit 25: 24 Bit 21: 20 Bit 17: 16 Bit 13: 12 Bit 9: 8 Bit 5: 4 Bit 1: 0	IOMCy	0x0	rw	GPIOx mode configuration (y=8~15) 00: Input mode (after reset) 01: Output mode, large sourcing/sinking strength 10: Output mode, normal sourcing/sinking strength 11: Output mode, maximum sourcing/sinking strength

Note: Some port registers have different reset values.

6.3.3 GPIO input register (GPIOx_IDT) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0.
Bit 15: 0	IDT	0xXXXX	ro	GPIOx input data Indicates the input status of I/O port. Each bit corresponds to an I/O.

6.3.4 GPIO output register (GPIOx_ODT) (x= A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Always 0.
Bit 15: 0	ODT	0x0000	rw	GPIOx output data Each bit represents an I/O port. As output: it indicates the output status of I/O port. 0: Low 1: High As input: it indicates the pull-up/pull-down status of I/O port. 0: Pull-down 1: Pull-up

6.3.5 GPIO set/clear register (GPIOx_SCR) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 16	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits
Bit 15: 0	IOSB	0x0000	wo	GPIOx set bit The corresponding ODT register bit is set by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Set the corresponding ODT bits

6.3.6 GPIO bit clear register (GPIOx_CLR) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	IOCB	0x0000	wo	GPIOx clear bit The corresponding ODT register bit is cleared by writing “1” to these bits. Otherwise, the corresponding ODT register bit remains unchanged, which acts as ODT register bit operations. 0: No action to the corresponding ODT bits 1: Clear the corresponding ODT bits

6.3.7 GPIO write protection register (GPIOx_WPR) (x=A...F)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	WPSEQ	0x0	rw	Write protect sequence Write protect enable sequence bit and WPEN bit must be enabled at the same time to achieve write protection for some I/O bits. Write protect enable bit is executed four times in the order below: write “1” -> write “0” -> write “1” -> read. Note that the value of WPEN bit cannot be modified during this period.
Bit 15: 0	WPEN	0x0000	rw	Write protect enable Each bit corresponds to an I/O port. 0: No effect. 1: Write protect

7 Multiplexed function I/Os (IOMUX)

7.1 Introduction

AT32F415 series support up to 55 bi-directional I/O pins, which are grouped as five categories, namely PA, PB, PC, PD and Pf. Each of the GPIO group provides up to 16 I/O pins that feature communication, control and data collection. In addition, their main features also include:

- Supports general-purpose I/O (GPIO) or multiplexed I/O (IOMUX), which will be detailed in this chapter.
- Can be configured as multiplexed function input/output by setting GPIOx_CFGLR or GPIOx_CFGHR register
- Most pins support output mapping for several peripherals. Select different peripheral input/output through IOMUX register
- Supports external interrupts

7.2 Functional overview

7.2.1 IOMUX structure

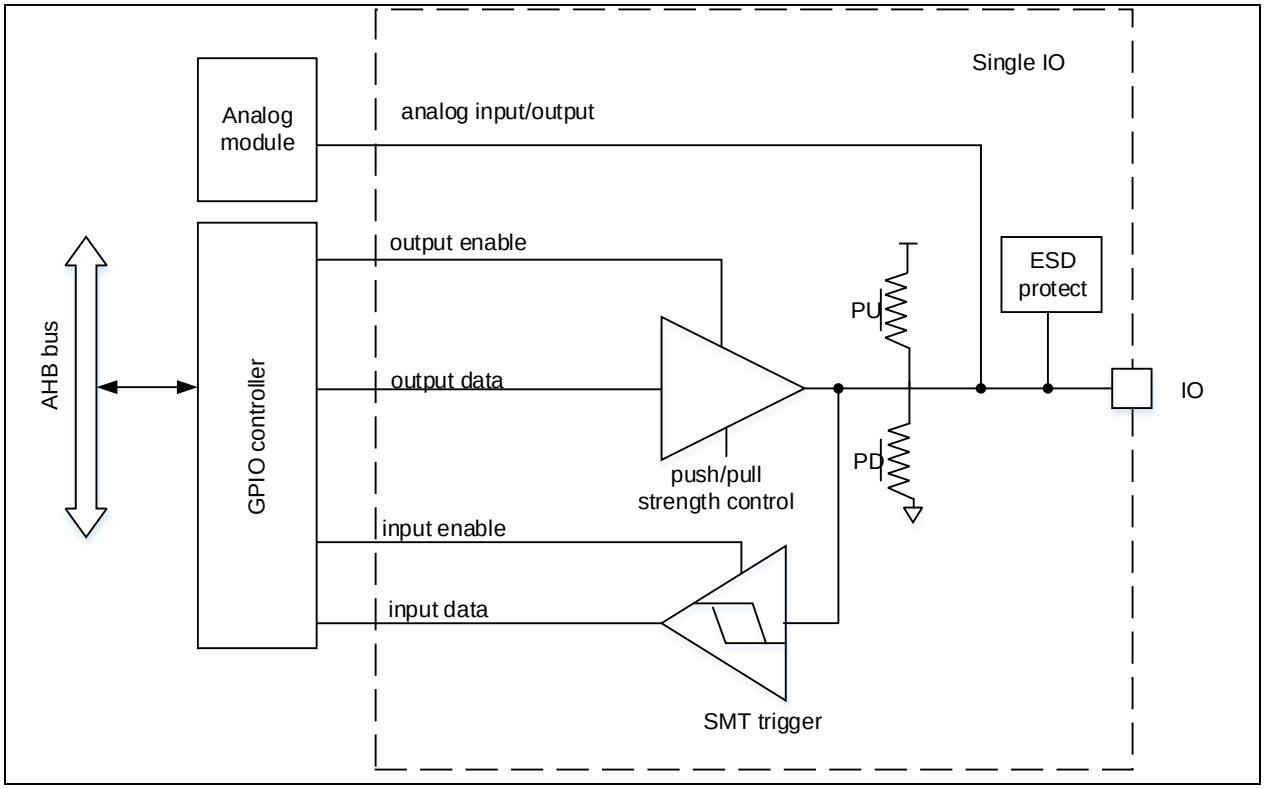
As multiplexed function input, the I/O port should be configured as input modes (floating, pull-up and pull-down input).

To enable multiplexed function output, the port must be configured as multiplexed function output mode (push-pull or open-drain) by setting GPIOx_CFGLR or GPIOx_CFGHR register. In this case, the pin is disconnected from GPIO controller, and controlled by IOMUX controller, instead.

To achieve bidirectional multiplexed function, the port needs to be configured as multiplexed function output modes (push-pull or open-drain), controlled by IOMUX controller.

In MUX output mode, it is possible that an I/O pin is used as an output for several peripherals. Select the required multiplexed function output through IOMUX registers. However, when a pin is programmed as MUX IO without activating the corresponding peripheral, its output will not specified.

Figure 7-1 Basic structure of IOMUX basic structure



7.2.2 MUX Input configuration

When I/O ports are configured as multiplexed function input:

- Get I/O pin state by reading input data registers
- The pin be configured as floating input, pull-up or pull-down input
- Schmitt-trigger input is activated.
- Pin output is disabled.

Table 7-1 IOMUX input configuration

Mode	IOFC	HDRV	IOMC[1]	IOMC[0]	ODT register
Floating input	01				Unused
Pull-down input	10		000		0
Pull-up input					1

7.2.3 MUX output or bidirectional MUX configuration

When an I/O port is configured as MUX output or a bidirectional MUX:

- I/O pin output depends on the peripherals.
- Schmitt-trigger input is activated.
- Pull-up/pull-down resistor is disabled.
- If the I/O pin is set as several MUX outputs by mistake, the pin output depends on map priority, refer to next section for details.
- In open-drain mode, get an I/O port state by reading input data register
- In push-pull mode, get an I/O port state by reading input data register

The MUX functions of some peripherals can be remapped to different pins. Therefore, it is necessary to select the number of the desired peripheral IOMUX functions in different packages. Pin mapping is achieved by setting the IOMUX_REMAP and IOMUX_REMAPx (x=2,3,8…)registers.

Table 7-2 IOMUX output configuration

Mode	IOFC	HDRV IOMC[1] IOMC[0]
Push-Pull	10	001: Output mode, large sourcing/sinking strength
Open-Drain	11	010: Output mode, normal sourcing/sinking strength 011: Output mode, normal sourcing/sinking strength 1xx: Output mode, maximum sourcing/sinking strength

Note: For MUX function output or bidirectional MUX function, IOMC[1: 0] > 00 must be met.

7.2.4 IOMUX map priority

When several peripheral MUX functions are mapped to the same pin, the priority below should be respected:

- Hardware preemption
- JTAG debug port
- Non-timer peripherals has priority over timer peripherals
- No priority applied among several non-timer peripherals, MUX function is overlapped to the same pin

7.2.4.1 Hardware preemption

Certain pins are occupied by specific hardware functions regardless of the GPIO configuration.

Table 7-3 Hardware preemption

Pin	Enable bit	Description
PA0	PWC_CTRLSTS[8] =1	Once enabled, PA0 pin acts as WKUP function of PWC.
PA11	CRM_APB1EN[23]=1	Once enabled, PA11 pin acts as USB_DM channel.
PA12	CRM_APB1EN[23]=1	Once enabled, PA12 pin acts as USB_DP channel.
PC13	CRM_APB1EN[27]=1& (BPR_CTRL[0]=1 | BPR_RTCCAL[8]=1 | BPR_RTCCAL[7]=1)	Once enabled, PC13 pin acts as RTC channel.
PC14	CRM_BPDC[0]=1	Once enabled, PC14 pin acts as LEXT channel.
PC15	CRM_BPDC[0]=1	Once enabled, PC15 pin acts as LEXT channel.

7.2.4.2 Debug port priority

The programmed debug pins will remain its state during device debugging, regardless of their GPIO register configuration. By doing this, can the debug port be free from disturbance imposed by other peripherals.

To utilize more pins during this period, the above-mentioned remap configuration can be changed by setting the SWJTAG_MUX [2:0] bit in the IOMUX_REMAP register and SWJTAG_GMUX [2:0] bit in the IOMUX_REMAP7 register.

Table 7-4 Debug port map

SWJTAG_MUX [2: 0] or SWJTAG_GMUX [2: 0]	SWJI/O pin allocation				
	PA13/JTMS/ SWDIO	PA14/JTCK/ SWCLK	PA15/JTDI	PB3/JTDO/ TRACESWO	PB4/NJTRST
000	√	√	√	√	√
001	√	√	√	√	x
010	√	√	x	x	x
100	x	x	x	x	x
Others	-	-	-	-	-

Note: ✓indicates that this pin is forcibly allocated to debug port, while x indicates that this pin can be released to other peripherals.

7.2.4.3 Other peripheral output priority

For other peripherals, their output priority are as follows:

- Non-timer peripherals have priority over timers. In other words, when other peripherals and timers are mapped to the same pin at the same time, the timer cannot be output.
- When multiple non-timer peripherals are mapped to the same pin, their output are overlapped to this pin.

7.2.5 External interrupt/wake-up lines

Each pin can be used as an external interrupt input. The corresponding pin should be configured as input mode.

7.3 IOMUX input/output

IP	IP pin multiplexed function	GPIO configuration
CAN	**CAN_MUX** 00: RX/PA11, TX/PA12 10: RX/ PB8, TX/ PB9 Others: Unused **CAN1_GMUX** 0000: RX/PA11, TX/PA12 0010: RX/ PB8, TX/ PB9 Others: Unused	CAN*_TX: Multiplexed push-pull output CAN*_RX: Floating input or pull-up input
ADC1	**ADC1_ETP_MUX** 0: ADC1 external trigger preempted conversion is connected to EXINT15; 1: ADC1 external trigger preempted conversion is connected to TMR8 channel 4. **ADC1_ETO_MUX** 0: ADC1 external trigger ordinary is connected to EXINT11; 1:ADC1 external trigger ordinary is connected toTMR8_TRGO.	ADC channel input pin: analog input
TMR1	**TMR1_MUX** 00: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PB12, CH1C/PB13, CH2C/PB14, CH3C/PB15; 01: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1; Others: Unused **TMR1_GMUX** 0000: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PB12, CH1C/PB13, CH2C/PB14, CH3C/PB15; 0001: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1; 0010: EXT/PA0, CH1/PC6, CH2/PC7, CH3/PC8, CH4/PC9, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1; Others: Unused	TMRx_CHx : Input capture channel x, configured as floating input Output compare channel x, configured as multiplexed push-pull output TMRx_CHxC: Configured as multiplexed push-pull output TMRx_BRK: Configured as floating input TMRx_EXT: Configured as floating input
TMR2	**TMR2_MUX** 00: CH1/EXT/PA0, CH2/PA1, CH3/PA2, CH4/PA3; 01: CH1/EXT/PA15, CH2/PB3, CH3/PA2, CH4/PA3; 10: CH1/EXT/PA0, CH2/PA1, CH3/PB10, CH4/PB11; 11: CH1/EXT/PA15, CH2/PB3, CH3/PB10, CH4/PB11. **TMR2_GMUX** 000: CH1_EXT/PA0 CH2/PA1 CH3/PA2 CH4/PA3 001: CH1_EXT/PA15 CH2/PB3 CH3/PA2 CH4/PA3 010: CH1_EXT/PA0 CH2/PA1 CH3/PB10 CH4/PB11 011: CH1_EXT/PA15 CH2/PB3 CH3/PB10 CH4/PB11	
TMR3	**TMR3_MUX**	

IP	IP pin multiplexed function	GPIO configuration
	00: CH1/PA6, CH2/PA7, CH3/PB0, CH4/PB1 01: Unused 10: CH1/PB4, CH2/PB5, CH3/PB0, CH4/PB1 11: CH1/PC6, CH2/PC7, CH3/PC8, CH4/PC9 Note: IO muxing does not affect the TMR3_EXT on PD2. **TMR3_GMUX** 0000: CH1/PA6 CH2/PA7 CH3/PB0 CH4/PB1 0010: CH1/PB4 CH2/PB5 CH3/PB0 CH4/PB1 0011: CH1/PC6 CH2/PC7 CH3/PC8 CH4/PC9 Others: Unused	
TMR4	NA	
TMR5	**TMR5CH4_MUX** 0: TMR5_CH4 is connected to PA3; 1: TMR5_CH4 is connected to LICK clock for LICK calibration. **TMR5_GMUX** 0000: CH1/PA0 CH2/PA1 CH3/PA2 CH4/PA3 0001: CH1/PF4 CH2/PF5 CH3/PA2 CH4/PA3 Others: Unused	
TRM9	**TMR9_GMUX** 0000: CH1/PA2 CH2/PA3 0010: CH1/PB14 CH2/PB15 Others: Unused	
TRM10	**TMR10_GMUX** 0000: CH1/PB8 0010: CH1/PA6 Others: Unused	
TRM11	**TMR11_GMUX** 0000: CH1/PB9 0010: CH1/PA7 Others: Unused	
USART1	**USART1_MUX** 0: TX/PA9, RX/PA10 1: TX/PB6, RX/PB7 **USART1_GMUX** 0000: TX/PA9, RX/PA10 0001: TX/PB6, RX/PB7 Others: Unused	USARTx_TX: Configured as multiplexed push-pull output USARTx_RX: Configured as floating input or pull-up input USARTx_CK: Configured as multiplexed push-pull output USARTx_RTS: Configured as multiplexed push-pull output USARTx_CTS: Configured as floating input or pull-up input
USART2	NA	
USART3	**USART3_MUX** 00: TX/PB10, RX/PB11, CK/PB12, CTS/PB13, RTS/PB14 01: TX/PC10, RX/PC11, CK/PC12, CTS/PB13, RTS/PB14 10: TX/PA7, RX/PA6, CK/PA5, CTS/PB1, RTS/PB0 11: Unused **USART3_GMUX** 0000: TX/PB10, RX/PB11, CK/PB12, CTS/PB13, RTS/PB14 0001: TX/PC10, RX/PC11, CK/PC12, CTS/PB13, RTS/PB14 0010: TX/PA7, RX/PA6, CK/PA5, CTS/PB1, RTS/PB0 Others: Unused	

IP	IP pin multiplexed function	GPIO configuration
UART4	**UART4_GMUX** 0000: TX/PC10 RX/PC11 0001: TX/PF4 RX/PF5 Others: Unused	
I2C1	**I2C1_MUX** 0: SCL/PB6, SDA/PB7 SMBA/PB5 1: SCL/PB8, SDA/PB9 SMBA/PB5 **I2C1_GMUX** 0000: SCL/PB6, SDA/PB7, SMBA/PB5 0001: SCL/PB8, SDA/PB9, SMBA/PB5 0010: SCL/PF6, SDA/PF7, SMBA/PB5 Others: Unused	I2Cx_SCL: Configured as multiplexed open-drain output I2Cx_SDA: Configured as multiplexed open-drain output
I2C2	**I2C2_GMUX** 0000: SCL/PB10, SDA/PB11, SMBA/PB12 0001: SCL/PA8, SDA/PC9, SMBA/PA9 0010: SCL/PA8, SDA/PB4, SMBA/PA9 0011: SCL/PF6, SDA/PF7, SMBA/PA9 Others: Unused	
SPI1	**SPI1_MUX** 00: CS/PA4, SCK/PA5, MISO/PA6, MOSI/PA7 MCK/PB0 01: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5 MCK/PB6 10, 11: Unused **SPI1_GMUX** 0000: CS/PA4, SCK/PA5, MISO/PA6, MOSI/PA7 MCK/PB0 0001: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5 MCK/PB6 Others: Unused	SPIx_SCK Master mode configured as multiplexed push-pull output Slave mode configured as floating input SPIx_MOSI Full-duplex/master mode or bidirectional data line/master mode configured as multiplexed push-pull output Full-duplex/slave mode configured as floating input or pull-up input SPIx_MISO Full-duplex/master mode configured as floating input or pull-up input Full-duplex/slave mode or bidirectional data line/slave mode configured as multiplexed push-pull output SPIx_CS Hardware master/slave mode configured as floating input, pull-up mode or pull-down mode Hardware master mode/CS output enable configured as multiplexed push-pull output
SPI2	**SPI2_GMUX** 0000: CS/PB12, SCK/PB13, MISO/PB14, MOSI/PB15 MCK/PC6 0001: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5 MCK/PC7 Others: Unused	
SDIO1	**SDIO1_GMUX** 0000: D0/PC8, D1/PC9, D2/PC10, D3/PC11, D4/PB8, D5/PB9, D6/PC6, D7/PC7, CK/PC12, CMD/PD2 0100: D0/PC0, D1/PC1, D2/PC2, D3/PC3, D4/PA4, D5/PA5, D6/PA6, D7/PA7, CK/PC4, CMD/PC5 0101: D0/PA4, D1/PA5, D2/PA6, D3/PA7, CK/PC4, CMD/PC5 0110: D0/PC0, D1/PC1, D2/PC2, D3/PC3, D4/PA4, D5/PA5, D6/PA6, D7/PA7, CK/PA2, CMD/PA3	SDIO_CK Configured as multiplexed push-pull output SDIO_CMD Configured as multiplexed push-pull output SDIO[D7:D0] Configured as multiplexed push-pull output

IP	IP pin multiplexed function	GPIO configuration
	0111: D0/PA4, D1/PA5, D2/PA6, D3/PA7, CK/PA2, CMD/PA3 Others: Unused	
USB	NA	Once the USB module is enabled, the USBFS1_D-/USBFS1_D+ will be connected to the internal USB receiver/transmitter automatically.
CMP	**CMP_MUX** When this bit is set to “00”, CMP1_OUT is connected to PA0, and CMP2_OUT is connected to PA2. When this bit is set to “01”, CMP1_OUT is connected to PA6, and CMP2_OUT is connected to PA7. When this bit is set to “10”, CMP1_OUT is connected to PA11, and CMP2_OUT is connected to PA12; Others: Reserved	CMP1_OUT Configured as multiplexed push-pull output CMP2_OUT Configured as multiplexed push-pull output Input channel Configured as analog input mode
TAMPER_RTC	NA	The configurations of BRKP_CTRL and BRKP_RTCCAL registers are forced by hardware.
CLKOUT	NA	Configured as multiplexed push-pull output
EXINT input line	NA	Configured as floating input, pull-up input or pull-down input

Note: “NA” represents no pin muxing mapping. Refer to the datasheet for the corresponding pins of peripherals.

7.4 IOMUX registers

Table 7-5 shows IOMUX register map and their reset values. These peripheral registers must be accessed by words (32 bits).

Table 7-5 IOMUX register map and reset value

Register	Offset	Reset value
IOMUX_EVTOUT	0x00	0x0000 0000
IOMUX_REMAP	0x04	0x0000 0000
IOMUX_EXINTC1	0x08	0x0000
IOMUX_EXINTC2	0x0C	0x0000
IOMUX_EXINTC3	0x10	0x0000
IOMUX_EXINTC4	0x14	0x0000
IOMUX_REMAP2	0x1C	0x0000 0000
IOMUX_REMAP3	0x20	0x0000 0000
IOMUX_REMAP4	0x24	0x0000 0000
IOMUX_REMAP5	0x28	0x0000 0000
IOMUX_REMAP6	0x2C	0x0000 0000
IOMUX_REMAP7	0x30	0x0000 0000
IOMUX_REMAP8	0x34	0x0000 0000

Note: IOMUX clock must be enabled before read/write access to IOMUX_EVCOUT, IOMUX_REMAPx and IOMUX_EXINTx registers.

7.4.1 Event output control register (IOMUX_EVTOUT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	EVOEN	0x0	rw	Event output enable Once enabled, the TXEV signal of Cortex®-M is directed to the allocated I/O port.
Bit 6: 4	SELPORT	0x0	rw	Selection IO port Select the GPIO port for EVENTOUT signal output: 000: GPIOA 001: GPIOB 010: GPIOC 011: GPIOD 101: GPIOF
Bit 3: 0	SELPIN	0x0	rw	Selection IO pin (x=A…E) Select the I/O pin of GPIOx for EVENTOUT output: 0000: Pin 0 0001: Pin 1 0010: Pin 2 0011: Pin 3 0100: Pin 4 0101: Pin 5 0110: Pin 6 0111: Pin 7 1000: Pin 8 1001: Pin 9 1010: Pin 10 1011: Pin 11 1100: Pin 12 1101: Pin 13 1110: Pin 14 1111: Pin 15

7.4.2 IOMUX remap register (IOMUX_REMAP)

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 27	Reserved	0x0	resd	Kept at its default value..
Bit 26: 24	SWJTAG_MUX	0x0	rw	SWD JTAG multiplexing These bits are used to configure SWJTGA-related I/Os as GPIOs. 000: Supports SWD and JTAG. All SWJTAG pins cannot be used as GPIOs. 001: Supports SWD and JTAG. NJTRST is disabled. PB4 can be used as GPIO. 010: Supports SWD but JTAG is disabled. PA15/PB3/PB4 can be used as GPIOs. 100: SWD and JTAG are disabled. All SWJTAG pins can be used as GPIOs. Others: No effect.
Bit 23: 19	Reserved	0x0	resd	Kept at its default value.
Bit 18	ADC1_ETO_MUX	0x0	rw	ADC1 external trigger regular conversion multiplexing Select external trigger input for ADC1 ordinary conversion. 0: ADC1 external trigger ordinary conversion is connected to EXINT11. 1: ADC1 external trigger ordinary conversion TMR1_TRGO.
Bit 17	ADC1_ETP_MUX	0x0	rw	ADC1 external trigger preempted conversion multiplexing This bit is used to select external trigger input for ADC1 preempted conversion. 0: ADC1 external trigger preempted conversion is connected to EXINT15; 1: ADC1 external trigger preempted conversion is connected to TMR1 channel 4.
Bit 16	TMR5CH4_MUX	0x0	rw	TMR5 channel4 multiplexing Select internal map for TMR5 channel 4. 0: TMR5_CH4 is connected to PA3. 1: TMR5_CH4 is connected to LICK. LICK can be calibrated.
Bit 15	PD01_MUX	0x0	rw	PD0/PD1 mapped on HEXT_IN / HEXT_OUT Select GPIO function map for PD0 and PD1. This is only applicable to 48-pin/64-pin packages. 0: Not PD0 and PD1 mapping 1: PD0 is mapped to HEXT_IN, while PD1 to HEXT_OUT.
Bit 14: 13	CAN_MUX	0x0	rw	CAN IO multiplexing Select IO multiplexing for CAN_TX and CAN_RX. 00: RX/PA11, TX/PA12 01: Unused 10: RX/ PB8, TX/ PB9 11: Unused
Bit 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 10	TMR3_MUX	0x0	rw	TMR3 IO multiplexing Select IO multiplexing for TMR3. 00: CH1/PA6, CH2/PA7, CH3/PB0 and CH4/PB1 01: Unused 10: CH1/PB4, CH2/PB5, CH3/PB0 and CH4/PB1 11: CH1/PC6, CH2/PC7, CH3/PC8 and CH4/PC9

				Note: IO multiplexing has no impact on TMR3_EXT on PD2.
Bit 9: 8	TMR2_MUX	0x0	rw	TMR2 IO multiplexing Select IO multiplexing for TMR2. 00: CH1/EXT/PA0, CH2/PA1, CH3/PA2 and CH4/PA3 01: CH1/EXT/PA15, CH2/PB3, CH3/PA2 and CH4/PA3 10: CH1/EXT/PA0, CH2/PA1, CH3/PB10 and CH4/PB11 11: CH1/EXT/PA15, CH2/PB3, CH3/PB10 and CH4/PB11
Bit 7: 6	TMR1_MUX	0x0	rw	TMR1 IO multiplexing Select IO multiplexing for TMR1. 00: EXT/PA12, CH1/PA8, CH2/PA9 and CH3/PA10 CH4/PA11, BRK/PB12, CH1C/PB13, CH2C/PB14, CH3C/PB15 01: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1 10: Unused 11: Unused
Bit 5: 4	USART3_MUX	0x0	rw	USART3 IO multiplexing Select IO multiplexing for USART3. 00: TX/PB10, RX/PB11, CK/PB12, CTS/PB13, RTS/PB14 01: TX/PC10, RX/PC11, CK/PC12, CTS/PB13, RTS/PB14 10: TX/PA7, RX/PA6, CK/PA5, CTS/PB1, RTS/PB0 11: Unused
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	USART1_MUX	0x0	rw	USART1 IO multiplexing Select USART1 IO multiplexing 0: TX/PA9, RX/PA10 1: TX/PB6, RX/PB7
Bit 1	I2C1_MUX	0x0	rw	I2C1 IO multiplexing Select I2C1 IO multiplexing. 0: SCL/PB6, SDA/PB7 SMBA/PB5 1: SCL/PB8, SDA/PB9 SMBA/PB5
Bit 0	SPI1_MUX	0x0	rw	SPI1 IO multiplexing Select SPI1 IO multiplexing. SPI1_MUX[1] is in bit 31. 00: CS/PA4, SCK/PA5, MISO/PA6, MOSI/PA7, MCK/PB0 01: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5, MCK/PB6 10, 11: Unused

7.4.3 IOMUX external interrupt configuration register1 (IOMUX_EXINTC1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT3	0x0000	rw	EXINT3 input source configuration Select the input source for EXINT3 external interrupt. 0000: GPIOA pin3 0001: GPIOB pin3 0010: GPIOC pin3 0011: GPIOD pin3 0100: GPIOF pin3 Others: Reserved.
Bit 11: 8	EXINT2	0x0000	rw	EXINT2 input source configuration

				Select the input source for EXINT2 external interrupt. 0000: GPIOA pin2 0001: GPIOB pin2 0010: GPIOC pin2 0011: GPIOD pin2 0100: GPIOF pin2 Others: Reserved.
Bit 7: 4	EXINT1	0x0000	rw	EXINT1 input source configuration Select the input source for EXINT1 external interrupt. 0000: GPIOA pin1 0001: GPIOB pin1 0010: GPIOC pin1 0011: GPIOD pin1 0100: GPIOF pin1 Others: Reserved.
Bit 3: 0	EXINT0	0x0000	rw	EXINT0 input source configuration Select the input source for EXINT0 external interrupt. 0000: GPIOA pin0 0001: GPIOB pin0 0010: GPIOC pin0 0011: GPIOD pin0 0100: GPIOF pin0 Others: Reserved.

7.4.4 IOMUX external interrupt configuration register2 (IOMUX_EXINTC2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT7	0x0000	rw	EXINT7 input source configuration Select the input source for EXINT7 external interrupt. 0000: GPIOA pin7 0001: GPIOB pin7 0010: GPIOC pin7 0011: GPIOD pin7 0100: GPIOF pin7 Others: Reserved.
Bit 11: 8	EXINT6	0x0000	rw	EXINT6 input source configuration Select the input source for EXINT6 external interrupt. 0000: GPIOA pin6 0001: GPIOB pin6 0010: GPIOC pin6 0011: GPIOD pin6 0100: GPIOF pin6 Others: Reserved.
Bit 7: 4	EXINT5	0x0000	rw	EXINT5 input source configuration Select the input source for EXINT5 external interrupt. 0000: GPIOA pin5 0001: GPIOB pin5 0010: GPIOC pin5 0011: GPIOD pin5 0100: GPIOF pin5 Others: Reserved.
Bit 3: 0	EXINT4	0x0000	rw	EXINT4 input source configuration

				Select the input source for EXINT4 external interrupt. 0000: GPIOA pin4 0001: GPIOB pin4 0010: GPIOC pin4 0011: GPIOD pin4 0100: GPIOF pin4 Others: Reserved.

7.4.5 IOMUX external interrupt configuration register3 (IOMUX_EXINTC3)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT11	0x0000	rw	EXINT11 input source configuration Select the input source for EXINT11 external interrupt. 0000: GPIOA pin11 0001: GPIOB pin11 0010: GPIOC pin11 0011: GPIOD pin11 0100: GPIOF pin11 Others: Reserved.
Bit 11: 8	EXINT10	0x0000	rw	EXINT10 input source configuration Select the input source for EXINT10 external interrupt. 0000: GPIOA pin10 0001: GPIOB pin10 0010: GPIOC pin10 0011: GPIOD pin10 0100: GPIOF pin10 Others: Reserved.
Bit 7: 4	EXINT9	0x0000	rw	EXINT9 input source configuration Select the input source for EXINT9 external interrupt. 0000: GPIOA pin9 0001: GPIOB pin9 0010: GPIOC pin9 0011: GPIOD pin9 0100: GPIOF pin9 Others: Reserved.
Bit 3: 0	EXINT8	0x0000	rw	EXINT8 input source configuration Select the input source for EXINT8 external interrupt. 0000: GPIOA pin8 0001: GPIOB pin8 0010: GPIOC pin8 0011: GPIOD pin8 0100: GPIOF pin8 Others: Reserved.

7.4.6 IOMUX external interrupt configuration register4 (IOMUX_EXINTC4)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 12	EXINT15	0x0000	rw	EXINT15 input source configuration Select the input source for EXINT15 external interrupt. 0000: GPIOA pin15 0001: GPIOB pin15 0010: GPIOC pin15 0011: GPIOD pin15 0100: GPIOF pin15 Others: Reserved.
Bit 11: 8	EXINT14	0x0000	rw	EXINT14 input source configuration Select the input source for EXINT14 external interrupt. 0000: GPIOA pin14 0001: GPIOB pin14 0010: GPIOC pin14 0011: GPIOD pin14 0100: GPIOF pin14 Others: Reserved.
Bit 7: 4	EXINT13	0x0000	rw	EXINT13 input source configuration Select the input source for EXINT13 external interrupt. 0000: GPIOA pin13 0001: GPIOB pin13 0010: GPIOC pin13 0011: GPIOD pin13 0100: GPIOF pin13 Others: Reserved.
Bit 3: 0	EXINT12	0x0000	rw	EXINT12 input source configuration Select the input source for EXINT12 external interrupt. 0000: GPIOA pin12 0001: GPIOB pin12 0010: GPIOC pin12 0011: GPIOD pin12 0100: GPIOF pin12 Others: Reserved.

7.4.7 IOMUX remap register2 (IOMUX_REMAP2)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x000	resd	Kept at its default value.
Bit 27: 26	CMP_MUX	0x0	w	CMP_MUX: CMP internal remap This field is set or cleared by software. It controls CMP internal remapping. 00: CMP1_OUT is connected to PA0, CMP2_OUT is connected to PA2; 01: CMP1_OUT is connected to PA6, CMP2_OUT is connected to PA7; 10: CMP1_OUT is connected to PA11, CMP2_OUT is connected to PA12; Others: Reserved.
Bit 25: 0	Reserved	0x00	resd	Kept at its default value.

7.4.8 IOMUX remap register3 (IOMUX_REMAP3)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x0000000	resd	Kept at its default value.
Bit 11: 8	TMR11_GMUX	0x0	rw	TMR11 IO general multiplexing Select IO multiplexing for TMR11. 0000: CH1/PB9 0010: CH1/PA7
Bit 7: 4	TMR10_GMUX	0x0	rw	TMR10 IO general multiplexing Select IO multiplexing for TMR10. 0000: CH1/PB8 0010: CH1/PA6
Bit 3: 0	TMR9_GMUX	0x0	rw	TMR9 IO general multiplexing Select IO multiplexing for TMR9. 0000: CH1/PA2, CH2/PA3 0010: CH1/PB14, CH2/PB15

7.4.9 IOMUX remap register4 (IOMUX_REMAP4)

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	TMR5CH4_GMUX	0x0	rw	TMR5 channel4 general multiplexing Select TMR5 channel4 general multiplexing 0: TMR5_CH4 is connected to PA3. 1: LICK is connected to TMR5_CH4 to get calibration.
Bit 18: 16	TMR5_GMUX	0x0	rw	TMR5 IO general multiplexing Select IO multiplexing for TMR4. 0000: CH1/PA0 CH2/PA1 CH3/PA2 CH4/PA3 0001: CH1/PF4 CH2/PF5 CH3/PA2 CH4/PA3
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 8	TMR3_GMUX	0x0	rw	TMR3 IO general multiplexing Select IO multiplexing for TMR3. 0000: CH1/PA6 CH2/PA7 CH3/PB0 CH4/PB1 0001: CH1/PB4 CH2/PB5 CH3/PB0 CH4/PB1
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	TMR2_GMUX	0x0	rw	TMR2 IO general multiplexing Select IO multiplexing for TMR2. 000: CH1_EXT/PA0 CH2/PA1 CH3/PA2 CH4/PA3 001: CH1_EXT/PA15 CH2/PB3 CH3/PA2 CH4/PA3 010: CH1_EXT/PA0 CH2/PA1 CH3/PB10 CH4/PB11 011: CH1_EXT/PA15 CH2/PB3 CH3/PB10 CH4/PB11
Bit 3: 0	TMR1_GMUX	0x0	rw	TMR1 IO general multiplexing Select IO multiplexing for TMR1. 0000: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PB12, CH1C/PB13, CH2C/PB14, CH3C/PB15; 0001: EXT/PA12, CH1/PA8, CH2/PA9, CH3/PA10, CH4/PA11, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1; 0010: EXT/PA0, CH1/PC6, CH2/PC7, CH3/PC8, CH4/PC9, BRK/PA6, CH1C/PA7, CH2C/PB0, CH3C/PB1; Others: Unused.

7.4.10 IOMUX remap register5 (IOMUX_REMAP5)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0	resd	Kept at its default value.
Bit 23: 20	SPI2_GMUX	0x0	rw	SPI2 IO general multiplexing Select IO multiplexing for SPI2. 0000: CS/PB12, SCK/PB13, MISO/PB14, MOSI/PB15 MCK/PC6 . 0001: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5 MCK/PC7 Others: Unused
Bit 19: 16	SPI1_GMUX	0x0	rw	SPI1 IO general multiplexing Select IO multiplexing for SPI1. 0000: CS/PA4, SCK/PA5, MISO/PA6, MOSI/PA7 MCK/PB0 0001: CS/PA15, SCK/PB3, MISO/PB4, MOSI/PB5 MCK/PB6 Others: Unused
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 8	I2C2_GMUX	0x0	rw	I2C2 IO general multiplexing Select IO multiplexing for I2C2. 0000: SCL/PB10, SDA/PB11, SMBA/PB12 0001: SCL/PA8, SDA/PC9, SMBA/PA9 0010: SCL/PA8, SDA/PB4, SMBA/PA9 0011: SCL/PF6, SDA/PF7, SMBA/PA9 Others: Unused
Bit 7: 4	I2C1_GMUX	0x0	rw	I2C1 IO general multiplexing Select IO multiplexing for I2C1. 0000: SCL/PB6, SDA/PB7, SMBA/PB5 0001: SCL/PB8, SDA/PB9, SMBA/PB5 0010: SCL/PF6, SDA/PF7, SMBA/PB5 Others: Unused
Bit 3: 0	Reserved	0x0	resd	Kept at its default value.

7.4.11 IOMUX remap register6 (IOMUX_REMAP6)

Bit	Register	Reset value	Type	Description
Bit 31: 28	UART4_GMUX	0x0	rw	IO general multiplexing Select IO multiplexing for UART4. 0000: TX/PC10 RX/PC11 0001: TX/PF4 RX/PF5 Others: Unused
Bit 27: 24	USART3_GMUX	0x0	rw	USART3 IO general multiplexing Select IO multiplexing for USART3. 0000: TX/PB10, RX/PB11, CK/PB12, CTS/PB13 RTS/PB14 0001: TX/PC10, RX/PC11, CK/PC12, CTS/PB13 RTS/PB14 0010: TX/PA7, RX/PA6, CK/PA5, CTS/PB1 RTS/PB0 Others: Unused
Bit 23: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 16	USART1_GMUX	0x0	rw	USART1 IO general multiplexing Select IO multiplexing for USART1. 0000: TX/PA9, RX/PA10 0001: TX/PB6, RX/PB7

				Others: Unused
Bit 15:12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 8	SDIO1_GMUX	0x0	rw	SDIO1 IO general muxing These bits are used to select SDIO IO general multiplexing. 0000: D0/PC8, D1/PC9, D2/PC10, D3/PC11, D4/PB8, D5/PB9, D6/PC6, D7/PC7, CK/PC12, CMD/PD2 0100: D0/PC0, D1/PC1, D2/PC2, D3/PC3, D4/PA4, D5/PA5, D6/PA6, D7/PA7, CK/PC4, CMD/PC5 0101: D0/PA4, D1/PA5, D2/PA6, D3/PA7, CK/PC4, CMD/PC5 0110: D0/PC0, D1/PC1, D2/PC2, D3/PC3, D4/PA4, D5/PA5, D6/PA6, D7/PA7, CK/PA2, CMD/PA3 0111: D0/PA4, D1/PA5, D2/PA6, D3/PA7, CK/PA2, CMD/PA3 Others: Unused
Bit 7: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3: 0	CAN1_GMUX	0x0	rw	CAN1 IO general multiplexing Select IO multiplexing for CAN1. 00: RX/PA11, TX/PA12 10: RX/ PB8, TX/ PB9 Others: Unused

7.4.12 IOMUX remap register7 (IOMUX_REMAP7)

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x0	resd	Kept at its default value.
Bit 20	PD01_GMUX	0x0	rw	PD0/PD1 mapped onto HEXT_IN / HEXT_OUT Select GPIO mapping for PD0 and PD1. This is applied to only 48-pin and 64-pin packages. 0: No PD0 and PD1 mapping 1: PD0 is mapped to HEXT_IN, and PD1 to HEXT_OUT
Bit 19	Reserved	0x0	resd	Kept at its default value.
Bit 18: 16	SWJTAG_GMUX	0x0	rw	SWD JTAG IO general multiplexing These bits are used to configure SWJTAG-related IOs as GPIO. 000: Supports SWD and JTAG. All SWJTAG pins cannot be used as GPIO. 001: Supports SWD and JTAG. NJTRST is disabled. PB4 can be used as GPIO. 010: Supports SWD. But JTAG is disabled. PA15/PB3/PB4 can be used as GPIO. 100: SWD and JTAG are disabled. All SWJTAG pins can be used as GPIO Others: No effect.
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	ADC1_ETO_GMUX	0x0	rw	ADC1 external trigger regular conversion general multiplexing Select the input source for ADC1 external trigger regular conversion. 0: ADC1 external trigger regular conversion is connected to EXINT11 1: ADC1 external trigger regular conversion is connected to TMR1_TRGO
Bit 4	ADC1_ETP_GMUX	0x0	rw	ADC1 External trigger preempted conversion general multiplexing

				This bit is set and cleared by software. It controls the trigger input connected to external triggers. When this bit is set, ADC1 external trigger preempted conversion is connected to TMR1 channel 4.
Bit 3: 0	Reserved	0x0	resd	Kept at its default value.

7.4.13 IOMUX remap register8 (IOMUX_REMAP8)

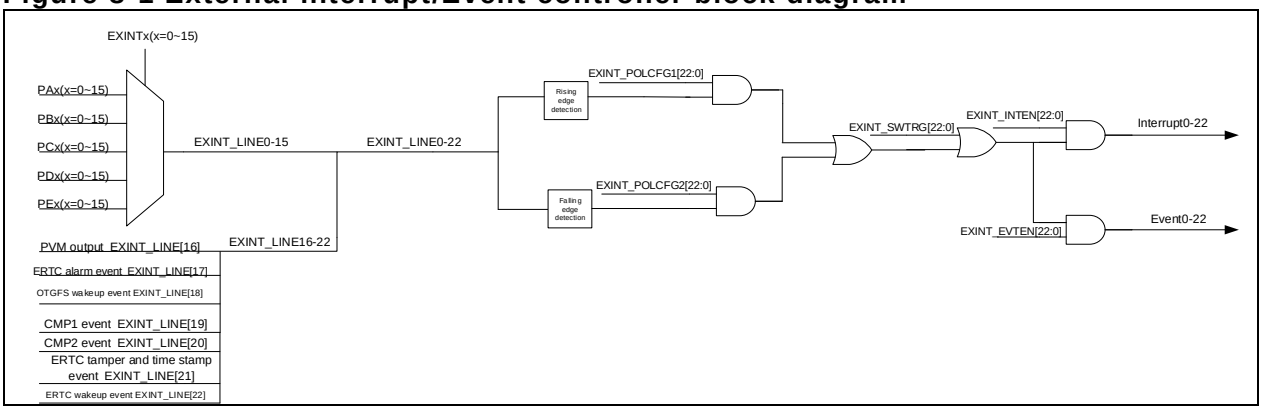
Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0	resd	Keep at its default value.
Bit 7: 6	TMR3_CH1_ CMP_GMUX	0x0	rw	TMR3 channel 1 internal mapping 00, 01: TMR3_GMUX IO signal is connected to TMR3 channel 1 10: CMP output signal is connected to TMR3 channel 1 11: Either CMP output signal or TMR3_GMUX IO signal is connected to TMR3 channel 1
Bit 5: 4	TMR2_CH4_ CMP_GMUX	0x0	rw	TMR2 channel 4 internal mapping 00, 01: TMR3_GMUX IO signal is connected to TMR2 channel 4 10: CMP output signal is connected to TMR2 channel 4 11: Either CMP output signal or TMR2_GMUX IO signal is connected to TMR2 channel 4
Bit 3: 2	TMR1_CH1_ CMP_GMUX	0x0	rw	TMR1 channel 1 internal mapping 00, 01: TMR1_GMUX IO signal is connected to TMR1 channel 1 10: CMP output signal is connected to TMR1 channel 1 11: Either CMP output signal or TMR1_GMUX IO signal is connected to TMR1 channel 1
Bit 1: 0	TMR1_BK1_ CMP_GMUX	0x0	rw	TMR1 break channel 1 internal mapping 00, 01: TMR1_GMUX IO signal is connected to TMR1 break channel 1 10: CMP output signal is connected to TMR1 break channel 1 11: Either CMP output signal or TMR1_GMUX IO signal is connected to TMR1 break channel 1

8 External interrupt/Event controller (EXINT)

8.1 EXINT introduction

EXINT consists of 23 interrupt lines EXINT_LINE[22:0], each of which can generate an interrupt or event by edge detection trigger or software trigger. EXINT can enable or disable an interrupt or event independently through software configuration, and utilizes different edge detection modes (rising edge, falling edge or both edges) as well as trigger modes (edge detection, software trigger or both triggers) to respond to the trigger source in order to generate an interrupt or event.

Figure 8-1 External interrupt/Event controller block diagram



Main features:

- EXINT 0~15 mapping IO can be configured independently
- Independent trigger selection on each interrupt line
- Independent enable bit on each interrupt
- Independent enable bit on each event
- Up to 23 software trigger that can be generated and cleared independently
- Independent status bit on each interrupt
- Each interrupt can be cleared independently.

8.2 Function overview and configuration procedure

With up to 23 interrupt lines EXINT_LINE[22:0], EXINT can detect not only GPIO external interrupt sources but also seven internal sources such as PVM output, ERTC alarm, OTGFS wakeup, CMP1 wakeup, CMP2 wakeup, ERTC tamper and time stamp events, and ERTC wakeup events. The GPIO interrupt sources can be selected with IOMUX_EXINTCx register. It should be noted that these input sources are mutually exclusive. For example, EXINT_LINE0 is allowed to select only one of PA0/PB0/PC0/PD0 pins, instead of taking both PA0 and PB0 as the input sources at the same time.

EXINT supports multiple edge detection modes, including rising edge, falling edge or both edges, selected by EXINT_POLCFG1 and EXINT_POLCFG2 register. Active edge trigger detected on the interrupt line can be used to generate an event or interrupt.

In addition, EXINT supports independent software trigger for the generation of an event or interrupt. This is achieved by setting the corresponding bits in the EXINT_SWTRG register.

EXINT can enable or disable an interrupt or event individually through software configuration such as EXINT_INTEN and EXINT_EVTEN registers, indicating that the corresponding interrupt or event control bit must be enabled in advance.

EXINT also features an independent interrupt status bit. Reading access to EXINT_INTSTS register can obtain the corresponding interrupt status. The status flag is cleared by writing “1” to this register.

- Writing “1” to the EXINT_INTSTS register to clear the interrupts generated, and the corresponding bits in the EXINT_SWTRG register.

Interrupt initialization procedure

1. Select an interrupt source by setting SCFG_EXINTCx register (This is required if GPIO is used as an interrupt source)
2. Select an trigger mode by setting EXINT_POLCFG1 and EXINT_POLCFG2 register
3. Enable interrupt or event by setting EXINT_INTEN and EXINT_EVTEN register
4. Generate software trigger by setting EXINT_SWTRG register (This is applied to software trigger interrupt only)

Note: if there is a need to modify interrupt source configuration, then switch off interrupt enable register and event enable register first before re-starting interrupt initialization configuration.

Interrupt clear procedure

- Writing "1" to the EXINT_INTSTS register to clear the interrupts generated, and the corresponding bits in the EXINT_SWTRG register.

8.3 EXINT registers

These peripheral registers must be accessed by words (32 bits).

Table 8-1 shows EXINT register map and their reset value.

Table 8-1 External interrupt/Event controller register map and reset value

Register	Offset	Reset value
EXINT_INTEN	0x00	0x0000 0000
EXINT_EVTEN	0x04	0x0000 0000
EXINT_ POLCFG1	0x08	0x0000 0000
EXINT_ POLCFG2	0x0C	0x0000 0000
EXINT_ SWTRG	0x10	0x0000 0000
EXINT_ INTSTS	0x14	0x0000 0000

8.3.1 Interrupt enable register (EXINT_INTEN)

Interrupt enable register (EXINT_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	INTENx	0x00000	rw	Interrupt enable or disable on line x 0: Interrupt request is disabled. 1: Interrupt request is enabled.

8.3.2 Event enable register (EXINT_EVTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	EVTENx	0x00000	rw	Event enable or disable on line x 0: Event request is disabled. 1: Event request is enabled.

8.3.3 Polarity configuration register1 (EXINT_ POLCFG1)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	RPx	0x00000	rw	Rising polarity configuration bit on line x These bits are used to select a rising edge to trigger an interrupt and event on line x. 0: Rising trigger on line x is disabled. 1: Rising trigger on line x is enable.

8.3.4 Polarity configuration register2 (EXINT_ POLCFG2)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to be 0 by hardware.
Bit 22: 0	FPx	0x00000	rw	Falling polarity configuration bit on line x These bits are used to select a falling edge to trigger an interrupt and event on line x. 0: Falling trigger on line x is disabled. 1: Falling trigger on line x is enabled..

8.3.5 Software trigger register (EXINT_ SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	SWTx	0x00000	rw	Software trigger on line x If the corresponding bit in EXINT_INTEN register is 1, the software writes to this bit. The hardware sets the corresponding bit in the EXINT_INTSTS automatically to generate an interrupt. If the corresponding bit in the EXINT_EVTEN register is 1, the software writes to this bit. The hardware generates an event on the corresponding interrupt line automatically. 0: Default value 1: Software trigger generated Note: This bit is cleared by writing 1 to the corresponding bit in the EXINT_INTSTS register.

8.3.6 Interrupt status register (EXINT_ INTSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 22: 0	LINEx	0x00000	rw1c	Line x status bit 0: No interrupt occurred. 1: Interrupt occurred. Note: This bit can be cleared by writing “1” to itself.

9 DMA controller (DMA)

9.1 Introduction

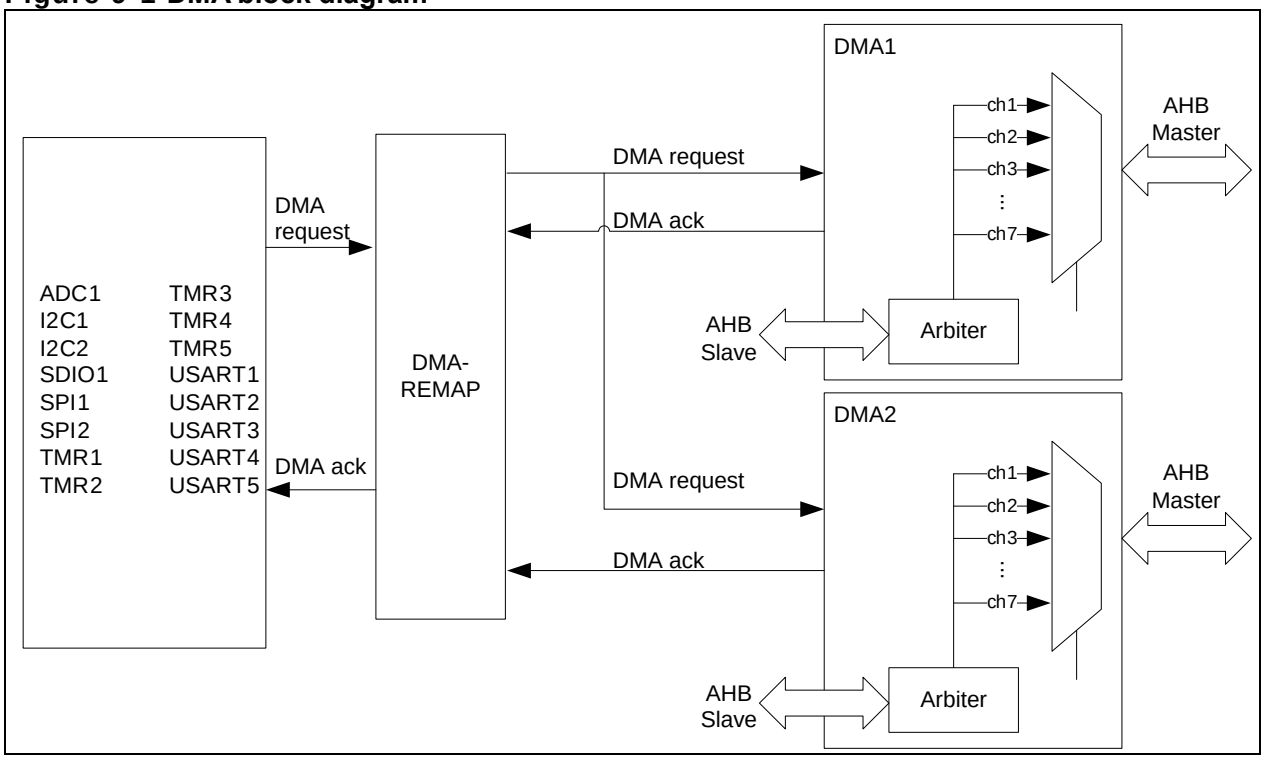
Direct memory access (DMA) controller is designed for 32-bit MCU applications with the aim of enhancing system performance and reducing the generation of interrupts.

Two DMA controllers are available in the microcontroller. Each controller contains 7 DMA channels. Each channel manages memory access requests from one or more peripherals. An arbiter is available for coordinating the priority of each DMA request.

9.2 Main features

- AMBA compliant (Rev. 2.0)
- Only support AHB OKAY and ERROR responses
- HBUSREQ and HGRANT of AHB master interface are not supported
- Support 7 channels
- Peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers
- Support hardware handshake
- Support 8-bit, 16-bit and 32- bit data transfers
- Programmable amount of data to be transferred: up to 65535
- Support flexible mapping

Figure 9-1 DMA block diagram



Note: The number of DMA peripherals in Figure 9-1 may decrease depending on different models.

9.3 Function overview

9.3.1 DMA configuration

1. **Set the peripheral address in the DMA_CxPADDR register**
   The initial peripheral address for data transfer remains unchanged during transmission.
2. **Set the memory address in the DMA_CxMADDR register**
   The initial memory address for data transfer remains unchanged during transmission.
3. **Configure the amount of data to be transferred in the DMA_CxDTCNT register**
   Programmable data transfer size is up to 65535. This value is decremented after each data transfer.
4. **Configure the channel setting in the DMA_CxCTRL register**
   Including channel priority, data transfer direction/width, address incremented mode, circular mode and interrupt mode

- **Channel priority (CHPL)**
  There are four levels, including very high priority, high priority, medium priority and low priority. If the two channels have the same priority level, then the channel with lower number will get priority over the one with higher number. For example, channel 1 has priority over channel 2.
- **Data transfer direction (DTD)**
  Memory-to-peripheral (M2P), peripheral-to-memory (P2M)
- **Address incremented mode (PINCM/MINCM)**
  In incremented mode, the subsequent transfer address is the previous address plus transfer width (PWIDTH/MWIDTH).
- **Circular mode (LM)**
  In circular mode, the contents in the DMA_CxDTCNT register is automatically reloaded with the initially programmed value after the completion of the last transfer.
- **Memory-to-memory mode (M2M)**
  This mode indicates that DMA channels perform data transfer without requests from peripherals. Circular mode and memory-to-memory mode cannot be used at the same time.

5. **Enable DMA transfer by setting the CHEN bit in the DMA_CxCTRL register**

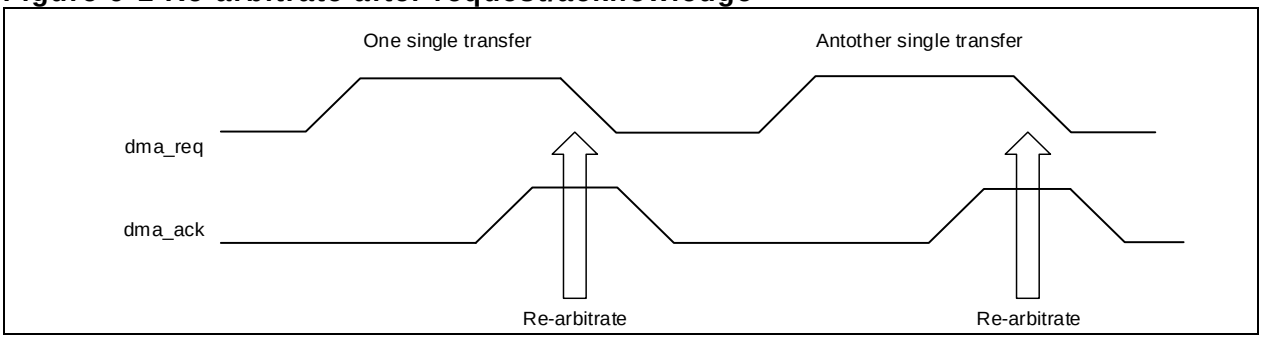
9.3.2 Handshake mechanism

In P2M and M2P mode, the peripherals need to send a request signal to the DMA controller. The DMA channel will send the peripheral transfer request (single) until the signal is acknowledged. After the completion of peripheral transmission, the DMA controller sends the acknowledge signal to the peripheral. The peripheral then releases its request as soon as it receives the acknowledge signal. At the same time, the DMA controller releases the acknowledge signal as well.

9.3.3 Arbiter

When several channels are enabled simultaneously, the arbiter will restart arbitration after full data transfer by the master controller. The channel with very high priority waits until the channel of the master controller has completed data transfers before taking control of it. The master controller will re-arbitrate to serve other channels as long as the channel completes a single transfer based on the master controller priority.

Figure 9-2 Re-arbitrate after request/acknowledge



9.3.4 Programmable data transfer width

Transfer width of the source data and destination data is programmable through the PWIDTH and MWIDTH bits in the DMA_CxCTRL register. When PWIDTH is not equal to MWIDTH, it can be aligned according to the settings of PWIDTH/ MWIDTH.

Figure 9-3 PWIDTH: byte, MWIDTH: half-word

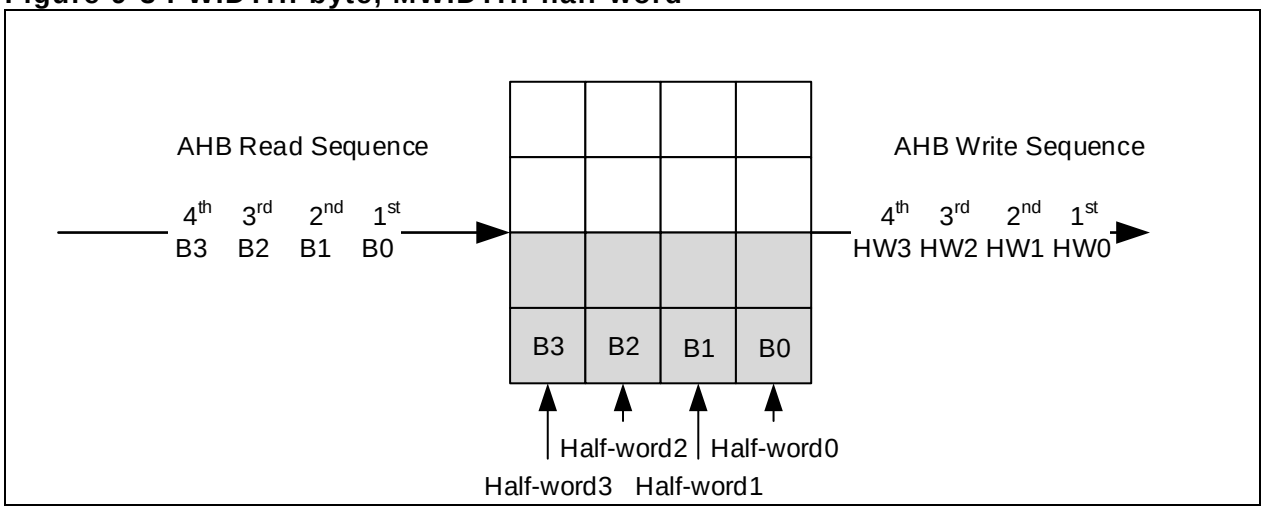


Figure 9-4 PWIDTH: half-word, MWIDTH: word

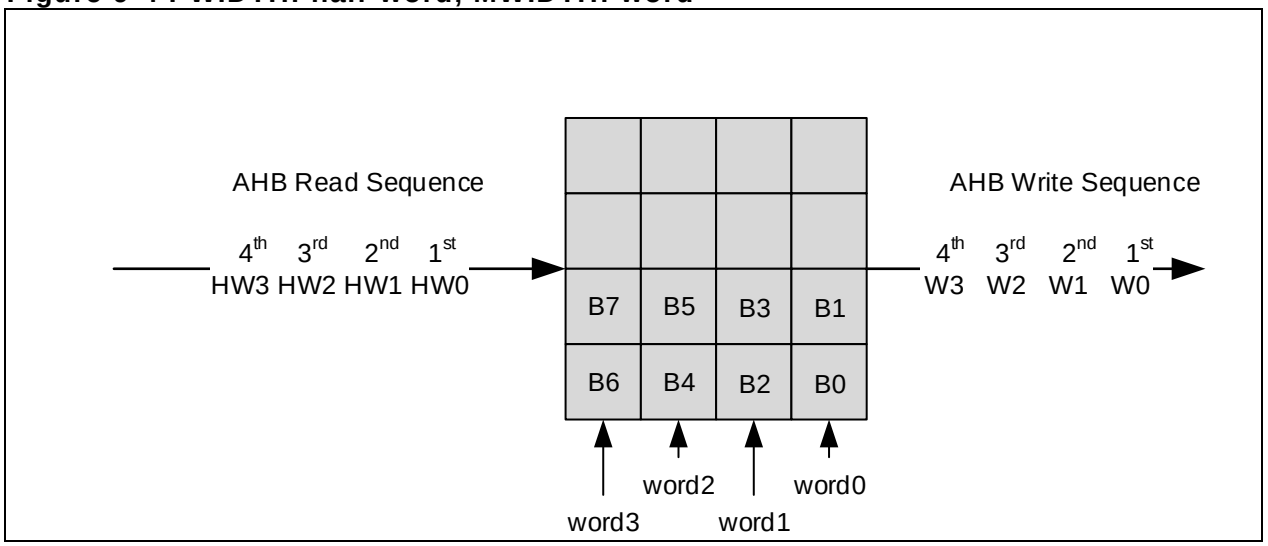
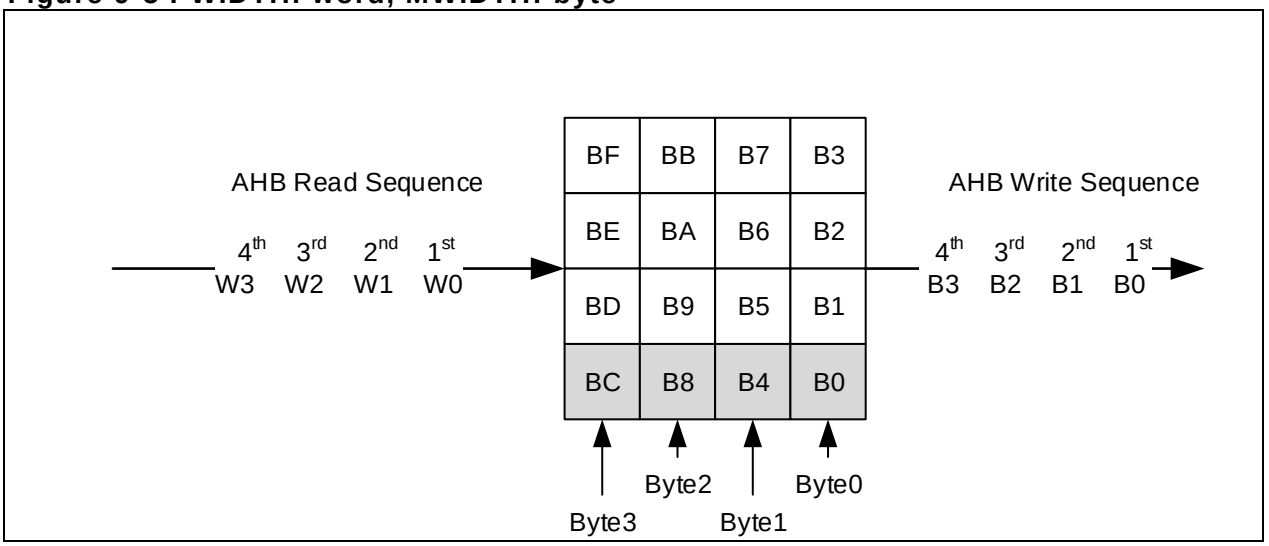


Figure 9-5 PWIDTH: word, MWIDTH: byte



9.3.5 Errors

Table 9-1 DMA error event

Error event	
Transfer error	AHB response error occurred during DMA read/write access

9.3.6 Interrupts

An interrupt can be generated on a DMA half-transfer, transfer complete and transfer error. Each channel has its specific interrupt flag, clear and enable bits, as shown in the table below.

Table 9-2 DMA interrupt requests

Interrupt event	Event flag bit	Clear control bit	Enable control bit
Half transfer	HDTF	HDTFC	HDTIEN
Transfer completed	FDTF	FDTFC	FDTIEN
Transfer error	DTERRF	DTERRFC	DTERRIEN

Note: DMA2 channel4/5, channel6/7 interrupts are mapped onto the same interrupt vector.

9.3.7 Fixed DMA request mapping

Several peripheral requests are mapped to the same DMA channel through logic ORed. This means that only one request can be enabled at a time.

The peripheral DMA requests can be independently activated/de-activated by setting the control bits in the corresponding peripheral registers.

Table 9-3 DMA1 requests for each channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC1	ADC1						
SPI/I²S		SPI1/I2S1_RX	SPI1/I2S1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX		
USART		USART3_TX	USART3_RX	USART1_TX	USART1_RX	USART2_RX	USART2_TX
I²C				I2C2_TX	I2C2_RX	I2C1_TX	I2C1_RX
TMR1		TMR1_CH1	TMR1_CH2	TMR1_CH4 TMR1_TRIG TMR1_HALL	TMR1_ OVERFLOW	TMR1_CH3	
TMR2	TMR2_CH3	TMR2_ OVERFLOW			TMR2_CH1		TMR2_CH2 TMR2_CH4
TMR3		TMR3_CH3	TMR3_CH4 TMR3_ OVERFLOW			TMR3_CH1 TMR3_TRIG	
TMR4	TMR4_CH1			TMR4_CH2	TMR4_CH3		TMR4_ OVERFLOW

Table 9-4 DMA2 requests for each channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
UART4			UART4_RX		UART4_TX		
SDIO1				SDIO1			
TMR5	TMR5_CH4 TMR5_TRIG	TMR5_CH3 TMR5_ OVERFLOW		TMR5_CH2	TMR5_CH1		

9.3.8 Flexible DMA request mapping

In flexible request mode (DMA_FLEX_EN = 1), the request source for each channel is selected through the CHx_SRC register (x=1~7). For example, to configure a DMA channel 1 as USART3_TX, and channel 3 to USART3_RX, others unused, then DMA_FLEX_EN=1, CH1_SRC=30, CH3_SRC=29, CH[2/4/5/6/7]_SRC=0 must be asserted.

Table 9-5 DMA flexible request sources

CHx_SRC	Request source	CHx_SRC	DMA source	CHx_SRC	Request source	CHx_SRC	Request source
0	No select	1	ADC1	2	reserved	3	reserved
4	reserved	5	reserved	6	reserved	7	reserved
8	reserved	9	SPI1_RX	10	SPI1_TX	11	SPI2_RX
12	SPI2_TX	13	reserved	14	reserved	15	reserved
16	reserved	17	reserved	18	reserved	19	reserved
20	reserved	21	reserved	22	reserved	23	reserved
24	reserved	25	USART1_RX	26	USART1_TX	27	USART2_RX
28	USART2_TX	29	USART3_RX	30	USART3_TX	31	UART4_RX
32	UART4_TX	33	UART5_RX	34	UART5_TX	35	reserved
36	reserved	37	reserved	38	reserved	39	reserved
40	reserved	41	I2C1_RX	42	I2C1_TX	43	I2C2_RX
44	I2C2_TX	45	reserved	46	reserved	47	reserved
48	reserved	49	SDIO1	50	reserved	51	reserved
52	reserved	53	TMR1_TRIG	54	TMR1_HALL	55	TMR1_OVERFLOW
56	TMR1_CH1	57	TMR1_CH2	58	TMR1_CH3	59	TMR1_CH4
60	reserved	61	TMR2_TRIG	62	reserved	63	TMR2_OVERFLOW
64	TMR2_CH1	65	TMR2_CH2	66	TMR2_CH3	67	TMR2_CH4
68	reserved	69	TMR3_TRIG	70	reserved	71	TMR3_OVERFLOW
72	TMR3_CH1	73	TMR3_CH2	74	TMR3_CH3	75	TMR3_CH4
76	reserved	77	TMR4_TRIG	78	reserved	79	TMR4_OVERFLOW
80	TMR4_CH1	81	TMR4_CH2	82	TMR4_CH3	83	TMR4_CH4
84	reserved	85	TMR5_TRIG	86	reserved	87	TMR5_OVERFLOW
88	TMR5_CH1	89	TMR5_CH2	90	TMR5_CH3	91	TMR5_CH4
92	reserved	93	reserved	94	reserved	95	reserved
96	reserved	97	reserved	98	reserved	99	reserved
100	reserved	101	reserved	102	reserved	103	reserved
104	reserved	105	reserved	106	reserved	107	reserved
108	reserved	109	reserved	110	reserved	111	reserved
112	reserved	113	reserved	114	reserved	115	reserved
116	reserved	117	reserved	118	reserved	119	reserved

9.4 DMA registers

Table 9-6 shows DMA register map and their reset values.
These peripheral registers must be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits).

Table 9-6 DMA register map and reset value

Register	Offset	Reset value
DMA_STS	0x00	0x0000 0000
DMA_CLR	0x04	0x0000 0000
DMA_C1CTRL	0x08	0x0000 0000
DMA_C1DTCNT	0x0C	0x0000 0000
DMA_C1PADDR	0x10	0x0000 0000
DMA_C1MADDR	0x14	0x0000 0000
DMA_C2CTRL	0x1C	0x0000 0000
DMA_C2DTCNT	0x20	0x0000 0000
DMA_C2PADDR	0x24	0x0000 0000
DMA_C2MADDR	0x28	0x0000 0000
DMA_C3CTRL	0x30	0x0000 0000
DMA_C3DTCNT	0x34	0x0000 0000
DMA_C3PADDR	0x38	0x0000 0000
DMA_C3MADDR	0x3C	0x0000 0000
DMA_C4CTRL	0x44	0x0000 0000
DMA_C4DTCNT	0x48	0x0000 0000
DMA_C4PADDR	0x4C	0x0000 0000
DMA_C4MADDR	0x50	0x0000 0000
DMA_C5CTRL	0x58	0x0000 0000
DMA_C5DTCNT	0x5C	0x0000 0000
DMA_C5PADDR	0x60	0x0000 0000
DMA_C5MADDR	0x64	0x0000 0000
DMA_C6CTRL	0x6C	0x0000 0000
DMA_C6DTCNT	0x70	0x0000 0000
DMA_C6PADDR	0x74	0x0000 0000
DMA_C6MADDR	0x78	0x0000 0000
DMA_C7CTRL	0x80	0x0000 0000
DMA_C7DTCNT	0x84	0x0000 0000
DMA_C7PADDR	0x88	0x0000 0000
DMA_C7MADDR	0x8C	0x0000 0000
DMA_SRC_SEL0	0xA0	0x0000 0000
DMA_SRC_SEL1	0xA4	0x0000 0000

Note: In the following registers, all bits related to channel 6 and channel 7 are not relevant for DMA 2 fixed request mapping since it has only 5 channels. They are applied to DMA 2 flexible request mapping, instead, supporting up to 7 channels.

9.4.1 DMA status register (DMA_STS)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	DTERRF7	0x0	ro	Channel 7 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 26	HDTF7	0x0	ro	Channel7 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 25	FDTF7	0x0	ro	Channel 7 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 24	GF7	0x0	ro	Channel7 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event occurred.
Bit 23	DTERRF6	0x0	ro	Channel 6 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 22	HDTF6	0x0	ro	Channel 6 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 21	FDTF6	0x0	ro	Channel 6 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 20	GF6	0x0	ro	Channel 6 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 19	DTERRF5	0x0	ro	Channel 5 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 18	HDTF5	0x0	ro	Channel 5 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 17	FDTF5	0x0	ro	Channel 5 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 16	GF5	0x0	ro	Channel 5 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 15	DTERRF4	0x0	ro	Channel 4 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 14	HDTF4	0x0	ro	Channel 4 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 13	FDTF4	0x0	ro	Channel 4 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 12	GF4	0x0	ro	Channel 4 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event

Bit 11	DTERRF3	0x0	ro	Channel 3 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 10	HDTF3	0x0	ro	Channel 3 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 9	FDTF3	0x0	ro	Channel 3 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 8	GF3	0x0	ro	Channel 3 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 7	DTERRF2	0x0	ro	Channel 2 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 6	HDTF2	0x0	ro	Channel 2 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 5	FDTF2	0x0	ro	Channel 2 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 4	GF2	0x0	ro	Channel 2 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event
Bit 3	DTERRF1	0x0	ro	Channel 1 data transfer error event flag 0: No transfer error occurred. 1: Transfer error occurred.
Bit 2	HDTF1	0x0	ro	Channel 1 half transfer event flag 0: No half-transfer event occurred. 1: Half-transfer event occurred.
Bit 1	FDTF1	0x0	ro	Channel 1 transfer complete event flag 0: No transfer complete event occurred. 1: Transfer complete event occurred.
Bit 0	GF1	0x0	ro	Channel 1 global event flag 0: No transfer error, half transfer or transfer complete event occurred. 1: Transfer error, half transfer or transfer complete event

9.4.2 DMA flag clear register (DMA_CLR)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
31: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	DTERRFC7	0x0	rw1c	Channel 7 data transfer error flag clear 0: No effect 1: Clear the DTERRF flag in the DMA_STS register
Bit 26	HDTFC7	0x0	rw1c	Channel 7 half transfer flag clear 0: No effect 1: Clear the HDTF7 flag in the DMA_STS register
Bit 25	FDTFC7	0x0	rw1c	Channel 7 transfer complete flag clear 0: No effect 1: Clear the FDTF7 flag in the DMA_STS register

Bit 24	GFC7	0x0	rw1c	Channel 7 global interrupt flag clear 0: No effect 1: Clear the DTERRF7, HDTF7, FDTF7 and GF7 flag in the DMA_STS register
Bit 23	DTERRFC6	0x0	rw1c	Channel 6 data transfer error flag clear 0: No effect 1: Clear the DTERRF6 flag in the DMA_STS register
Bit 22	HDTFC6	0x0	rw1c	Channel 6 half transfer flag clear 0: No effect 1: Clear the HDTF6 flag in the DMA_STS register
Bit 21	FDTFC6	0x0	rw1c	Channel 6 transfer complete flag clear 0: No effect 1: Clear the FDTF6 flag in the DMA_STS register
Bit 20	GFC6	0x0	rw1c	Channel 6 global interrupt flag clear 0: No effect 1: Clear the DTERRF6, HDTF6, FDTF6 and GF6 flag in the DMA_STS register
Bit 19	DTERRFC5	0x0	rw1c	Channel 5 data transfer error flag clear 0: No effect 1: Clear the DTERRF5 flag in the DMA_STS register
Bit 18	HDTFC5	0x0	rw1c	Channel 5 half transfer flag clear 0: No effect 1: Clear the HDTF5 flag in the DMA_STS register
Bit 17	FDTFC5	0x0	rw1c	Channel 5 transfer complete flag clear 0: No effect 1: Clear the FDTF5 flag in the DMA_STS register
Bit 16	GFC5	0x0	rw1c	Channel 5 global interrupt flag clear 0: No effect 1: Clear the DTERRF5, HDTF5, FDTF5 and GF5 in the DMA_STS register
Bit 15	DTERRFC4	0x0	rw1c	Channel 4 data transfer error flag clear 0: No effect 1: Clear the DTERRF4 flag in the DMA_STS register
Bit 14	HDTFC4	0x0	rw1c	Channel 4 half transfer flag clear 0: No effect 1: Clear the HDTF4 flag in the DMA_STS register
Bit 13	FDTFC4	0x0	rw1c	Channel 4 transfer complete flag clear 0: No effect 1: Clear the FDTF4 flag in the DMA_STS register
Bit 12	GFC4	0x0	rw1c	Channel 4 global interrupt flag clear 0: No effect 1: Clear the DTERRF4, HDTF4, FDTF4 and GF4 flag in the DMA_STS register
Bit 11	DTERRFC3	0x0	rw1c	Channel 7 data transfer error flag clear 0: No effect 1: Clear the DTERRF7 flag in the DMA_STS register
Bit 10	HDTFC3	0x0	rw1c	Channel 7 half transfer flag clear 0: No effect 1: Clear the HDTF7 flag in the DMA_STS register
Bit 9	FDTFC3	0x0	rw1c	Channel 3 transfer complete flag clear 0: No effect 1: Clear the FDTF3 flag in the DMA_STS register
Bit 8	GFC3	0x0	rw1c	Channel 3 global interrupt flag clear 0: No effect 1: Clear the DTERRF3, HDTF3, FDTF3 and GF3 flag in the DMA_STS register

Bit 7	DTERRFC2	0x0	rw1c	Channel 2 data transfer error flag clear 0: No effect 1: Clear the DTERRF2 flag in the DMA_STS register
Bit 6	HDTFC2	0x0	rw1c	Channel 2 half transfer flag clear 0: No effect 1: Clear the HDTF2 flag in the DMA_STS register
Bit 5	FDTFC2	0x0	rw1c	Channel 2 transfer complete flag clear 0: No effect 1: Clear the FDTF2 flag in the DMA_STS register
Bit 4	GFC2	0x0	rw1c	Channel 2 global interrupt flag clear 0: No effect 1: Clear the DTERRF2, HDTF2, FDTF2 and GF2 in the DMA_STS register
Bit 3	DTERRFC1	0x0	rw1c	Channel 1 data transfer error flag clear 0: No effect 1: Clear the DTERRF1 flag in the DMA_STS register
Bit 2	HDTFC1	0x0	rw1c	Channel 1 half transfer flag clear 0: No effect 1: Clear the HDTF1 flag in the DMA_STS register
Bit 1	FDTFC1	0x0	rw1c	Channel 1 transfer complete flag clear 0: No effect 1: Clear the FDTF1 flag in the DMA_STS register
Bit 0	GFC1	0x0	rw1c	Channel 1 global interrupt flag clear 0: No effect 1: Clear the DTERRF1, HDTF1, FDTF1 and GF1 in the DMA_STS register

9.4.3 DMA channel-x configuration register (DMA_CxCTRL) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Kept at its default value.
Bit 14	M2M	0x0	rw	Memory to memory mode 0: Disabled 1: Enabled.
Bit 13: 12	CHPL	0x0	rw	Channel priority level 00: Low 01: Medium 10: High 11: Very high
Bit 11: 10	MWIDTH	0x0	rw	Memory data bit width 00: 8 bits 01: 16 bits 10: 32 bits 11: Reserved
Bit 9: 8	PWIDTH	0x0	rw	Peripheral data bit width 00: 8 bits 01: 16 bits 10: 32 bits 1: Reserved
Bit 7	MINCM	0x0	rw	Memory address increment mode 0: Disabled 1: Enabled.
Bit 6	PINCM	0x0	rw	Peripheral address increment mode 0: Disabled 1: Enabled.

Bit 5	LM	0x0	rw	Circular mode 0: Disabled 1: Enabled.
Bit 4	DTD	0x0	rw	Data transfer direction 0: Read from peripherals 1: Read from memory
Bit 3	DTERRIEN	0x0	rw	Data transfer error interrupt enable 0: Disabled 1: Enabled.
Bit 2	HDTIEN	0x0	rw	Half-transfer interrupt enable 0: Disabled 1: Enabled.
Bit 1	FDTIEN	0x0	rw	Transfer complete interrupt enable 0: Disabled 1: Enabled.
Bit 0	CHEN	0x0	rw	Channel enable 0: Disabled 1: Enabled.

9.4.4 DMA channel-x number of data register (DMA_CxDTCNT) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CNT	0x0000	rw	Number of data to transfer The number of data to transfer is from 0x0 to 0xFFFF. This register can only be written when the CHEN bit in the corresponding channel is set 0. The value is decremented after each DMA transfer. Note: This register holds the number of data to transfer, instead of transfer size. The transfer size is calculated by data width.

9.4.5 DMA channel-x peripheral address register (DMA_CxPADDR) (x = 1…7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PADDR	0x0000 0000	rw	Peripheral base address Base address of peripheral data register is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

9.4.6 DMA channel-x memory address register (DMA_CxMADDR) (x = 1···7)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 0	MADDR	0x0000 0000	rw	Memory base address Memory address is the source or destination of data transfer. Note: The register can only be written when the CHEN bit in the corresponding channel is set 0.

9.4.7 DMA channel source register (DMA_SRC_SEL0)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	CH4_SRC	0x00	rw	CH4 source select When DMA_FLEX_EN=1, channel 4 is selected by the CH4_SRC. Refer to *Section 9.3.8* for more information.
Bit 23: 16	CH3_SRC	0x00	rw	CH3 source select When DMA_FLEX_EN=1, channel 3 is selected by the CH3_SRC. Refer to *Section 9.3.8* for more information.
Bit 15: 8	CH2_SRC	0x00	rw	CH2 source select When DMA_FLEX_EN=1, channel 2 is selected by the CH2_SRC. Refer to *Section 9.3.8* for more information.
Bit 7: 0	CH1_SRC	0x00	rw	CH1 source select When DMA_FLEX_EN=1, channel 1 is selected by the CH1_SRC. Refer to *Section 9.3.8* for more information.

9.4.8 DMA channel source register1 (DMA_SRC_SEL1)

Access: 0 wait state, accessible by bytes, half-words or words.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24	DMA_FLEX_EN	0x00	rw	DMA flexible mapping mode selection 0: Fixed mapping mode 1: Flexible mapping mode
Bit 23: 16	CH7_SRC	0x00	rw	CH7 source select When DMA_FLEX_EN=1, channel 7 is selected by the CH7_SRC. Refer to *Section 9.3.8* for more information.
Bit 15: 8	CH6_SRC	0x00	rw	CH6 source select When DMA_FLEX_EN=1, channel 6 is selected by the CH6_SRC. Refer to *Section 9.3.8* for more information.
Bit 7: 0	CH5_SRC	0x00	rw	CH5 source select When DMA_FLEX_EN=1, channel 5 is selected by the CH5_SRC. Refer to *Section 9.3.8* for more information.

10 CRC calculation unit (CRC)

10.1 CRC introduction

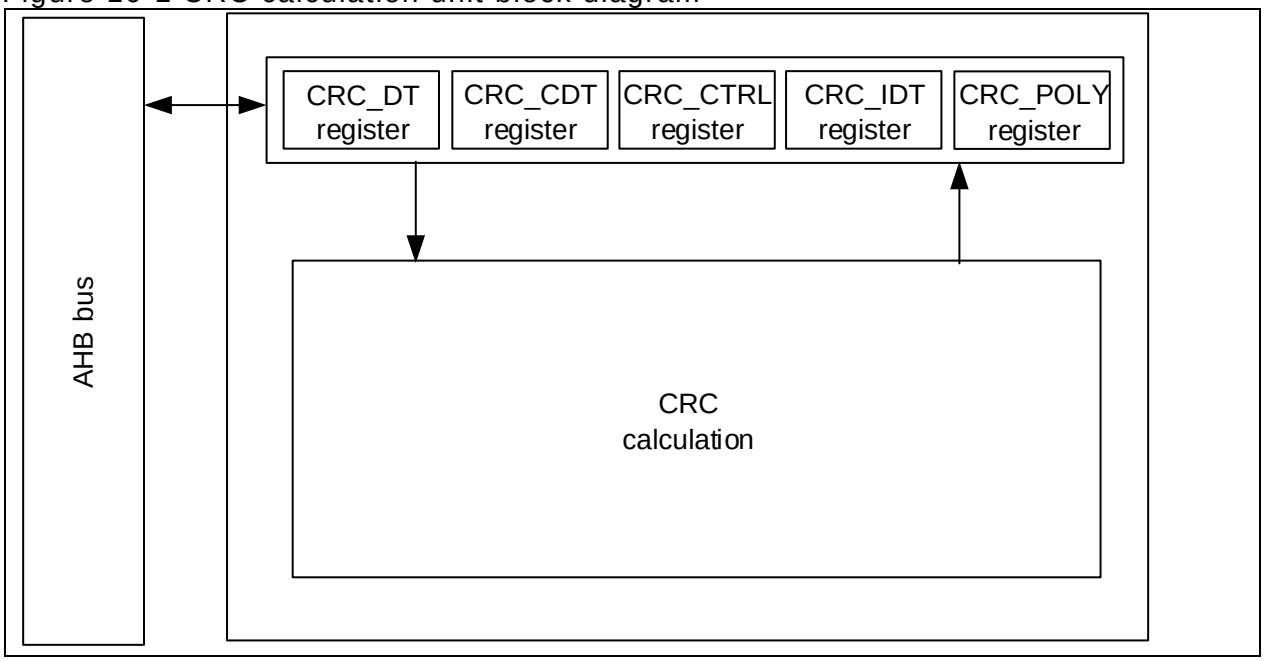
The Cyclic Redundancy Check (CRC) is an independent peripheral with CRC check feature. It follows CRC32/MPEG-2 standard.

The CRC_CTRL register is used to select output data reverse (word, REVOD=1) or input data reverse (byte, REVID=01; half-word, REVID=10; word: REVID=11). CRC calculation unit is also equipped with initialization function. After each reset, the value in the CRC_IDT register is written into the data register (CRC_DT) by CRC.

The CRC_POLY register is used to set different polynomial coefficient. The polynomial size can be set as 7 bits, 8 bits, 16 bits or 32 bits through the POLY-SIZE bit in the CRC_CTR register.

Users can write the data to go through CRC check and read the calculated result through CRC_DT register. Note that the calculation result is the combination of the previous result and the current value to be calculated.

Figure 10-1 CRC calculation unit block diagram



Main features

- Use CRC-32 code
- Support the generation of polynomial
- 4 HCLK cycles for each CRC calculation
- Support input/output data format toggle
- Perform write/read operation through CRC_DT register
- Set an initialization value with the CRC_IDT register. The value is loaded with CRC_DT register after each CRC reset.

10.2 CRC functional description

According to CRC calculation principle: the input data is taken as dividend, and the generator polynominal as a division. Using mod 2 division logic, the input data divided by the generator polynominal gets a remainder, that is, the CRC value.

CRC calculation procedure

- Input data reverse. After data input, reverse input data depending on the REVID value in CRC_CTRL register

- Initialization. The first data input needs to be XOR-ed with the initial value defined in the CRC_IDT register. If it is not the first data input, the initial value is the previously calculated result.
- CRC calculation. Dividing the input data by the generator polynominal (0x4C11DB7) using mod 2 division method produces a remainder, that is, CRC value.
- Output data toggle. Select whether to perform word toggle before output CRC value through the REVOD bit in the CRC_CTRL register
- XOR calculation. The XOR-ed result is fixed at 0x0000 0000

CRC-32/MPEG-2 parameters

- Generator polynominal: 0x4C11DB7

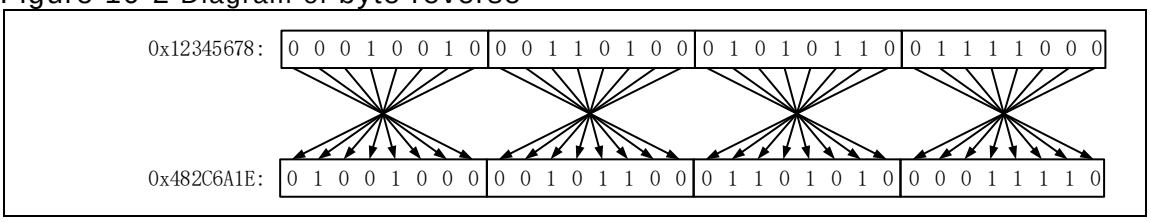
that is, $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$

- Initial value: 0xFFFF FFFF, in order to avoid that 1-byte 0x00 data to be calculated has the same result as that of multiple-byte 0x00.
- XOR-ed value: 0x0000 0000, indicating that the CRC result will not be XOR-ed.

Toggle function

- Byte reverse, 8 bits in a group, and sequence is reversed within a group. As shown in figure below, if the original data is 0x12345678, it is reversed as 0x482C6A1E.
- Half-word reverse, 16 bits in a group, and sequence is reversed within a group
- Word reverse, 32 bits in a group, and sequence is reversed within a group

Figure 10-2 Diagram of byte reverse



10.3 CRC registers

CRC_DT register can be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits). Other registers have to be accessed by words (32 bits).

Table 10-1 CRC register map and reset value

Register	Offset	Reset value
CRC_DT	0x00	0xFFFF FFFF
CRC_CDT	0x04	0x0000 0000
CRC_CTRL	0x08	0x0000 0000
CRC_IDT	0x10	0xFFFF FFFF
CRC_POLY	0x14	0x04C1 1DB7

10.3.1 Data register (CRC_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0xFFFF FFFF	rw	Data value Used as input register when writing new data into the CRC calculator. It returns CRC calculation results when it is read.

10.3.2 Common data register (CRC_CDT)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7: 0	CDT	0x00	rw	Common 8-bit data value This field is used to store one byte data temporarily. This register is not affected by the CRC reset generated by the RST bit in the CRC_CTRL register.

10.3.3 Control register (CRC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	REVOD	0x0	resd	Reverse output data Set and cleared by software. This bit is used to control whether or not to reverse output data. 0: No effect 1: Word reverse
Bit 6: 5	REVID	0x0	rw	Reverse input data Set and cleared by software. This bit is used to control how to reverse input data. 00: No effect 01: Byte reverse 10: Half-word reverse 11: Word reverse
Bit 4: 3	POLY_SIZE	0x0	rw	Polynomial size This field is used to set the size of polynomial. It is used in conjunction with the CRC_POLY register. 00: 32 bits 01: 16 bits 10: 8 bits 11: 7 bits
Bit 2: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	RST	0x0	wo	Reset CRC calculation unit Set by software. Cleared by hardware. To reset CRC calculation unit, the data register is set as 0xFFFF FFFF. 0: No effect 1: Reset

10.3.4 Initialization register (CRC_IDT)

Bit	Register	Reset value	Type	Description
Bit 31: 0	IDT	0xFFFF FFFF	rw	Initialization data register When CRC reset is triggered by the RST bit in the CRC_CTRL register, the value in the initialization register is written into the CRC_DT register as an initial value.

10.3.5 Polynomial register (CRC_POLY)

Bit	Register	Reset value	Type	Description
Bit 31: 0	POLY	0x04C1 1DB7	rw	Polynomial coefficient The generated polynomial is a divisor in CRC calculation. Using CRC32 mode, this polynomial coefficient is 0x4C11DB7. Users can also set the polynomial coefficient according to their needs.

11 I²C interface

11.1 I²C introduction

I²C (inter-integrated circuit) bus interface manages the communication between the microcontroller and serial I²C bus. It supports master and slave modes, with up to 400 Kbit/s of communication speed.

11.2 I²C main features

- I2C bus
  - Master and slave modes
  - Multimaster capability
  - Stand speed (100 kHz), fast speed (400 kHz)
  - 7-bit and 10-bit address modes
  - Broadcast call mode
  - Status flag
  - Error flag
  - Clock stretching capability
  - Communication event interrupts
  - Error interrupts
- Support DMA transfer
- Programmable digital noise filter
- Support SMBus2.protocol
  - PEC generation and verification
  - SMBus reminder capability
  - ARP(address resolution protocol)
  - Timeout detection
- PMBus

Note: I2S frequency can be up to 1 MHz. For details on this, please contact your local or nearest ARTERY sales office for further support.

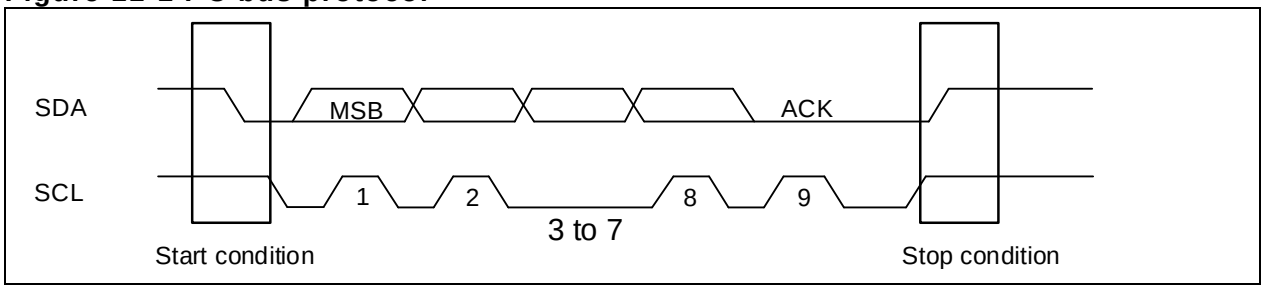
11.3 I²C function overview

I²C bus consists of a data line (SDA) and clock line (SCL). It can achieve a maximum of 100 kHz speed in standard mode, whereas up to 400kHz in fast mode. A frame of data transmission begins with a Start condition and ends with a Stop condition. The bus is kept in busy state after receiving a Start condition, and becomes idle as long as it receives a Stop condition.

Start condition: SDA switches from high to low when SCL is set high.

Stop condition: SDA switches from low to high when SCL is set high.

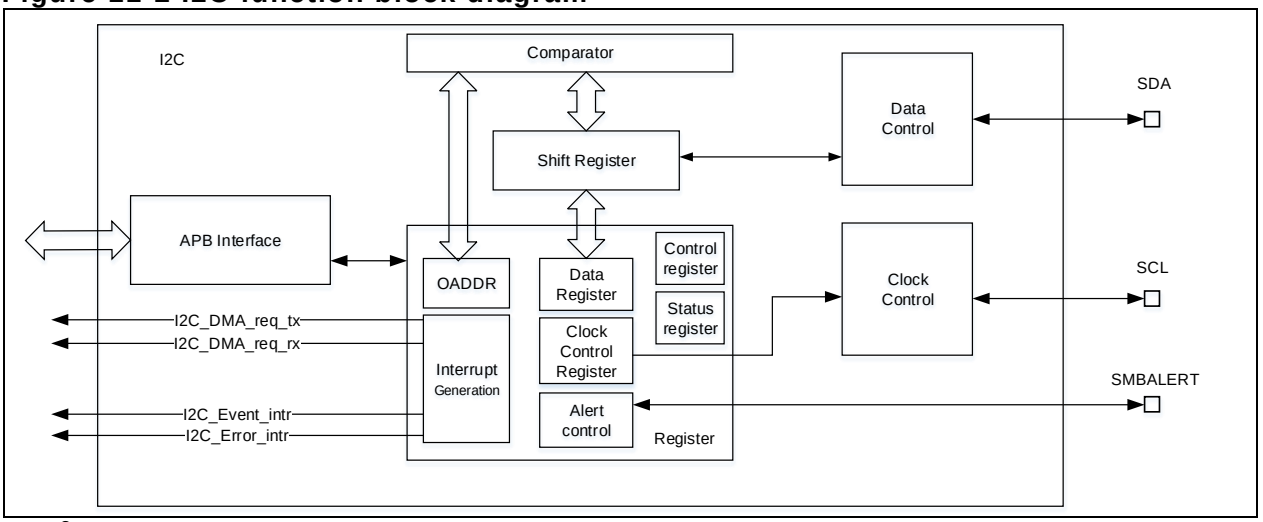
Figure 11-1 I²C bus protocol



11.4 I²C interface

Figure 11-2 shows the block diagram of I²C function

Figure 11-2 I2C function block diagram



1. I²C clock

I²C is clocked by either APB1 or APB2. The I²C clock division is achieved by setting the CLKFREQ[7:0] in the I2C_CTRL2 register. The minimum clock frequency varies from one mode to another, that is, at least 2 MHz in standard mode, but 4 MHz in fast mode.

2. Operating mode

I²C bus interface can operate both in master mode and slave mode. Switching from master mode to slave mode, vice versa, is supported as well. By default, the interface operates in slave mode. When GENSTART=1 is set (Start condition is activated), the I²C bus interface switches from slave mode to master mode, and returns to slave mode automatically at the end of data transfer (Stop condition is triggered).

- Master transmitter
- Master receiver
- Slave transmitter
- Slave receiver

3. Communication process

- Master mode communication:
  1. Start condition generation
  2. Address transmission
  3. Data Tx or Rx
  4. Stop condition generation
  5. End of communication
- Slave mode communication:
  1. Wait until the address is matched.
  2. Data Tx or Rx
  3. Wait for the generation of Stop condition
  4. End of communication

4. Address control

Both master and slave support 7-bit and 10-bit addressing modes.

Slave address mode:

- In 7-bit mode
  - ADDR2EN=0 stands for a single address mode: only matches OADDR1
  - DUALEN=1stands for dual address mode: matches OADDR1 and OADDR2

- In 10-bit mode
  - Only matches OADDR1

Support special slave address:

- Broadcast call address (0b0000000x): This address is enabled when GCAEN=1.
- SMBus device default address (0b1100001x): This address is enabled for SMBus address resolution protocol in SMBus device mode.
- SMBus master default address (0b0001000x): This address is enabled for SMBus master notification protocol in SMBus master mode.
- SMBus alert address (0b0001100x): This address is enabled for SMBus alert response address protocol in SMBus master mode when SMBALERT = 1

Refer to SMBus2.0 protocol for more information.

Slave address matching procedure:

- Receive a Start condition
- Address matching
- The slave sends an ACK if address is matched.
- ADDR7F is set, with DIRF indicating the transmission direction
  - When DIRF =0, slave enters receiver mode, starting receiving data.
  - When DIRF =1, slave enters transmitter mode, starting transmitting data

5. Clock stretching capability

Clock stretching is enabled by setting the STRETCH bit in the I2C_CTRL1 register. Once enabled, when the slave cannot process data in a timely manner on certain conditions, it will pull down SCL line to low level to stop communication in order to prevent data loss.

- Transmitter mode:
  - Clock stretching enable: If no data is written to the I2C_DT register before the next byte transmission (the first SCL rising edge of the next data), the I^2C interface will pull down SCL bus and wait until the data is written to the I2C_DT
  - Clock stretching disable: if no data is written to the I2C_DT register before the next byte transmission (the first SCL rising edge of next data), an underrun error will happen.
- Receiver mode
  - Clock stretching enable: When the shift register has received another byte before the data in the I2C_DT register is read, the I^2C will hold the SCL bus low to wait for the software to read I2C_DT register
  - Clock stretching disable: The data in the I2C_DT register is not yet read when the shift register receives another byte. In this case, if another data is received, an overrun error occurs.

11.4.1 I²C slave communication flow

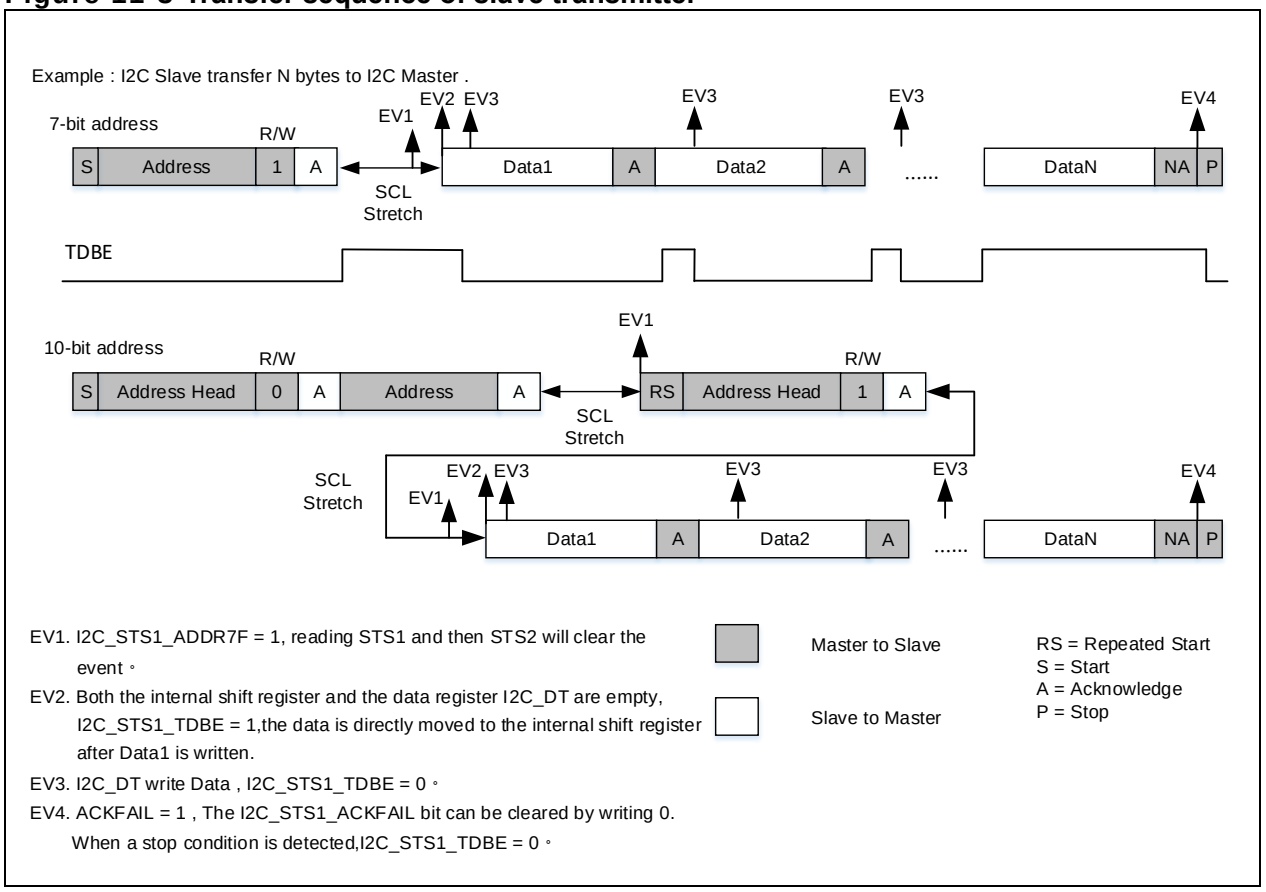
Initialization

Enable I²C peripheral clock, and configure the clock-related bits in the I2C_CTRL2 register for a correct timing, and then wait for I²C master to send a Start condition.

Transmitter

Figure 11-3 shows the transfer sequence of slave transmitter.

Figure 11-3 Transfer sequence of slave transmitter



7-bit address mode:

1. Wait for the master to send addresses
2. EV1: Address is matched (ADDR7F=1), and then the slave pulls the SCL bus low. Read STS1 and then STS2 by software will clear the ADDR7F bit. At this point, it enters transmission stage, in which both DT register and internal shift register are empty. The TDBE bit is set 1 by hardware.
3. EV2: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
4. EV3: At this point, the DT register is empty, but the shift register is not. Writing to the DT register will clear the TDBE bit.
5. EV4: After receiving the ACKFAIL event from the master, the ACKFIAL=1. Writing 0 to the ACKFIAL will clear the event.
6. End of communication.

10-bit address mode:

1. Wait for the master to send address
2. EV1: Address is matched (ADDR7F=1), and then the slave pulls the SCL bus low. Read STS1 and then STS2 by software will clear the ADDR7F bit. Wait for the master to re-send Start condition.
3. EV1: Address is matched (ADDR7F=1). Read STS1 and then STS2 will re-clear the ADDR7F bit. At this point, it enters transmission stage. Both DT register and shift register are empty. The TDBE

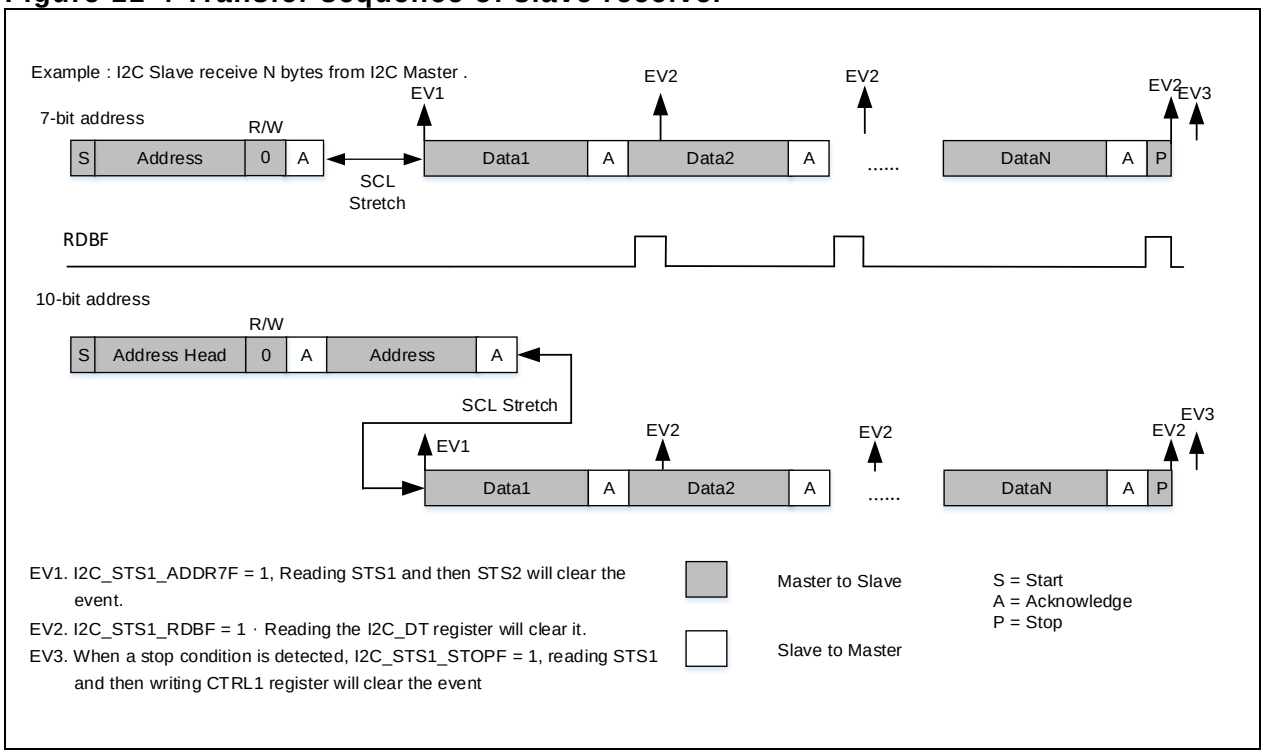
is set 1 by hardware.

4. EV2: When the data is written to DT register, it is directly moved to the shift register, and SCL bus is released. The TDBE is still set 1 at this time.
5. EV3: At this point, the DT register is empty but the shift register is not. Writing to the DT register will clear the TDBE bit.
6. EV4: After receiving the ACKFAIL event from the master, ACKFIAL=1. Writing 0 to the ACKFIAL bit will clear the event.
7. End of communication.

Slave receiver

Figure 11-4 shows the transfer sequence of slave receiver.

Figure 11-4 Transfer sequence of slave receiver



7-bit address mode:

1. Wait for the master to send address.
2. EV1: Address is matched (ADDR7F=1), and the slave pulls the SCL bus low. Read STS1 and then STS2 by software will clear the ADDR7F bit. At this point, the SCL bus is released, and enters receive stage.
3. The internal shift register receives the bus data and stores it to DT register.
4. EV2: After receiving the byte, the RDBF bit is set 1. Read the I2C_DT register will clear the RDBF bit.
5. EV3: After receiving the Stop condition from the master, STOPF=1. Read STS1 and then write to CTRL1 register will clear the event.
6. End of communication.

10-bit address mode:

1. Wait for the master to send address.
2. EV1: Address is matched (ADDR7F=1). The slave pulls the SCL bus low. Read STS1 and then STS2 by software will clear the ADDR7F bit. At this point, the SCL bus is released, and enters receive stage.
3. The internal shift register receives the bus data and stores it to DT register.
4. EV2: After receiving the byte, the RDBF bit is set 1. Read the I2C_DT register will clear the

RDBF bit.

5. EV3: After receiving the Stop condition from the master, STOPF=1. Read STS1 and then write to CTRL1 register will clear the event.
6. End of communication.

11.4.2 I²C master communication flow

Master mode Initialization

1. Program input clock to generate correct timing through the CLKFREQ bit in the I2C_CTRL2 register;
2. Program I²C communication speed through the I2C_CLKCTRL bit in the clock control register;
3. Program the maximum rising time of bus through the I2C_TMRISE register;
4. Program the control register1 I2C_CTRL1;
5. Enable peripherals, if the GENSTART bit is set, a Start condition is generated on the bus, and the device enters master mode.

Slave address transmission

Slave address is divided into 7-bit and 10-bit modes. Whether it is transmitter mode or receiver mode depends on the lowest address bit.

- 7-bit address mode:

  Transmitter: When the lowest bit of the address sent is 0, the master enters transmitter mode.

  Receiver: When the lowest bit of the address sent is 1, the master enters receiver mode.

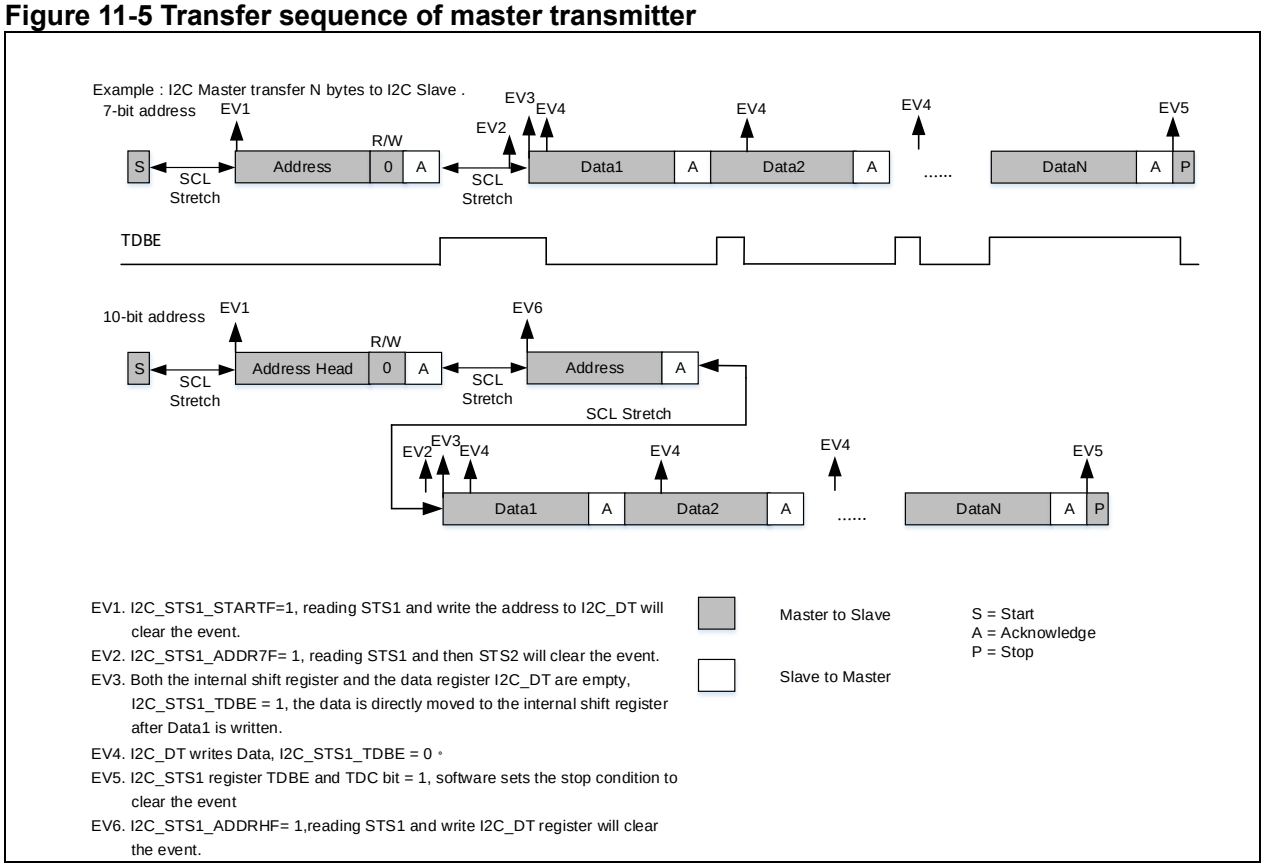
- 10-bit address mode:

  Transmitter: First send address head 0b11110xx0 (where xx refers to address [9: 8]), and then slave address [7: 0], the master enters transmitter mode.

  Receiver: First send slave address head 0b11110xx0 (where xx refers to address [9:8]) and then address [7: 0], followed by the address head 0b11110xx1 (where xx refers to address [9: 8]), the master enters receiver mode.

Master transmitter

Figure 11-5 Transfer sequence of master transmitter



- **7-bit address mode:**

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit. In this case, the master enters transmit stage, and both DT register and internal shift register are empty. The TDBE bit is set 1 by hardware.
4. EV3: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
5. EV4: At this point, the DT register is empty but the shift register is full. Writing to the DT register will clear the TDBE bit.
6. The TDBE bit is set only after the second-to-last byte is sent.
7. EV5: TDC=1 indicates that the byte transmission is complete. The master sends Stop condition (STOPF=1). The TDBE bit and TDC bit is cleared by hardware.
8. End of communication.

- **10-bit address mode:**

1. Generate Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV6: 10-bit address head sequence is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
4. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit. In this case, the master enters transmit stage, and both DT register and internal shift register are empty. The TDBE bit is set 1 by hardware.
5. EV3: When the data is written to the DT register, it is directly moved to the shift register and the SCL bus is released. The TDBE bit is still set 1 at this time.
6. EV4: At this point, the DT register is empty but the shift register is full. Writing to the DT register will clear the TDBE bit.
7. The TDBE bit is set only after the second-to-last byte is sent.
8. EV5: TDC=1 indicates that the byte transmission is complete. The master sends Stop condition (STOPF=1). The TDBE bit and TDC bit is cleared by hardware.
9. End of communication.

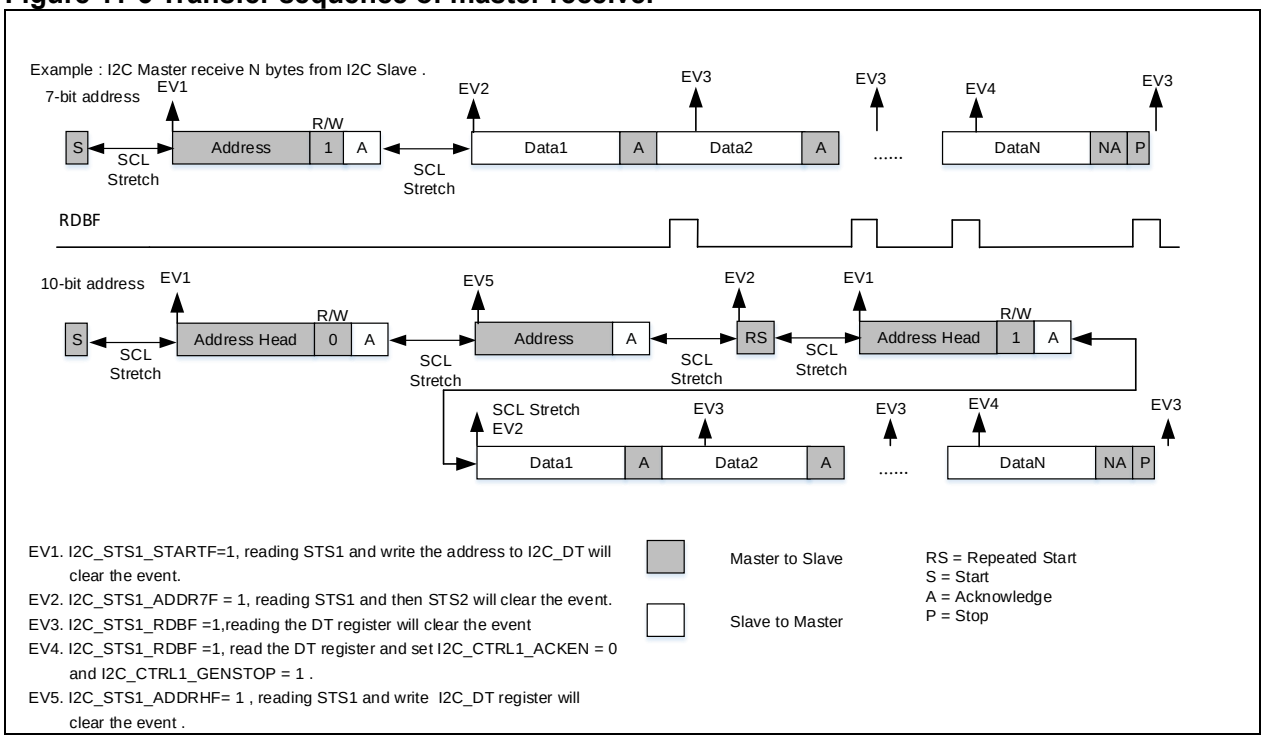
Master receiver

Data reception depends on I^2C interrupt priority:

1. **Very high priority**

- When the second-to-last byte is being read, clear the ACKEN bit and set the GENSTOP bit in the I2C_CTRL1 register to generate Stop condition.
- If only one byte is received, clear the ADDR7F flag and set the ACKEN and GENSTOP bit in the I2C_CTRL1 register.
- After the byte is received, the I2C_STS1_RDBF bit is set 1 by hardware, and it is cleared after the software read the I2C_DT register.

Figure 11-6 Transfer sequence of master receiver



- **7-bit address mode:**

1. Generate a Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit. In this case, the master enters receive stage.
4. EV3: The RDBF bit is set 1 after the byte is received. It is cleared when the I2C_DT register is read.
5. EV4: The ACKEN bit is cleared and the GENSTOP is set as soon as the second-to-last byte is received.
6. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
7. End of communication.

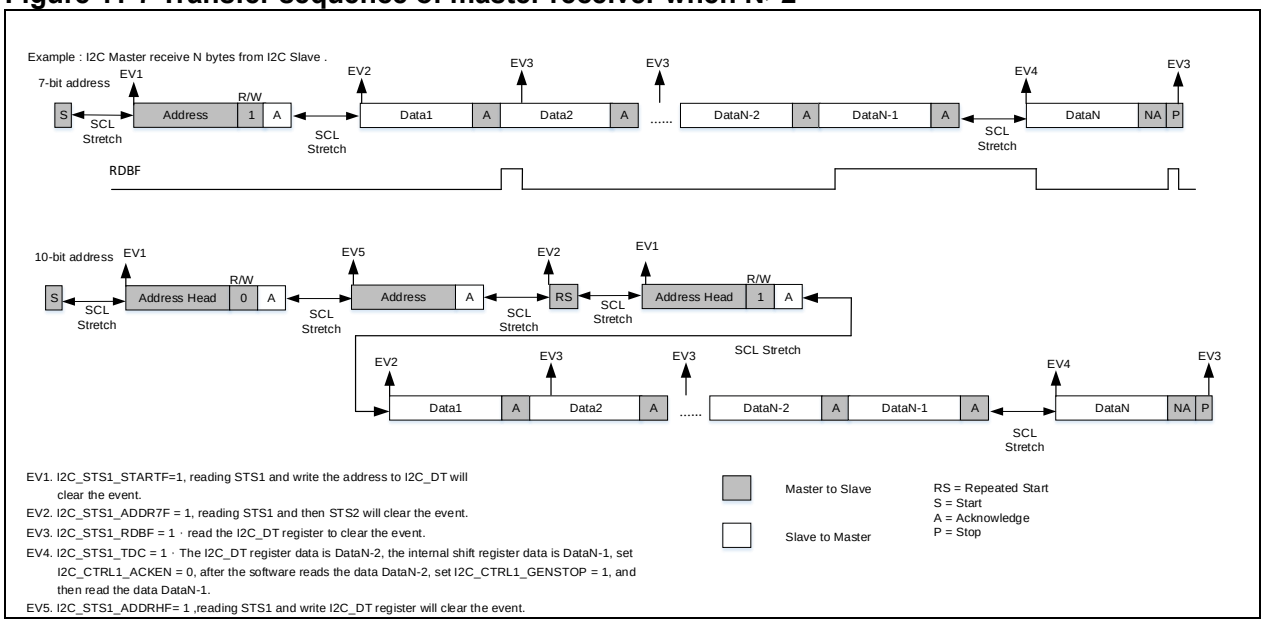
- **10-bit address mode:**

1. Generate Start condition (GENSTART=1)
2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
3. EV5: 10-bit address head sequence is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
4. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit, and the master re-send Start condition (GENSTART=1).
5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
6. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit. The master enters receive stage at this time.
7. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
8. EV4: The ACKEN bit is cleared and the GENSTOP is set as soon as the second-to-last byte is received.
9. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
10. End of communication.

2. **When I2C interrupt priority is not very high but the number of bytes to receive is greater than 2**

- The third-to-last byte (N-2) is not read when being received. It is read only after the ACKEN bit in the I2C_CTRL1 register is cleared when the second-to-last byte (N-1) is received. Then the second-to-last byte (N-1) is read after the GENSTOP bit in the I2C_CTRL1 register is set. Afterwards, the bus starts to receive the last one byte.

Figure 11-7 Transfer sequence of master receiver when N>2



- **7-bit address mode:**
  1. Generate a Start condition (GENSTART=1)
  2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  3. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit, and the master enters receive stage.
  4. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
  5. EV4: TDC=1, the contents in the I2C_DT is N-2, and that of the shift register is N-1. The ACKEN is set 0 by software and the data N-2 is read, afterwards, the GENSTOP=1, and data N-1 is read.
  6. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
  7. End of communication.
- **10-bit address mode:**
  1. Generate Start condition (GENSTART=1)
  2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  3. EV5: 10-bit address head sequence is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
  4. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit, and the master re-send Start condition.
  5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.
  6. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit, and the master enters receive stage.
  7. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
  8. EV4: TDC=1, the contents in the I2C_DT is N-2, and that of the shift register is N-1. The ACKEN is set 0 by software and the data N-2 is read, afterwards, the GENSTOP=1, and data

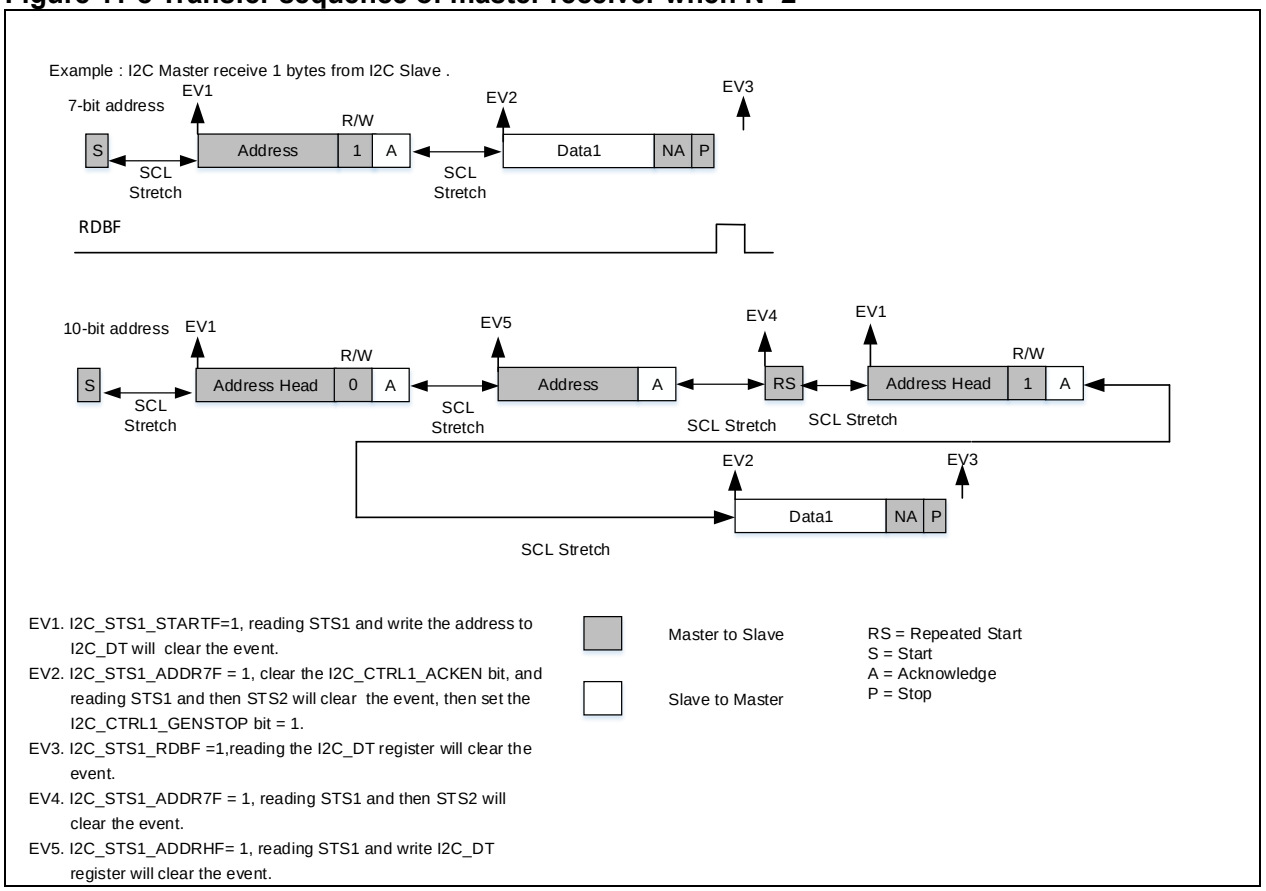
N-1 is read.

9. EV3: The RDBF bit is set 1 after receiving the byte, and it is cleared when the I2C_DT register is read.
10. End of communication.

3. **When I2C interrupt priority is not very high but the number of bytes to receive is equal to 2**

- Set the MACKCTRL bit in the I2C_CTRL1 register before data reception. When the address is matched, clear ACKEN bit and then the ADDR7F bit. When the TDC bit is set 1, set the GENSTOP bit in the I2C_CTRL1 register, and then read the DT register.

Figure 11-8 Transfer sequence of master receiver when N=2



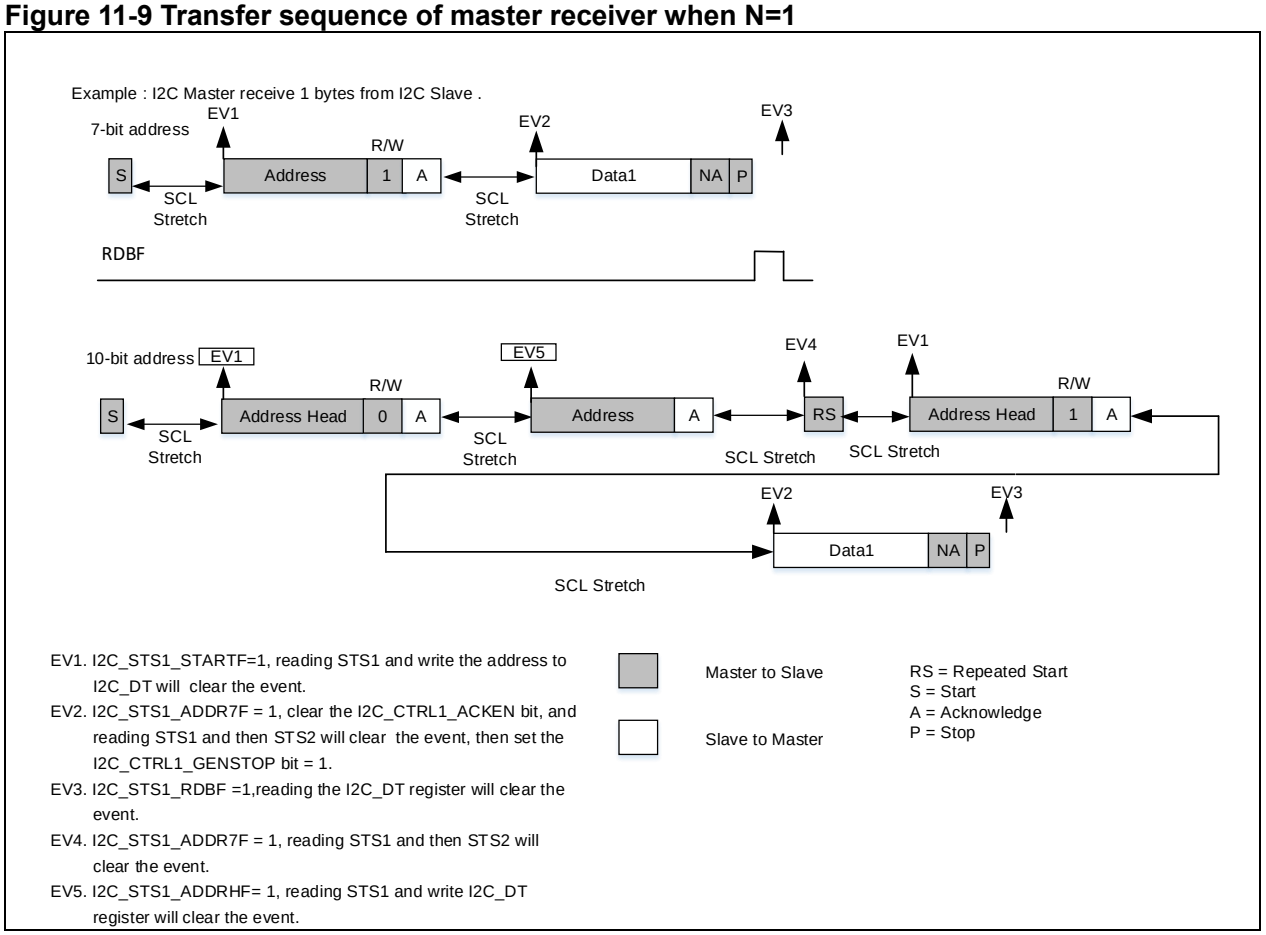
- **7-bit address mode:**
  1. Set MACKCTRL=1 in the I2C_CTRL1 register
  2. Generate Start condition (GENSTART=1)
  3. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  4. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit and read STS1 before reading STS2 and clearing ADDR7F bit, the master enters receive state at this time.
  5. EV2: TDC=1, set GENSTOP=1, and then read the I2C_DT register twice.
  6. End of communication.
- **10-bit address mode:**
  1. Set MACKCTRL=1 in the I2C_CTRL1 register
  2. Generate Start condition (GENSTART=1)
  3. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  4. EV4: 10-bit address head is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
  5. EV2: Address is matched successfully (ADDR7F=1). Read STS1 and then STS2 will clear the ADDR7F bit, and the master re-send Start condition.
  6. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.

7. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit and read STS1 before reading STS2 and clearing ADDR7F bit, the master enters receive state at this time.
8. EV3: TDC=1, set GENSTOP=1, and then read the I2C_DT register twice.
9. End of communication.

4. **When I2C interrupt priority is not very high but the number of bytes to receive is equal to 1**

- After the address is matched, clear the ACKEN bit and then ADDR7F bit, then set the GENSTOP bit in the I2C_CTRL1 register. Wait until the RDBF bit is set 1 before reading the DT register.

Figure 11-9 Transfer sequence of master receiver when N=1



- **7-bit address mode:**
  1. Generate a Start condition (GENSTART=1)
  2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  3. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit, read STS1 and then STS2 will clear the ADDR7F bit. Afterwards, set GENSTOP=1, the master enters receive stage at this time.
  4. EV3: RDBF=1. It is cleared when the I2C_DT register is read.
  5. End of communication.
- **10-bit address mode:**
  1. Generate a Start condition (GENSTART=1)
  2. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to DT register.
  3. EV5: 10-bit address head is sent. Read STS1 and write to DT register can clear the ADDRHF bit.
  4. EV4: Address is matched successfully (ADDR7F=1). Read STS1 and STS2 will clear the ADDR7F bit, the master re-sends Start condition (GENSTART=1).
  5. EV1: Start condition is ready (STARTF=1). Read STS1 and write the address to the DT register.
  6. EV2: Address is matched successfully (ADDR7F=1). Clear the ACKEN bit, read STS1 and

then STS2 will clear the ADDR7F bit. Afterwards, set GENSTOP=1, the master enters receive stage at this time.

7. EV3: RDBF=1. It is cleared when the I2C_DT register is read.
8. End of communication.

11.4.3 Data transfer using DMA

I²C data transfer can be done using DMA controller. An interrupt is generated by enabling the transfer complete interrupt bit. The DATAIEN bit in the I2C_CTRL2 register must be set 0 when using DMA for data transfer. The following sequence is for data transfer with DMA.

Transmission using DMA

1. Set the peripheral address (DMA_CxPADDR= I2C_DT address)
2. Set the memory address (DMA_CxMADDR=data memory address)
3. The transmission direction is set from memory to peripheral (DTD=1 in the DMA_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc in the DMA_CHCTRL register
6. Enable the DMA channel by setting CHEN=1 in the DMA_CxCTRL register
7. Enable I²C DMA request by setting DMAEN=1 in the I2C_CTRL2 register. Once the TDBE bit in the I2C_STS1 register is set, the data is loaded from the programmed memory to the I2C_DT register through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master transmitter: Once the TDC flag is set, the STOP condition is generated, indicating that transfer is complete.

   Slave transmitter: Once the ACKFAIL flag is set, clear the ACKFAIL flag, transfer is complete.

Reception using DMA

1. Set the peripheral address (DMA_CxPADDR = I2C_DT address)
2. Set the memory address (DMA_CxMADDR = memory address)
3. The transmission directions set from peripheral to memory (DTD=0 in the DMA_CHCTRL register)
4. Configure the total number of bytes to be transferred in the DMA_CxDTCNT register
5. Configure other parameters such as priority, memory data width, peripheral data width, interrupts, etc in the DMA_CHCTRL register
6. Enable the DMA channel by setting CHEN=1 in the DMA_CxCTRL register
7. Enable I²C DMA request by setting DMAEN=1 in the I2C_CTRL2 register. Once the RDBE bit in the I2C_STS1 register is set, the data is loaded from the I2C_DT register to the programmed memory through DMA
8. When the number of data transfers, programmed in the DMA controller, is reached (DMA_CxDTCNT=0), the data transfer is complete (An interrupt is generated if enabled).
9. Master receiver: Clear the ACKFAIL flag, the STOP condition is generated, indicating that the transfer is complete (when the number of bytes to be transferred is greater >=2 and DMAEND=1, the NACK signal is generated automatically after transfer complete (DMA_CxDTCNT=0))

   Slave receiver: Once the STOPF flag is set, clear the STOPF flag, and the transfer is complete.

11.4.4 SMBus

The System Management Bus (SMBus) is a two-wire interface through which various devices can communicate with each other. It is based on I²C. With SMBus, the device can provide manufacturer information, tell the system its model/part number, report different types of errors and accept control parameters and so on. For more information, refer to SMBus 2.0 protocol.

Differences between SMBus and I²C

1. SMBus requires a minimum speed of 10 kHz for the purpose of management and monitor. It is quite easy to know whether the bus is in Idle state or not as long as a parameter is input while running on a certain transmission speed, without the need of detecting the STOP signals one after another, or even keeping STOP and other parameter monitor. There is no limit for I²C.
2. SMBus transmission speed ranges from 10 kHz to 100 kHz. In contrast, I2C has no minimum requirement, and its maximum speed varies from one mode to another, namely, 100 kHz in standard mode and 400 kHz in fast mode.
3. After reset, SMBus needs timeout (35ms), but there is no limit for I²C in this regard.

SMBus applications

1. The I²C interface is set in SMBus mode by setting PERMODE=1 in the I2C_CTRL1 register.
2. Select SMBus mode:
   SMBMODE=1: SMBus host
   SMBMODE=0: SMBus device
3. Other configurations are the same as those of I²C.

SMBus protocols are implemented by the user software, while I²C interface only provide the address identification of these protocols.

SMBus address resolution protocol (ARP)

SMBus address conflicts can be resolved by dynamically assigning a new unique address to each device. Refer to SMBus 2.0 protocol for more information about ARP.

Setting the ARPEN bit can enable the I²C interface to recognize the default device address (0b1100001x). However, unique device identifier (UDID) and the detailed protocol implementation should be handled by software.

SMBus host notify protocol

The slave device can send data to the master device through SMBus host notify protocol. For example, the slave can notify the host to implement ARP with this protocol. Refer to SMBus 2.0 protocol for details on SMBus host notify protocol.

When the ARP mode is enabled (ARPEN=1) in host mode (SMBMODE=1), the I²C interface is enabled to recognize the 0b0001000x (default host address)

SMBus Alert

SMBALERT is an optional signal that connects the ALERT pin between the host and the salve. With this signal, the slave notifies the host to access the slave. SMBALERT is a wired-AND signal. For more information about SMBus Alert, refer to SMBus2.0 protocol.

The detailed sequences are as follows:

SMBus host:

1. Enable SMBus Alert mode by setting SMBALERT=1
2. Enable ALERT interrupt if necessary
3. When an alert event occurs on the ALERT pin (ALERT pin changes from high to low)
4. The host will generate ALERT interrupt if enabled
5. The host then processes the interrupt and accesses to all devices through ARA (Alert Response Address 0001100x) so as to get the slave addresses. Only the devices with pulled-down SMBALERT can acknowledge ARA.
6. The host then continues to operate based on the slave addresses available.

SMBus slave:

1. When an alert event occurs and the ALERT pin changes from high to low (SMBALERT=1), the slave responses to ARA (Alert Response Address) address (0001100x)
2. Enable ALERT interrupt if necessary (an interrupt is generated when receiving ARA address)
3. Wait until the host gets the slave addresses through ARA
4. Report its own address, but it continues to wait if the arbitration is lost.
5. Address is reported properly, and the ALERT pin is released (SMBALERT=0).

Packet error checking (PEC)

Packet error checking (PEC) is used to guarantee the correctness and integrity of data transfer. This is done by using CRC-8 polynomial:

$C(x) = x^8 + x^2 + x + 1$

PEC calculation is enabled when PECEN=1 to check address and data. It becomes invalid when the arbitration is lost.

PEC transmission:

- Common mode: Set PECTRA=1 after the last TDBE event so that PEC is transferred after the last transmitted byte.
- DMA mode: The PEC is transferred automatically after the last transmitted byte. For example, if the number of data to be transferred is 8, then DMA_TCNTx=8.

PEC reception:

- Common mode: Set the PECTRA bit after the last RDBF event. The PECTRA must be set before the ACK pulse of the current byte is received.
- DMA mode: When receiving, it will automatically consider the last byte as PECVAL and check it. For example, if the number of data to be transferred is 8, then DMA_TCNTx=9.

In reception mode, the NACK will be generated when PEC fails.

11.4.5 I²C interrupt requests

The following table lists all the I²C interrupt requests.

Interrupt event	Event flag	Enable control bit
Start condition sent (Host)	STARTF	EVTIEN
Address sent (host) or address matched (slave)	ADDR7F	
10-bit address head sent (host)	ADDRHF	
Data transfer complete	TDC	
Stop condition received (slave)	STOPF	
Transmit data buffer empty	TDBE	EVTIEN and DATAIEN
Receive data buffer full	RDBF	
SMBus alert	ALERTF	ERRIEN
Timeout error	TMOUT	
PEC error	PECERR	
Overload/underload	OUF	
Acknowledge failure	ACKFAIL	
Arbitration lost	ARLOST	
Bus error	BUSERR	

11.4.6 I²C debug mode

When the microcontroller enters debug mode (Cortex®-M4 halted), the SMBUS timeout either continues to work or stops, depending on the I2Cx_SMBUS_TIMEOUT configuration bit in the DEBUG module.

11.5 I²C registers

These peripheral registers must be accessed by words (32 bits).

Table 11-1 I²C register map and reset values

Register	Offset	Reset value
I2C_CTRL1	0x00	0x0000
I2C_CTRL2	0x04	0x0000
I2C_OADDR1	0x08	0x0000
I2C_OADDR2	0x0C	0x0000
I2C_DT	0x10	0x0000
I2C_STS1	0x14	0x0000
I2C_STS2	0x18	0x0000
I2C_CLKCTRL	0x1C	0x0000
I2C_TMRISE	0x20	0x0002

11.5.1 Control register1 (I2C_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15	RESET	0x0	rw	I²C peripheral reset 0: I²C peripheral is not at reset state. 1: I²C peripheral is at reset state. Note: This bit can be used only when the BUSYF bit is “1”, and no Stop condition is detected on the bus.
Bit 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	SMBALERT	0x0	rw	SMBus alert pin set This bit is set or cleared by software. It is cleared by hardware when I2CEN=0. 0: SMBus alert pin high. 1: SMBus alert pin low.
Bit 12	PECTEN	0x0	rw	Request PEC transfer enable This bit is set or cleared by software. It is cleared by hardware after PECTEN is sent, or under Start/Stop condition. 0: No PEC transfer 1: PEC transfer
Bit 11	MACKCTRL	0x0	rw	Master receive mode acknowledge control 0: ACKEN bit controls ACK of the current byte being transferred 1: ACKEN bit controls ACK of the next byte to be transferred. This bit is used only when the number of bytes to receive is equal to 2 so as to ensure that the host responds to ACK in time.
Bit 10	ACKEN	0x0	rw	Acknowledge enable This bit is set or cleared by software. 0: Disabled (no acknowledge sent) 1: Enabled (acknowledge sent)
Bit 9	GENSTOP	0x0	rw	Generate stop condition This bit is set or cleared by software. It is cleared when a Stop condition is detected. It is set by hardware when a timeout error is detected. 0: No Stop condition is generated. 1: Stop condition is generate. The salve releases the SCL and SDA lines when this bit is set in slave mode.
Bit 8	GENSTART	0x0	rw	Generate start condition This bit is set or cleared by software. It is cleared when a Start condition is sent. 0: No Start condition is generated. 1: Start condition is generated.
Bit 7	STRETCH	0x0	rw	Clock stretching mode 0: Enabled 1: Disabled Note: This feature applies to slave mode only.
Bit 6	GCAEN	0x0	rw	General call address enable 0: Enabled 1: Disabled
Bit 5	PECEN	0x0	rw	PEC calculation enable 0: Disabled 1: Enabled

Bit 4	ARPEN	0x0	rw	SMBus address resolution protocol enable 0: Disabled 1: Enabled SMBus host: response to host address 0001000x SMBus slave: response to default device address 0001100x
Bit 3	SMBMODE	0x0	rw	SMBus device mode 0: SMBus slave 1: SMBus host
Bit 2	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 1	PERMODE	0x0	rw	I^2C peripheral mode 0: I^2C mode 1: SMBus mode
Bit 0	I2CEN	0x0	rw	I^2C peripheral enable 0: Disabled 1: Enabled All bits are cleared as I2CEN=0 at the end of the communication. In master mode, this bit must not be cleared before the end of the communication.

Note: When the GENSTART, GENSTP or PECTEN bit is set, the I2C_CTRL1 cannot be written by software until the corresponding bit has been cleared by hardware, otherwise, a second GENSTART, GENSTP or PECTEN request may be set.

11.5.2 Control register2 (I2C_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 12	DMAEND	0x0	rw	End of DMA transfer 0: The next DMA transfer is no the last one. 1: The next DMA transfer is the last one.
Bit 11	DMAEN	0x0	rw	DMA transfer enable 0: Disabled 1: Enabled
Bit 10	DATAIEN	0x0	rw	Data transfer interrupt enable An interrupt is generated when TDBE =1 or RDBF=1. 0: Disabled 1: Enabled
Bit 9	EVTIEN	0x0	rw	Event interrupt enable 0: Disabled 1: Enabled An interrupt is generated in the following conditions: – STARTF = 1 (Master mode) – ADDR7F = 1 (Master/slave mode) – ADDRHF= 1 (Master mode) – STOPF = 1 (Slave mode) – TDC = 1, but no TDBE or RDBF event – If DATAIEN = 1, the TDBE event is 1. – If DATAIEN = 1, the RDBF event is 1.
Bit 8	ERRIEN	0x0	rw	Error interrupt enable 0: Disabled 1: Enabled An interrupt is generated in the following conditions: – BUSERR = 1 – ARLOST = 1

				– ACKFAIL = 1 – OVER = 1 – PECERR = 1 – TMOUT = 1 – ALERTF = 1
Bit 7: 0	CLKFREQ	0x00	rw	I²C input clock frequency Correct input clock frequency must be set to generate correct timings. The range allowed is between 2 MHz and 120 MHz. 2: 2MHz 3: 3MHz …… 120: 120MHz

11.5.3 Own address register1 (I2C_OADDR1)

Bit	Register	Reset value	Type	Description
Bit 15	ADDR1MODE	0x0	rw	Address mode 0: 7-bit address 1: 10-bit address
Bit 14: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 0	ADDR1	0x000	rw	Own address1 In 7-bit address mode, bit 0 and bit [9: 8] don't care.

11.5.4 Own address register2 (I2C_OADDR2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 1	ADDR2	0x00	rw	Own address 2 7-bit address
Bit 0	ADDR2EN	0x0	rw	Own address 2 enable 0: In 7-bit address mode, only OADDR1 is recognized. 1: In 7-bit address mode, both OADDR1 and OADDR2 are recognized.

11.5.5 Data register (I2C_DT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value
Bit 7: 0	DT[7: 0]	0x00	rw	This field is used to store data received or to be transferred. Transmitter mode: Data transfer starts automatically when a byte is written to the DT register. Once the transfer starts (TDE=1), I²C will keep a continuous data transfer flow if the next data to be transferred is written to the DT register in a timely manner. Receiver mode: Btyes received are copied into the DT register (RDNE=1). A continuous data transfer flow can be maintained if the DT register is read before the next word is received (RDNE=1). Note: If an ARLOST event occurs on ACK pulse, the received byte is not copied into the data register, so it cannot be read.

11.5.6 Status register1 (I2C_STS1)

Bit	Register	Reset value	Type	Description
Bit 15	ALERTF	0x0	rw0c	SMBus alert flag In SMBus host mode: 0: No SMBus alert 1: SMBus alert event is received. In SMBus slave mode: It indicates the receiving status of the default device address (0001100x) 0: Default device address is not received. 1: Default device address is received. This bit is cleared by software, or by hardware when I2CEN=0.
Bit 14	TMOUT	0x0	rw0c	SMBus timeout flag 0: No timeout error. 1: Timeout This bit is cleared by software, or by hardware when I2CEN=0. Note: This function is valid only in SMBUS mode.
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	PECERR	0x0	rw0c	PEC receive error flag 0: No PEC error 1: PEC error occurs. This bit is cleared by software.
Bit 11	OUF	0x0	rw0c	Overload / underload flag In transmission mode: 0: Normal 1: Underload In reception mode: 0: Normal 1: Overload This bit is cleared by software, or by hardware when I2CEN=0.
Bit 10	ACKFAIL	0x0	rw0c	Acknowledge failure flag 0: No acknowledge failure 1: Acknowledge failure occurs. Set by hardware when no acknowledge is returned. This bit is cleared by software, or by hardware when I2CEN=0.
Bit 9	ARLOST	0x0	rw0c	Arbitration lost flag 0: No arbitration lost is detected. 1: Arbitration lost is detected. This bit is cleared by software, or by hardware when I2CEN=0. On ARLOST even, the I^2C interface switches to slave mode automatically.
Bit 8	BUSERR	0x0	rw0c	Bus error flag 0: No Bus error occurs. 1: Bus error occurs. Set by hardware when the interface detects a misplaced Start or Stop condition. This bit is cleared by software, or by hardware when I2CEN=0.
Bit 7	TDBE	0x0	ro	Transmit data buffer empty flag

				0: The data is being transferred from the DT register to the shift register (the data is still loaded with the data at this point.) 1: The data has been moved from the DT register to the shift register. The data register is empty now. This flag is set when the DT register is empty, and cleared when writing to the DT register. Note: The TDBE bit is not cleared by writing the first data to be transmitted, or by writing data when the TDC is set, since the data register is still empty at this time.
Bit 6	RDBF	0x0	ro	Receive data buffer full flag 0: Data register is empty. 1: Data register is full (data received) This flag is cleared when the DT register is read. The RDBF bit is not set at ARLOST event.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	STOPF	0x0	ro	Stop condition generation complete flag 0: No Stop condition is detected. 1: Stop condition is detected. This bit is set by hardware when a Stop condition is detected on the bus by the slave if ACKEN=1. It is cleared by reading STS1 register followed by writing to the CTRL1 register.
Bit 3	ADDRHF	0x0	ro	Master 9~8 bit address head match flag 0: Master 9~8 bit address head mismatch 1: Master 9~8 bit address head match Set by hardware when the first byte is sent by master in 10-bit address mode. Cleared by a write to the CTRL1 register after the STS1 register is read by software, or by hardware when PEN=0. Note: The ADDR10 bit is not set after a NACK reception.
Bit 2	TDC	0x0	ro	Data transfer complete flag 0: Data transfer is not completed yet (the shift register still holds data) 1: Data transfer is completed (shift register is empty) This bit is cleared automatically by read or write access to the DT register, or when a Start or Stop condition is received. When STRETCH=0 In reception mode, when a new byte (including ACK pulse) is received and the data register is not read yet (RDBF=1) In transmission mode, when a new byte is sent and the data register is not written yet (TDBE=1) The TDC is set under both conditions.
Bit 1	ADDR7F	0x0	ro	0~7 bit address match flag 0: Address is not sent in host ode or received in slave mode 1: Address is sent in host mode or address is received in slave mode. Cleared by read access to STS2 register after the software reads STS1 register. Note: the ADDR7F bit is not set after a NACK reception.
Bit 0	STARTF	0x0	ro	Start condition generation complete flag 0: No Start condition is generated. 1: Start condition is generated. Cleared by write access to the DT register after the

				software reads the STS1 register.

11.5.7 Status register2 (I2C_STS2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	PECVAL	0x00	ro	PEC value Cleared when PECEN is reset.
Bit 7	ADDR2F	0x0	ro	Received address 2 flag 0: Received address matches the contents of OADDR1 1: Received address matches the contents of OADDR2 Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 6	HOSTADDRF	0x0	ro	SMBus host address reception flag 0: SMBus host address is not received. 1: SMBus host address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 5	DEVADDRF	0x0	ro	SMBus device address reception flag 0: SMBus device address is not received. 1: SMBus device address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 4	GCADDRF	0x0	ro	General call address reception flag 0: General call address is not received. 1: General call address is received. Cleared when a Stop/Start condition is received, or by hardware when I2CEN=0.
Bit 3	Reserved	0x0	resd	Keep at its default value.
Bit 2	DIRF	0x0	ro	Transmission direction flag 0: Data reception 1: Data transmission Cleared by hardware when a Stop condition is received.
Bit 1	BUSYF	0x0	ro	Bus busy flag transmission mode 0: Bus idle 1: Bus busy Set by hardware on detection of SDA/SCL low, and cleared by hardware on detection of a Stop condition.
Bit 0	TRMODE	0x0	ro	Transmission mode 0: Slave mode 1: Master mode Set by hardware when the GENSTART is set and a Start condition is sent. Cleared by hardware when a Stop condition is detected.

11.5.8 Clock control register (I2C_ CLKCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	SPEEDMODE	0x0	rw	Speed mode selection 0: Standard mode (up to 100 kHz) 1: Fast mode (up to 400 kHz) In fast mode, an accurate 400kHz clock is generated when the I^2C clock frequency is an integer multiple of 10MHz.
Bit 14	DUTYMODE	0x0	rw	Fast mode duty cycle 0: The ratio of High to low is 1:2. 1: The ratio of low to high is 9:16.
Bit 13: 12	Reserved	0x0	resd	Kept at its default value.

Bit 11: 0	SPEED	0x000	rw	I²C bus speed config In standard mode: High level= SPEED x T_{I2C_CLK} Low level= SPEED x T_{I2C_CLK} In fast mode: DUTYMODE = 0: High level= SPEED x T_{I2C_CLK} x 1 Low level= SPEED x T_{I2C_CLK} x 2 DUTYMODE = 1: High level= SPEED x T_{I2C_CLK} x 9 Low level= SPEED x T_{I2C_CLK} x 16 The minimum value allowed in standard mode is 4. In fast mode, the minimum value allowed is 1. The CLKCTRL register can be configured only when the I2C is disabled (I2CEN=0).

Note: The I2C_CLKCTRL register can be configured only when the I2C is disabled (I2CEN=0).

11.5.9 Clock rise register (I2C_TMRISE)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x000	resd	Forced to 0 by hardware.
Bit 5: 0	RISETIME	0x02	rw	I2C bus rise time Time= RISETIME x T_{I2C_CLK} In standard mode, I2C protocol stand is 1000ns, and the formula as follows: RISETIME = F_{I2C_CLK} +1 For example, when I2C clock is 48MHz, RISETIME = 48+1 In fast mode, I2C protocol stand is 300ns, and the formula as follows: RISETIME = F_{I2C_CLK} x0.3+1 For example, when I2C clock is 48MHz, RISETIME = 48x0.3+1 Note: RISETIME[5:0] can be configured only when I2CEN=0.

12 Universal synchronous/asynchronous receiver/transmitter (USART)

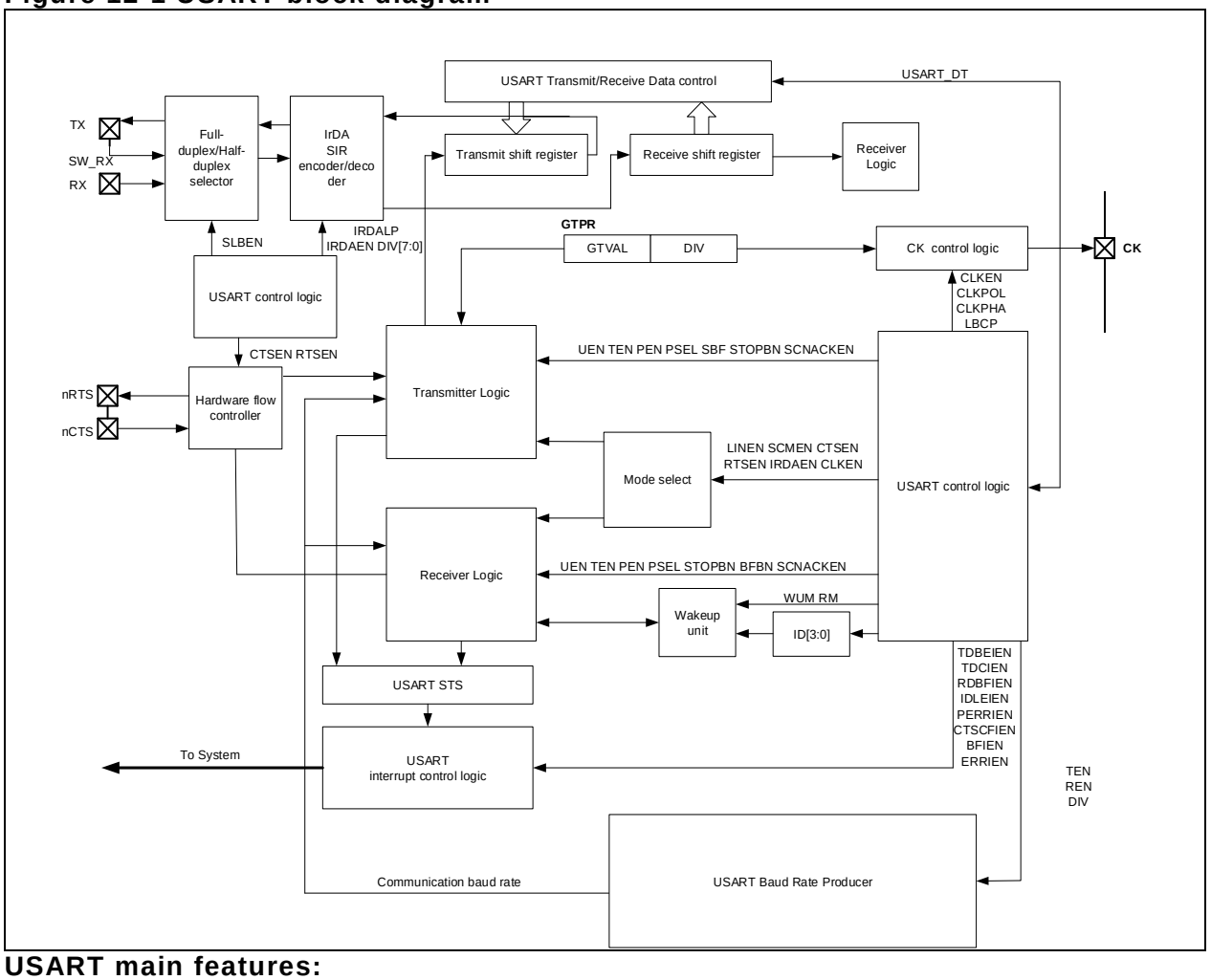
12.1 USART introduction

The universal synchronous/asynchronous receiver/transmitter (USART) acts as a general-purpose interface for communication by means of various configurations and peripherals with different data formats. It supports asynchronous full-duplex and half-duplex as well as synchronous transfer. With a programmable baud rate generator, USART offers up to 4.6875 MBits/s of baud rate by setting the system frequency and frequency divider, which is also convenient for users to configure the required communication frequency.

In addition to standard NRZ asynchronous and synchronous receiver/transmitter communication protocols, USART also supports widely-used serial communication protocols such as LIN (Local Interconnection Network), IrDA (Infrared Data Association) SIRENDEC specification, Asynchronous SmartCard protocol defined in ISO7816-3 standard, and CTS/RTS (Clear To Send/Request To Send) hardware flow operation.

It also allows multi-processor communication, and supports silent mode waken up by idle frames or ID matching to build up a USART network. Meanwhile, high-speed communication is possible by using DMA.

Figure 12-1 USART block diagram



USART main features:

- Programmable full-duplex or half-duplex communication
  - Full-duplex, asynchronous communication
  - Half-duplex, single communication

- Programmable communication modes
  - NRZ standard format (Mark/Space)
  - LIN (Local Interconnection Network):
  - IrDA SIR:
  - Asynchronous SmartCard protocol defined in ISO7816-3 standard: Support 0.5 or 1.5 stop bits in Smartcard mode
  - RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation
  - Multi-processor communication with silent mode (waken up by configuring ID match and bus idle frame)
  - Synchronous mode
- Programmable baud rate generator
  - Shared by transmission and reception, up to 4.6875 MBits/s
- Programmable frame format
  - Programmable data word length (8 bits or 9 bits)
  - Programmable stop bits-support 1 or 2 stop bits
  - Programmable parity control: transmitter with parity bit transmission capability, and receiver with received data parity check capability
- Programmable DMA multi-processor communication
- Programmable separate enable bits for transmitter and receiver
- Programmable output CLK phase, polarity and frequency
- Detection flags
  - Receive buffer full
  - Transmit buffer empty
  - Transfer complete flag
- Four error detection flags
  - Overrun error
  - Noise error
  - Framing error
  - Parity error
- Programmable 10 interrupt sources with flags
  - CTSF changes
  - LIN break detection
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle bus detected
  - Overrun error
  - Framing error
  - Noise error
  - Parity error

12.2 Full-duplex/half-duplex selector

The full-duplex and half-duplex selector enables USART to perform data exchanges with peripherals in full-duplex or half-duplex mode, which is achieved by setting the corresponding registers.

In two-wire unidirectional full-duplex mode (by default), TX pin is used for data output, while the RX pin is used for data input. Since the transmitter and receiver are independent of each other, USART is allowed to send/receive data at the same time so as to achieve full-duplex communication.

When the HALFSEL is set 1, the single-wire bidirectional half-duplex mode is selected for communication. In this case, the LINEN, CLKEN, SCMEN and IRDAEN bits must be set 0. RX pin is inactive, while TX and SW_RX are interconnected inside the USART. For the USART part, TX pins is used for data output, and SW_RX for data input. For the peripheral part, bidirectional data transfer is executed through IO mapped by TX pin.

12.3 Mode selector

12.3.1 Introduction

USART mode selector allows USART to work in different operation modes through software configuration so as to enable data exchanges between USART and peripherals with different communication protocols.

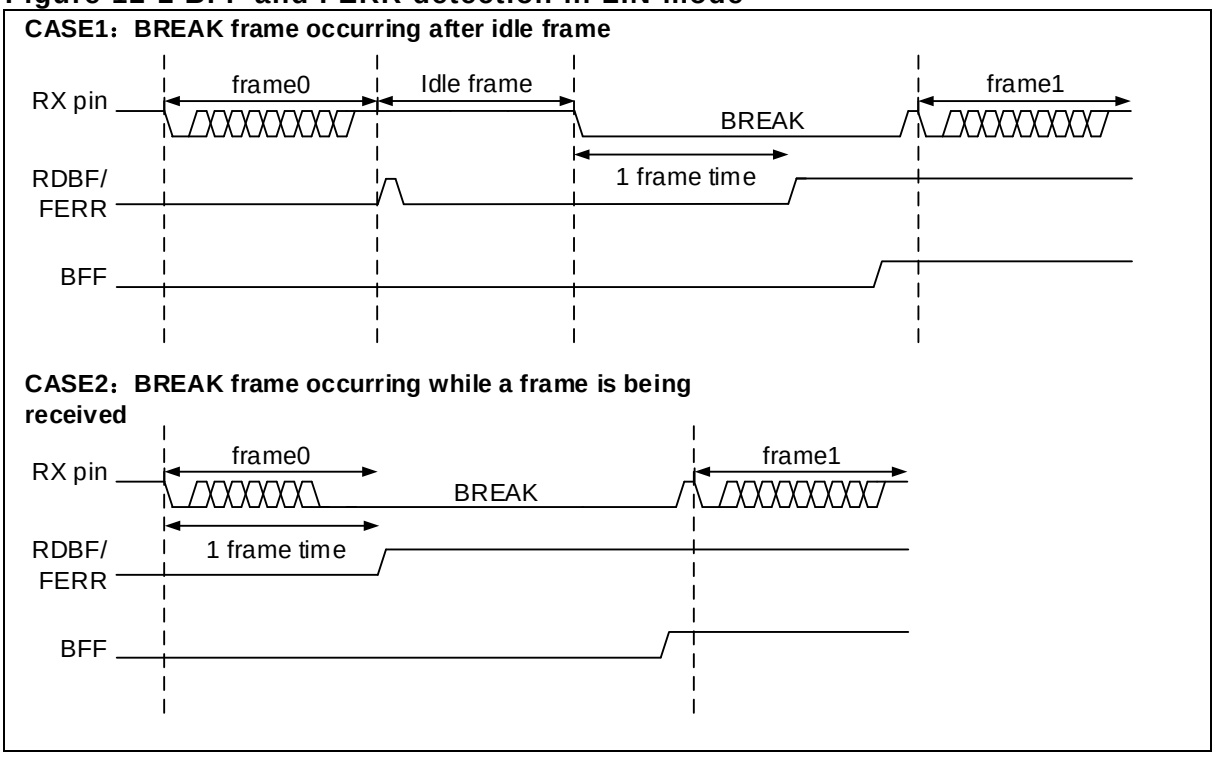
USART supports NRZ standard format (Mark/Space), by default. It also supports LIN (Local Interconnection Network), IrDA SIR (Serial Infrared), Asynchronous Smartcard protocol in ISO7816-3 standard, RS-232 CTS/RTS (Clear To Send/Request To Send) hardware flow operation, silent mode and synchronous mode, depending on USART mode selection configuration.

12.3.2 Configuration procedure

Selection of operation mode is done by following the configuration process listed below. In addition, such configuration method, along with that of receiver and transmitter described in the subsequent sections, are used to make USART initialization configuration.

1. **LIN mode:**
   Parameters configuration: LINEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0, SLHDEN=0, IRDAEN=0 and DBN=0.
   LIN master has break frame transmission capability, and thus it is able to send 13-bit low-level LIN synchronous break frame by setting SBF=1.
   Similarly, LIN slave has break frame detection capability, and thus it is able to detect 11-bit or 10-bit break fame, depending on whether BFBN=1 or BFBN=0.

Figure 12-2 BFF and FERR detection in LIN mode



2. **Smartcard mode:**

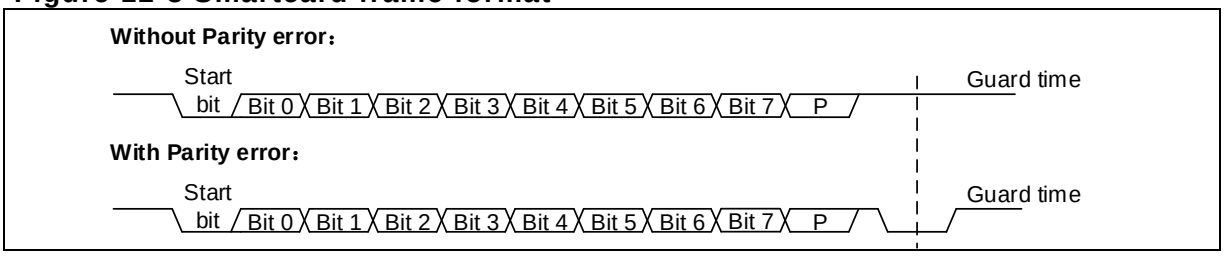
Parameters configuration: SCMEN=1, LINEN=0, SLHDEN=0, IRDAEN=0, CLKEN=1, DBN=1, PEN=1, and STOPBN[1: 0]=11.

The polarity, phase and pulse number of the clock can be configured by setting the CLKPOL, CLKPHA and LBCP bits (Refer to Synchronous mode for details).

The assertion of the TDC flag can be delayed by setting the SCGT[7: 0] bit (guard time bit). The TDF bit can be asserted high after the guard time counter reaches the value programmed in the SCGT[7: 0] bit.

The Smartcard is a single-wire half duplex communication protocol. The SCNACKEN bit is used to select whether to send NACK when a parity error occurs. This is to indicate to the Smarcard that the data has not been correctly received.

Figure 12-3 Smartcard frame format

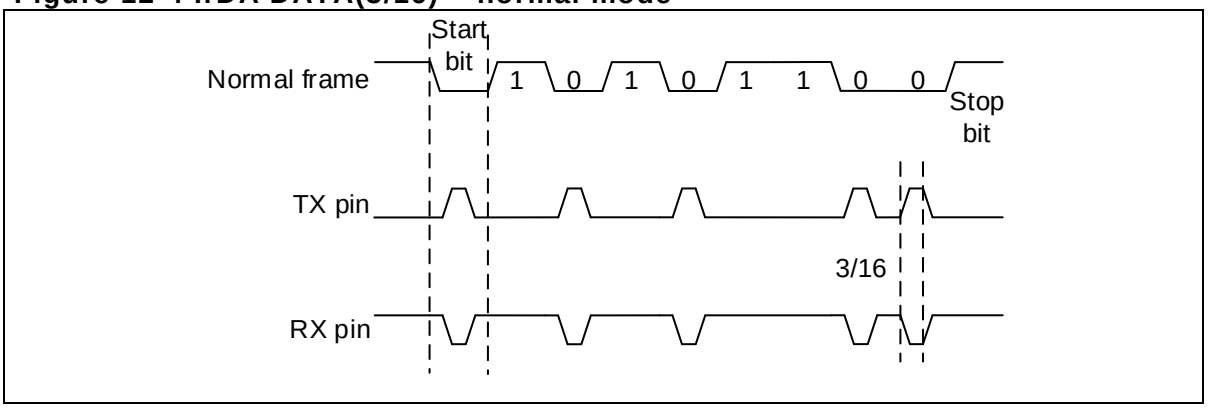


3. **Infrared mode:**

Parameters configuration: IRDAEN=1, CLKEN=0, STOPBN[1: 0]=0, SCMEN=0 and SLHDEN=0.

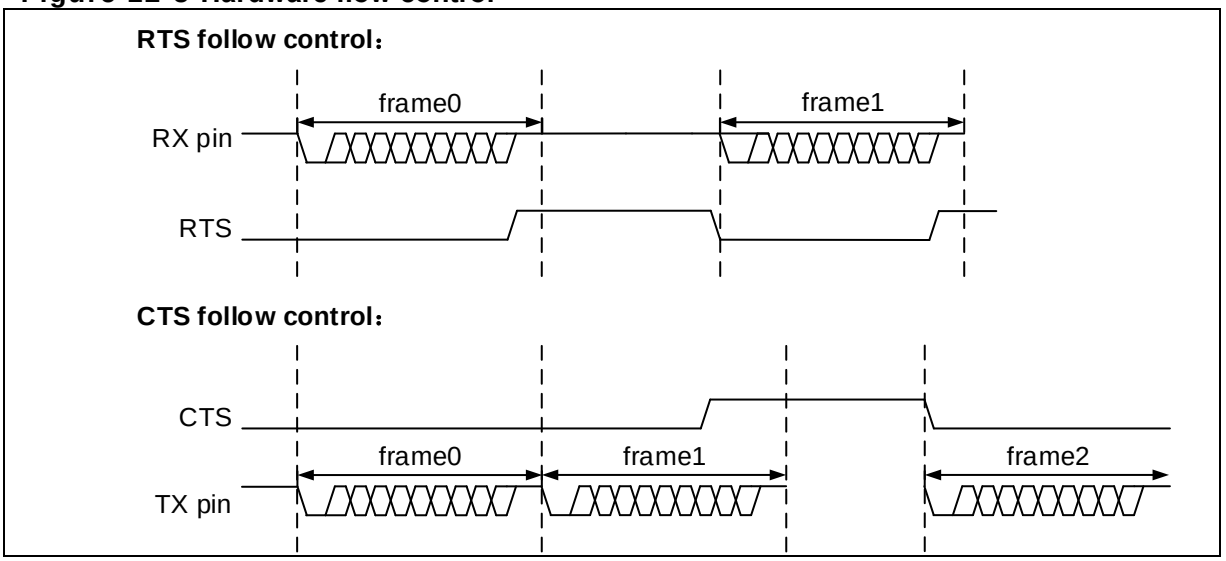
The infrared low-power mode can be enabled by setting IRDALP=1. In normal mode the transmitted pulse width is specified as 3/16 bit. In infrared low-power mode, the pulse width can be configurable. And the ISDIV[7:0] bit can be used to achieve the desired low-power frequency.

Figure 12-4 IrDA DATA(3/16) – normal mode



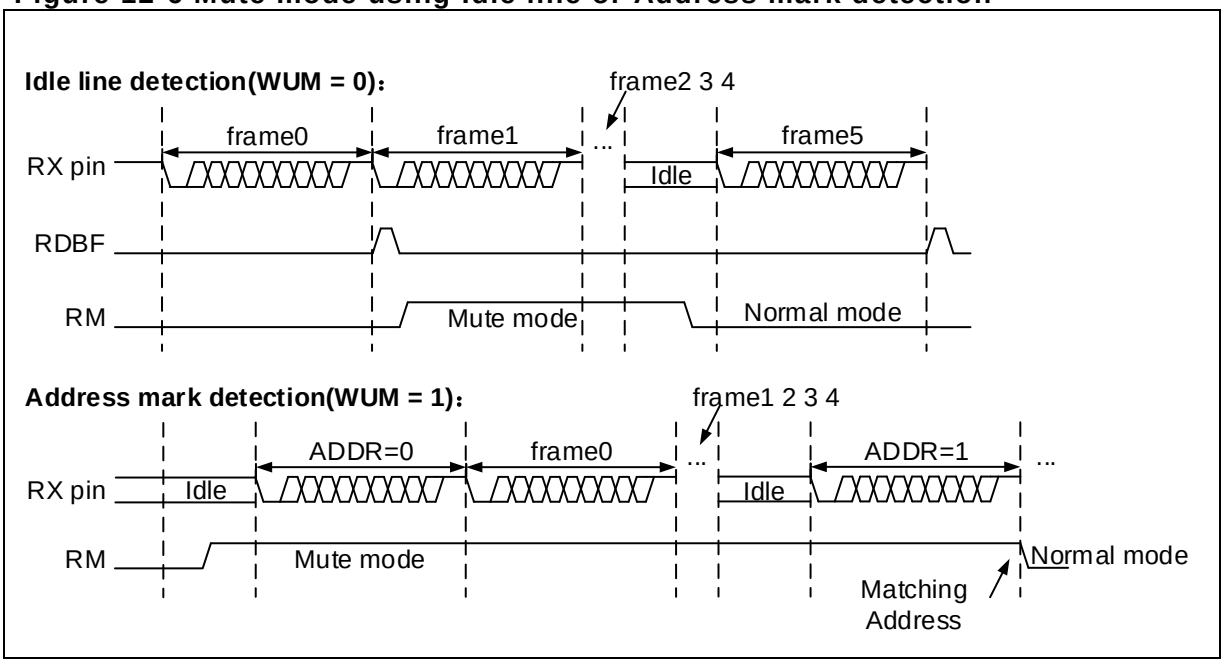
4. **Hardware flow control mode:**
RTS and CTS flow control can be enabled by setting RTSEN=1 and CTSEN=1, respectively. This is to control serial data flow between two devices.
RTS: the RTS becomes active (pull-down means low) as soon as the USART receiver is ready to receive a data. When the data has arrived (starts at each STOP bit) in the receive register, the RTS bit is set, indicating request to stop data transfer at the end of current frame.
CTS: the USART transmitter checks the CTS input before sending next frame. The next data is sent if CTS is active (when low); if CTS becomes inactive (when high) during transmission, it stops sending at the end of current transfer.

Figure 12-5 Hardware flow control



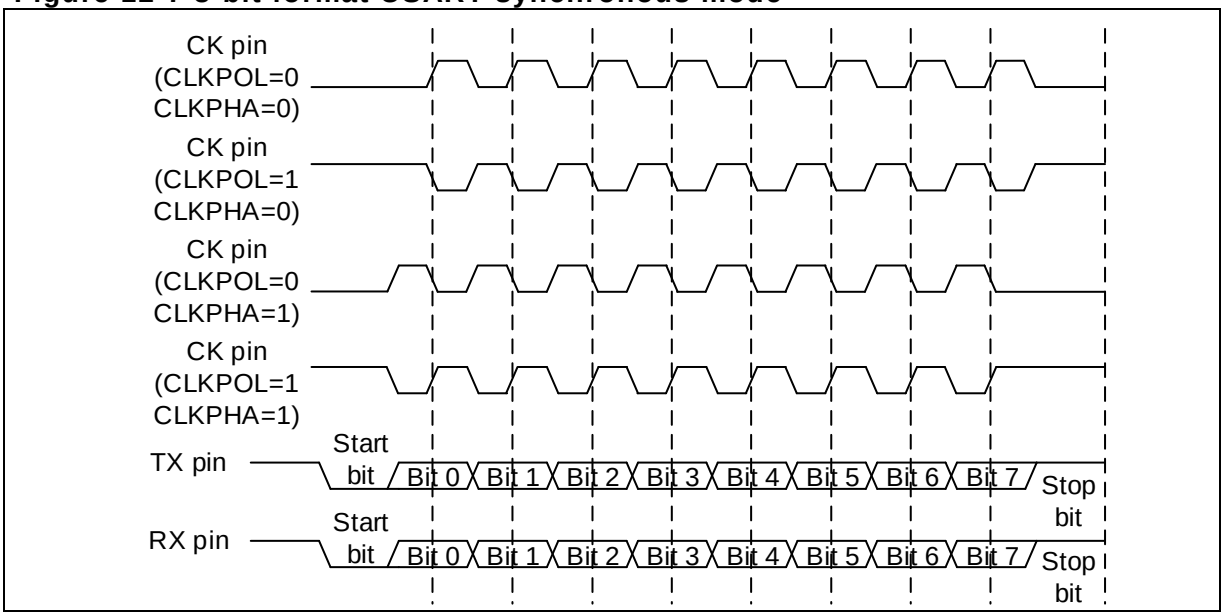
5. **Silent mode:**
Silent mode can be entered by setting RM=1. It is possible to wake up from silent mode by setting WUM=1 (ID match) and WUM=0 (idle bus), respectively. The ID[3: 0] is configurable. When ID match is selected, if the MSB of data bit is set to 1, it indicates that the current data is ID, and the four LSB represent ID value.

Figure 12-6 Mute mode using Idle line or Address mark detection



6. **Synchronous mode:**
By setting the CLKEN bit to 1, synchronous mode and clock pin output are enabled. Select CK pin high or low in idle state by setting the CLKPOL bit (1 or 0). Whether to sample data on the second or the first edge of the clock depends on the CLKPHA bit (1 or 0). The LBCP bit (1 or 0) is used to select whether to output clock on the last data bit. And the ISDIV[4: 0] is used to select the required clock output frequency.

Figure 12-7 8-bit format USART synchronous mode



12.4 USART frame format and configuration

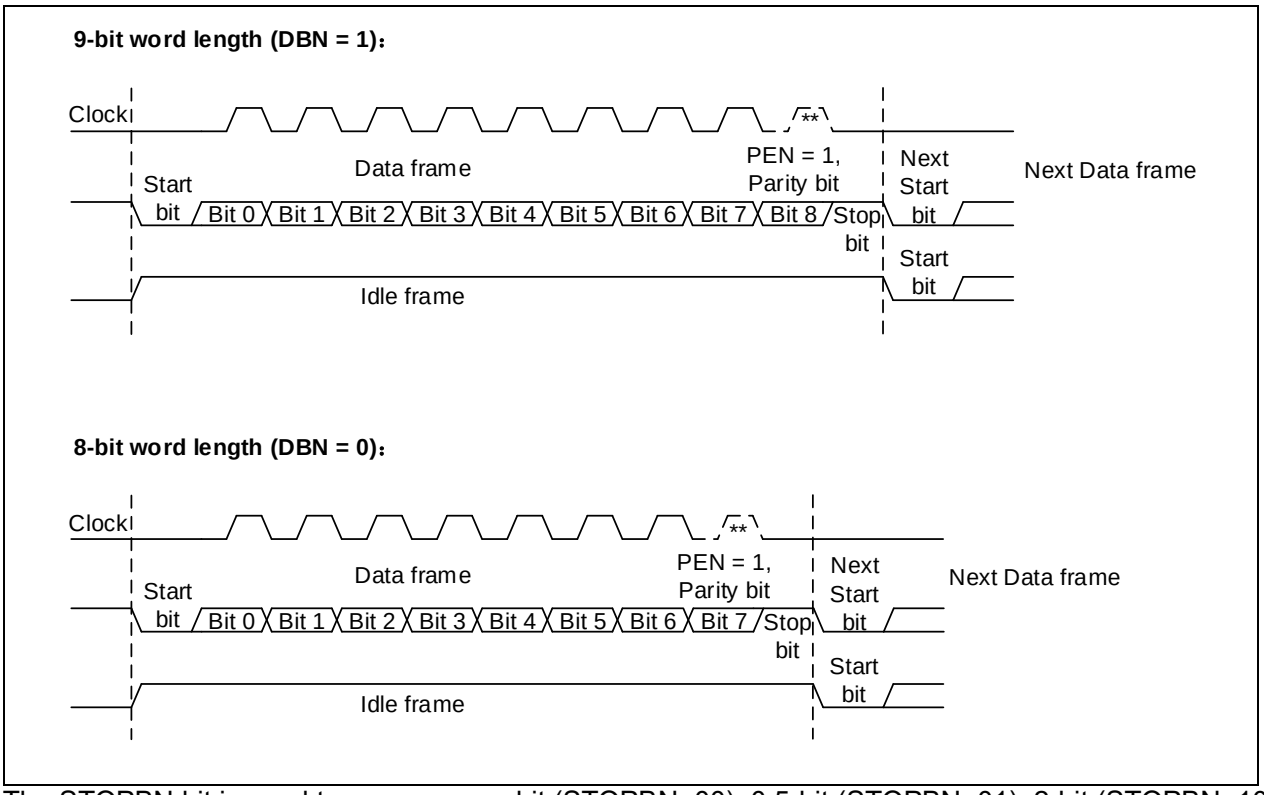
USART data frame consists of start bit, data bit and stop bit, with the last data bit being as a parity bit.

USART idle frame size is equal to that of the data frame under current configuration, but all bits are 1.

USART break frame size is the current data frame size plus its stop bit. All bits before the stop bit are 0.

The DBN bit is used to program 8-bit (DBN=0) or 9-bit (DBN=1) data bits.

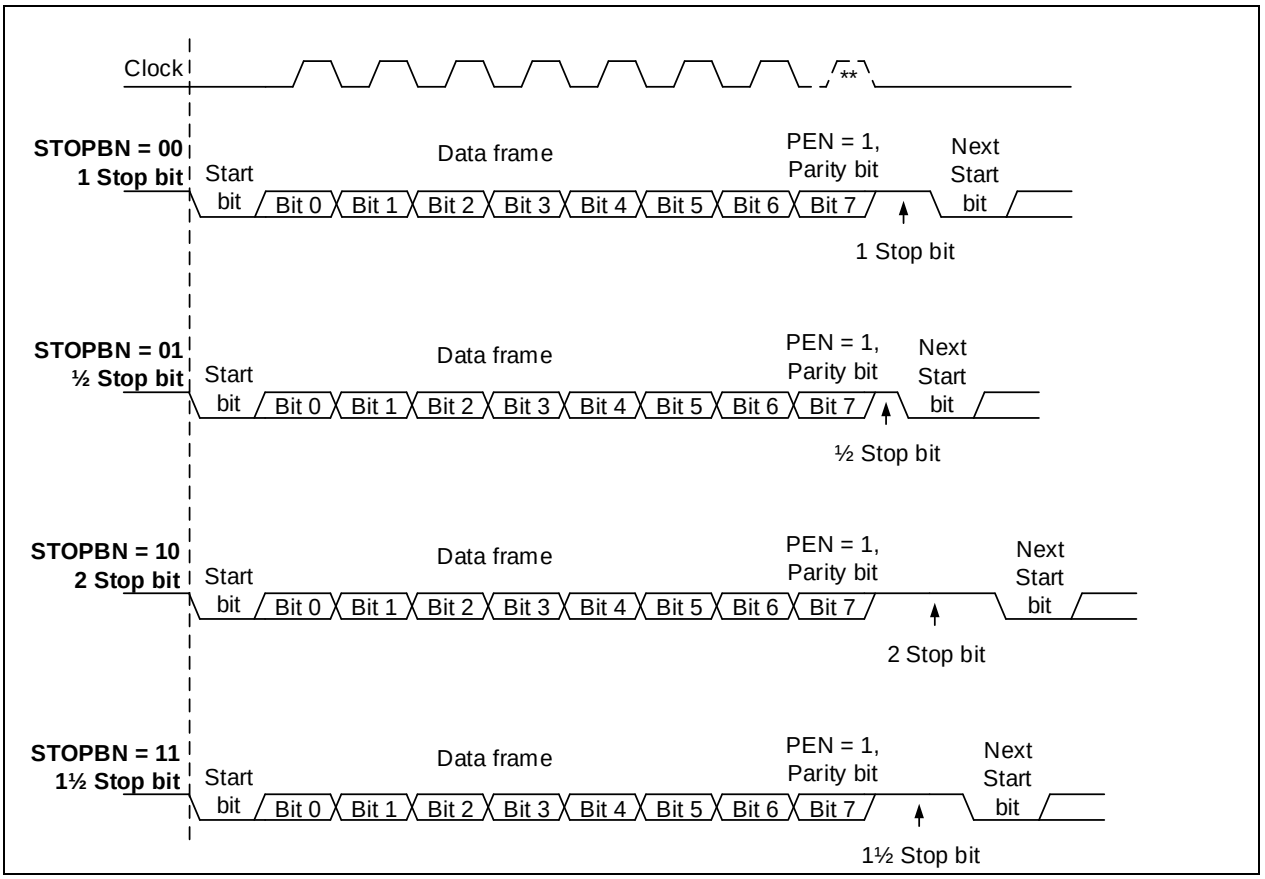
Figure 12-8 Word length



The STOPBN bit is used to program one bit (STOPBN=00), 0.5-bit (STOPBN=01), 2-bit (STOPBN=10) and 1.5-bit (STOPBN=11) stop bits.

Set the PEN bit will enable parity control. PSEL=1 indicates Odd parity, while PSEL=0 for Even parity. Once the parity control is enabled, the MSB of the data bit will be replaced with parity bit, that is, the significant bits is reduced by one bit.

Figure 12-9 Stop bit configuration



12.5 DMA transfer introduction

Enable transmit data buffer and receive data buffer using DMA to achieve continuous high-speed transmission for USART, which is detailed in subsequent sections. For more information on specific DMA configuration, refer to DMA chapter.

12.5.1 Transmission using DMA

1. Select a DMA channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the USART_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
3. Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the USART_DT register from the memory address after transmit request is received by DMA.
4. Configure the total number of bytes to be transferred in the DMA control register (TMRx_DMACTRL).
5. Configure the channel priority of DMA transfer in the DMA control register (TMRx_DMACTRL).
6. Configure DMA interrupt generation after half or full transfer in the DMA control register (TMRx_DMACTRL).
7. Enable DMA transfer channel in the DMA control register (TMRx_DMACTRL).

12.5.2 Reception using DMA

1. Select a DMA transfer channel: Select a DMA channel from DMA channel map table described in DMA chapter.
2. Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the USART_DT register to the programmed destination after reception request is received by DMA.
3. Configure the source of DMA transfer: Configure the USART_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the USART_DT register to the programmed destination after reception request is received by DMA.
4. Configure the total number of bytes to be transferred in the DMA control register (TMRx_DMACTRL).
5. Configure the channel priority of DMA transfer in the DMA control register (TMRx_DMACTRL).
6. Configure DMA interrupt generation after half or full transfer in the DMA control register (TMRx_DMACTRL).
7. Enable a DMA transfer channel in the DMA control register (TMRx_DMACTRL).

12.6 Baud rate generation

12.6.1 Introduction

USART baud rate generator uses an internal counter based on PCLK. The DIV (USART_BAUDR [15:0] register) represents the overflow value of the counter. Each time the counter is full, it denotes one-bit data. Thus each data bit width refers to PCLK cycles x DIV.

The receiver and transmitter of USART share the same baud rate generator, and the receiver splits each data bit into 16 equal parts to achieve oversampling, so the data bit width should not be less than 16 PCLK periods, that is, the DIV value must be equal to or greater than 16.

12.6.2 Configuration

User can program the desired baud rate by setting different system clocks and writing different values into the USART_BAUDR register. The calculation format is as follows:

$$\frac{\mathrm{TX}}{\mathrm{RX}} baud\ \ rate = \frac{f_{CK}}{\mathrm{DIV}}$$

Where, f_{CK} refers to the system clock of USART (i.e. PCLK1/PCLK2)

Note: 1. Write access to the USART_BAUDR register before UEN. The baud rate register value should not be altered when UEN=1.

2. When USART receiver or transmitter is disabled, the internal counter will be reset, and baud rate interrupt will occur.

Table 12-1 Error calculation for programmed baud rate

Baud		fPCLK=36MHz			fPCLK=72MHz		
No.	Kbps	Actual	Value programmed in the baud register	Error%	Actual	Value programmed in the baud register	Error%
1	2.4	2.4	15000	0%	2.4	30000	0%
2	9.6	9.6	3750	0%	9.6	7500	0%
3	19.2	19.2	1875	0%	19.2	3750	0%
4	57.6	57.6	625	0%	57.6	1250	0%
5	115.2	115.384	312	0.15%	115.2	625	0%
6	230.4	230.769	156	0.16%	230.769	312	0.16%
7	460.8	461.538	78	0.16%	461.538	156	0.16%
8	921.6	923.076	39	0.16%	923.076	78	0.16%
9	2250	2250	16	0%	2250	32	0%
10	4500	NA	NA	NA	4500	16	0%

Taking a baud rate of 115.2Kbps as an example, if fPCLK=36MHz, the value in the baud register should be set to 312(0x38). Based on formula, the calculated baud rate (actual) is 36000000 / 312 = 115384 = 115.384Kbps.

The % error between the desired and actual value is calculated based on the formula: (Calculated actual result-Desired)/desired baud rate*100%, that is, (115.384 - 115.2) / 115.2 * 100% = 0.15%.

12.7 Transmitter

12.7.1 Transmitter introduction

USART transmitter has its individual TEN control bit. The transmitter and receiver share the same baud rate that is programmable. There is a transmit data buffer (TDR) and a transmit shift register in the USART. The TDBE bit is set whenever the TDR is empty, and an interrupt is generated if the TDBEIEN is set.

The data written by software is stored in the TDR register. When the shift register is empty, the data will be moved from the TDR register to the shift register so that the data in the transmit shift register is output on the TX pin in LSB mode. The output format depends on the programmed frame format.

If synchronous transfer or clock output is selected, the clock pulse is output on the CK pin. If the hardware flow control is selected, the control signal is input on the CTS pin.

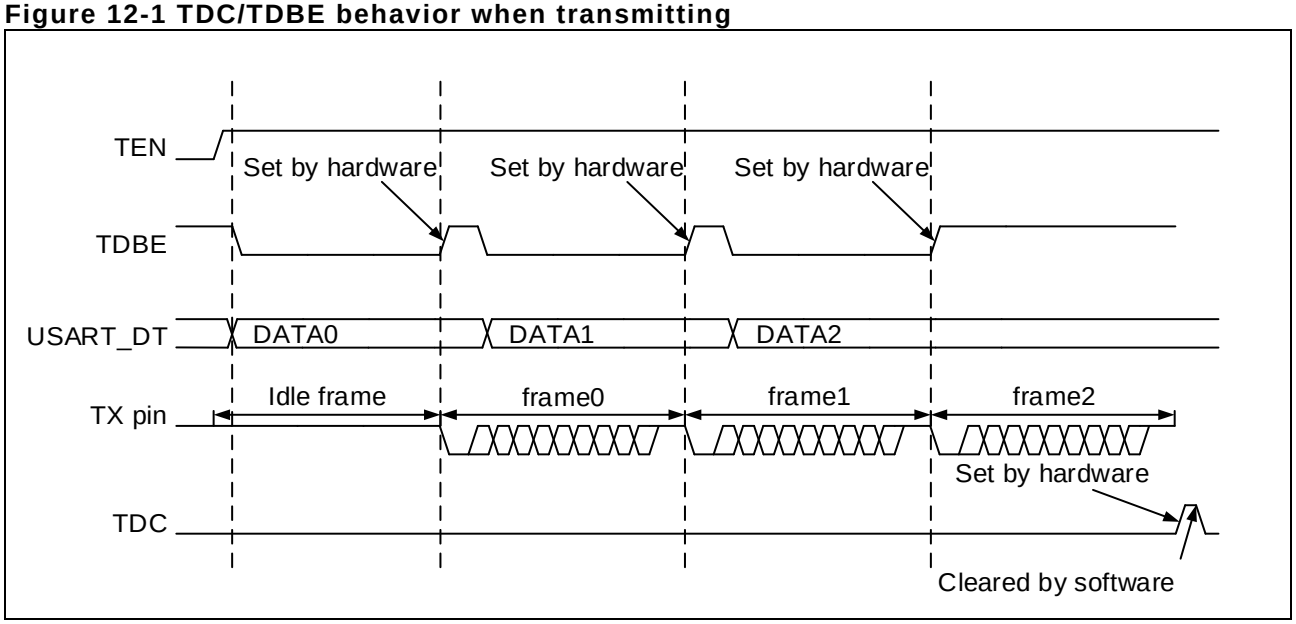
Note: 1. The TEN bit cannot be reset during data transfer, or the data on the TX pin will be corrupted.

2. After the TEN bit is enabled, the USART will automatically send an idle frame.

12.7.2 Transmitter configuration

1. USART enable: Set the UEN bit.
2. Full-duplex/half-duplex configuration: Refer to *12.2 Full-duplex/half-duplex selector*.
3. Mode configuration: Refer to *12.3 Mode selector*.
4. Frame format configuration: Refer to *12.4 USART frame format and configuration*.
5. Interrupt configuration: Refer to *12.9 Interrupt requests*.
6. DMA transmission configuration: If the DMA mode is selected, the DMATEN bit (bit 7 in the USART_CTRL3register) is set, and configure DMA register accordingly.
7. Baud rate configuration: Refer to *12.6 Baud rate generation*.
8. Transmitter enable: When the TEN bit is set, the USART transmitter will send an idle frame.
9. Write operation: Wait until the TDBE bit is set, the data to be transferred will be loaded into the USART_DT register (This operation will clear the TDBE bit). Repeat this step in non-DMA mode.
10. After the last data expected to be transferred is written, wait until the TDC is set, indicating the end of transfer. The USART cannot be disabled before the flag is set, or transfer error will occur.
11. When TDC=1, read access to the USART_STS register and write access to the USART_DT register will clear the TDC bit; This bit can also be cleared by writing “0”, but this is valid only in DMA mode.

Figure 12-1 TDC/TDBE behavior when transmitting



12.8 Receiver

12.8.1 Receiver introduction

USART receiver has its individual REN control bit (bit 2 in the USART_CTRL1 register). The transmitter and receiver share the same baud rate that is programmable. There is a receive data buffer (RDR) and a receive shift register in the USART.

The data is input on the RX pin of the USART. When a valid start bit is detected, the receiver ports the data received into the receive shift register in LSB mode. After a full data frame is received, based on the programmed frame format, it will be moved from the receive shift register to the receive data buffer, and the RDBF is set accordingly. An interrupt is generated if the RDBFIEN is set.

If hardware flow control is selected, the control signal is output on the RTS pin.

During data reception, the USART receiver will detect whether there are errors to occur, including framing error, overrun error, parity check error or noise error, depending on software configuration, and whether there are interrupts to generate using the interrupt enable bits.

12.8.2 Receiver configuration

Configuration procedure:

1. USART enable: UEN bit is set.
2. Full-duplex/half-duplex configuration: Refer to *12.2 Full-duplex/half-duplex selector*.
3. Mode configuration: Refer to *12.3 Mode selector*.
4. Frame format configuration: Refer to *12.4 USART frame format and configuration*.
5. Interrupt configuration: Refer to *12.9 Interrupt requests*.
6. Reception using DMA: If the DMA mode is selected, the DMAREN bit is set, and configure DMA register accordingly.
7. Baud rate configuration: Refer to *12.6 Baud rate generation*.
8. Receiver enable: REN bit is set.

Character reception:

- The RDBF bit is set. It indicates that the content of the shift register is transferred to the RDR (Receiver Data Register). In other words, data is received and can be read (including its associated error flags)
- An interrupt is generated when the RDBFIEN is set.
- The error flag is set when a framing error, noise error or overrun error is detected during reception.
- In DMA mode, the RDBF bit is set after every byte is received, and it is cleared when the data register is read by DMA.
- In non-DMA mode, the RDBF bit is cleared when read access to the USART_DT register by software. The RDBF flag can also be cleared by writing 0 to it. The RDBF bit must be cleared before the end of next frame reception to avoid overrun error.

Break frame reception:

- Non-LIN mode: It is handled as a framing error, and the FERR is set. An interrupt is generated if the corresponding interrupt bit is enabled. Refer to framing error described below for details.
- LIN mode: It is handled as a break frame, and the BFF bit is set. An interrupt is generated if the BFIEN is set.

Idle frame reception:

- It is handled as a data frame, and the IDLEF bit is set. An interrupt is generated if the IDLEIEN is set.

When a framing error occurs:

- The FERR bit is set.
- The USART receiver moves the invalid data from the receive shift register to the receive data buffer.
- In non-DMA mode, both FERR and RDBF are set at the same time. The latter will generate an interrupt. In DMA mode, an interrupt is generated if the ERRIEN.

When an overrun error occurs:

- The ROERR bit is set.
- The data in the receive data buffer is not lost. The previous data is still available when the USART_DT register is read.
- The content in the receive shift register is overwritten. Afterwards, any data received will be lost.
- An interrupt is generated if the RDBFIEN is set or both ERRIEN and DMAREN are set.

- The ROERR bit is cleared by reading the USART_STS register and then USART_DT register in order.

Note: If ROERR is set, it indicates that at least one piece of data is lost, with two possibilities:

If RDBF=1, it indicates that the last valid data is still stored in the receive data buffer, and can be read.

If RDBF=0, it indicates that the last valid data in the receive data buffer has already been read.

Note: The REN bit cannot be reset during data reception, or the byte that is currently being received will be lost.

12.8.3 Start bit and noise detection

A start bit detection occurs when the REN bit is set. With the oversampling techniques, the USART receiver samples data on the 3rd, 5th, 7th, 8th, 9th and 10th bits to detect the valid start bit and noise. *Table 12-2* shows the data sampling over start bit and noise detection.

Table 12-2 Data sampling over start bit and noise detection

Sampled value (3·5·7)	Sampled value (8·9·10)	NERR bit	Start bit validity
000	000	0	Valid
001/010/100	001/010/100	1	Valid
001/010/100	000	1	Valid
000	001/010/100	1	Valid
111/110/101/011	Any value	0	Invalid
Any value	111/110/101/011	0	Invalid

Note: If the sampling values on the 3rd, 5th, 7th, 8th, 9th, and 10th bits do not match the above mentioned requirements, the USART receiver does not think that a correct start bit is received, and thus it will abort the start bit detection and return to idle state waiting for a falling edge.

The USART receiver has the ability to detect noise. In the non-synchronous mode, the USART receiver samples data on the 7th, 8th and 9th bits, with its oversampling techniques, to distinguish valid data input from noise based on different sampling values, and recover data as well as set NERR (Noise Error Flag) bit.

Table 12-3 Data sampling over valid data and noise detection

Sampled value	NERR bit	Received bit value	Data validity
000	0	0	Valid
001	1	0	Invalid
010	1	0	Invalid
011	1	1	Invalid
100	1	0	Invalid
101	1	1	Invalid
110	1	1	Invalid
111	0	1	Valid

USART is able to receive data under the maximum allowable deviation condition. Its value depends on the DBN bit of the USART_CTRL1 register and the DIV[3: 0] of the USART_BAUDR register.

Note: The maximum allowable deviations stated in the table below are calculated based on 115.2Kbps. The actual deviations may vary with the settings of buad rate. In other words, the greater the buad rate is, the smaller the maximum allowable deviation; in contrast, when the baud rate gets smaller, the maximum allowable deviation will get bigger.

Table 12-4 Maximum allowable deviation

DBN	DIV[3:0] = 0	DIV[3:0] != 0
0	3.75%	3.33%
1	3.41%	3.03%

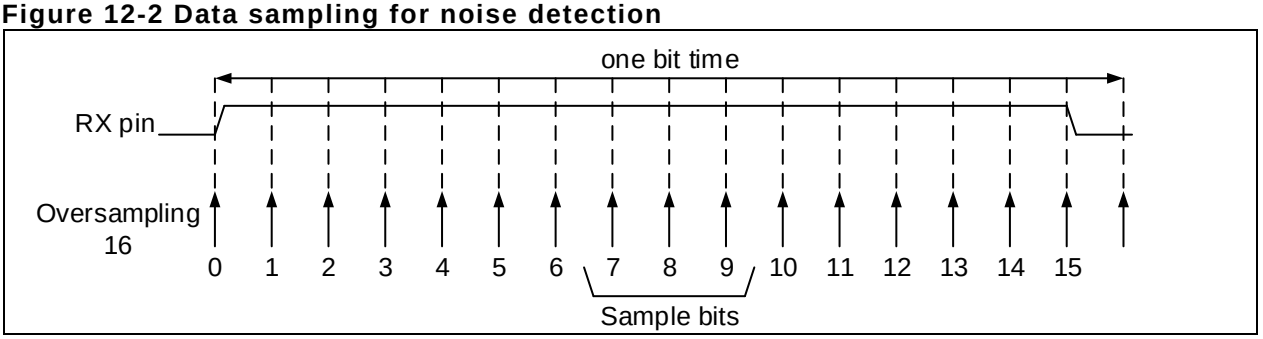
When noise is detected in a data frame:

- The NERR bit is set at the same time as the RDBF bit
- The invalid data is transferred from the receive shift register to the receive data buffer.

- No interrupt is generated in non-DMA mode. However, since the NERR bit is set at the same time as the RDBF bit, the RDBF bit will generate an interrupt. In DMA mode, an interrupt will be issued if the ERRIEN is set.

The NERR bit is cleared by read access to USART_STS register followed by the USART_DT read operation.

Figure 12-2 Data sampling for noise detection



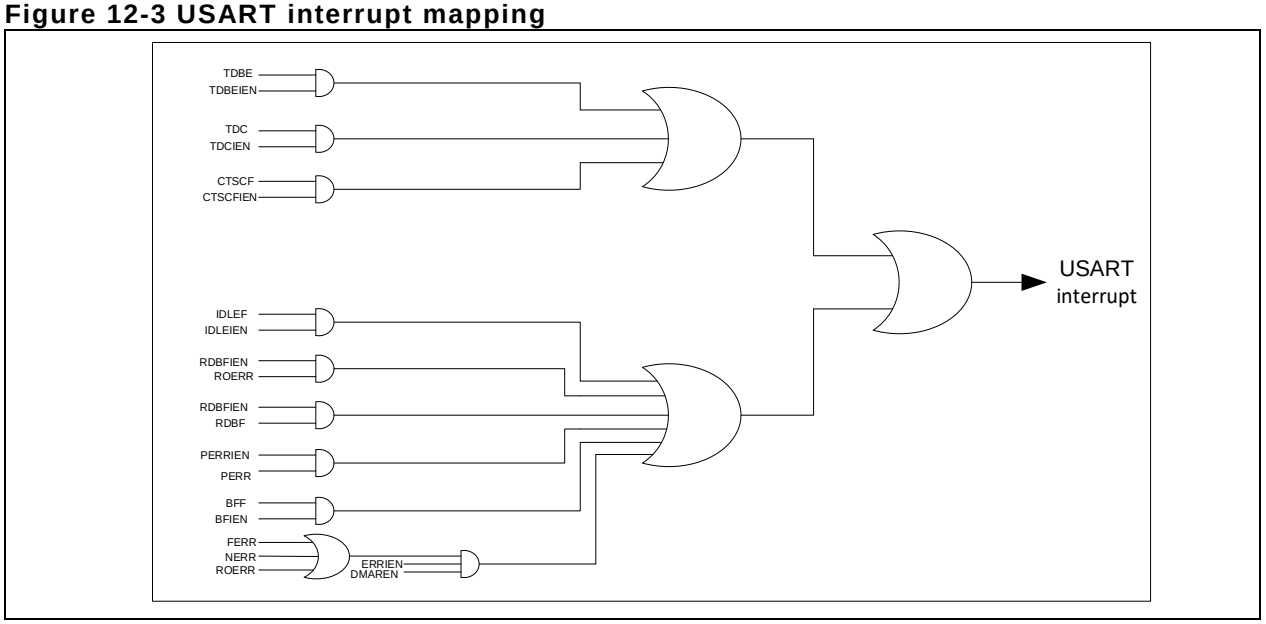
12.9 Interrupt requests

USART interrupt generator serves as a control center of USART interrupts. It is used to monitor the interrupt source inside the USART in real time and the generation of interrupts according to the programmed interrupt control bits. Table 12-4 shows the USART interrupt source and interrupt enable control bit. An interrupt will be generated over an event when the corresponding interrupt enable bit is set.

Table 12-5 USART interrupt request

Interrupt event	Event flag	Enable bit
Transmit data register empty	TDBE	TDBEIEN
CTS flag	CTSCF	CTSCFIEN
Transmit data complete	TDC	TDCIEN
Receive data buffer full	RDBF	RDBFIEN
Receiver overflow error	ROERR	
Idle flag	IDLEF	IDLEIEN
Parity error	PERR	PERRIEN
Break frame flag	BFF	BFIEN
Noise error, overflow error or framing error	NERR or ROERR or FERR	ERRIEN(1)

Figure 12-3 USART interrupt mapping



12.10 I/O pin control

The following five interfaces are used for USART communication.

RX: Serial data input.

TX: Serial data output. In single-wire half-duplex and Smartcard mode, the TX pin is used as an I/O for data transmission and reception.

CK: Transmitter clock output. The output CLK phase, polarity and frequency can be programmable.

CTS: Transmitter input. Send enable signal in hardware flow control mode.

RTS: Receiver output. Send request signal in hardware flow control mode.

12.11 USART registers

These peripheral registers must be accessed by words (32 bits).

Table 12-6 USART register map and reset value

Register	Offset	Reset value
USART_STS	0x00	0x0000 00C0
USART_DT	0x04	0x0000 0000
USART_BAUDR	0x08	0x0000 0000
USART_CTRL1	0x0C	0x0000 0000
USART_CTRL2	0x10	0x0000 0000
USART_CTRL3	0x14	0x0000 0000
USART_GDIV	0x18	0x0000 0000

12.11.1 Status register (USART_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 9	CTSCF	0x0	rw0c	CTS change flag This bit is set by hardware when the CTS status line changes. It is cleared by software. 0: No change on the CTS status line 1: A change occurs on the CTS status line.
Bit 8	BFF	0x0	rw0c	Break frame flag This bit is set by hardware when a break frame is detected. It is cleared by software. 0: Break frame is not detected. 1: Break frame is detected.
Bit 7	TDBE	0x1	ro	Transmit data buffer empty This bit is set by hardware when the transmit data buffer is empty. It is cleared by a USART_DT register write operation. 0: Data is not transferred to the shift register. 1: Data is transferred to the shift register.
Bit 6	TDC	0x1	rw0c	Transmit data complete This bit is set by hardware at the end of transmission. It is cleared by software. (Option 1: read access to USART_STS register followed by a USART_DT write operation; Option 2: Write “0” to this bit) 0: Transmission is not completed. 1: Transmission is completed.
Bit 5	RDBF	0x0	rw0c	Receive data buffer full This bit is set by hardware when the data is transferred from the shift register to the USART_DT register. It is cleared by software. (Option 1: read USART_DT register; Option 2: write “0” to this bit) 0: Data is not received. 1: Data is received.
Bit 4	IDLEF	0x0	ro	Idle flag This bit is set by hardware when an idle line is detected. It is cleared by software. (Read USART_DT register

				followed by a USART_DT read operation) 0: No idle line is detected. 1: Idle line is detected.
Bit 3	ROERR	0x0	ro	Receiver overflow error This bit is set by hardware when the data is received while the RDNE is still set. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No overflow error 1: Overflow error is detected. Note: When this bit is set, the DT register content will not be lost, but the subsequent data will be overwritten.
Bit 2	NERR	0x0	ro	Noise error This bit is set by hardware when noise is detect on a received frame. It is cleared by software. (Read USART_STS register followed by a USART_DT read operation) 0: No noise is detected. 1: Noise is detected.
Bit 1	FERR	0x0	ro	Framing error This bit is set by hardware when a stop bit error (low), excessive noise or break frame is detected. It is cleared by software. USART_STS register followed by a USART_DT read operation) 0: No framing error is detected. 1: Framing error is detected.
Bit 0	PERR	0x0	ro	Parity error This bit is set by hardware when parity error occurs. It is cleared by software. USART_STS register followed by a USART_DT read operation) 0: No parity error occurs. 1: Parity error occurs.

12.11.2 Data register (USART_DT)

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8: 0	DT	0x00	rw	Data value This register provides read and write function. When transmitting with the parity bit enabled, the value written in the MSB bit will be replaced by the parity bit. When receiving with the parity bit enabled, the value in the MSB bit is the received parity bit.

12.11.3 Baud rate register (USART_BAUDR)

Note: If the TE and RE are disabled, the baud counter stops counting.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	DIV	0x0000	rw	Divider This field define the USART divider.

12.11.4 Control register1 (USART_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Forced to be 0 by hardware.
Bit 13	UEN	0x0	rw	USART enable 0: USART is disabled. 1: USART is enable.
Bit 12	DBN	0x0	rw	Data bit num This bit is used to program the number of data bits. 0: 8 data bits 1: 9 data bits

Bit 11	WUM	0x0	rw	Wakeup mode This bit determines the way to wake up silent mode. 0: Waken up by idle line 1: Waken up by ID match
Bit 10	PEN	0x0	rw	Parity enable This bit is used to enable hardware parity control (generation of parity bit for transmission; detection of parity bit for reception). When this bit is enabled, the MSB bit of the transmitted data is replaced with the parity bit; Check whether the parity bit of the received data is correct. 0: Parity control is disabled. 1: Parity control is enabled.
Bit 9	PSEL	0x0	rw	Parity selection This bit selects the odd or even parity after the parity control is enabled. 0: Even parity 1: Odd parity
Bit 8	PERRIEN	0x0	rw	PERR interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 7	TDBEIEN	0x0	rw	TDBE interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 6	TDCIEN	0x0	rw	TDC interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 5	RDBFIEN	0x0	rw	RDBF interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 4	IDLEIEN	0x0	rw	IDLE interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 3	TEN	0x0	rw	Transmitter enable This bit enables the transmitter. 0: Transmitter is disabled. 1: Transmitter is enabled.
Bit 2	REN	0x0	rw	Receiver enable This bit enables the receiver. 0: Receiver is disabled. 1: Receiver is enabled.
Bit 1	RM	0x0	rw	Receiver mute This bit determines if the receiver is in mute mode or not. It is set or cleared by software. When the idle line is used to wake up from mute mode, this bit is cleared by hardware after wake up. When the address match is used to wake up from mute mode, it is cleared by hardware after wake up. When address mismatches, this bit is set by hardware to enter mute mode again. 0: Receiver is in active mode. 1: Receiver is in mute mode.
Bit 0	SBF	0x0	rw	Send break frame This bit is used to send a break frame. It can be set or cleared by software. Generally speaking, it is set by software and cleared by hardware at the end of break

	frame transmission. 0: No break frame is transmitted. 1: Break frame is transmitted.

12.11.5 Control register2 (USART_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 15	Reserved	0x00000	resd	Forced to be 0 by hardware.
Bit 14	LINEN	0x0	rw	LIN mode enable 0: LIN mode is disabled. 1: LIN mode is enabled.
Bit 13: 12	STOPBN	0x0	rw	STOP bit num These bits are used to program the number of stop bits. 00: 1 stop bit 01: 0.5 stop bit 10: 2 stop bits 11: 1.5 stop bits
Bit 11	CLKEN	0x0	rw	Clock enable This bit is used to enable the clock pin for synchronous mode or Smartcard mode. 0: Clock is disabled. 1: Clock is enabled.
Bit 10	CLKPOL	0x0	rw	Clock polarity In synchronous mode or Smartcard mode, this bit is used to select the polarity of the clock output on the clock pin in idle state. 0: Clock output low 1: Clock output high
Bit 9	CLKPHA	0x0	rw	Clock phase This bit is used to select the phase of the clock output on the clock pin in synchronous mode or Smartcard mode. 0: Data capture is done on the first clock edge. 1: Data capture is done on the second clock edge.
Bit 8	LBCP	0x0	rw	Last bit clock pulse This bit is used to select whether the clock pulse of the last data bit transmitted is output on the clock pin in synchronous mode. 0: The clock pulse of the last data bit is no output on the clock pin. 1: The clock pulse of the last data bit is output on the clock pin.
Bit 7	Reserved	0x0	resd	Keep at its default value.
Bit 6	BFIEN	0x0	rw	Break frame interrupt enable 0: Disabled 1: Enabled
Bit 5	BFBN	0x0	rw	Break frame bit num This bit is used to select 11-bit or 10-bit break frame. 0: 10-bit break frame 1: 11-bit break frame
Bit 4	Reserved	0x0	resd	Keep at its default value.
Bit 3: 0	ID	0x0	rw	USART identification Configurable USART ID.

Note: These three bits (CLKPOL, CLKPHA and LBCP) should not be changed while the transmission is enabled.

12.11.6 Control register3 (USART_CTRL3)

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Forced to be 0 by hardware.
Bit 10	CTSCFIEN	0x0	rw	CTSCF interrupt enable 0: Interrupt is disabled. 1: Interrupt is enabled.
Bit 9	CTSEN	0x0	rw	CTS enable 0: CTS is disabled. 1: CTS is enabled.
Bit 8	RTSEN	0x0	rw	RTS enable 0: RTS is disabled. 1: RTS is enabled.
Bit 7	DMATEN	0x0	rw	DMA transmitter enable 0: DMA transmitter is disabled. 1: DMA transmitter is enabled.
Bit 6	DMAREN	0x0	rw	DMA receiver enable 0: DMA receiver is disabled. 1: DMA receiver is enabled.
Bit 5	SCMEN	0x0	rw	Smartcard mode enable 0: Smartcard mode is disabled. 1: Smartcard mode is enabled.
Bit 4	SCNACKEN	0x0	rw	Smartcard NACK enable This bit is used to send NACK when parity error occurs. 0: NACK is disabled when parity error occurs. 1: NACK is enabled when parity error occurs.
Bit 3	SLBEN	0x0	rw	Single-wire bidirectional half-duplex enable 0: Single-wire bidirectional half-duplex is disabled. 1: Single-wire bidirectional half-duplex is enabled.
Bit 2	IRDALP	0x0	rw	IrDA low-power mode This bit is used to configure IrDA low-power mode. 0: IrDA low-power mode is disabled. 1: IrDA low-power mode is enabled.
Bit 1	IRDAEN	0x0	rw	IrDA enable 0: IrDA is disabled. 1: IrDA is enabled.
Bit 0	ERRIEN	0x0	rw	Error interrupt enable An interrupt is generated when a framing error, overflow error or noise error occurs. 0: Error interrupt is disabled. 1: Error interrupt is enabled.

12.11.7 Guard time and divider register (USART_GDIV)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced 0 by hardware.
Bit 15: 8	SCGT	0x00	rw	Smartcard guard time value This field specifies the guard time value. The transmission complete flag is set after this guard time in smartcard mode.
Bit 7: 0	ISDIV	0x00	rw	IrDA/smartcard division In IrDA mode: 8 bit [7: 0] is valid. It is valid in common mode and must be set to 00000001. In low-power mode, it divides the peripheral clock to serve as the period base of the pulse width; 00000000: Reserved–Do not write. 00000001: Divided by 1 00000010: Divided by 2 …… Smartcard mode: The lower 5 bit [4: 0] is valid. This division is used to divide the peripheral clock to provide clock for the Smartcard. Configured as follows: 00000: Reserved–Do not write. 00001: Divided by 2 00010: Divided by 4 00011: Divided by 6 ……

13 Serial peripheral interface (SPI)

13.1 SPI introduction

The SPI interface supports either the SPI protocol or the I²S protocol, depending on software configuration. This chapter gives an introduction of the main features and continuation procedure of SPI used as SPI or I²S.

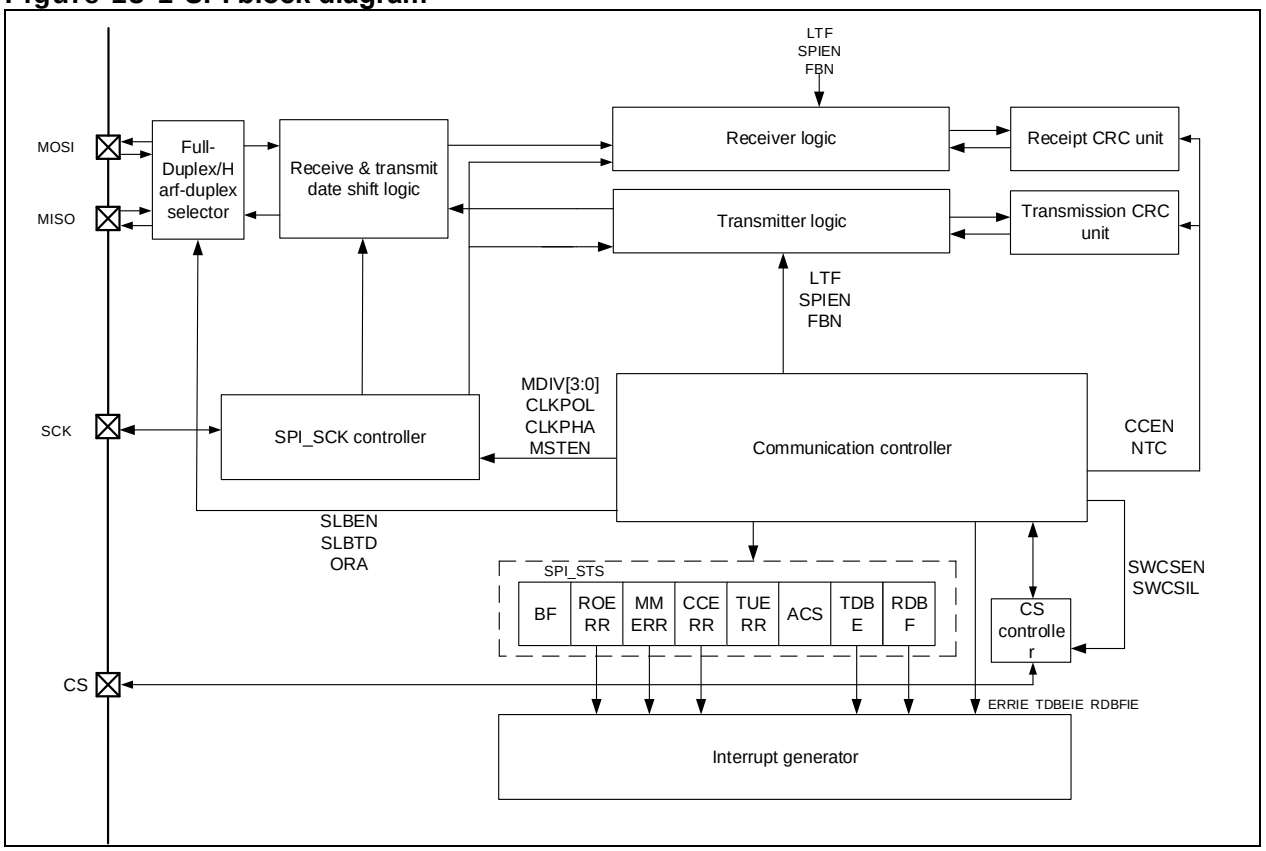
13.2 Functional overview

13.2.1 SPI description

The SPI can be configured as host or slave based on software configuration, supporting full-duplex, reception-only full-duplex and transmission-only/reception-only half-duplex modes, DMA transfer, and automatic CRC function of SPI internal hardware.

SPI block diagram:

Figure 13-1 SPI block diagram



Main features as SPI:

- Full-duplex or half-duplex communication
  - Full-duplex synchronous communication (supporting reception-only mode to release IO for transmission)
  - Half-duplex synchronous communication (transfer direction is configurable: receive or transmit)
- Master or slave mode
- CS signal processing mode
  - CS signal processing by hardware
  - CS signal processing by software
- 8-bit or 16-bit frame format
- Communication frequency and prescalers (Frequency up to 48M, and prescalers up to $f_{PCLK}/2$)
- Programmable clock polarity and phase

- Programmable data transfer order (MSB-first or LSB-first)
- Programmable error interrupt flags (receiver overflow error, master mode error and CRC error)
- Programmable transmit data buffer empty interrupt and receive data buffer full interrupt
- Support transmission and reception using DMA
- Support hardware CRC transmission and error checking
- Busy status flag

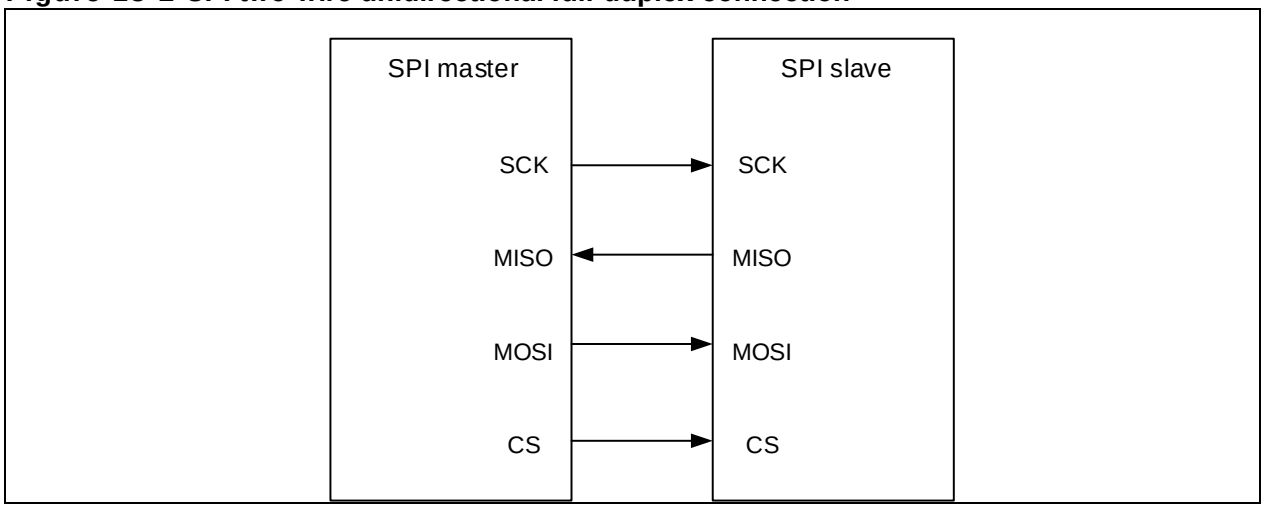
13.2.2 Full-duplex/half-duplex selector

When used as an SPI interface, it supports four synchronous modes: two-wire unidirectional full-duplex, single-wire unidirectional receive only, single-wire bidirectional half-duplex transmit and single-wire bidirectional half-duplex receive.

Figure 13-2 shows the two-wire unidirectional full-duplex mode and SPI IO connection:

The SPI operates in two-wire unidirectional full-duplex mode when the SLBEN bit and the ORA bit is both 0. In this case, the SPI supports data transmission and reception at the same time. IO connection is as follows:

Figure 13-2 SPI two-wire unidirectional full-duplex connection



In either master or slave mode, it is required to wait until the RDBF bit and TDBE bit is set, and BF=0 before disabling the SPI or entering power-saving mode (or disabling SPI system clock).

Figure 13-3 shows the single-wire unidirectional receive-only mode and SPI IO connection

The SPI operates in single-wire unidirectional receive-only mode when the SLBEN is 0 and the ORA is set. In this case, the SPI can be used only for data reception (transmission is not supported). The MISO pin transmits data in slave mode and receives data in master mode. The MOSI pin transmits data in master mode and receives data in salve mode.

Figure 13-3 Single-wire unidirectional receive only in SPI master mode

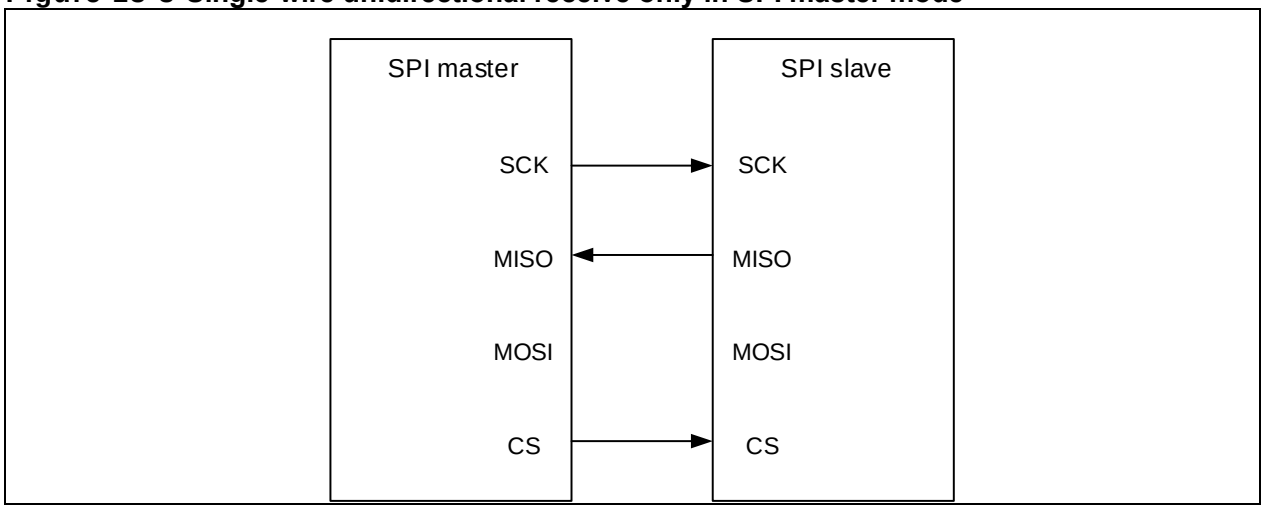
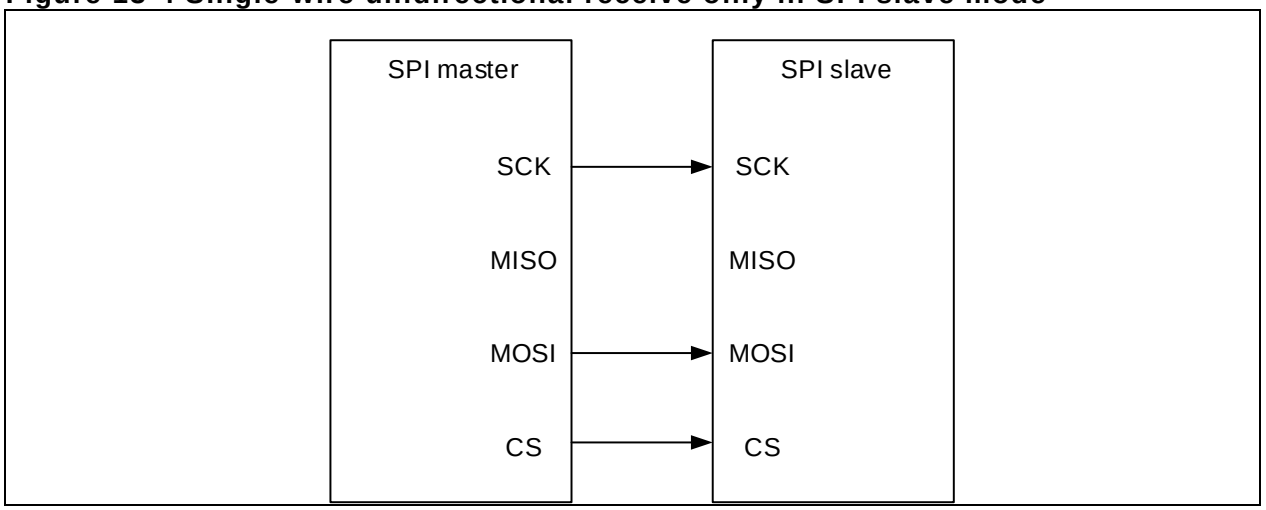


Figure 13-4 Single-wire unidirectional receive only in SPI slave mode



In master mode, it is necessary to wait until the second-to-last RDBF bit is set and then another SPI_CPK period before disabling the SPI. The last RDBF must be set before entering power-saving mode (or disabling SPI system clock).

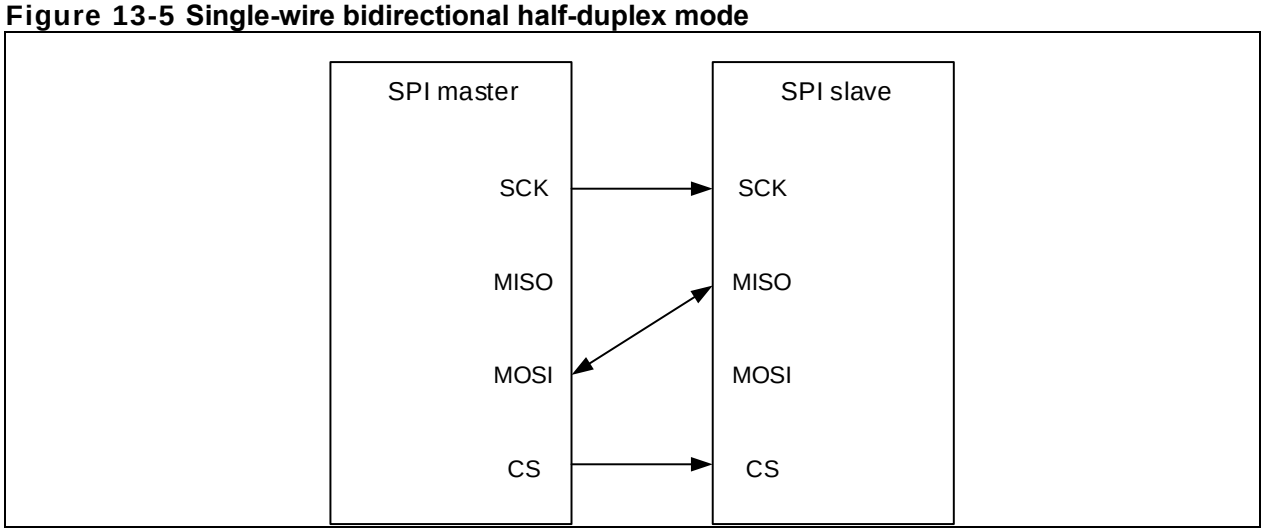
In slave mode, there is no need to check any flag before disabling the SPI. However, it is required to wait until the BF becomes 0 before entering power-saving mode.

Figure 13-5 shows single-wire bidirectional half-duplex mode and SPI IO connection

When the SLBEN is set, the SPI operates in single-wire bidirectional half-duplex mode. In this case, the SPI supports data reception and transmission alternately. In master mode, the MOSI pin transmits or receives data in master mode, while the MISO pin is released. In slave mode, the MISO pin transmits or receives data, but the MOSI pin is released.

The SLBTD bit is used by software to configure transfer direction. When the SLBTD bit is set, the SPI can be used only for data transmission; when the SLBTD bit is 0, the SPI can be used only for data reception.

Figure 13-5 Single-wire bidirectional half-duplex mode



When the SPI is selected for data transmission in single-wire bidirectional half-duplex mode (master or slave), the TDBE bit must be set, and the BF must be 0 before disabling the SPI. The power-saving mode (or disabling SPI system clock) cannot be entered unless the SPI is disabled.

In master mode, when the SPI is selected for data reception in single-wire bidirectional half-duplex mode, it is required to wait until the second-to-last RDBF is set and then another SPI_SCK period before disabling the SPI. And the last RDBF must be set before entering power-saving mode (or disabling SPI system clock).

In slave mode, when the SPI is selected for data reception in single-wire bidirectional half-duplex mode, there is no need to check any flags before disabling the SPI. However, the BT must be 0 before entering power-saving mode (or disabling SPI system clock).

13.2.3 Chip select controller

The Chip select controller (CS) is used to enable hardware or software control for chip select signals through software configuration. This controller is used to select master/slave device in multi-processor mode, and to avoid conflicts on the data lines by enabling the SCK signal output followed by CS signal. The hardware and software configuration procedure is detailed as follows, along with their respective input/output in master and slave mode.

CS hardware configuration procedure:

In master mode with CS being as an output, HWCSOE=1, SWCSEN=0, the CS hardware control is enabled. If the SPI is enabled, low level is output on the CS pin. The CS signal is then released after the SPI is disabled and the transmission is complete.

In master mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. At this point, the SPI is automatically disabled by hardware and enters slave mode as soon as the CS pin low is detected by master SPI. The mode error flag (MMERR bit) is set at the same time. An interrupt is generated if ERRIE=1. When the MMERR is set, the SPIEN and MSTEN cannot be set by software. The MMERR is cleared by read or write access to the SPI_STS register followed by write operation to the SPI_CTRL1 register.

In slave mode with CS being as an input, HWCSOE=0, SWCSEN=0, the CS hardware control is enabled. The slave selects whether to transmit / receive data based on the level on the CS pin. The slave is selected for data reception and transmission only when the CS pin is low.

CS software configuration procedure:

In master mode with CS being as an input, SWCSEN=1, the CS software control is enabled. When SWCSIL=0, the SPI is automatically disabled by hardware and enters slave mode. The mode error flag (MMERR bit) is set at this time. An interrupt is generated if ERRIE=1. When the MMERR bit is set, the SPIEN and MSTEN bits cannot be set by software. The MMERR bit is cleared by read or write access to the SPI_STS register followed by write operation to the SPI_CTRL1 register.

In slave mode with CS being as an input, SWCSEN=1, the CS software control is enabled. The SPI judges the CS signal with the SWCSIL bit, instead of CS pin. When SWCSIL=0, the slave is selected for data reception and transmission.

13.2.4 SPI_SCK controller

The SPI protocol adopts synchronous transmission. In master mode with the SPI being used as SPI, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by peripherals, and is input to the SPI via IO. In all, the SPI_SCK controller is used for the generation and distribution of SPI_SCK, with the configuration procedure detailed as follows:

SPI_SCK controller configuration procedure:

- Clock polarity and clock phase selection: It is selected by setting the CLKPOL and CLKPHA bit.
- Clock prescaler selection: Select the desired PCLK frequency by setting the CRM bit. Select the desired prescaler by setting the MDIV[3: 0] bit.
- Master/slave selection: Select SPI as master or slave by setting the MSTEN bit.

Note that the clock output is activated after the SPI is enabled in master reception-only mode, and it remains output until when the SPI is disabled and the reception is complete.

13.2.5 CRC introduction

There is an independent transmission and reception CRC calculation unit in the SPI. When used as SPI through software configuration, the SPI enables CRC calculation and CRC check automatically while the user is reading or writing through DMA or CPU. During the transmission, if the received data is not consistent with, detected by hardware, the data in the SPI_RCRC register, and such data is exactly the CRC value, then the CCERR bit will be set. An interrupt is generated if ERRIE=1.

The CRC function and configuration procedure of the SPI are described as follows.

CRC configuration procedure

- CRC calculation polynomial is configured by setting the SPI_CPOLY register.
- CRC enable: The CRC calculation is enabled by setting the CCEN bit. This operation will reset the SPI_RCRC and SPI_TCRC registers.
- Select if or when the NTC bit is set, depending on DMA or CPU data register. See the following descriptions.

Transmission using DMA

When DMA is used to write the data to be transmitted, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI_CPOLY register and each transmitted data, and sends the CRC value at the end of the last data transmission. This result is regarded as the value of the SPI_TCRC register.

Reception using DMA

When DMA is used to read the data to be received, if the CCEN bit is enabled, the hardware calculates the CRC value automatically according to the value in the SPI_CPOLY register and each received data, and waits until the completion of CRC data reception at the end of the last data reception before comparing the received CRC value with the value of the SPI_RCRC register. If check error occurs, the CCERR flag is set. An interrupt is generated if the ERRIE bit is enabled.

Transmission using CPU

Unlike DMA mode, after writing the last data to be transmitted, the CPU mode requires the NTC bit to be set by software before the end of the last data transmission.

Reception using CPU

In two-wire unidirectional full-duplex mode, follow CPU transmission mode to operate the NTC bit, the CRC calculation and check in CPU reception mode will be completed automatically.

In single-wire unidirectional reception-only mode and single-wire bidirectional reception-only mode, it is required to set the NTC bit before the software receives the last data when the second-to-last data is received.

13.2.6 DMA transfer

The SPI supports write and read operations with DMA. Refer to the following configuration procedure.

Special attention should be paid to: when the CRC calculation and check is enabled, the number of data transferred by DMA is configured as the number of the data to be transferred. The number of data read with DMA is configured as the number of the data to be received. In this case, the hardware will send CRC automatically at the end of full transfer, and the receiver will also perform CRC check. Note that the received CRC data will be moved into the SPI_DT register by hardware, with the RDBF being set, and the DMA read request will be sent if then DAM transfer is enabled. Hence, it is recommended to read the SPI_DT register to get the CRC value at the end of CRC reception in order to avoid the upcoming transfer error.

Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.

- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

13.2.7 Transmitter

The SPI transmitter is clocked by SPI_SCK controller. It can output different data frame formats, depending on software configuration. There is a SPI_DT register available in the SPI that is used to be written with the data to be transmitted. When the transmitter is clocked, the contents in the SPI_DT register are copied into the data buffer (Unlike SPI_DT, it is driven by SPI_SCK, and controlled by hardware, instead of software), and sent out in order based on the programmed frame format.

Both DMA and CPU can be used for write operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the TDBE bit. The reset value of this bit is 1, indicating that the SPI_DT register is empty. If the TDBEIE bit is set, an interrupt is generated. After the data is written, the TDBE is pulled low until the data is moved to the transmit data buffer before the TDBE is set once again. This means that the user can be allowed to write the data to be transmitted only when the TDBE is set.

After the transmitter is configured and the SPI is enabled, the SPI is ready for data transmission. Before going forward, it is necessary for the users to refer to full-duplex / half-duplex chapter to get detailed configuration information, go to the Chip select controller chapter for specific chip select mode, check the SPI_SCK controller chapter for information on communication clock, and refer to CRC and DMA transfer chapter to configure CRC and DMA (if necessary). The recommended configuration procedure are as follows.

Transmitter configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable transmit data interrupt (TDBEIE =1) through the TDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN

13.2.8 Receiver

The SPI receiver is clocked by the SPI_SCK controller. It can output different data frame formats through software configuration. There is a receive data buffer register, driven by the SPI_SCK, in the SPI receiver.

At the last CLK of each transfer, the data is moved from the shift register to the receive data buffer register. Then the transmitter sets the receive data complete flag to the SPI logic. When the flag is detected by the SPI logic, the data in the receive data buffer is copied into the SPI_DT register, with the RDBF being set. This means that the data is received, and it is already stored into the SPI_DT. In this case, read access to the SPI_DT register will clear the RDBF bit.

Both DMA and CPU can be used for read operation. For DMA transfer, refer to DMA transfer section for more details. For CPU transfer, attention should be paid to the RDBE bit. The reset value of this bit is 0, indicating that the SPI_DT register is empty. If the data is received and moved into the SPI_DT, the RDBF is set, meaning that there are some data to be read in the SPI_DT register. An interrupt is

generated if the RDBFIE bit is set.

When the next received data is ready to be moved to the SPI_DT register, if the previous received data is still not read (RDBF=1), then the data overflow occurs. The previous receive data is not lost, but the next received data will do. At this point, the ROERR is set. An interrupt is generated if the ERRIE is set. Read SPI_DT register and then the SPI_STS register will clear the ROERR bit. The recommended configuration procedure is as follows.

Receiver configuration procedure:

- Configure full-duplex/half-duplex selector
- Configure chip select controller
- Configure SPI_SCK controller
- Configure CRC (if necessary)
- Configure DMA transfer (if necessary)
- If the DMA transfer mode is not used, the software will check whether to enable receive data interrupt (RDBEIE =1) through the RDBE bit.
- Configure frame format: select MSB/LSB mode with the LTF bit, and select 8/16-bit data with the FBN bit
- Enable SPI by setting the SPIEN

13.2.9 Motorola mode

This section describes the SPI communication timings, which includes full-duplex and half-duplex master/slave timings.

Full-duplex communication – master mode

Configured as follows:

MSTEN=1: Master enable

SLBEN=0: Full-duplex mode

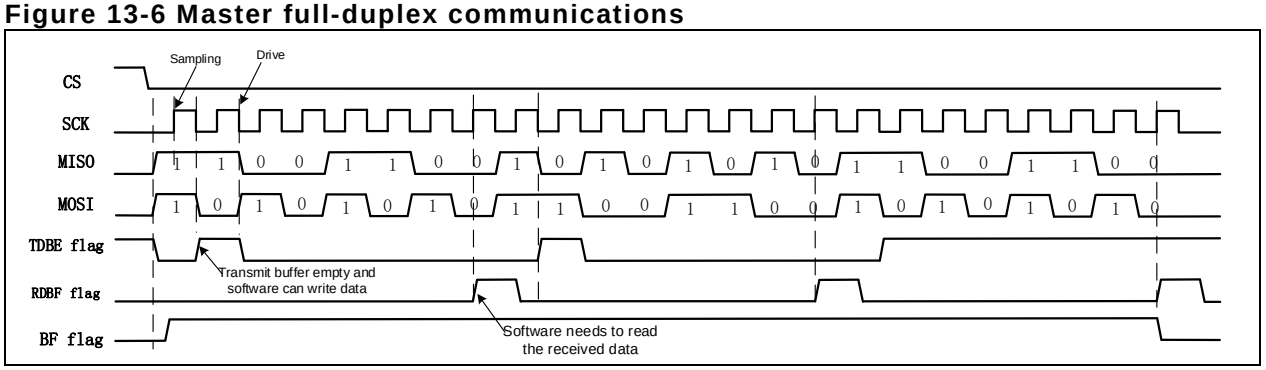
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

Figure 13-6 Master full-duplex communications



Full-duplex communication – slave mode

Configured as follows:

MSTEN=0: Slave enable

SLBEN=0: Full-duplex mode

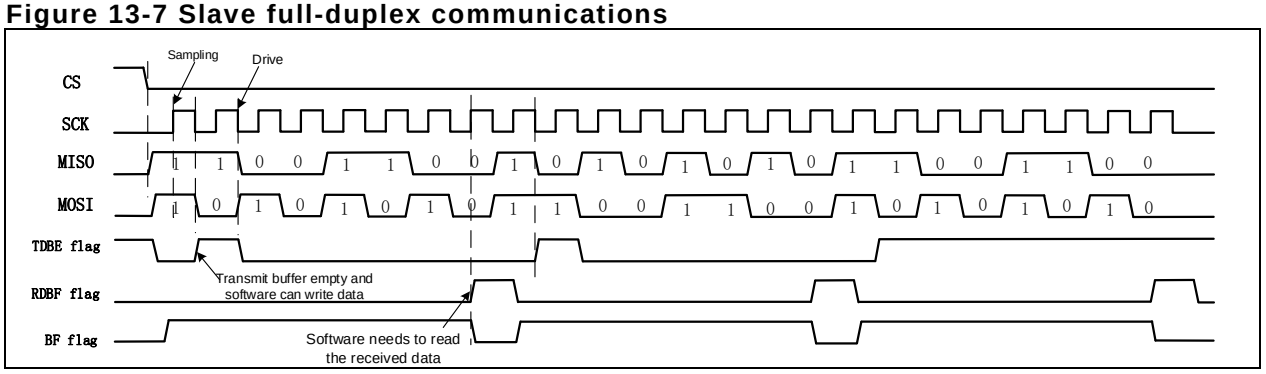
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Slave transmit (MISO): 0xcc, 0xaa, 0xcc

Figure 13-7 Slave full-duplex communications



Half-duplex communication – master transmit

Configured as follows:

MSTEN=1: Master enable

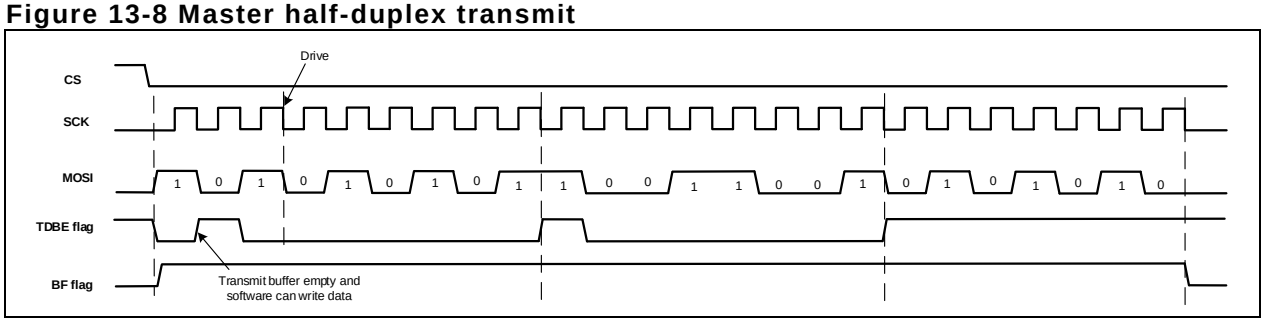
SLBEN=1: Single line bidirectional mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Master transmit (MOSI): 0xaa, 0xcc, 0xaa

Figure 13-8 Master half-duplex transmit



Half-duplex communication – slave receive

Configured as follows:

MSTEN=0: Slave enable

SLBEN=1: Single line bidirectional mode

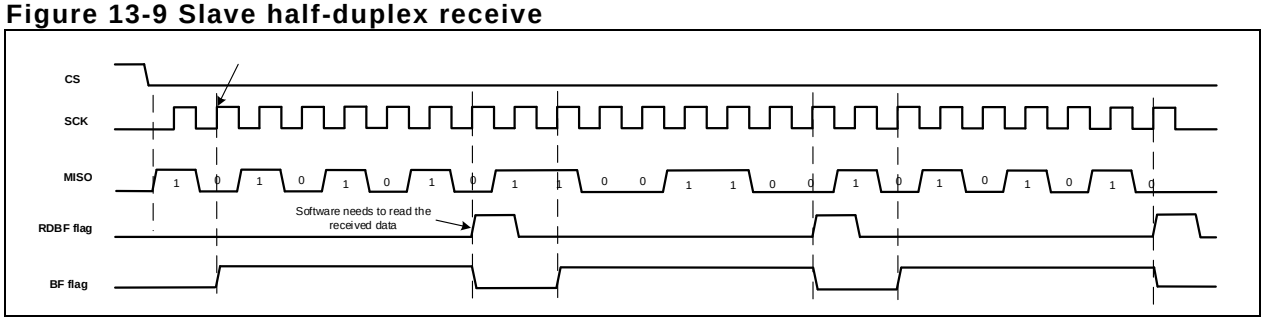
SLBTD=0: Receive mode

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave receive: 0xaa, 0xcc, 0xaa

Figure 13-9 Slave half-duplex receive



Half-duplex communication – slave transmit

Configured as follows:

MSTEN=0: Slave enable

SLBEN=1: Single line bidirectional mode

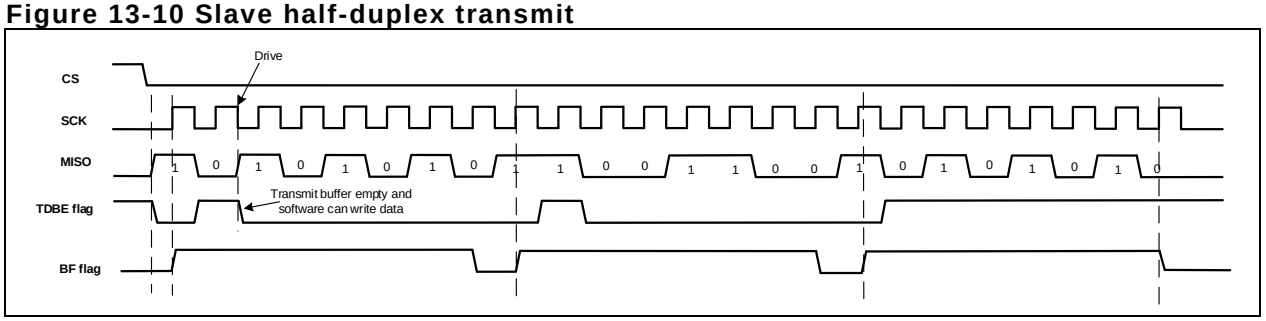
SLBTD=1: Transmit enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

Slave transmit: 0xaa, 0xcc, 0xaa

Figure 13-10 Slave half-duplex transmit



Half-duplex communication – master receive

Configured as follows:

MSTEN=1: Master enable

SLBEN=1: Single line bidirectional mode

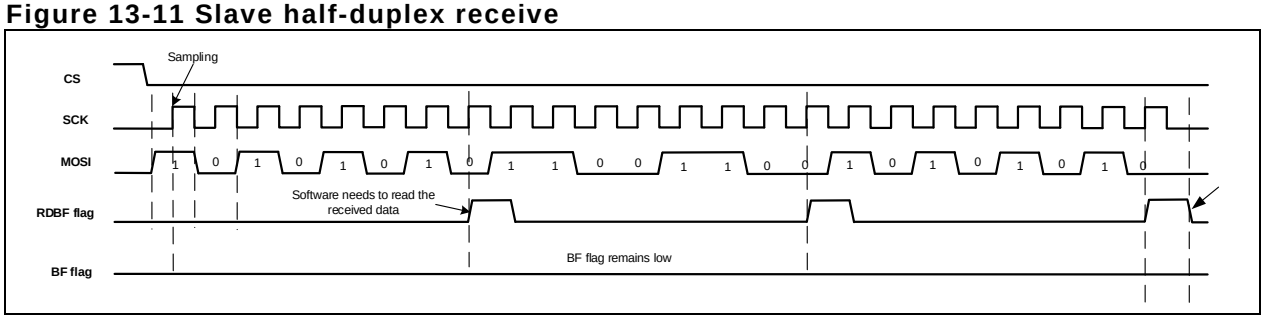
SLBTD=0: Receive enable

CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

FBN=0: 8-bit frame

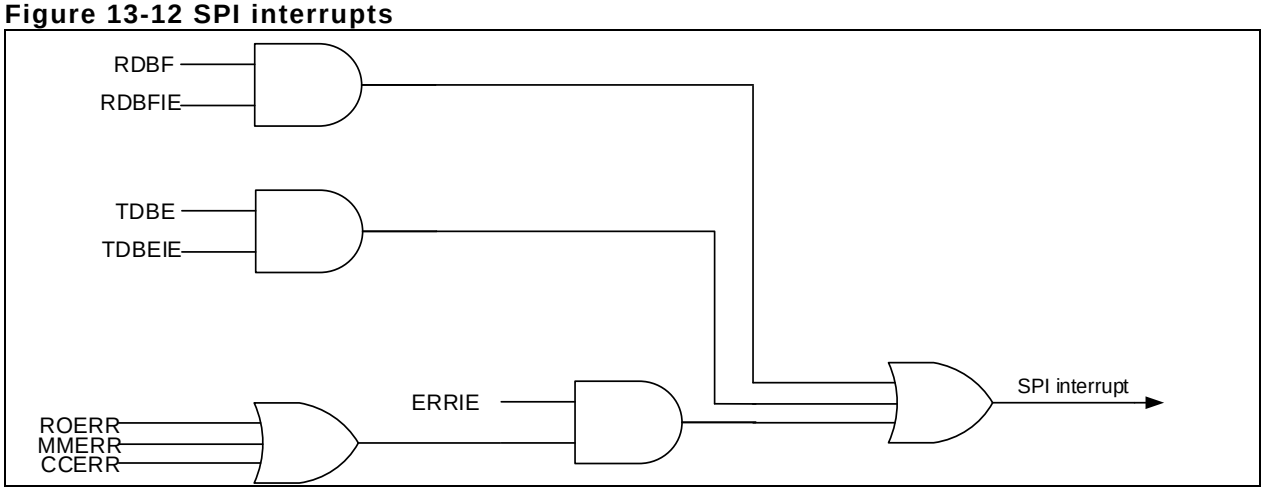
Master receive: 0xaa, 0xcc, 0xaa

Figure 13-11 Slave half-duplex receive



13.2.10 Interrupts

Figure 13-12 SPI interrupts



13.2.11 IO pin control

Usually, the SPI is connected to external devices through four pins.

- MISO: Master In/Slave Out. The pin receives data in master mode, and transmits data in slave mode.
- MOSI: Master Out/Slave In. The pin transmits data in master mode, and receives data in slave mode.
- SCK: SPI communication clock. The pin serves as output in master mode, and input in slave mode.
- CS: Chip Select. This is an optional pin which selects master/slave mode.

13.2.12 Precautions

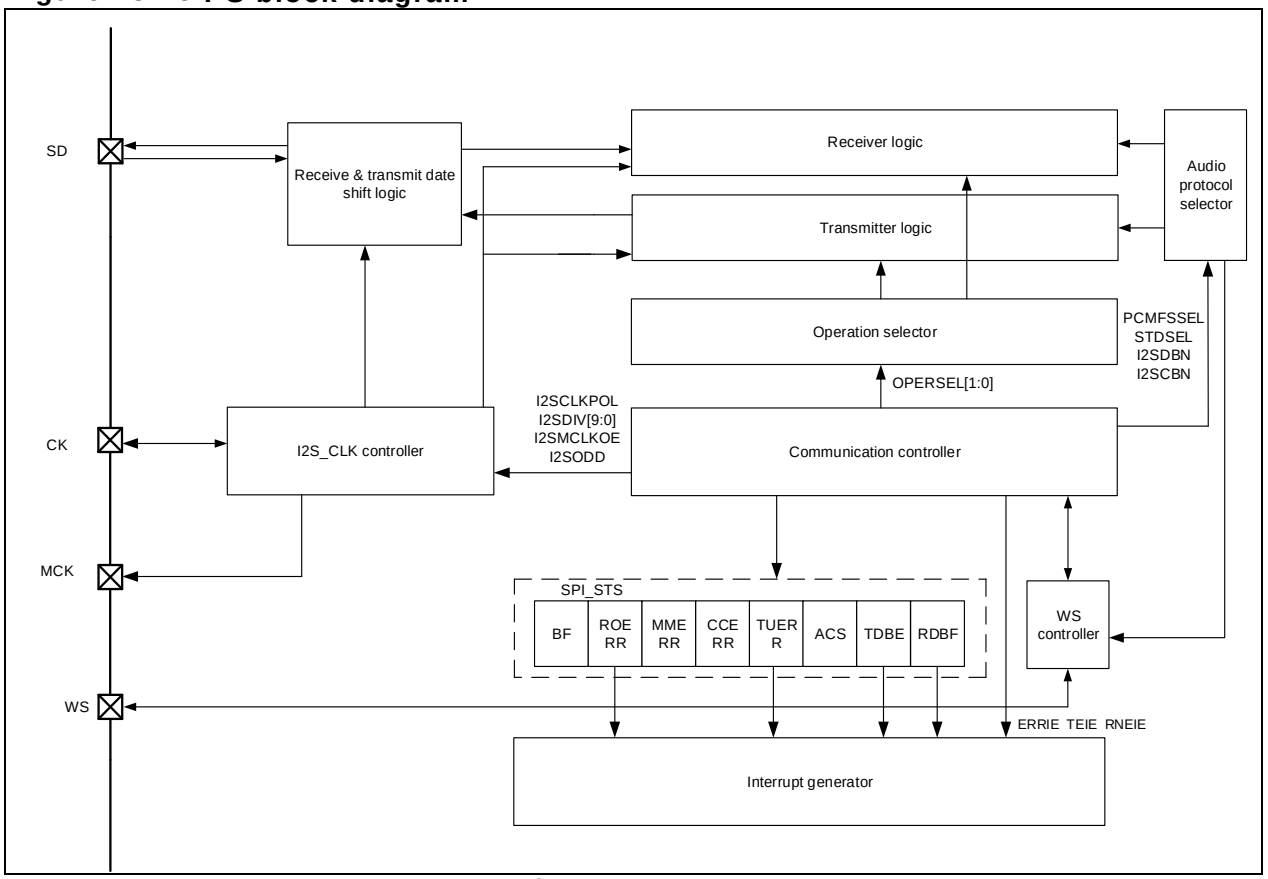
CRC value is obtained by software reading DT register at the end of CRC reception

13.3 I²S functional description

13.3.1 I²S introduction

The I²S can be configured by software as master reception/transmission, and slave reception/transmission, supporting four kinds of audio protocols including Philips standard, MSB-aligned standard, LSB-aligned standard and PCM standard, respectively. The DMA transfer is also supported.

Figure 13-13 I²S block diagram



Main features when the SPI is used as I²S:

- Programmable operation mode
  - Slave device transmission
  - Slave device reception
  - Master device transmission
  - Master device reception
- Programmable clock polarity
- Programmable clock frequency (8 KHz to 192 KHz)

- Programmable data bits (16 bit, 24 bit, 32 bit)
- Programmable channel bits (16 bit, 24 bit)
- Programmable audio protocol
  - I²S Philips standard
  - MSB-aligned standard (left-aligned)
  - LSB-aligned standard (right-aligned)
  - PCM standard (long or short frame)
- DMA transfer

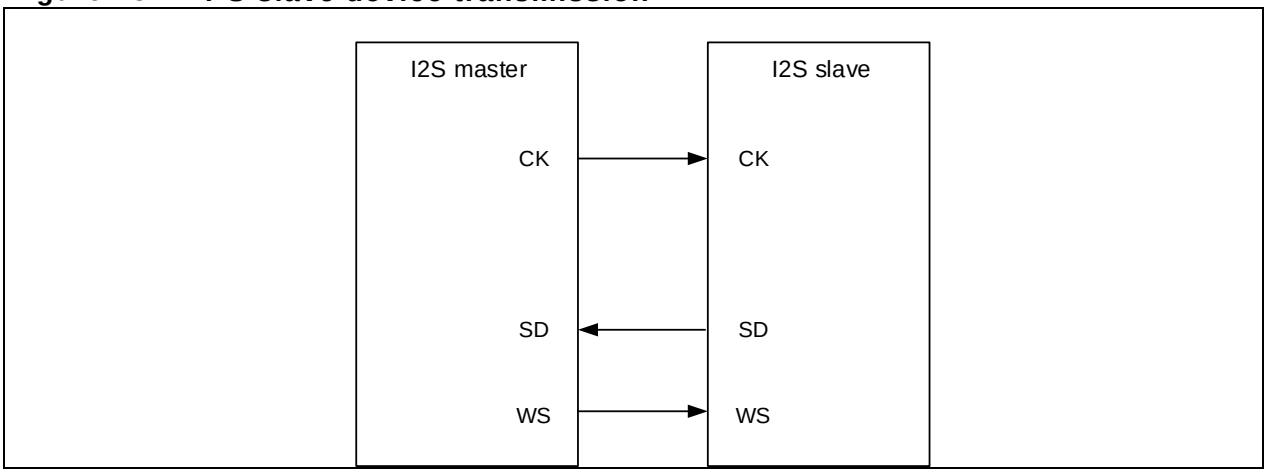
13.3.2 Operation mode selector

The SPI, used as I²S selector, offers multiple operation modes for selection, namely, slave device transmission, slave device reception, master device transmission and master device reception. This is done by software configuration.

Slave device transmission:

Set the I2SMSEL bit, and OPERSEL[1:0] =00, the I2S will work in slave device transmission mode.

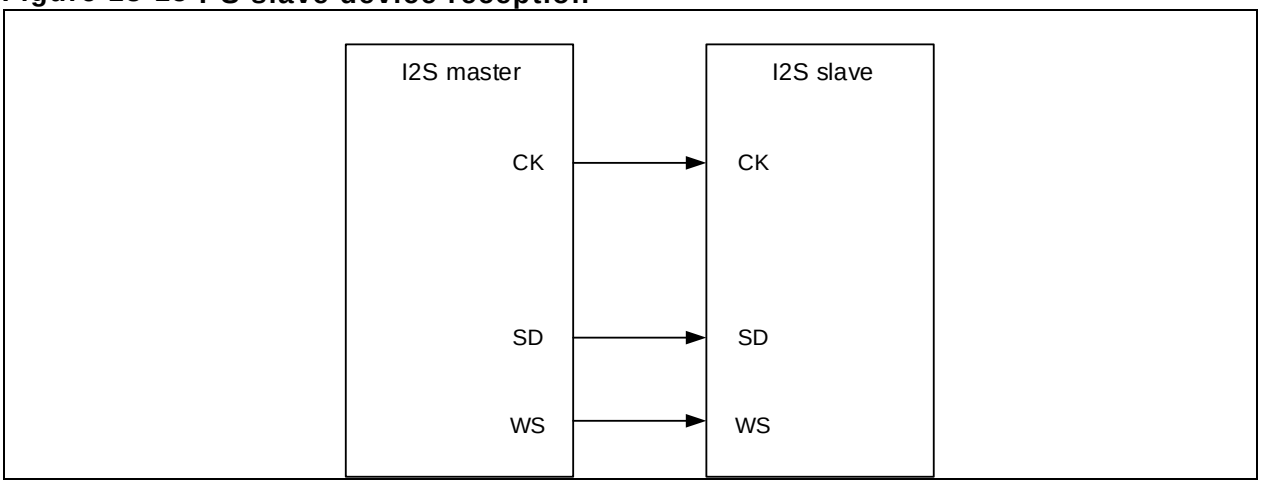
Figure 13-14 I²S slave device transmission



Slave device reception:

Set the I2SMSEL bit, and OPERSEL[1:0]=01, the I²S will work in slave device reception mode.

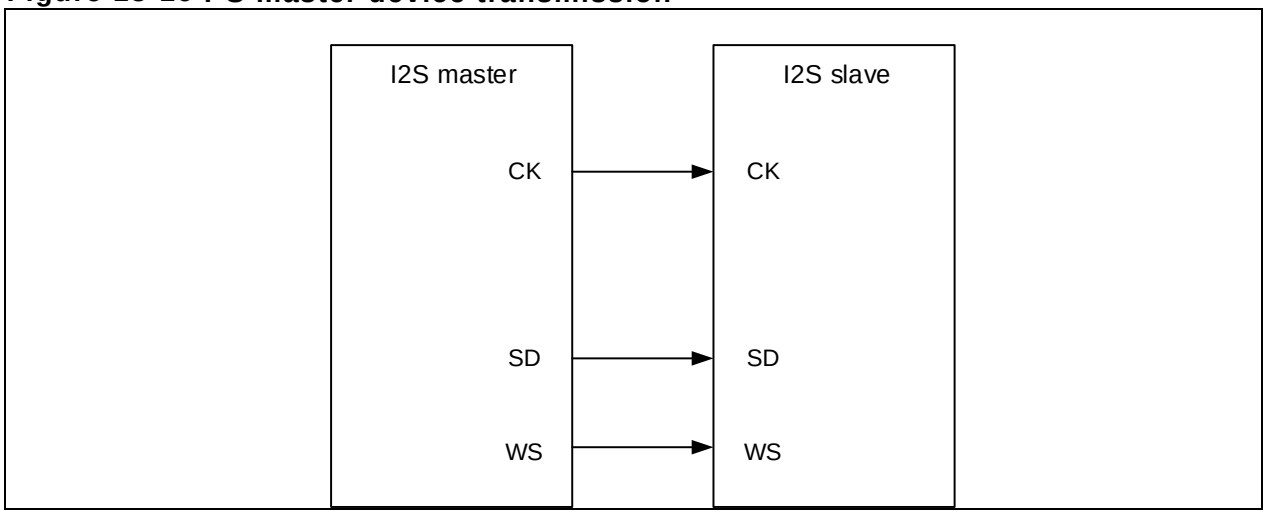
Figure 13-15 I²S slave device reception



Master device transmission:

Set the I2SMSEL bit, and OPERSEL[1:0]=10, the I²S will work in master device transmission mode.

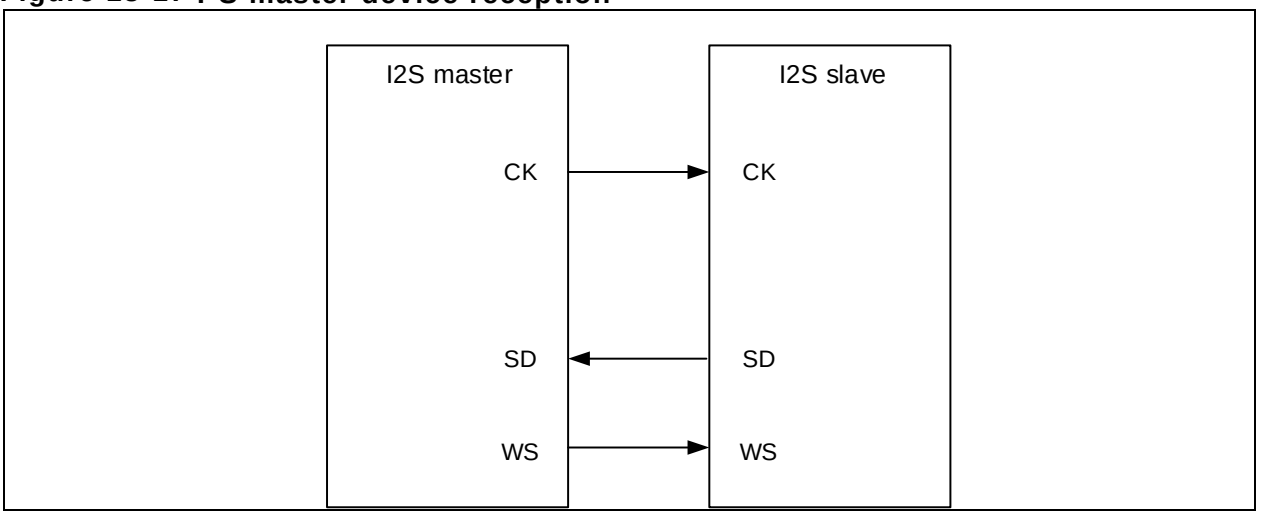
Figure 13-16 I²S master device transmission



Master device reception:

Set the I2SMSEL bit, and OPERSEL[1: 0]=11, the I²S will work in master device reception mode.

Figure 13-17 I²S master device reception



13.3.3 Audio protocol selector

While being used as I²S, the SPI supports multiple audio protocols. The user can control the audio protocol selector through software configuration to select the desired audio protocol, with the data bits and channel bits being controlled by the audio protocol selector. Besides, the user can also select the data bits and channel bits through software configuration. Meanwhile, the audio protocol selector manages the WS controller, output or detect the WS signal that meets the protocol requirements.

- Select audio protocol by setting the STDSEL bit
  STDSLE=00: Philips standard
  STDSLE=01: MSB-aligned standard (left-aligned)
  STDSLE=10: LSB-aligned standard (right-aligned)
  STDSLE=11: PCM standard
- Select PCM frame synchronization format: PCMFSSEL=1 for PCM long frame synchronization, PCMFSSEL=0 for short frame synchronization (this step is required when selecting PCM protocol)
- Select data bits by setting the I2SDBN bit
  I2SDBN=00: 16 bit
  I2SDBN =01: 24 bit
  I2SDBN =10: 32 bit

- Select channel bits by setting the I2SCBN bit
  I2SDBN =0: 16 bit
  I2SDBN =1: 32 bit

Note: Read/Write operation mode depends on the selected audio protocols, data bits and channel bits. The following lists all possible configuration combinations and their respective read and write operation mode.

- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 16-bit data and 16-bit channel

  The data bit is the same as the channel bit. Each channel requires one read/write operation from/ to the SPI_DT register, and the number of DMA transfer is 1.
- Philips standard, PCM standard or MSB-aligned standard, 16-bit data and 32-bit channel

  The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The first 16 bits (MSB) are the significant bits, and the 16-bit LSB is forced to 0 by hardware.
- Philips standard, PCM standard or MSB-aligned standard, 24-bit data and 32-bit channel

  The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. The 16-bit MSB transmits and receives the first 16-bit data, the 16-bit LSB transmits and receives the 8-bit MSB data, with 8-bit LSB data being forced to 0 by hardware.
- Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 32-bit data and 32-bit channel

  The data bit is the same as the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. These 32-bit data are proceeded in two times, with 16-bit data each time.
- LSB-aligned standard, 16-bit data and 32-bit channel

  The data bit is different from the channel bit. Each channel requires one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The 16 bits (LSB) are the significant bits while the first 16-bit data (MSB) are forced to 0 by hardware.
- LSB-aligned standard, 24-bit data and 32-bit channel
  The data bit is different from the channel bit. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. For the first 16-bit data, its 8-bit LSB are the significant bits, with the 8-bit MSB forced to 0 by hardware; the subsequent 16 bits transmit and receive the second 16-bit data.

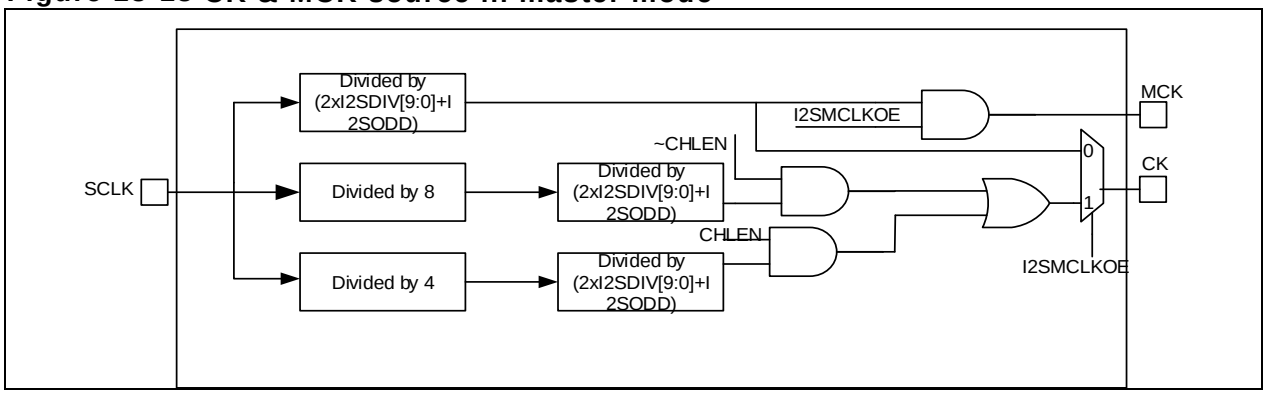
13.3.4 I2S_CLK controller

The audio protocols the SPI supports adopts synchronous transmission. In master mode, it is required to generate a communication clock for data reception and transmission on the SPI, and the communication clock should be output to the slave via IO for data reception and transmission. In slave mode, the communication clock is provided by master, and is input to the SPI via IO. In all, the I2S_SCK controller is used for the generation and distribution of I2S_SCK, with the configuration procedure detailed as follows:

When used as I^2S master, the SPI can provide communication clock (CK) and main peripheral clock (MCK) shown in Figure 13-18. The CK and MCK are generated by HCLK divider, with the prescaler of the MCK determined by I2SDIV and I2SODD. The calculation formula is seen in Figure 13-18.

The prescaler of the CK depends on whether to provide the main clock for peripherals. To ensure that the main clock is always 256 times larger than the audio sampling frequency, The channel bits should be taken into account. When the main clock is needed, the CK should be divided by 8 (I2SCBN=0) or 4 (I2SCBN=1), then divided again by the same prescaler as that of the MCK, that is the final communication clock; When the main clock is not needed, the prescaler of the CK is determined by I2SDIV and I2SODD, shown in Figure 13-18.

Figure 13-18 CK & MCK source in master mode



Apart from the above-mentioned configuration, the following table lists the values of I2SDIV and I2SODD corresponding to some specific frequencies, as well as their respective error for the users to configure the I2SDIV and I2SODD.

Table 13-1 Audio frequency precision using system clock

SCLK (MHz)	MCLK	Target Fs (Hz)	16bit				32bit			
			I2S DIV	I2S_ODD	RealFs	Error	I2S DIV	I2S_ODD	RealFs	Error
150	NO	192000	12	0	1953125	1.73%	6	0	1953125	1.73%
150	NO	96000	24	1	95663	0.35%	12	0	97656	1.73%
150	NO	48000	49	0	47831	0.35%	24	1	47831	0.35%
150	NO	44100	53	0	44221	0.28%	26	1	442217	0.28%
150	NO	32000	73	0	32106	0.33%	36	1	32106	0.33%
150	NO	22050	106	1	22007	0.19%	53	1	22110	0.28%
150	NO	16000	146	1	15998	0.01%	73	0	16053	0.33%
150	NO	11025	212	1	11029	0.04%	106	1	11003	0.19%
150	NO	8000	293	0	7999	0.01%	146	1	7999	0.01%
150	YES	96000	3	0	97656	1.73%	3	0	97656	1.73%
150	YES	48000	6	0	48828	1.73%	6	0	48828	1.73%
150	YES	44100	6	1	45072	2.20%	6	1	45072	2.20%
150	YES	32000	9	0	32552	1.73%	9	0	32552	1.73%
150	YES	22050	13	1	21701	1.58%	13	1	21701	1.58%
150	YES	16000	18	1	15836	1.02%	18	1	15836	1.02%
150	YES	11025	26	1	11055	0.28%	26	1	11055	0.28%
150	YES	8000	36	2	8026	0.28%	36	2	8026	0.28%
100	No	192000	8	0	195312.5	1.73%	4	0	195312.5	1.73%
100	No	96000	16	1	94696.97	1.36%	8	0	97656.25	1.73%
100	No	48000	32	1	48076.92	0.16%	16	1	47348.48	1.36%
100	No	44100	35	1	44014.08	0.19%	17	1	44642.86	1.23%
100	No	32000	49	0	31887.76	0.35%	24	1	31887.76	0.35%
100	No	22050	71	0	22007.04	0.19%	35	1	22007.04	0.19%
100	No	16000	97	1	16025.64	0.16%	49	0	15943.88	0.35%
100	No	11025	141	1	11042.4	0.16%	71	0	11003.52	0.19%
100	No	8000	195	1	7992.327	0.10%	97	1	8012.821	0.16%
100	Yes	96000	2	0	97656.25	1.73%	2	0	97656.25	1.73%
100	Yes	48000	4	0	48828.13	1.73%	4	0	48828.13	1.73%
100	Yes	44100	4	1	43402.78	1.58%	4	1	43402.78	1.58%
100	Yes	32000	6	0	32552.08	1.73%	6	0	32552.08	1.73%
100	Yes	22050	9	0	21701.39	1.58%	9	0	21701.39	1.58%
100	Yes	16000	12	0	16276.04	1.73%	12	0	16276.04	1.73%
100	Yes	11025	17	1	11160.71	1.23%	17	1	11160.71	1.23%

SCLK (MHz)	MCLK	Target Fs (Hz)	16bit				32bit			
			I2S DIV	I2S_ODD	RealFs	Error	I2S DIV	I2S_ODD	RealFs	Error
100	Yes	8000	24	1	7971.939	0.35%	24	1	7971.939	0.35%
72	No	192000	6	0	187500	2.34%	3	0	187500	2.34%
72	No	96000	11	1	97826.09	1.90%	6	0	93750	2.34%
72	No	48000	32	1	34615.38	27.88%	11	1	48913.04	1.90%
72	No	44100	25	1	44117.65	0.04%	13	0	43269.23	1.88%
72	No	32000	35	0	32142.86	0.45%	17	1	32142.86	0.45%
72	No	22050	51	0	22058.82	0.04%	25	1	22058.82	0.04%
72	No	16000	70	1	15957.45	0.27%	35	0	16071.43	0.45%
72	No	11025	102	0	11029.41	0.04%	51	0	11029.41	0.04%
72	No	8000	140	1	8007.117	0.09%	70	1	7978.723	0.27%
72	Yes	96000	2	0	70312.5	26.76%	2	0	70312.5	26.76%
72	Yes	48000	3	0	46875	2.34%	3	0	46875	2.34%
72	Yes	44100	3	0	46875	6.29%	3	0	46875	6.29%
72	Yes	32000	4	1	31250	2.34%	4	1	31250	2.34%
72	Yes	22050	6	1	21634.62	1.88%	6	1	21634.62	1.88%
72	Yes	16000	9	0	15625	2.34%	9	0	15625	2.34%
72	Yes	11025	13	0	10817.31	1.88%	13	0	10817.31	1.88%
72	Yes	8000	17	1	8035.714	0.45%	17	1	8035.714	0.45%

13.3.5 DMA transfer

The SPI supports write and read operations with DMA. Whether used as SPI or I²S, read/write request using DMA comes from the same peripheral. As a result, their configuration procedure are the same, described as follows.

Transmission with DMA

- Select DMA channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the SPI_DT register address as the destination address bit of DMA transfer in the DMA control register. Data will be sent to this address after transmit request is received by DMA.
- Configure the source of DMA transfer: Configure the memory address as the source of DMA transfer in the DMA control register. Data will be loaded into the SPI_DT register from the memory address after transmit request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the channel priority of DMA transfer in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register.
- Enable DMA transfer channel in the DMA control register.

Reception with DMA

- Select DMA transfer channel: Select a DMA channel for the current SPI from DMA channel map table described in DMA chapter.
- Configure the destination of DMA transfer: Configure the memory address as the destination of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the source of DMA transfer: Configure the SPI_DT register address as the source of DMA transfer in the DMA control register. Data will be loaded from the SPI_DT register to the programmed destination after reception request is received by DMA.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure the total number of bytes to be transferred in the DMA control register.
- Configure DMA interrupt generation after half or full transfer in the DMA control register
- Enable DMA transfer channel in the DMA control register.

13.3.6 Transmitter/Receiver

Whether being used as SPI or I²S, there is no difference for CPU. The SPI (in whatever mode) shares the same base address, the same SPI_DT register, the same transmitter and receiver. The SPI transmitter and receiver is responsible for sending and receiving the desired data frame according to the configuration of the communication controller. Thus their status flags such as TDBE, RDBF and ROERR, and their interrupt enable bits including TDBEIE, RDBFIE and ERRIE are identical.

Special attention must be paid to:

- CRC check is not available on the I²S. Any operation linked to CRC, including CCERR flag and the corresponding interrupts, is not supported.
- I²S protocol needs decode the current channel status. The ACS bit is used to judge whether the current transfer occurs on the left channel (ACS=0) or the right channel (ACS=1).
- TUERR bit indicates whether an underrun occurs. TUERR=1 means an underrun error occurs on the transmitter. An interrupt is generated when the ERRIE is set.
- Read/write operation to the SPI_DT register is different under different audio protocols, data bits and channel bits. Refer to the audio protocol selector section for more information.
- Pay more attention to the I²S disable operation under different configurations, shown as follows:
  — I2SDBN=00, I2SCBN=1, STDSLE=10: wait for the second-to-last RDBF=1 and 17 CK periods before disabling the I²S
  — I2SDBN=00, I2SCBN=1, STDSLE=00 or STDSLE=01 or STDSLE=11: wait for the last RDBF=1 and one CK period before the I²S
  — I2SDBN, I2SCBN,STDSLE combination: wait for the second-to-last RDBF=1 and one CK period before disabling the I²S.

I²S transmitter configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I²S
- Follow above steps to configure the I²SxEXT (For I²S full-duplex mode)

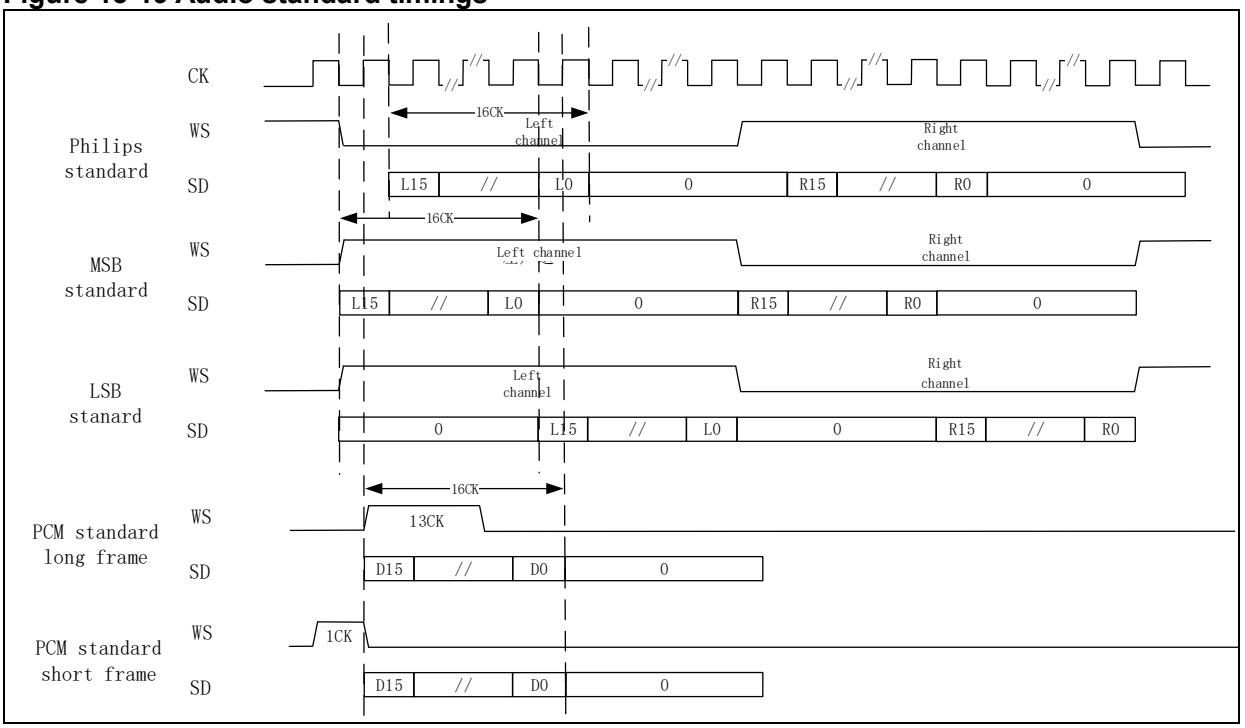
I²S receiver configuration procedure:

- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I²S
- Follow above steps to configure the I²SxEXT (For I²S full-duplex mode)

13.3.7 I2S communication timings

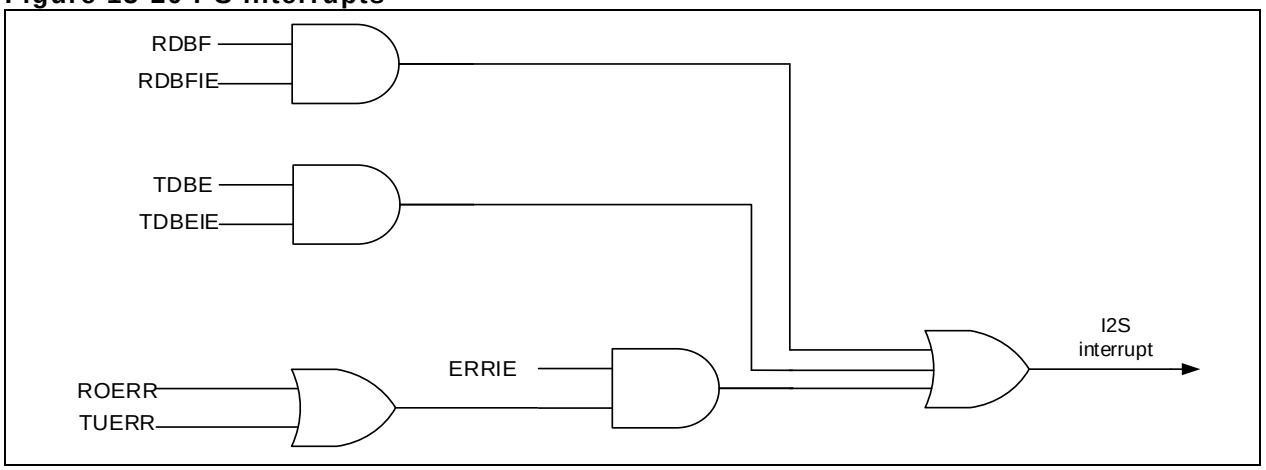
I2S can address four different audio standards: Philips standard, the most significant byte (left-aligned) and the least significant byte (right-aligned) standards, and the PCM standard. *Figure 13-19* shows their respective timgins.

Figure 13-19 Audio standard timings



13.3.8 Interrupt

Figure 13-20 I²S interrupts



13.3.9 IO pin control

The I²S needs three pins for transfer operation, namely, the SD, WS and CK. The MCLK pin is also required if need to provide main clock for peripherals. The I²S shares some pins with the SPI, described as follows:

- SD: Serial data (mapped on the MOSI pin) for bidirectional data transmission and reception.
- WS: Word select (mapped on the CS pin) for data control signal output in master mode, and input in slave mode.
- CK: Communication clock (mapped on the SCK pin) as clock signal output in master mode, and input in slave mode.
- MCLK: Master clock (mapped independently) is used to provide main clock for peripherals. The frequency of output clock signal is set to 256x Fs (audio sampling frequency)

13.4 SPI registers

These peripheral registers must be accessed by half-word (16 bits) or word (32 bits).

Table 13-2 SPI register map and reset value

Register	Offset	Reset value
SPI_CTRL1	0x00	0x0000
SPI_CTRL2	0x04	0x0000
SPI_STS	0x08	0x0002
SPI_DT	0x0C	0x0000
SPI_CPOLY	0x10	0x0007
SPI_RCRC	0x14	0x0000
SPI_TCRC	0x18	0x0000
SPI_I2SCTRL	0x1C	0x0000
SPI_I2SCLKP	0x20	0x0002

13.4.1 SPI control register1 (SPI_CTRL1) (Not used in I²S mode)

Bit	Register	Reset value	Type	Description
Bit 15	SLBEN	0x0	rw	Single line bidirectional half-duplex enable 0: Disabled 1: Enabled
Bit 14	SLBTD	0x0	rw	Single line bidirectional half-duplex transmission direction This bit and the SLBEN bit together determine the data output direction in “Single line bidirectional half-duplex” mode. 0: Receive-only mode 1: Transmit-only mode
Bit 13	CCEN	0x0	rw	RC calculation enable 0: Disabled 1: Enabled
Bit 12	NTC	0x0	rw	Transmit CRC next When this bit is set, it indicates that the next data transferred is CRC value. 0: Next transmitted data is the normal value 1: Next transmitted data is CRC value
Bit 11	FBN	0x0	rw	Frame bit num This bit is used to configure the number of data frame bit for transmission/reception. 0: 8-bit data frame 1: 16-bit data frame
Bit 10	ORA	0x0	rw	Receive-only active In two-wire unidirectional mode, when this bit is set, it indicates that Receive-only is active, but the transmit is not allowed. 0: Transmission and reception 1: Receive-only mode
Bit 9	SWCSEN	0x0	rw	Software CS enable When this bit is set, the CS pin level is determined by the SWCSIL bit. The status of I/O level on the CK pin is invalid. 0: Disabled 1: Enabled
Bit 8	SWCSIL	0x0	rw	Software CS internal level This bit is valid only when the SWCSEN is set. It determines the level on the CS pin. In master mode, this bit must be set. 0: Low level 1: High level

Bit 7	LTF	0x0	rw	LSB transmit first This bit is used to select for MST transfer first or LSB transfer first. 0: MSB 1: LSB
Bit 6	SPIEN	0x0	rw	SPI enable 0: Disabled 1: Enabled
Bit 5: 3	MDIV	0x0	rw	Master clock frequency division In master mode, the peripheral clock divided by the prescaler is used as SPI clock. The MDIV[3] bit is in the SPI_CTRL2 register, MDIV[3: 0]: 0000: Divided by 2 0001: Divided by 4 0010: Divided by 8 0011: Divided by 16 0100: Divided by 32 0101: Divided by 64 0110: Divided by 128 0111: Divided by 256 1000: Divided by 512 1001: Divided by 1024
Bit 2	MSTEN	0x0	rw	Master enable 0: Disabled (Slave) 1: Enabled (Master)
Bit 1	CLKPOL	0x0	rw	Clock polarity Indicates the polarity of clock output in idle state. 0: Low level 1: High level
Bit 0	CLKPHA	0x0	rw	Clock phase 0: Data capture starts from the first clock edge 1: Data capture starts from the second clock edge

Note: The SPI_CTRL1 register must be 0 in I^2S mode.

13.4.2 SPI control register2 (SPI_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Forced to be 0 by hardware.
Bit 8	MDIV	0x0	rw	Master clock frequency division Refer to the MDIV[2: 0] of the SPI_CTRL1 register.
Bit 7	TDBEIE	0x0	rw	Transmit data buffer empty interrupt enable 0: Disabled 1: Enabled
Bit 6	RDBFIE	0x0	rw	Receive data buffer full interrupt enable 0: Disabled 1: Enabled
Bit 5	ERRIE	0x0	rw	Error interrupt enable This bit controls interrupt generation when errors occur (CCERR, MMERR, ROERR and TUERR) 0: Disabled 1: Enabled
Bit 4: 3	Reserved	0x0	resd	Kept at its default value
Bit 2	HWCSOE	0x0	rw	Hardware CS output enable This bit is valid only in master mode. When this bit is set, the I/O output on the CS pin is low; when this bit is 0, the I/O input on the CS pin must be set high. 0: Disabled 1: Enabled
Bit 1	DMATEN	0x0	rw	DMA transmit enable 0: Disabled 1: Enabled
Bit 0	DMAREN	0x0	rw	DMA receive enable 0: Disabled 1: Enabled

13.4.3 SPI status register (SPI_STS)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Forced to be 0 by hardware
Bit 7	BF	0x0	ro	Busy flag 0: SPI is not busy. 1: SPI is busy.
Bit 6	ROERR	0x0	ro	Receiver overflow error 0: No overflow error 1: Overflow error occurs.
Bit 5	MMERR	0x0	ro	Master mode error This bit is set by hardware and cleared by software (read/write access to the SPI_STS register, followed by write operation to the SPI_CTRL1 register) 0: No mode error 1: Mode error occurs.
Bit 4	CCERR	0x0	rw0c	CRC error Set by hardware, and cleared by software. 0: No CRC error 1: CRC error occurs.
Bit 3	TUERR	0x0	ro	Transmitter underload error Set by hardware, and cleared by software (read the SPI_STS register). 0: No underload error 1: Underload error occurs. Note: This bit is only used in I^2S mode.
Bit 2	ACS	0x0	ro	Audio channel state This bit indicates the status of the current audio channel. 0: Left channel 1: Right channel Note: This bit is only used in I^2S mode.
Bit 1	TDBE	0x1	ro	Transmit data buffer empty 0: Transmit data buffer is not empty. 1: Transmit data buffer is not empty.
Bit 0	RDBF	0x0	ro	Receive data buffer full 0: Transmit data buffer is not full. 1: Transmit data buffer is full.

13.4.4 SPI data register (SPI_DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DT	0x0000	rw	Data value This register controls read and write operations. When the data bit is set as 8 bit, only the 8-bit LSB [7: 0] is valid.

13.4.5 SPICRC register (SPI_CPOLY)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CPOLY	0x0007	rw	CRC polynomial This register contains the polynomial used for CRC calculation. Note: This register is valid only in SPI mode.

13.4.6 SPIRxCRC register (SPI_RCRC)

Bit	Register	Reset value	Type	Description
Bit 15: 0	RCRC	0x0000	ro	Receive CRC When CRC calculation is enabled, this register contains the CRC value computed based on the received data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared. When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard. Note: This register is only used in SPI mode.

13.4.7 SPITxCRC register (SPI_TCRC)

Bit	Register	Reset value	Type	Description
Bit 15: 0	TCRC	0x0000	ro	Transmit CRC When CRC calculation is enabled, this register contains the CRC value computed based on the transmitted data. This register is reset when the CCEN bit in the SPI_CTRL1 register is cleared. When the data frame format is set to 8-bit data, only the 8-bit LSB ([7: 0]) are calculated based on CRC8 standard; when 16-bit data bit is selected, follow CRC16 standard. Note: This register is only used in SPI mode.

13.4.8 SPI_I2S configuration register (SPI_I2SCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced to be 0 by hardware.
Bit 11	I2SMSEL	0x0	rw	I^2S mode select 0: SPI mode 1: I^2S mode
Bit 10	I2SEN	0x0	rw	I^2S enable 0: Disabled 1: Enabled
Bit 9: 8	OPERSEL	0x0	rw	I^2S operation mode select 00: Slave transmission 01: Slave reception 10: Master transmission 11: Master reception
Bit 7	PCMFSSEL	0x0	rw	PCM frame synchronization This bit is valid only when the PCM standard is used.

				0: Short frame synchronization 1: Long frame synchronization
Bit 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	STDSEL	0x0	rw	I²S standard select 00: Philips standard 01: MSB-aligned standard (left-aligned) 10: LSB-aligned standard (right-aligned) 11: PCM standard
Bit 3	I2SCLKPOL	0x0	rw	I²S clock polarity This bit indicates the clock polarity on the clock pin in idle state. 0: Low 1: High
Bit 2: 1	I2SDBN	0x0	rw	I²S data bit num 00: 16-bit data length 01: 24-bit data length 10: 32-bit data length 11: Not allowed.
Bit 0	I2SCBN	0x0	rw	I²S channel bit num This bit can be configured only when the I²S is set to 16-bit data; otherwise, it is fixed to 32-bit by hardware. 0: 16-bit wide 1: 32-bit wide

13.4.9 SPI_I2S prescaler register (SPI_I2SCLKP)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced to be 0
Bit 9	I2SMCLKOE	0x0	rw	I²S Master clock output enable 0: Disabled 1: Enabled
Bit 8	I2SODD	0x0	rw	Odd factor for I²S division 0: Actual divider factor =I2SDIV*2 1: Actual divider factor =(I2SDIV*2)+1
Bit 11: 10 Bit 7: 0	I2SDIV	0x02	rw	I²S division It is not allowed to configure I2SDIV[9: 0]=0 or I2SDIV[9: 0]=1

14 Timer

AT32F415 timers include basic timers, general-purpose timers, and advanced timers.

Please refer to *Section 14.1* ~ *Section 14.3* for the detailed function modes. All functions of different timers are shown in the following tables.

Table 14-1 TMR functional comparison

Timer type	Timer	Counter bit	Count mode	Repetition	Prescaler	DMA requests	Capture/ compare channel	PWM input mode	EXT input	Break input
Advanced-control timer	TMR1	16	Up Down Up/Down	8-bit	1~65536	O	4	O	O	O
General-purpose timer	TMR2 TMR5	16/32	Up Down Up/Down	X	1~65536	O	4	O	TMR2 only	X
	TMR3 TMR4	16	Up Down Up/Down	X	1~65536	O	4	O	TMR3 only	X
	TMR9	16	Up	X	1~65536	X	2	O	X	X
	TMR10 TMR11	16	Up	X	1~65536	X	1	X	X	X

Timer type	Timer	Counter bit	Count mode	PWM output	Single pulse output	Complementary output	Dead-time	Encoder interface connection	Interfacing with hall sensors	Linkage peripheral
Advanced-control timer	TMR1	16	Up Down Up/Down	O	O	O	O	O	O	Timer synchronization ADC
General-purpose timer	TMR2 TMR5	16/32	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization ADC
	TMR3 TMR4	16	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization ADC
	TMR9	16	Up	O	O	X	X	X	X	Timer synchronization ADC
	TMR10 TMR11	16	Up	O	O	X	X	X	X	NA

14.1 General-purpose timer (TMR2 to TMR5)

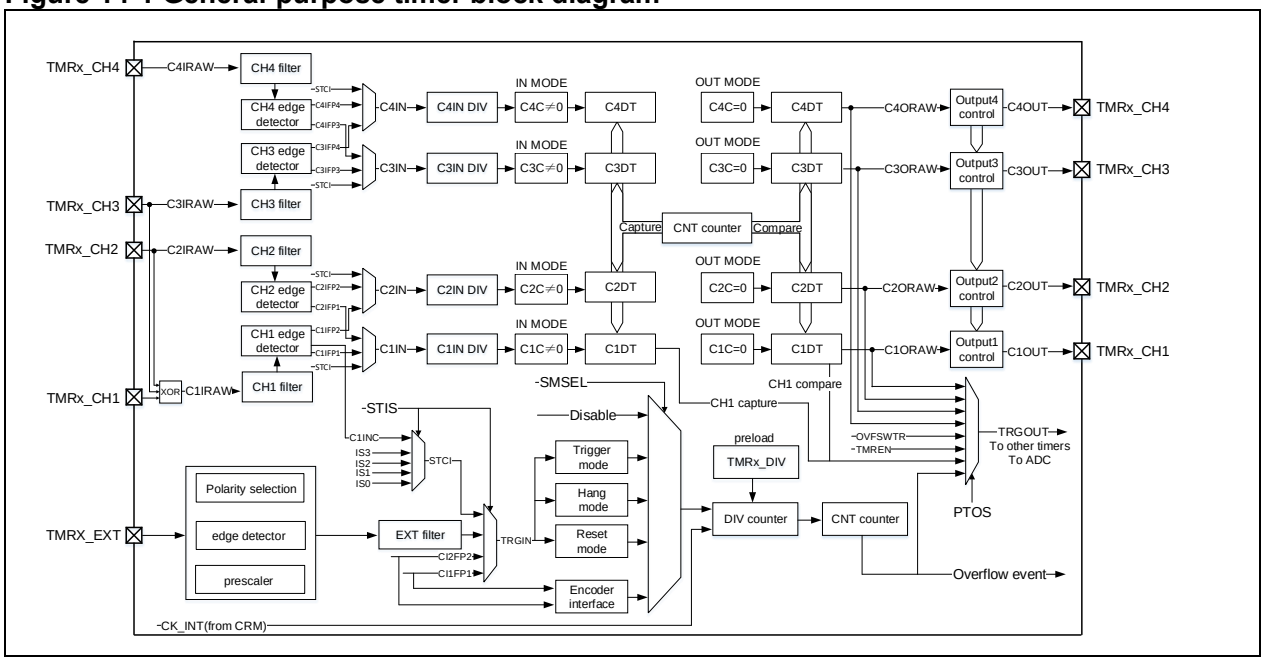
14.1.1 TMRx introduction

The general-purpose timer (TMR2 to TMR5) consists of a 16-bit counter supporting up, down, up/down (bidirectional) counting modes, four capture/compare registers, and four independent channels to achieve input capture and programmable PWM output.

14.1.2 TMRx main features

- Source of count clock is selectable : internal clock, external clock and internal trigger
- 16-bit up, down, up/down and encoder mode counter (TMR2/5 can be extended to 32-bit)
- 4 independent channels for input capture, output compare, PWM generation and one-pulse mode output
- Synchronization control between master and slave timers
- Interrupt/DMA is generated at overflow event, trigger event and channel event
- Support TMR burst DMA transfer

Figure 14-1 General-purpose timer block diagram

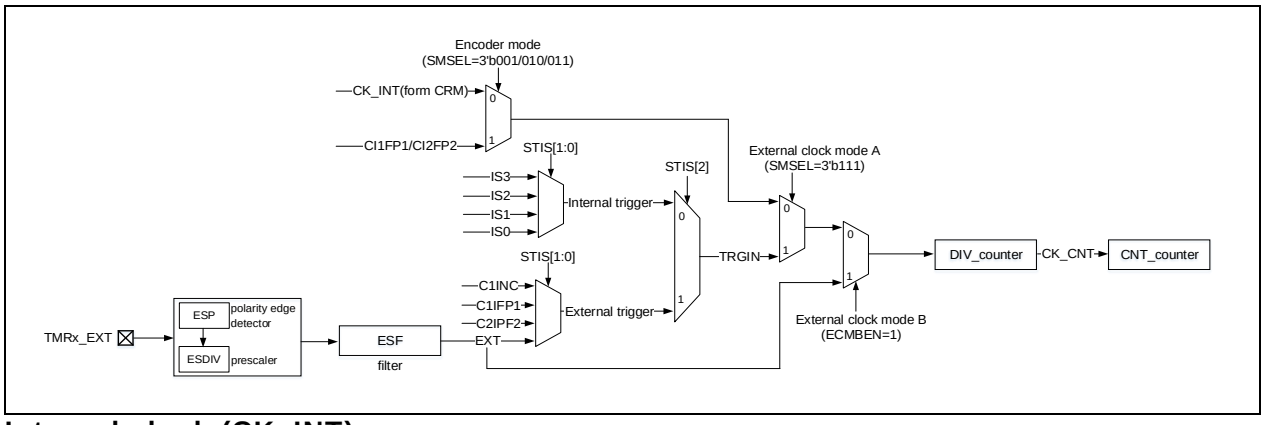


14.1.3 TMRx functional overview

14.1.3.1 Counting clock

The count clock of TMR2~TMR5 can be provided by the internal clock (CK_INT), external clock (external clock mode A and B) and internal trigger input (ISx)

Figure 14-2 Counting clock



Internal clock (CK_INT)

By default, the CK_INT divided by a prescaler is used to drive the counter to start counting. When TMR's APB clock prescaler factor is 1, the CK_INT frequency is equal to that of APB, otherwise, it doubles the APB clock frequency.

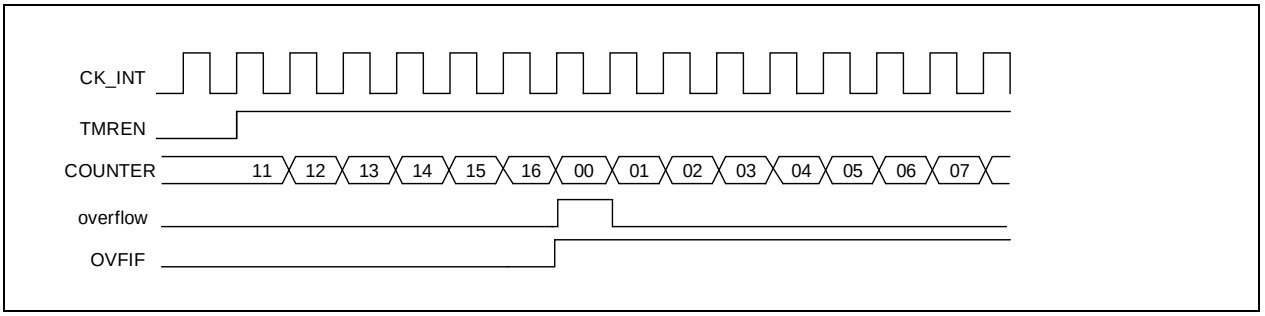
The configuration process is as follows:

- Select count mode through the TWCMSEL[1:0] bit in the TMRx_CTRL1 register and. If the one-way

count direction is set, configure OWCDIR bit in the TMRx_CTRL1 register to select the specific direction.

- Configure the TMRx_DIV register and set counting frequency.
- Configure the TMRx_PR register and set counting period.
- Configure the TMREN bit in the TMRx_CTRL1 register and enable the counter.

Figure 14-3 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by setting the STIS[2: 0] bit to drive the counter to start counting.

The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, the channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, a channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal with polarity selection, frequency division and filtering).

ECMBEN=1: External clock mode B is selected. The counter is driven by external input that has gone through polarity selection, frequency division and filtering. The external clock mode B is equivalent to the external clock mode A which selects EXT signal as an external force TRGIN.

To use external clock mode A, follow the steps below:

-Set external source TRGIN parameters

If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter (C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).

- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register

-Enable counter through the TMREN bit in TMRx_CTRL1 register

To use external clock mode B, follow the steps below:

-Set external signal polarity through the ESP bit in TMRx_STCTRL register

-Set external signal frequency division through the ESDIV[1:0] bit in TMRx_STCTRL register

-Set external signal filter through the ESF[3:0] bit in TMRx_STCTRL register

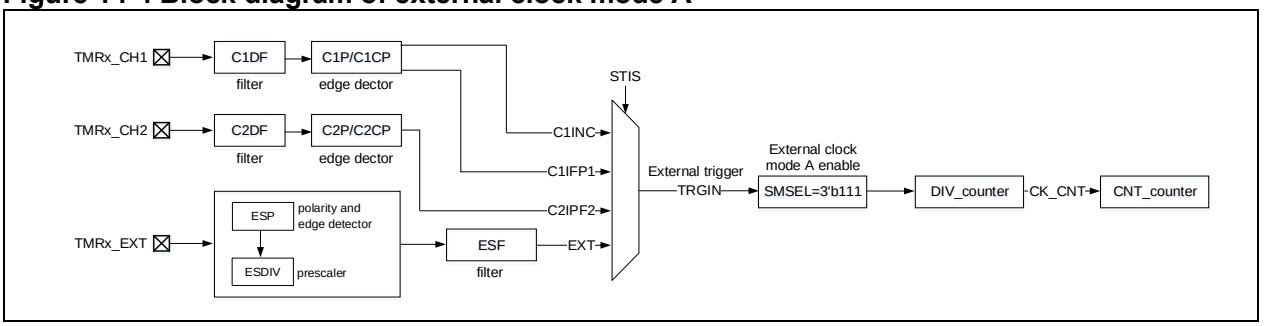
-Enable external clock mode B through the ECMBEN bit in TMRx_STCTR register

-Set counting frequency through the DIV[15:0] bit in TMRx_DIV register

-Set counting period through the PR[15:0] bit in TMRx_PR register

-Enable counter through the TMREN in TMRx_CTRL1 register

Figure 14-4 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-5 Counting in external clock mode A, PR=0x32, DIV=0x0

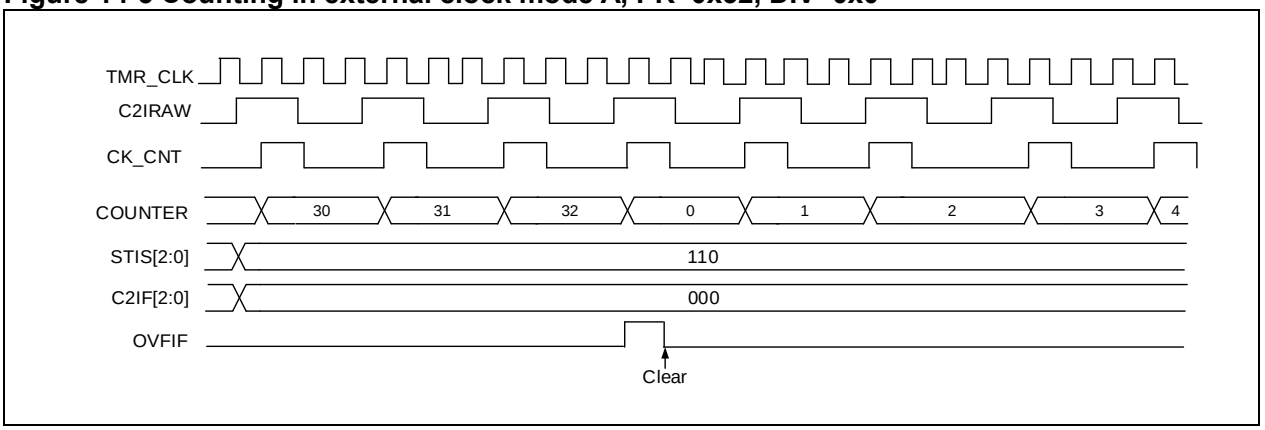
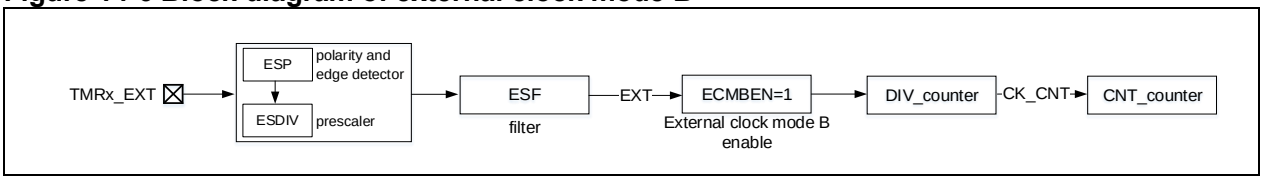
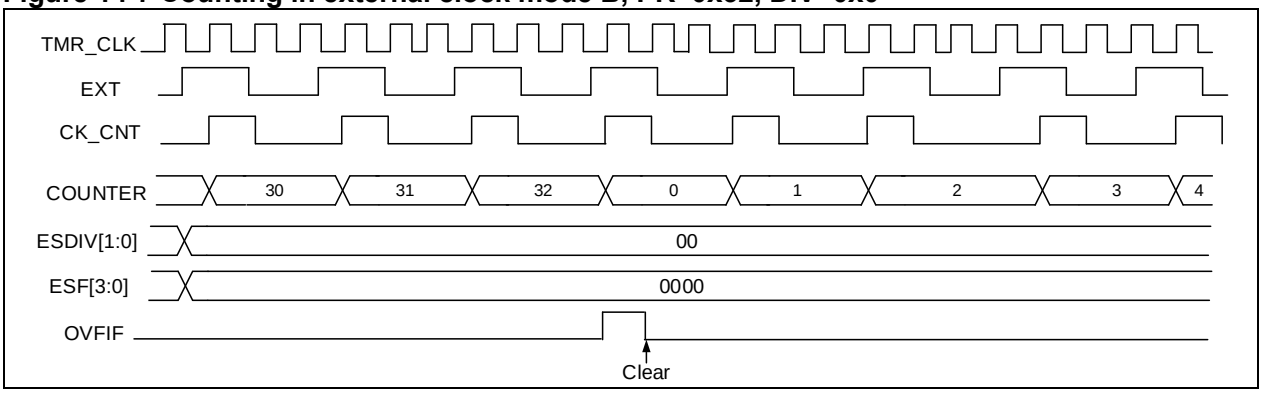


Figure 14-6 Block diagram of external clock mode B



Note: The delay between the EXT signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-7 Counting in external clock mode B, PR=0x32, DIV=0x0



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each timer (TMR2 to TMR5) consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

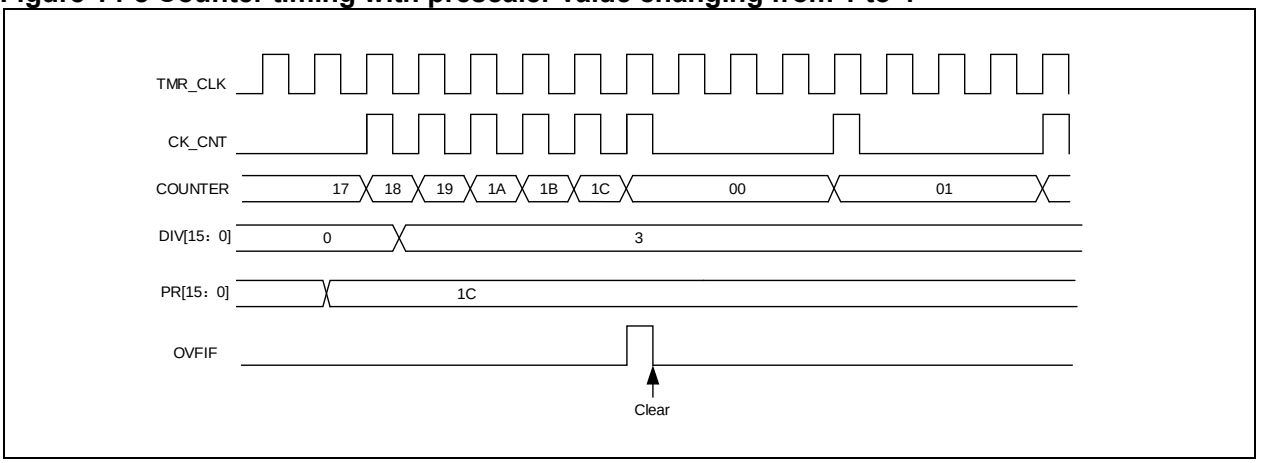
- Select counting period through the TMRx_PR register
- Select counting frequency through the TMRx_DIV register to
- Select count mode through the TWCMSEL[1:0] bit in the TMRx_CTRL1 register
- Select internal trigger by setting the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register
- Select external clock mode A by setting SMSEL[2:0]=3'b111 in the TMRx_STCTRL register
- Enable TMRx counter through the TMREN bit in the TMRx_CTRL1 register

Table 14-2 TMRx internal trigger connection

Slave timer	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR2	TMR1	USB_SOF	TMR3	TMR4
TMR3	TMR1	TMR2	TMR5	TMR4
TMR4	TMR1	TMR2	TMR3	-
TMR5	TMR2	TMR3	TMR4	-

Note 1: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

Figure 14-8 Counter timing with prescaler value changing from 1 to 4



14.1.3.2 Counting mode

The timer (TMR2 to TMR5) supports several counting modes to meet different application scenarios. Each timer has an internal 16-bit upcounter, downcounter, upcounter/downcounter. TMR2/5 can be extended to 32-bit by setting the PMEN bit.

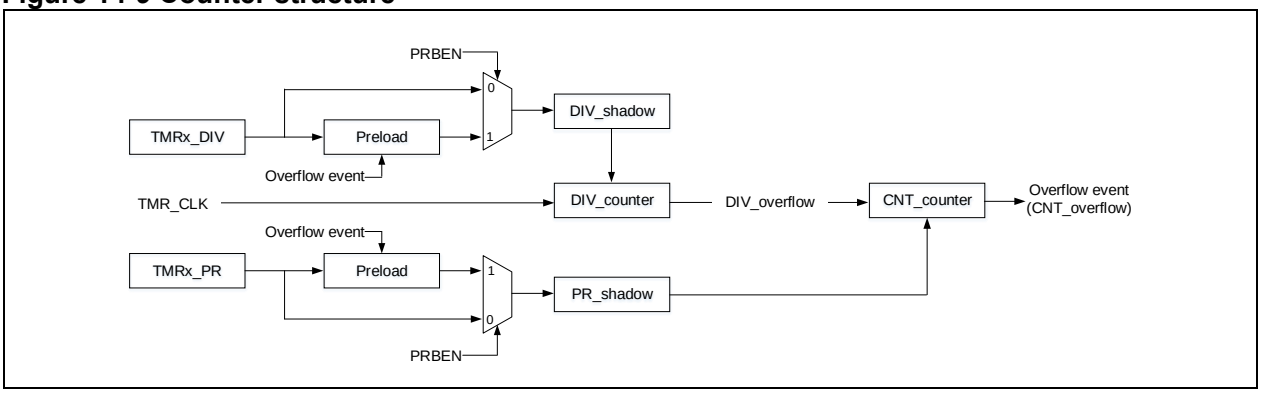
The TMRx_PR register is used to set the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event.

The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.
An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-9 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event, otherwise, the counter is updated with the preload value and period value at an overflow event.

Figure 14-10 Overflow event when PRBEN=0

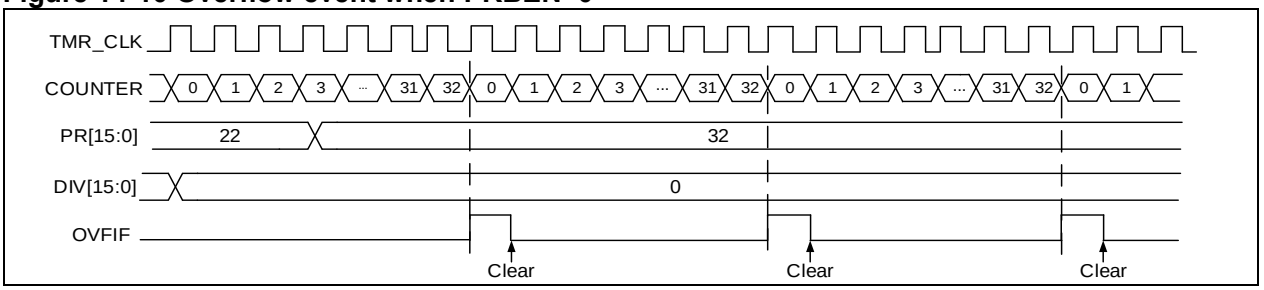
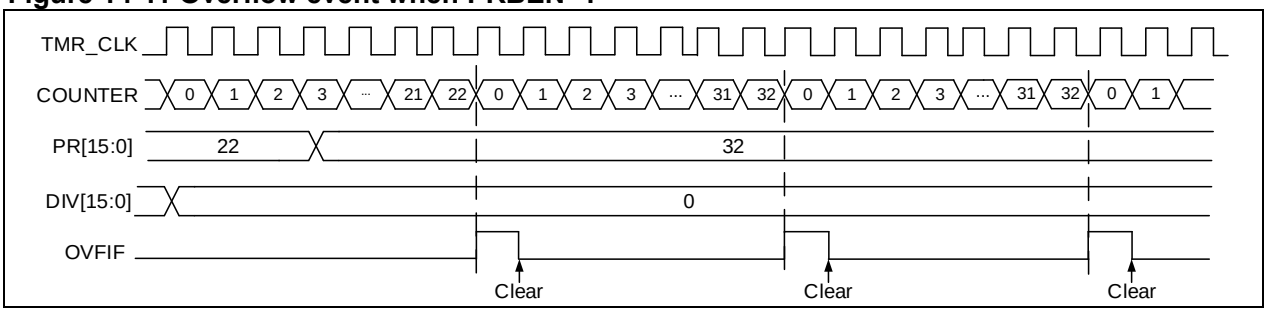


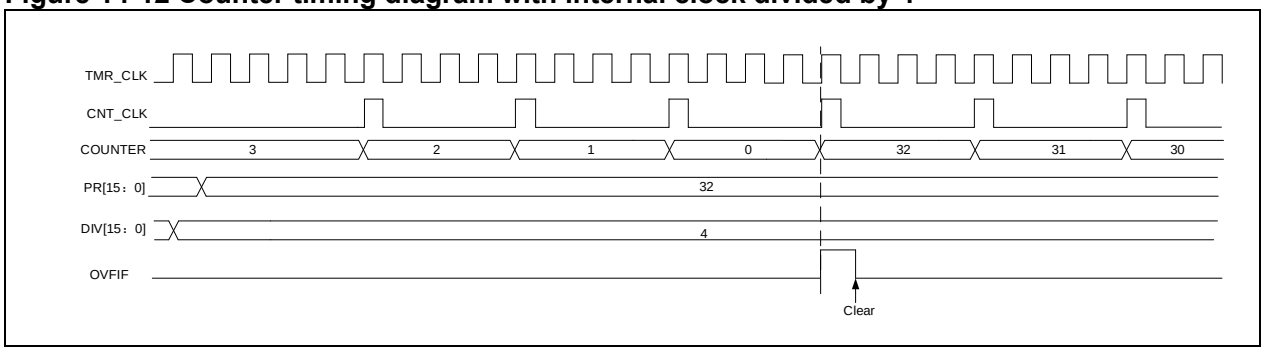
Figure 14-11 Overflow event when PRBEN=1



Downcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register to enable downcounting mode. In this mode, the counter counts from the value programmed in the TMRx_PR register down to 0, and restarts from the value programmed, and generates a counter underflow event.

Figure 14-12 Counter timing diagram with internal clock divided by 4



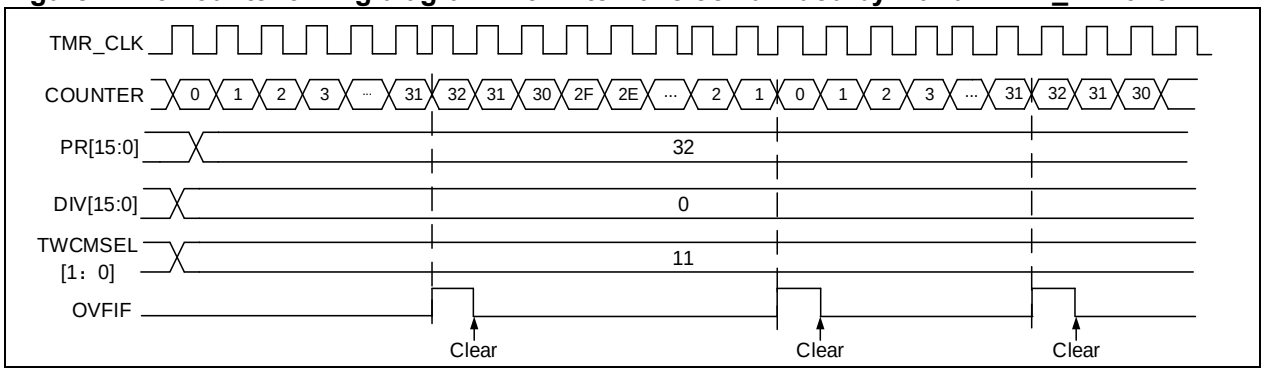
Up/down counting mode

Set CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register to enable up/down counting mode. In this mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the TMRx_PR register down to 1, an underflow event is generated, and then restarts counting from 0. When the counter counts from 0 to the value of the TMRx_PR register -1, an overflow event is generated, and then restarts counting from the value of the TMRx_PR register. The OWCDIR bit indicates the current counting direction.

The TWCMSEL[1:0] bit in the TMRx_CTRL1 register is also used to select the CxIF flag setting method in up/down counting mode. In up/down counting mode 1 (TWCMSEL[1:0]=2'b01), CxIF flag can only be set when the counter counts down; in up/down counting mode 2 (TWCMSEL[1:0]=2'b10), CxIF flag can only be set when the counter counts up; in up/down counting mode 3 (TWCMSEL[1:0]=2'b11), CxIF flag can be set when the counter counts up/down.

Note: The OWCDIR is ready-only in up/down counting mode.

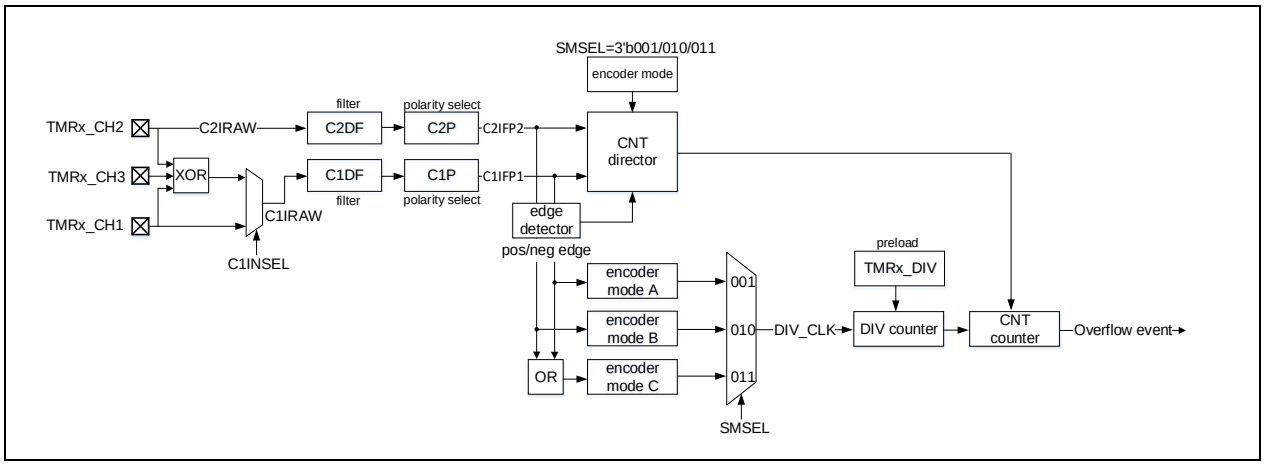
Figure 14-13 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Encoder interface mode

In this mode, the two input (TMRx_CH1 and TMRx_CH2) signals are required. Depending on the level on one input, the counter counts up or down on the edge of the other input signal. The OWCDIR bit indicates the direction of the counter, as shown in the table below:

Figure 14-14 Encoder mode structure



Encoder mode A: SMSEL=3'b001. The counter counts on the selected C1IFP1 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C1IFP1 and the level of C2IFP2.

Encoder mode B: SMSEL=3'b010. The counter counts on the selected C2IFP2 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C2IFP2 and the level of C1IFP1.

Encoder mode C: SMSEL=3'b011. The counter counts on both C1IFP1 and C2IFP2 edges (rising and falling edges). The counting direction is dependent on the C1IFP1 edge direction and C2IFP2 level, and C2IFP2 edge direction and C1IFP1 level.

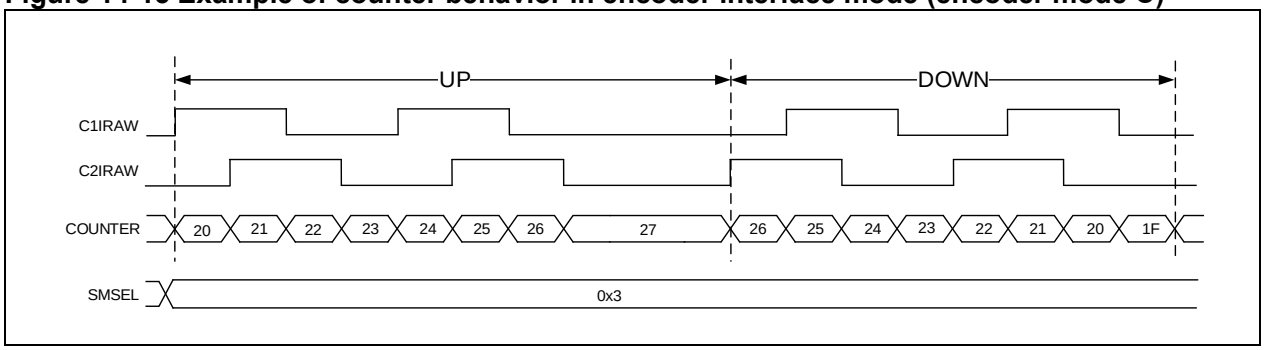
To use encoder mode, follow the procedures below:

- Set channel 1 input signal filtering through the C1DF[3:0] bit in the TMRx_CM1 register;
  Set channel 1 input signal active level through the C1P bit in the TMRx_CCTRL register
- Set channel 2 input signal filtering through the C2DF[3:0] bit in the TMRx_CM1 register;
  Set channel 2 input signal active level through the C2P bit in the TMRx_CCTRL register
- Set channel 1 as input mode through the C1C[1:0] bit in the TMRx_CM1 register;
  Set channel 2 as input mode through the C2C[1:0] bit in the TMRx_CM1 register
- Select encoder mode A (SMSEL=3'b001), encoder mode B (SMSEL=3'b010), or encoder mode C (SMSEL=3'b011) by setting the SMSEL[2:0] bit in the TMRx_STCTRL register
- Set counting cycles through the PR[15:0] bit in the TMRx_PR register
- Set counting frequency through the DIV[15:0] bit in the TMRx_DIV register
- Configure the corresponding IOs of TMRx_CH1 and TMRx_CH2 as multiplexed mode
- Enable counter through the TMREN bit in the TMRx_CTRL1 register

Table 14-3 Counting direction versus encoder signals

Active edge	Level on opposite signal (C1IFP1 to C2IFP2, C2IFP2 to C1IFP1)	C1IFP1 signal		C2IFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IFP1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IFP2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Count on both C1IFP1 and C2IFP2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Figure 14-15 Example of counter behavior in encoder interface mode (encoder mode C)



14.1.3.3TMR input function

Each of timers (TMR2 and TMR5) has four independent channels, with each channel being configured as input or output.

As input, each channel input signal is handled as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSEL bit is used to select the source of C1IRAW from TMRx_CH1 or the XOR-ed TMRx_CH1, TMRx_CH2 and TMRx_CH3.

  The sources of C2IRAW, C3IRAW and C4IRAW are TMRx_CH2, TMRx_CH3 and TMRx_CH4, respectively.
- CxIRAW inputs digital filter and outputs filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF inputs edge detector, and outputs the CxIFPx signal after edge selection. The edge selection depends on both CxP and CxCP bits. It is possible to select input rising edge, falling edge or both edges.
- CxIFPx inputs capture signal selector, and outputs the CxIN signal after capture signal selection. The capture signal selection is defined by CxC bit. It is possible to select CxIFPx, CyIFPx or STCI as CxIN source. Of those, CyIFPx (x≠y) is the CyIFPy signal that is from Y channel and processed by channel-x edge detector (for example, C1IFP2 is the channel 1's C1IFP1 signal that passed through channel 2 edge detection). The STCI comes from slave timer controller, and its source is selected by STIS bit.
- CxIN outputs the CxIPS signal that is divided by input channel divider. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit. It can be used for filtering, selection, division and input capture of input signals.

Figure 14-16 Input/output channel 1 main circuit

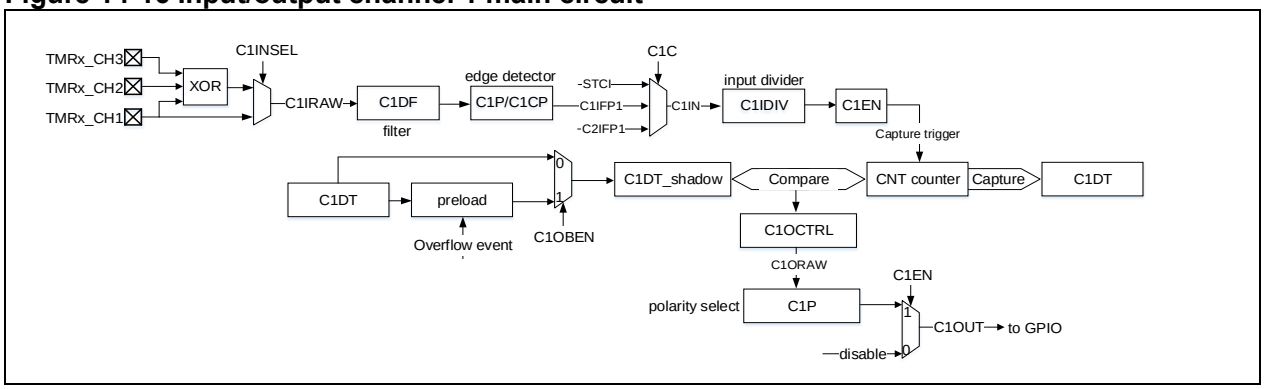
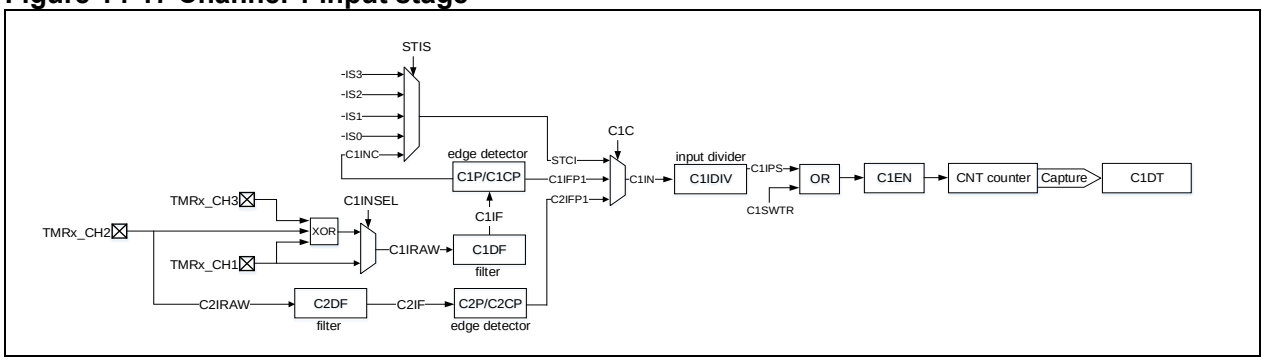


Figure 14-17 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt or a DMA request will be generated if the CxIEN and CxDEN bits are enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTRL register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMRx_IDEN register or the C1DEN bit in the TMRx_IDEN register

Timer Input XOR function

The 3 timer input pins (TMRx_CH1, TMRx_CH2 and TMRx_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMRx_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For example, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

PWM input

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2‘b01 to set C1IN as C1IFP1;
- Set C1P=1’b0 to set C1IFP1 rising edge active;
- Set C2C=2‘b10 to set C2IN as C1IFP2;
- Set C2P=1’b1 to set C1IFP2 falling edge active;
- Set STIS=3’b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3‘b100 to set the slave timer in reset mode;
- Set C1EN=1’b1 and C2EN=1’b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-18 Example of PWM input mode configuration

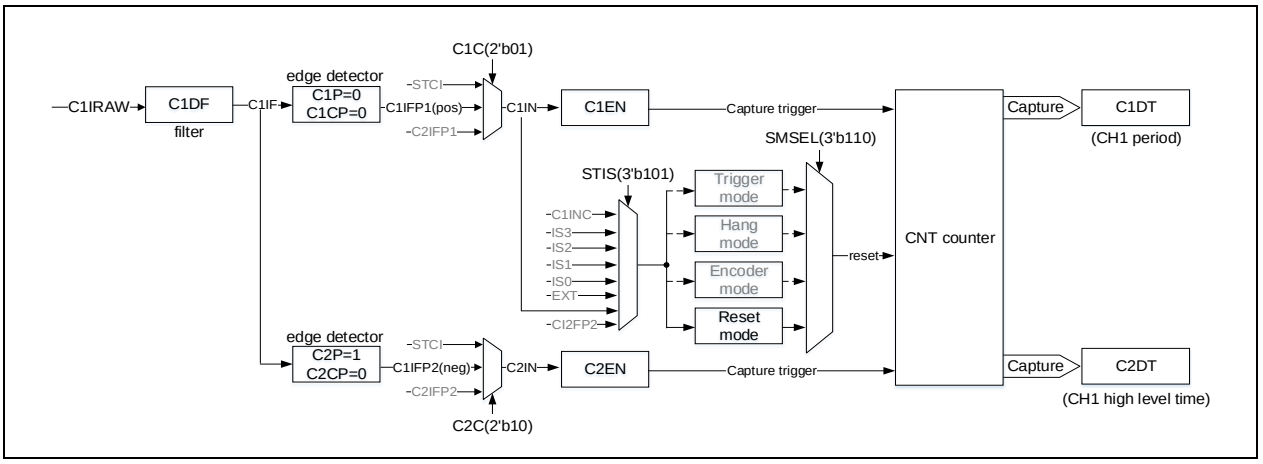
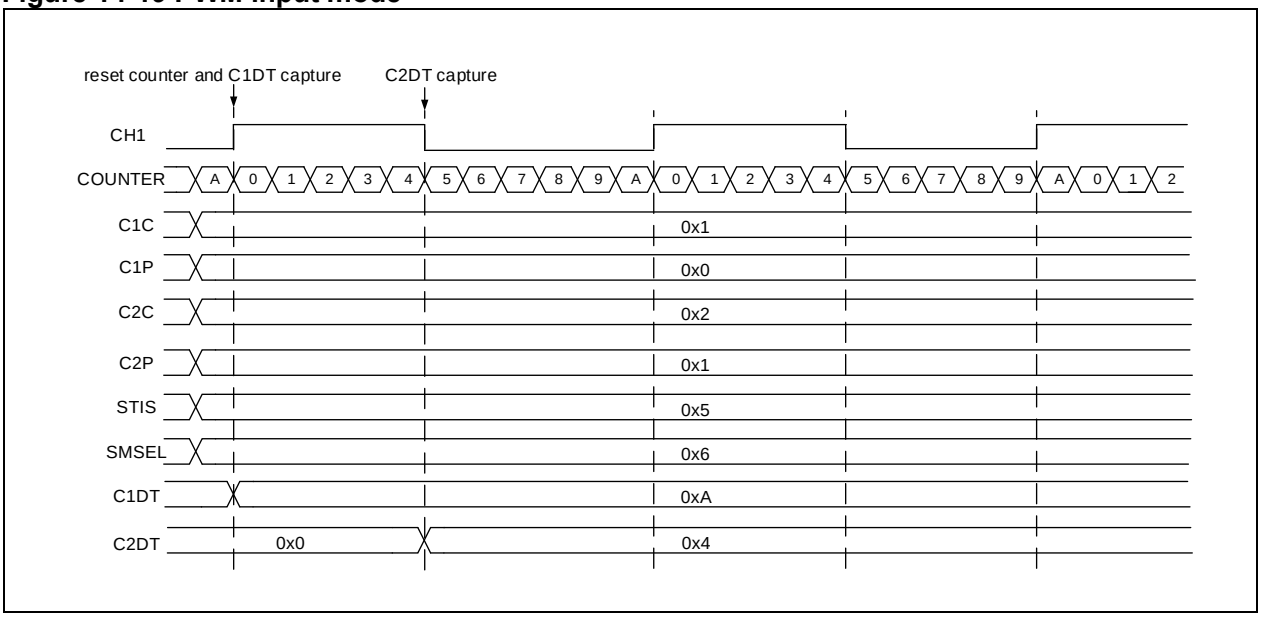


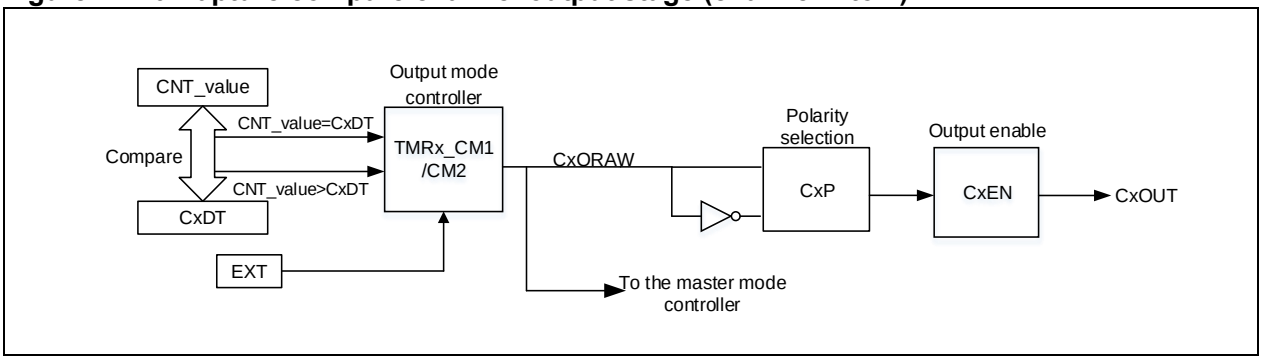
Figure 14-19 PWM input mode



14.1.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

Figure 14-20 Capture/compare channel output stage (channel 1 to 4)



Output mode

Write CxC[2: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A:** Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in the TMRx_CM1/CM2 register to set output mode as PWM mode A;
  - Set the TMRx_DIV register and set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to set the count mode;
  - Set CxP bit and CxCP bit in the TMRx_CCTRL register to set output polarity;
  - Set CxEN bit and CxCEN bit in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B:** Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode:** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode:** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 21 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 22 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 23 gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 24 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-21 C1ORAW toggles when counter value matches the C1DT value

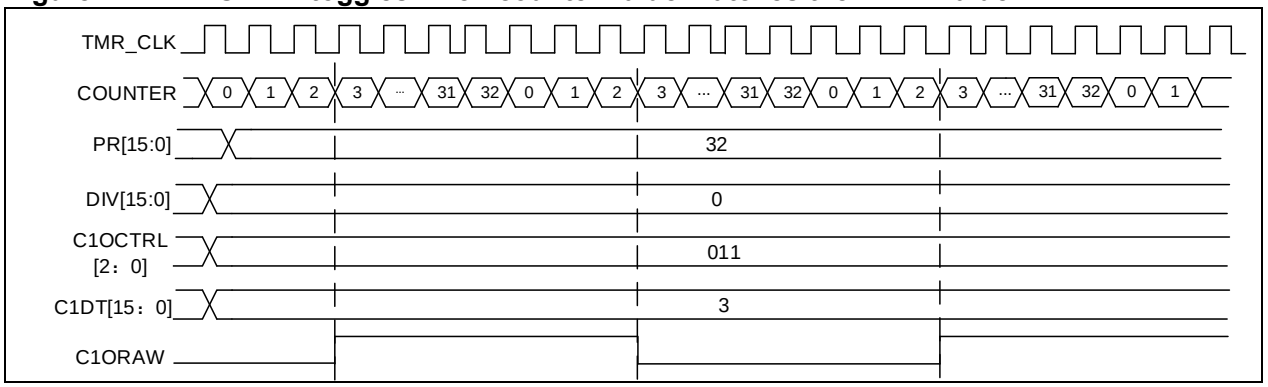


Figure 14-22 Upcounting mode and PWM mode A

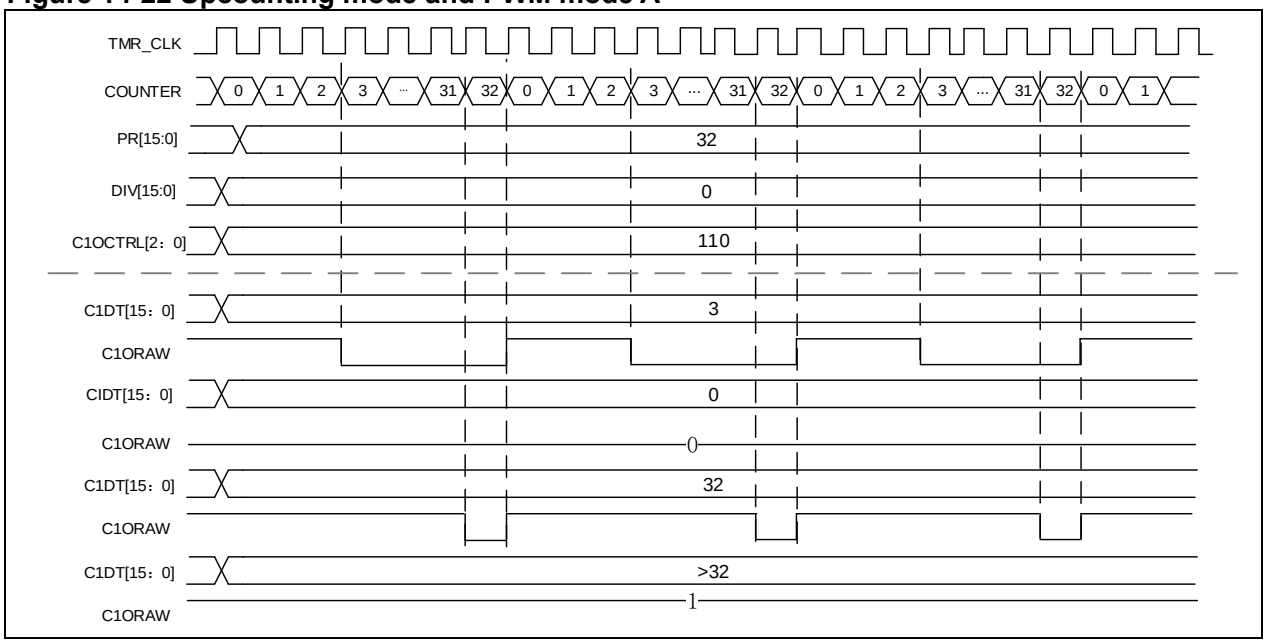


Figure 14-23 Up/down counting mode and PWM mode A

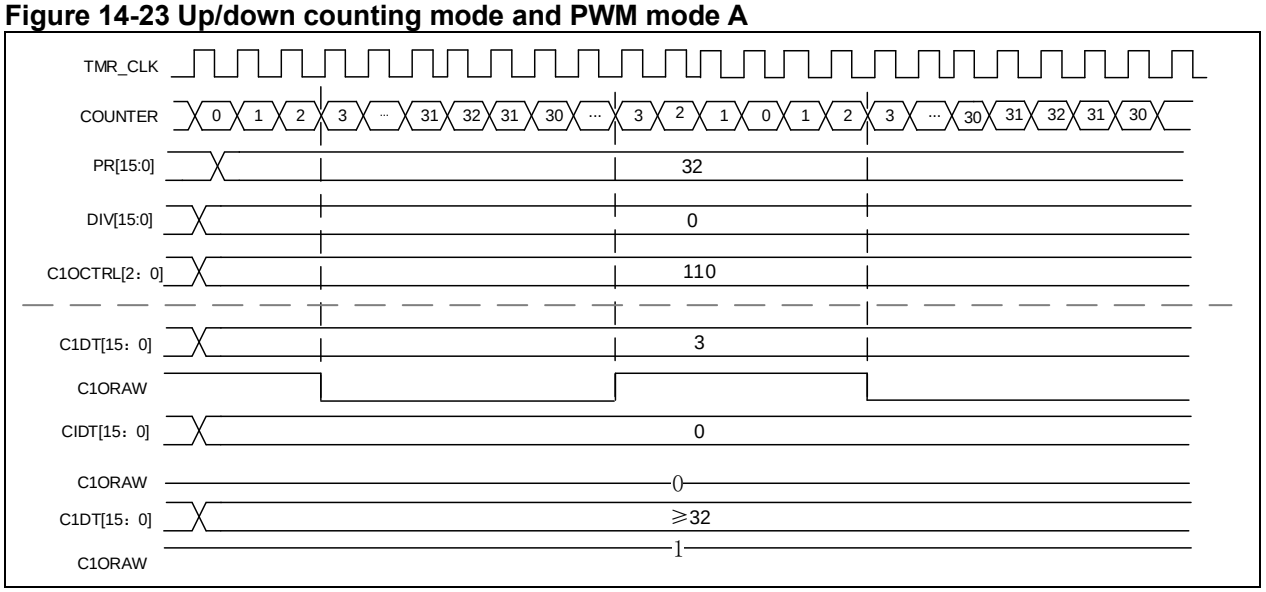
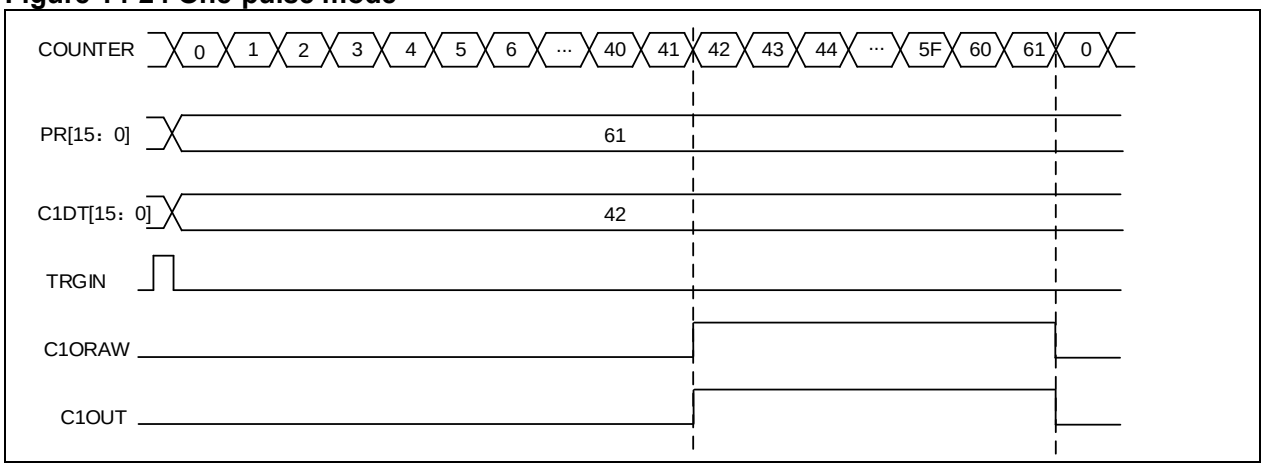


Figure 14-24 One-pulse mode



Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRxCTRL2 register.

-PTOS=3'b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx_SWEVT

register).

-PTOS=3'b001, TRGOUT outputs counter enable signal.

-PTOS=3'b010, TRGOUT outputs counter overflow event.

-PTOS=3'b011, TRGOUT outputs capture and compare event.

-PTOS=3'b100, TRGOUT outputs C1ORAW signal.

-PTOS=3'b101, TRGOUT outputs C2ORAW signal.

-PTOS=3'b110, TRGOUT outputs C3ORAW signal.

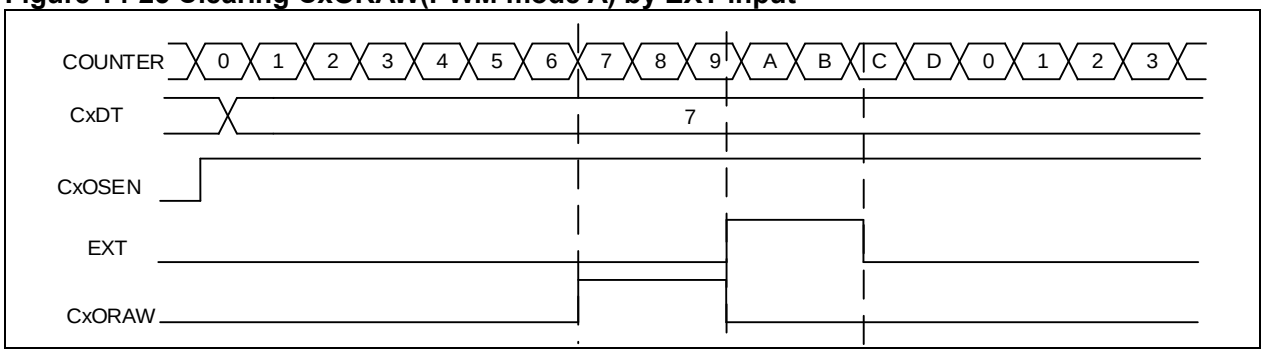
-PTOS=3'b111, TRGOUT outputs C4ORAW signal.

CxORAW clear

When the CxOSEN bit is set to 1, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced mode. *Figure 5* shows the example of clearing CxORAW signal. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

Figure 14-25 Clearing CxORAW(PWM mode A) by EXT input



14.1.3.5TMR synchronization

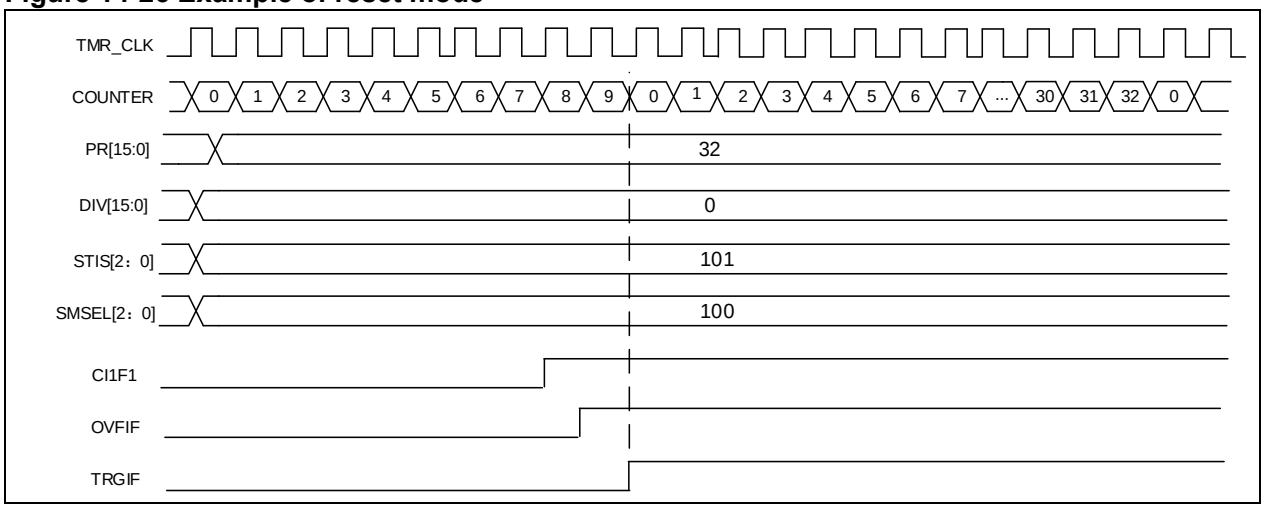
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave mode include:

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

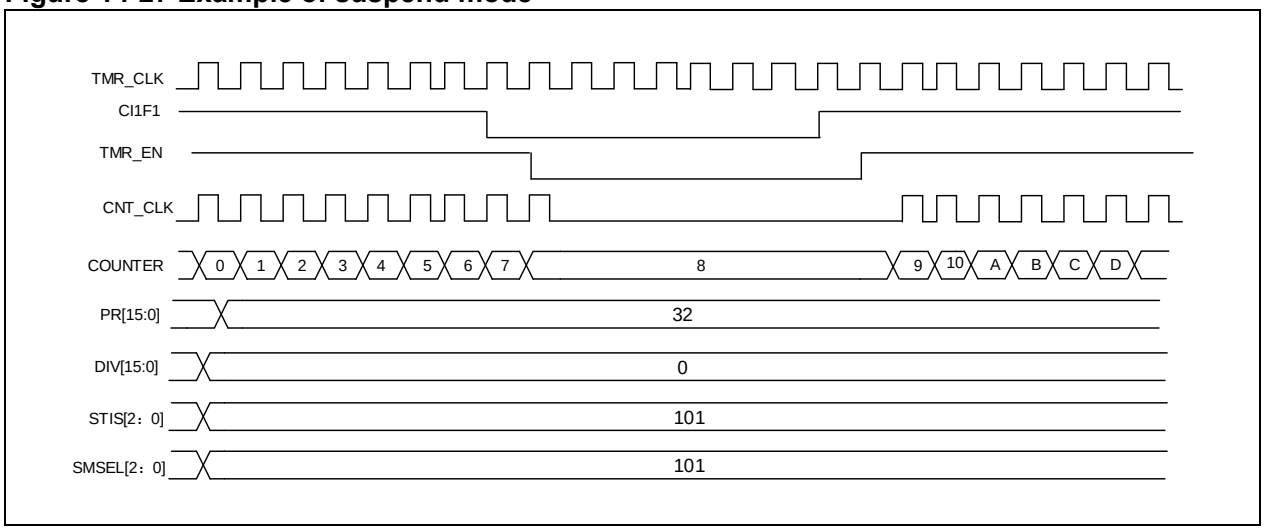
Figure 14-26 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

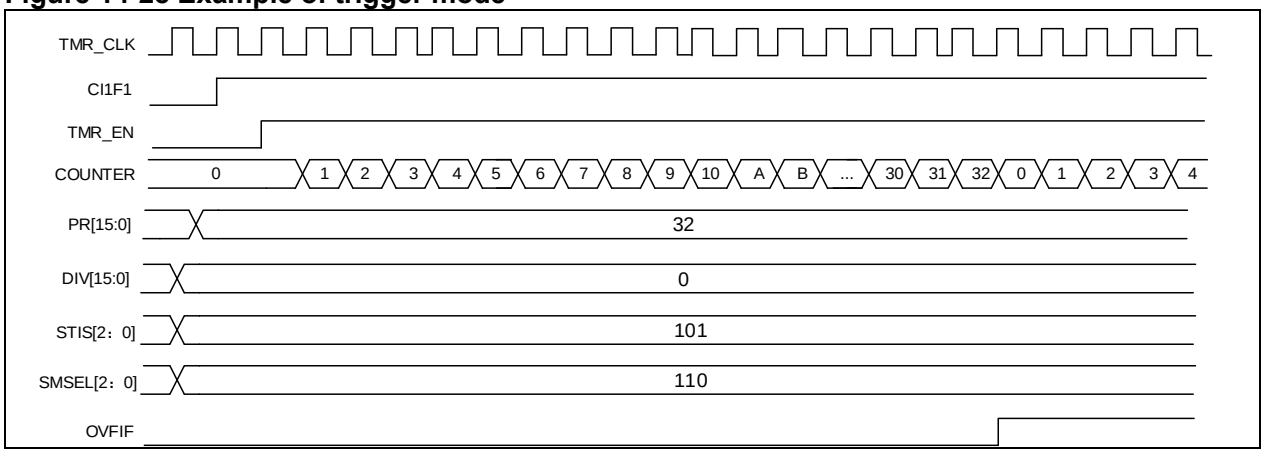
Figure 14-27 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

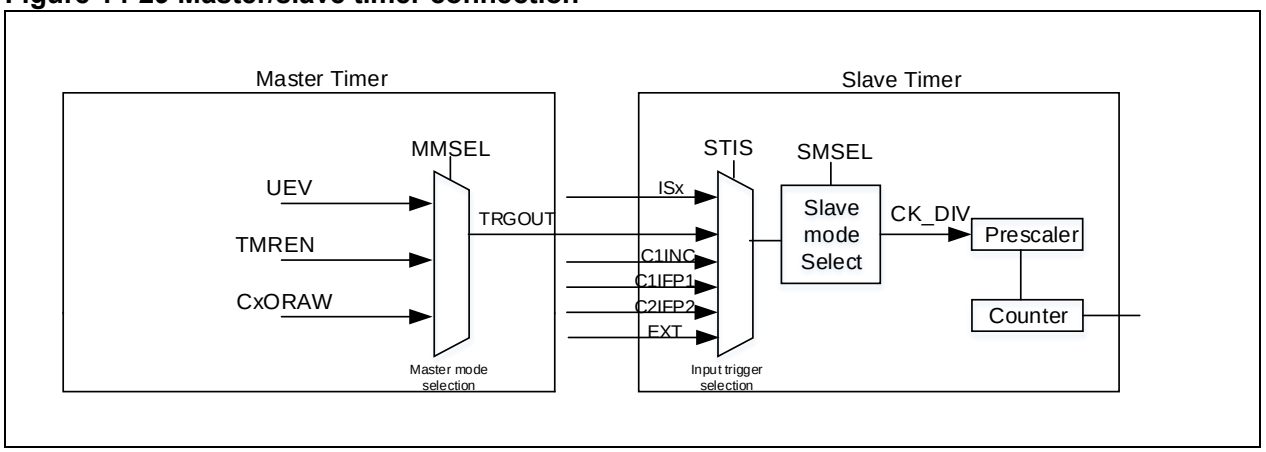
Figure 14-28 Example of trigger mode



Master/slave timer interconnection

Both Master and slave timer can be configured in different master and slave modes respectively. The combination of both them can be used for various purposes. *Figure 9* provides an example of interconnection between master timer and slave timer.

Figure 14-29 Master/slave timer connection



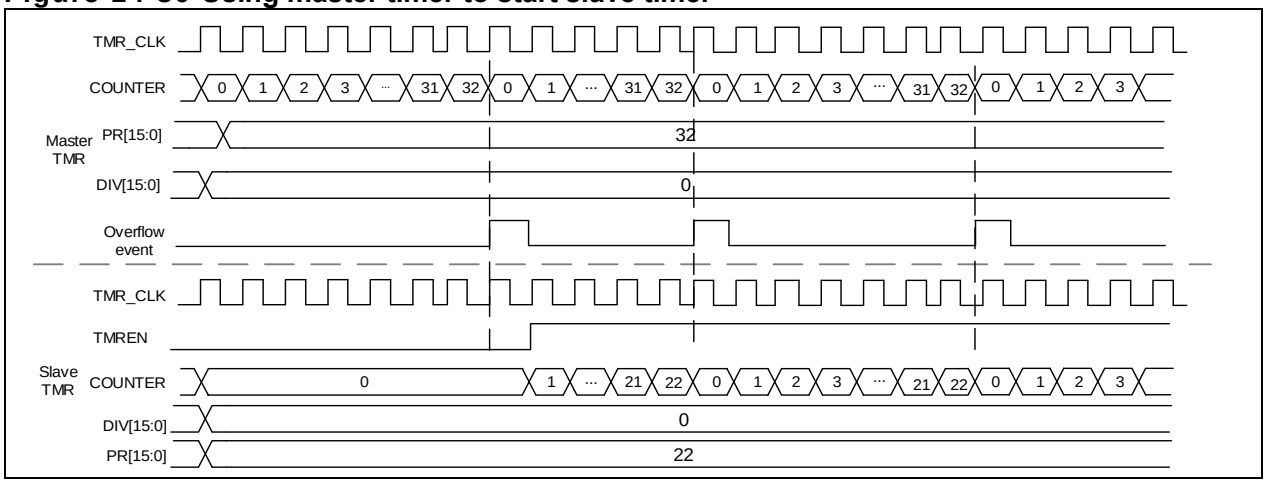
Using master timer to clock the slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3’b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the master timer counting period (TMRx_PR registers)
- Configure the slave timer trigger input signal TRGIN as master timer output (STIS[2: 0] in the TMRx_STCTRL register)
- Configure the slave timer to use external clock mode A (SMSEL[2: 0]=3’b111 in the TMRx_STCTRL register)
- Set TMREN =1 in both master timer and slave timer to enable them

Using master timer to start slave timer:

- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3’b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure master timer counting period (TMRx_PR registers)
- Configure slave timer trigger input signal TRGIN as master timer input
- Configure slave timer as trigger mode (SMSEL=3’b110 in the TMR2_STCTRL register)
- Set TMREN=1 to enable master timer.

Figure 14-30 Using master timer to start slave timer

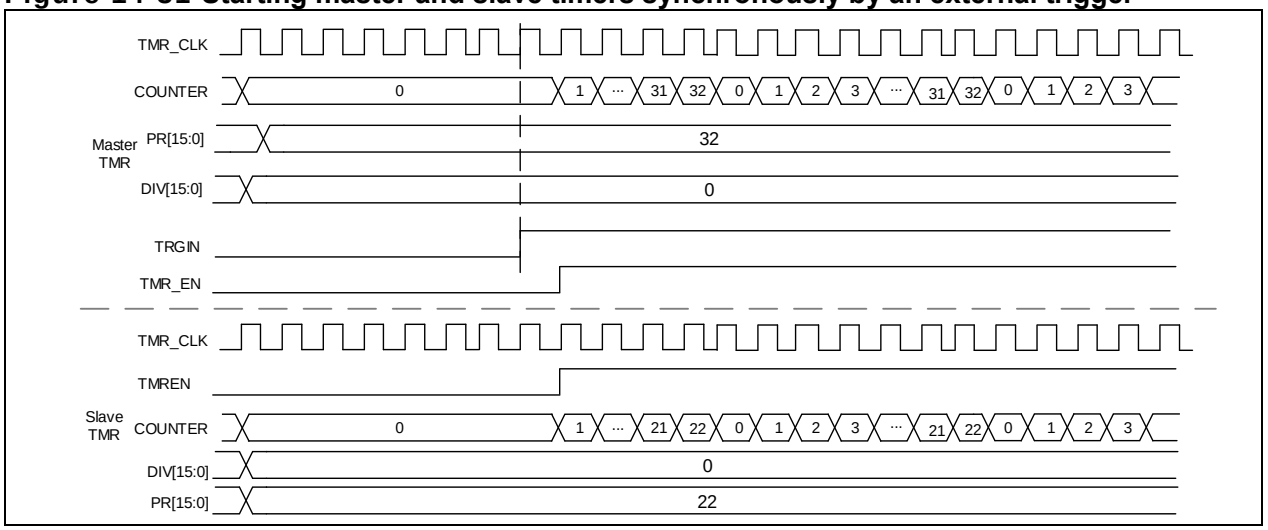


Starting master and slave timers synchronously by an external trigger:

In this example, configure the master timer as master/slave mode synchronously and enable its slave timer synchronization function. This mode is used for synchronization between master timer and slave timer.

- Set the STS bit of the master timer.
- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3’b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.
- Configure the slave timer mode of the master timer as trigger mode, and select C1IN as trigger source
- Configure slave timer trigger input signal TRGIN as master timer output
- Configure slave timer as trigger mode (SMSEL=3’b110 in the TMR2_STCTRL register)

Figure 14-31 Starting master and slave timers synchronously by an external trigger



14.1.3.6 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module.

14.1.4 TMRx registers

These peripheral registers must be accessed by word (32 bits).

TMRx register are mapped into a 16-bit addressable space.

Table 14-4 TMRx register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000
TMRx_STCTRL	0x08	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CM2	0x1C	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000 0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000 0000
TMRx_C1DT	0x34	0x0000 0000
TMRx_C2DT	0x38	0x0000 0000
TMRx_C3DT	0x3C	0x0000 0000
TMRx_C4DT	0x40	0x0000 0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000

14.1.4.1 Control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x00	resd	Kept at its default value.
Bit 10	PMEN	0x0	rw	Plus Mode Enable This bit is used to enable TMRx plus mode. In this mode, TMRx_CVAL, TMRx_PR and TMRx_CxDT are extended from 16-bit to 32-bit. 0: Disabled 1: Enabled Note: This function is only valid for TMR2 and TMR5. It is not applicable to other TMRs. In plus mode or when disabled, only 16-bit value can be written to TMRx_CVAL, TMRx_PR and TMRx_CxDT registers.
Bit 9: 8	CLKDIV	0x0	rw	Clock division This field is used to define the relationship between digital filter sampling frequency (f_{DTS}) and timer clock frequency (f_{CK_INT}). 00: No division, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 5	TWCMSEL	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the CxIF bit is set only when the counter counts down 10: Two-way counting mode2, count up and down alternately, the CxIF bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the CxIF bit is set when the counter counts up / down
Bit 4	OWCDIR	0x0	rw	One-way count direction 0: Up 1: Down
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an overflow event 0: The counter does not stop at an overflow event 1: The counter stops at an overflow event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.1.4.2 Control register2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	C1INSEL	0x0	rw	C1IN selection

				0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable 010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2: 0	Reserved	0x0	resd	Kept at its default value.

14.1.4.3 Slave timer control register (TMRx_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4 11: Divided by 8
Bit 11: 8	ESF	0x0	rw	External signal filter This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times 0000: No filter, sampling by f_{DTS} 0001: $f_{SAMPLING}$ =f_{CK_INT}, N=2 0010: $f_{SAMPLING}$ =f_{CK_INT}, N=4 0011: $f_{SAMPLING}$ =f_{CK_INT}, N=8 0100: $f_{SAMPLING}$ =f_{DTS}/2, N=6 0101: $f_{SAMPLING}$ =f_{DTS}/2, N=8 0110: $f_{SAMPLING}$ =f_{DTS}/4, N=6 0111: $f_{SAMPLING}$=f_{DTS}/4, N=8 1000: $f_{SAMPLING}$=f_{DTS}/8, N=6 1001: $f_{SAMPLING}$=f_{DTS}/8, N=8 1010: $f_{SAMPLING}$=f_{DTS}/16, N=5 1011: $f_{SAMPLING}$=f_{DTS}/16, N=6 1100: $f_{SAMPLING}$=f_{DTS}/16, N=8 1101: $f_{SAMPLING}$=f_{DTS}/32, N=5 1110: $f_{SAMPLING}$=f_{DTS}/32, N=6 1111: $f_{SAMPLING}$=f_{DTS}/32, N=8
Bit 7	STS	0x0	rw	Subordinate TMR synchronization If enabled, master and slave timer can be synchronized. 0: Disabled 1: Enabled
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1)

				010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: External input (EXT) Please refer to *Table 14-2* for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for the details on encoder mode A/B/C.

14.1.4.4DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at its default value
Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled

				1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.1.4.5 Interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	Reserved	0x0	resd	Kept at its default value
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register: − An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; − An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

14.1.4.6 Software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.1.4.7 Channel mode register1 (TMRx_CM1)

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input,C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	C1OSEN	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT 1: Once high level is detect on EXT input, clear C1ORAW.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A －OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; －OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B

				—OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; —OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6

				0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.1.4.8 Channel mode register2 (TMRx_CM2)

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
Bit 1: 0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter

Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider
Bit 1:0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.1.4.9 Channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11: 10	Reserved	0x0	resd	Kept at its default value.
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit 0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-5 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0, Cx_EN=0)
1	CxOUT = CxORAW + polarity, Cx_EN=1

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.1.4.10 Counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	CVAL	0x0000	rw	Counter value When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the CVAL is expanded to 32 bits.
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.1.4.11 Division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.1.4.12 Period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 31: 16	PR	0x0000	rw	Period value When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the PR is expanded to 32 bits.
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.1.4.13 Channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C1DT	0x0000	rw	Channel 1 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C1DT is expanded to 32 bits.
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.1.4.14 Channel 2 data register (TMRx_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C2DT	0x0000	rw	Channel 2 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C2DT is expanded to 32 bits.
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.1.4.15 Channel 3 data register (TMRx_C3DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C3DT	0x0000	rw	Channel 3 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C3DT is expanded to 32 bits.
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

14.1.4.16 Channel 4 data register (TMRx_C4DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C4DT	0x0000	rw	Channel 4 data register When TMR2 or TMR5 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C4DT is expanded to 32 bits.
Bit 15: 0	C4DT	0x0000	rw	Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN) When the channel 4 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

14.1.4.17 DMA control register (TMRx_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12: 8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte 00001: 2 bytes 00010: 3 bytes 00011: 4 bytes .... 10000: 17 bytes 10001: 18 bytes
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register. 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

14.1.4.18 DMA data register (TMRx_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A read or write operation to the DMADT register accesses the TMR registers at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4.

14.2 General-purpose timer (TMR9 to TMR11)

14.2.1 TMRx introduction

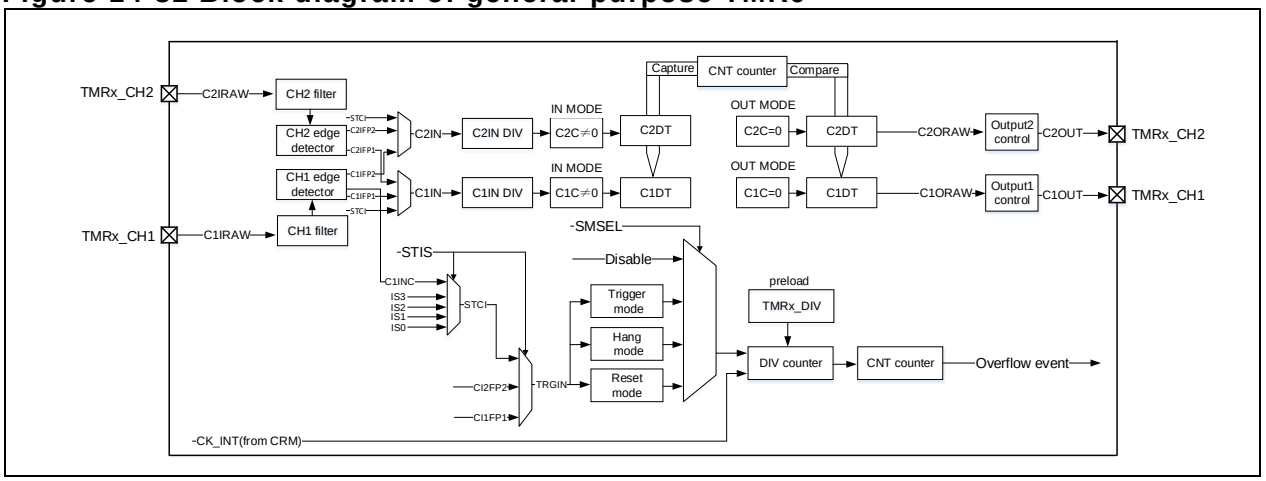
The general-purpose timer (TMR9 to TMR11) consists of a 16-bit counter supporting upcounting mode. These timers can be synchronized.

14.2.2 TMRx main features

14.2.2.1 TMR9 main features

- Source of counter clock: internal clock and external clock
- 16-bit up counter
- 2 independent channels for input capture, output compare, PWM generation and one-pulse mode output
- Synchronization control between master and slave timers
- Interrupt is generated at overflow event, trigger event and channel event

Figure 14-32 Block diagram of general-purpose TMR9

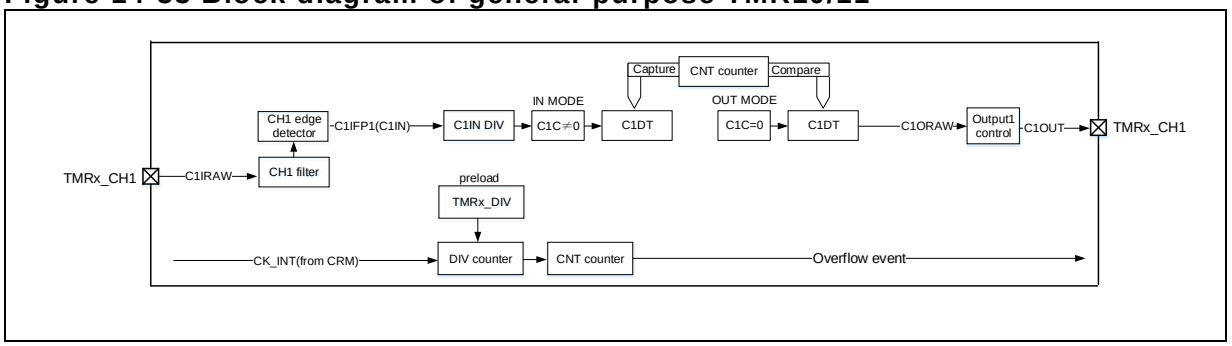


14.2.2.2 TMR10 and TMR11 main features

The main functions of general-purpose TMRx (TMR10 and TMR11) include:

- Source of counter clock: internal clock
- 16-bit up counter
- 1 independent channel for input capture, output compare, PWM generation
- Synchronization control between master and slave timers
- Interrupt is generated at overflow event and channel event

Figure 14-33 Block diagram of general-purpose TMR10/11

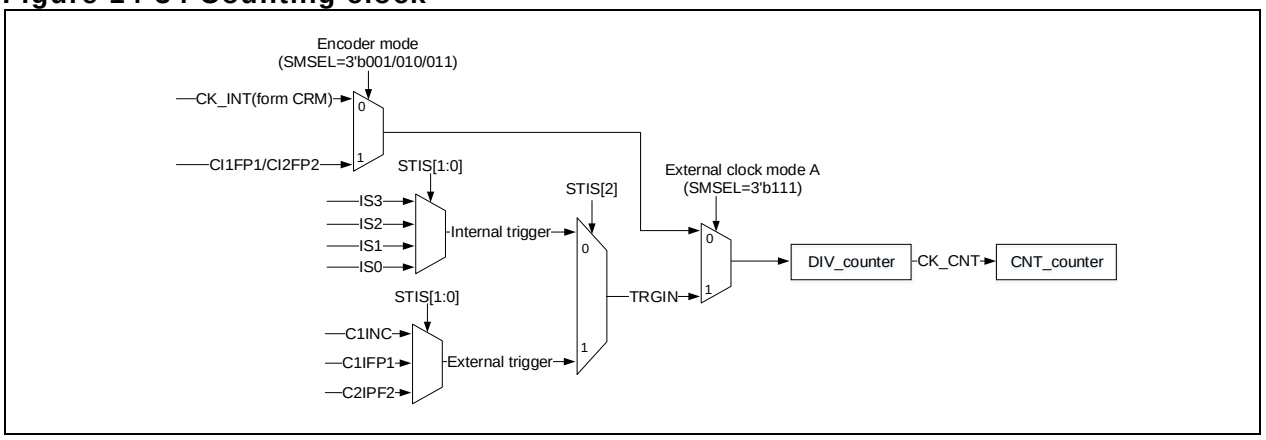


14.2.3 TMRx functional overview

14.2.3.1 Counting clock

The count clock of general-purpose timers can be provided by the internal clock (CK_INT), external clock (external clock mode A) and internal trigger input (ISx)

Figure 14-34 Counting clock



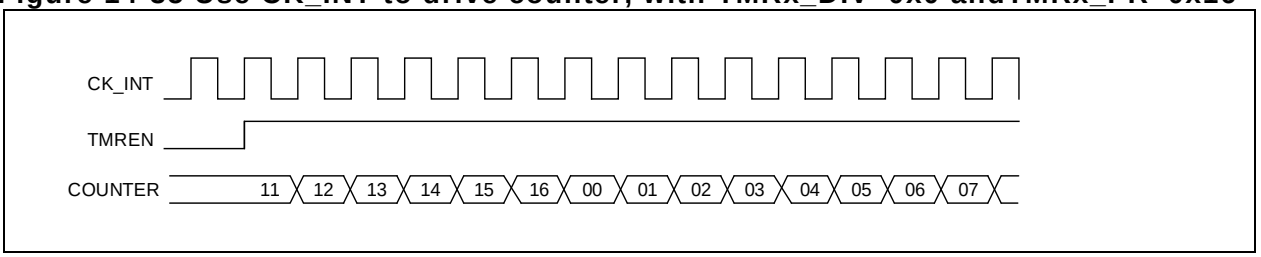
Internal clock (CK_INT)

By default, the CK_INT divided by the prescaler is used to drive the counter to start counting. When TMR's APB clock prescaler factor is 1, the CK_INT frequency is equal to that of APB, otherwise, it doubles the APB clock frequency.

The configuration process is as follows:

- Set the CLKDIV[1:0] bit in the TMRx_CTRL1 register to set the CK_INT frequency;
- Set the TMRx_DIV register to set the counting frequency;
- Set the TMRx_PR register to set the counting period;
- Set the TMREN it in the TMRx_CTRL1 register to enable the counter.

Figure 14-35 Use CK_INT to drive counter, with TMRx_DIV=0x0 andTMRx_PR=0x16



External clock (TMR9 only)

The counter clock can be provided by TRGIN signal.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by setting the STIS[2: 0] bit to drive the counter to start counting.

The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, the channel 1 signal with filtering and polarity selection) and C2IFP2 (STIS=3'b110, a channel 2 signal with filtering and polarity selection).

To use external clock mode A, follow the steps below:

-Set external source TRGIN parameters

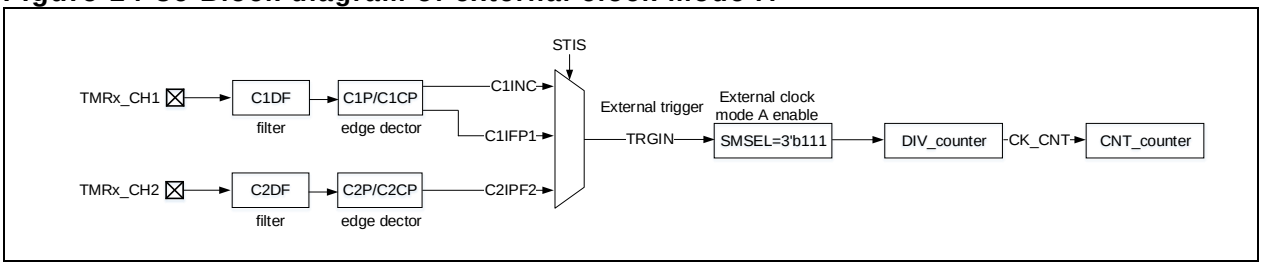
If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter (C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register

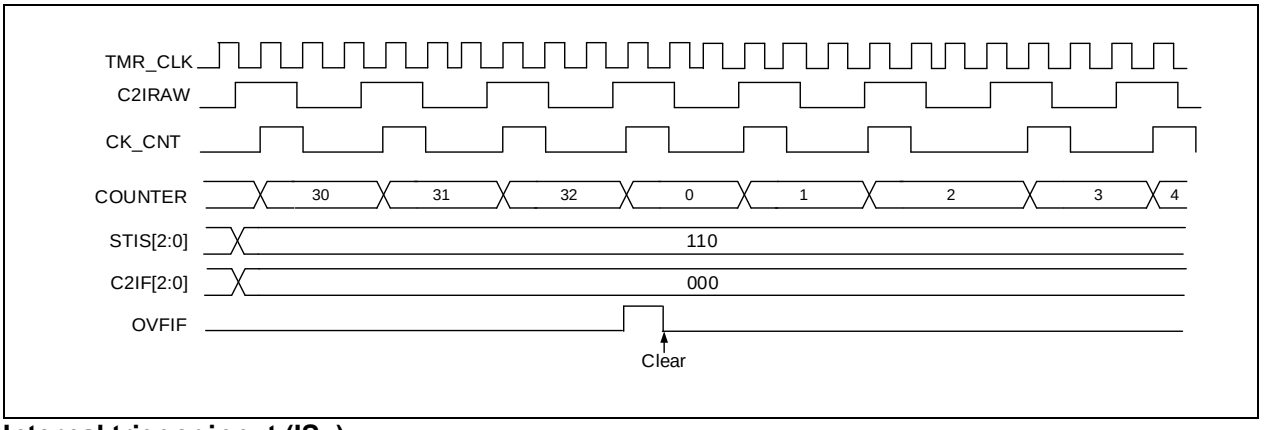
-Enable counter through the TMREN bit in TMRx_CTRL1 register

Figure 14-36 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-37 Counting in external clock mode A



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each general-purpose timer consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

The internal trigger input is configured as follows:

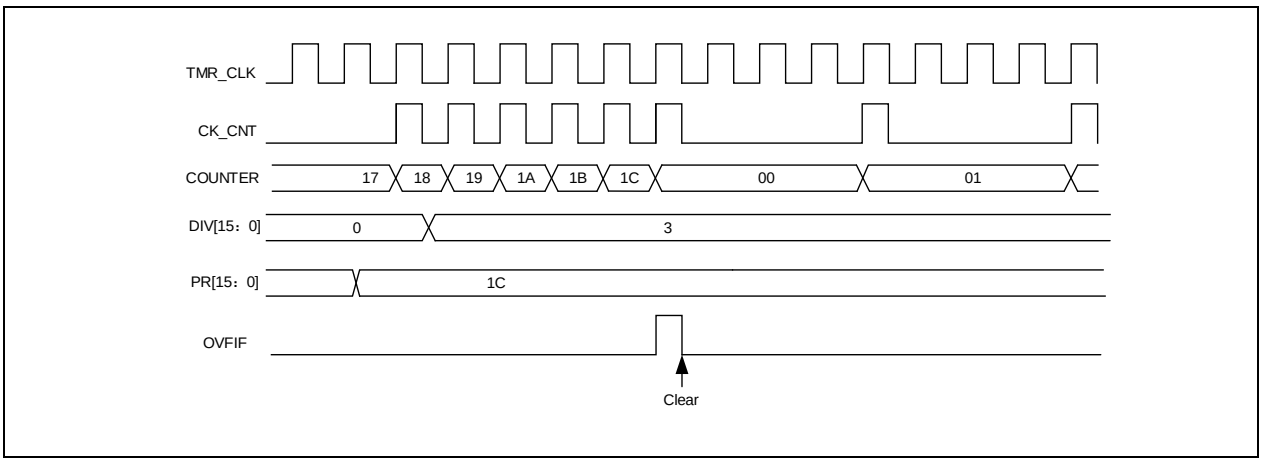
- Set the TMRx_PR register to set the counting period;
- Set the TMRx_DIV register to set the counting frequency;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.

Table 14-6 TMRx internal trigger connection

Slave controller	IS0 (STIS=000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR9	TMR2	TMR3	TMR10_OC	TMR11_OC

Note: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

Figure 14-38 Counter timing with prescaler value changing from 1 to 4



14.2.3.2 Counting mode

The timer (TMR9 ~ TMR11) supports upcounting mode only, and each timer has an internal 16-bit up counter.

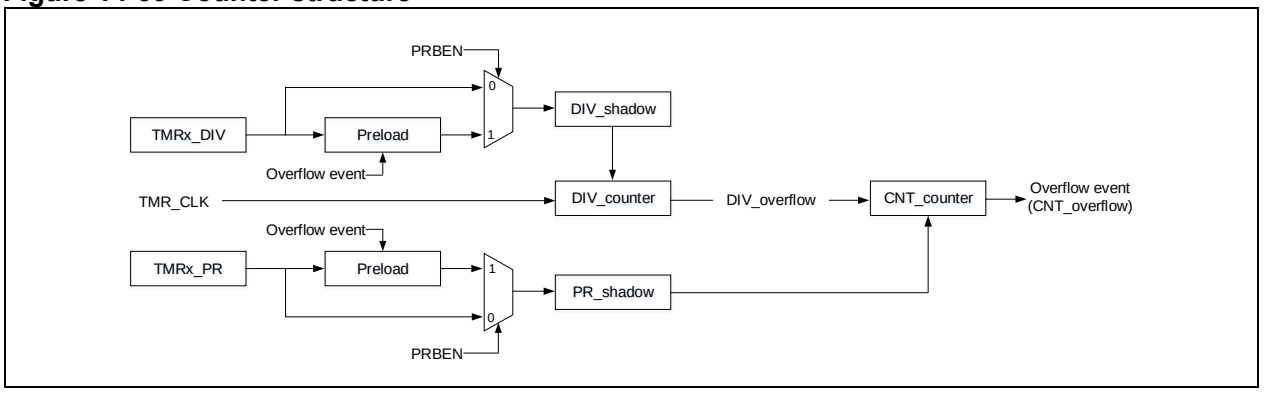
The TMRx_PR register is used to configure the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register only at an overflow event.

The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.
Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-39 Counter structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, then restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event, otherwise, the counter is updated with the preload value and period value on an overflow event.

Figure 14-40 Overflow event when PRBEN=0

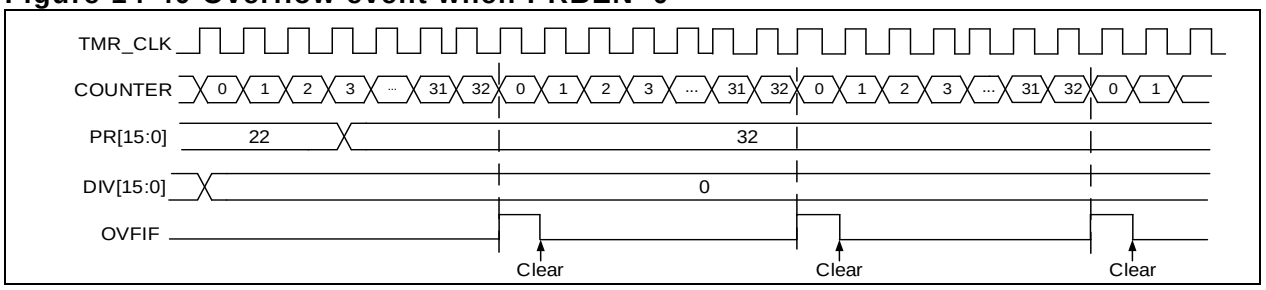
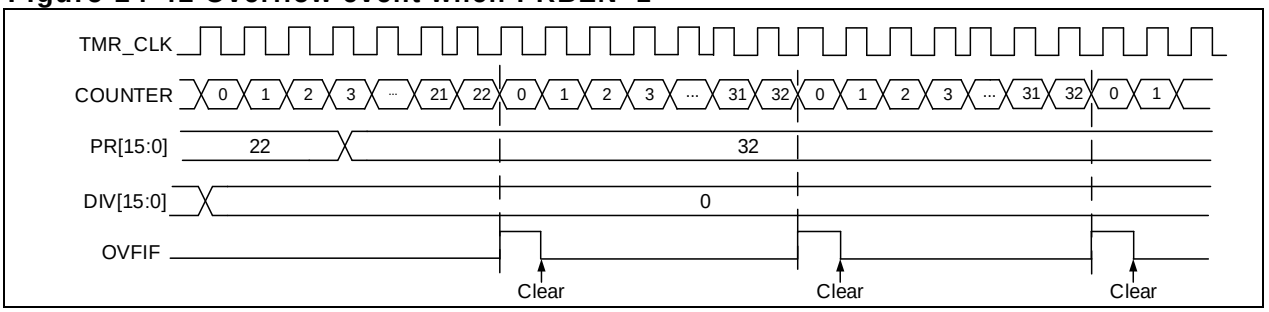


Figure 14-41 Overflow event when PRBEN=1



14.2.3.3TMR input function

TMR9 has two independent channels. Each timer of TMR10 and TMR11 has an independent channel. Each channel can be configured as input or output.

As input, each channel input is handle as follows:

- TMRx_CHx outputs CxIRAW after being preprocessed. Select the TMRx_CHx for CxIRAW through the C1INSEL bit
- CxIRAW inputs digital filter and outputs filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF inputs edge detector, and outputs the CxIFPx signal after edge selection. The edge selection depends on both CxP and CxCP bits. It is possible to select input rising edge, falling edge or both edges.
- CxIFPx inputs capture signal selector, and outputs the CxIN signal after capture sigal selection. The capture signal selection is defined by CxC bits. It is possible to select CxIFPx, CyIFPx or STCI as CxIN source. Of those, CyIFPx (x≠y) is the CyIFPy signal that is from Y channel and processed by channel-x edge detector (For example, the C1IFP2 is the Channle 1's C1IFP1 signal that passed through channel 2 edge detection). The STCI comes from slave timer controller, and its source is selected by STIS bit. For a single channel TMR, only CxIFPx can be selected as the source of CxIN.
- CxIN outputs the CxIPS signal that is divided by input channel divider. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit.

Figure 14-42 Input/output channel 1 main circuit

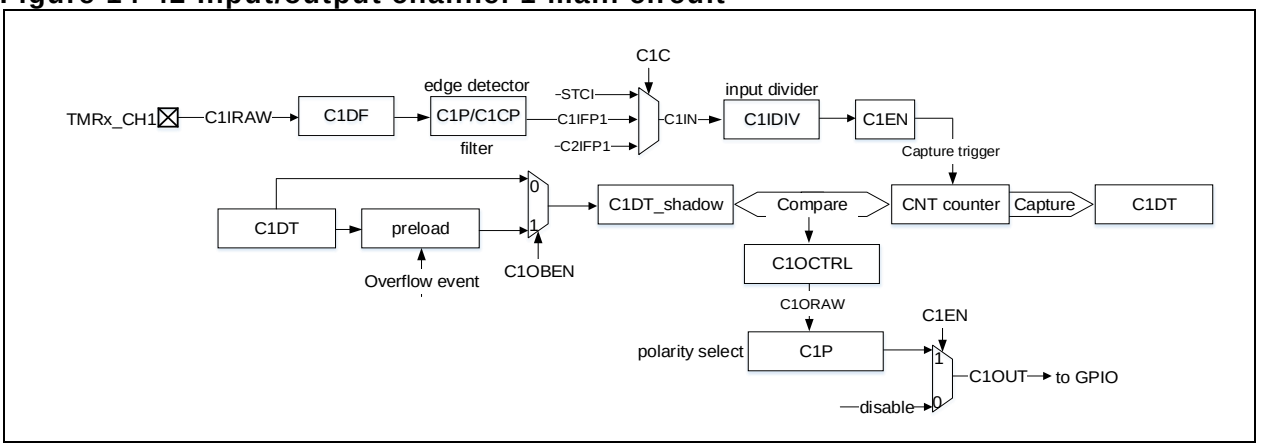
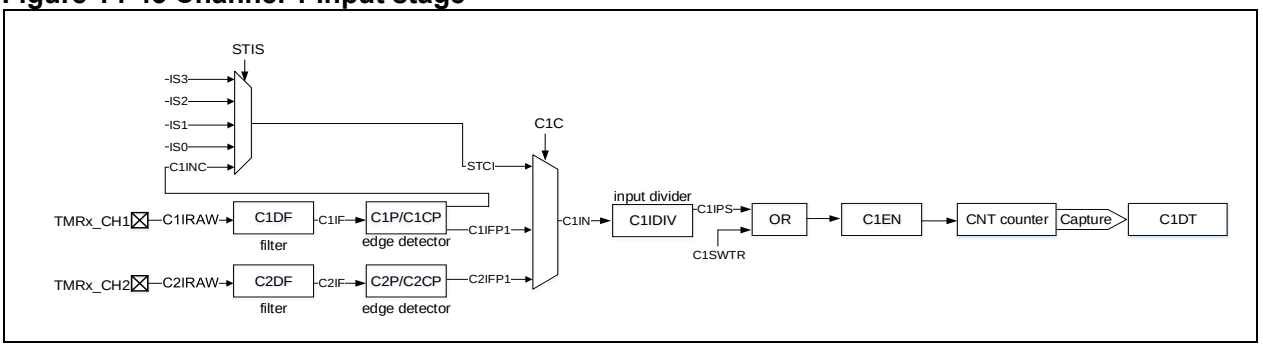


Figure 14-43 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt or a DMA request will be generated if the CxIEN or CxDEN bit is enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR register
- Program the capture frequency of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt by setting the C1IEN bit in the TMRx_IDEN register

PWM input (TMR9)

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2‘b01 and set C1IN as C1IFP1;
- Set C1P=1’b0 and set C1IFP1 rising edge active;
- Set C2C=2‘b10 and set C2IN as C1IFP2;
- Set C2P=1’b1 and set C1IFP2 falling edge active;
- Set STIS=3’b101 and set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3‘b100 and set the slave timer in reset mode;
- Set C1EN=1’b1 and C2EN=1’b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-44 Example of PWM input mode configuration

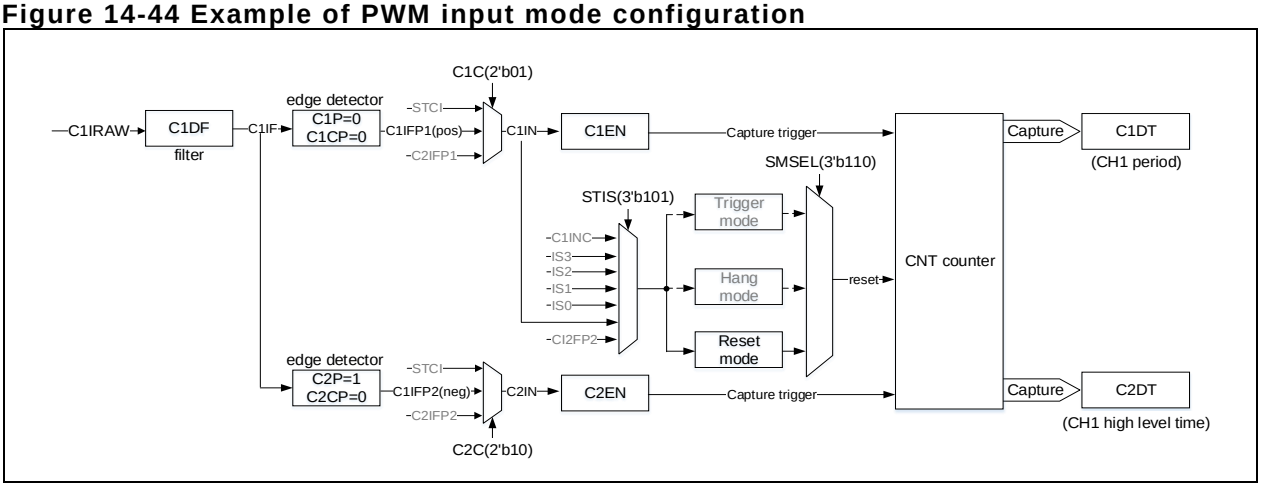
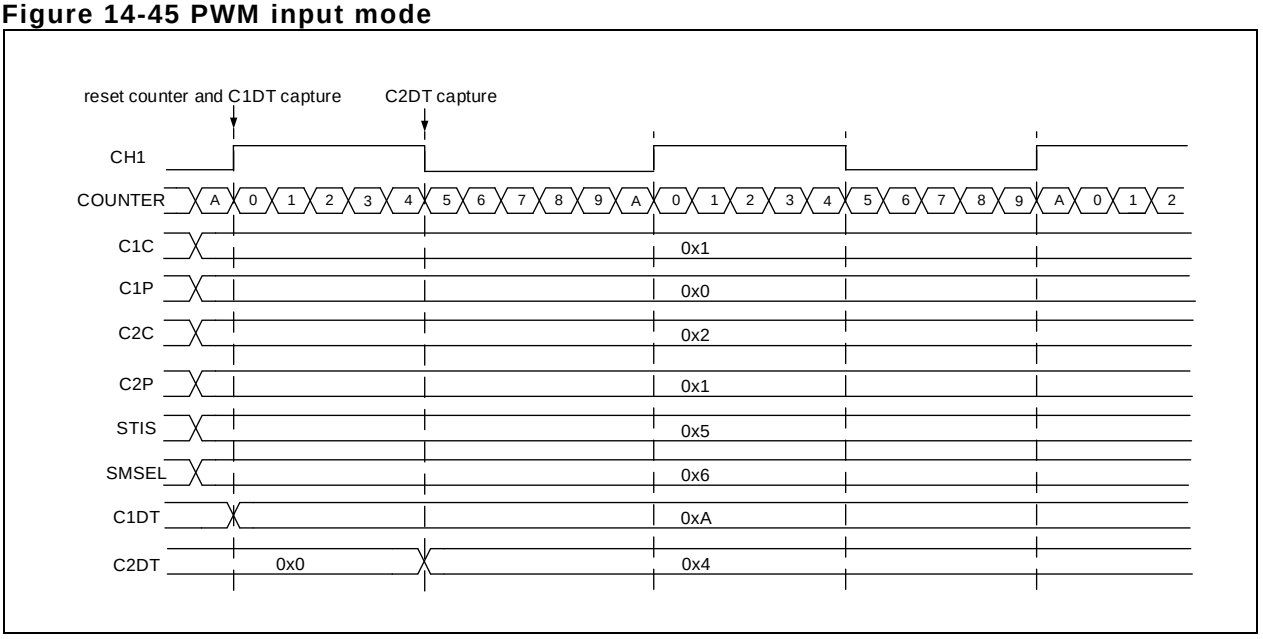


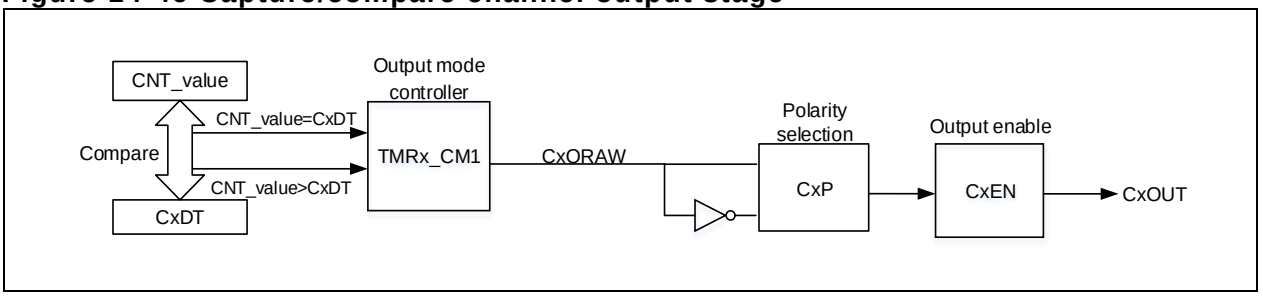
Figure 14-45 PWM input mode



14.2.3.4 TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal.

Figure 14-46 Capture/compare channel output stage



Output mode

Write CxC[2: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A**: Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in TMRx_CM1/CM2 register and set output mode as PWM mode A;
  - Set the TMRx_DIV register to set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 to set the count mode;
  - Set the CxP and CxCP bits in the TMRx_CCTRL register to set the output polarity;
  - Set the CxEN and CxCEN bits in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B**: Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=3'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.
- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode (TMR9/12 only):** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode (TMR9/12 only):** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 47 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 48 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 49 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-47 C1ORAW toggles when counter value matches the C1DT value

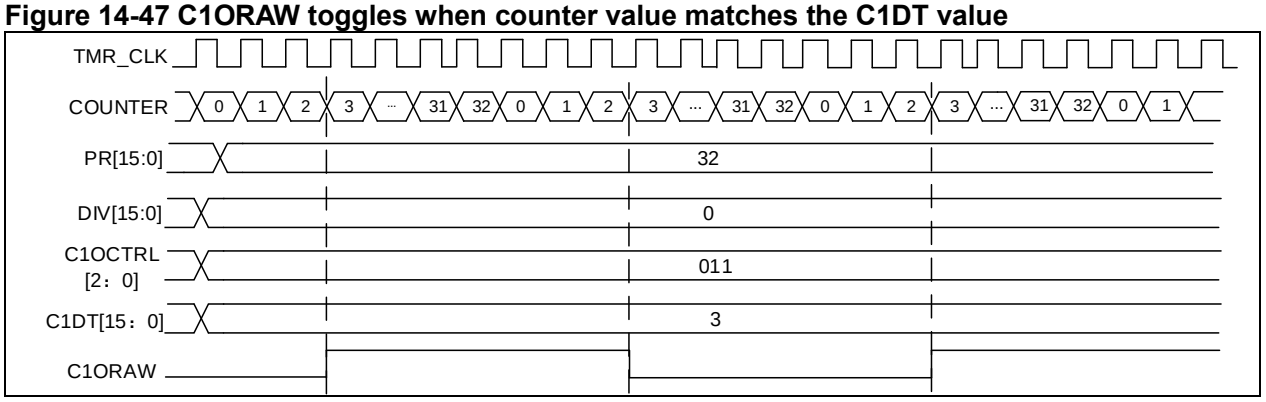


Figure 14-48 Upcounting mode and PWM mode A

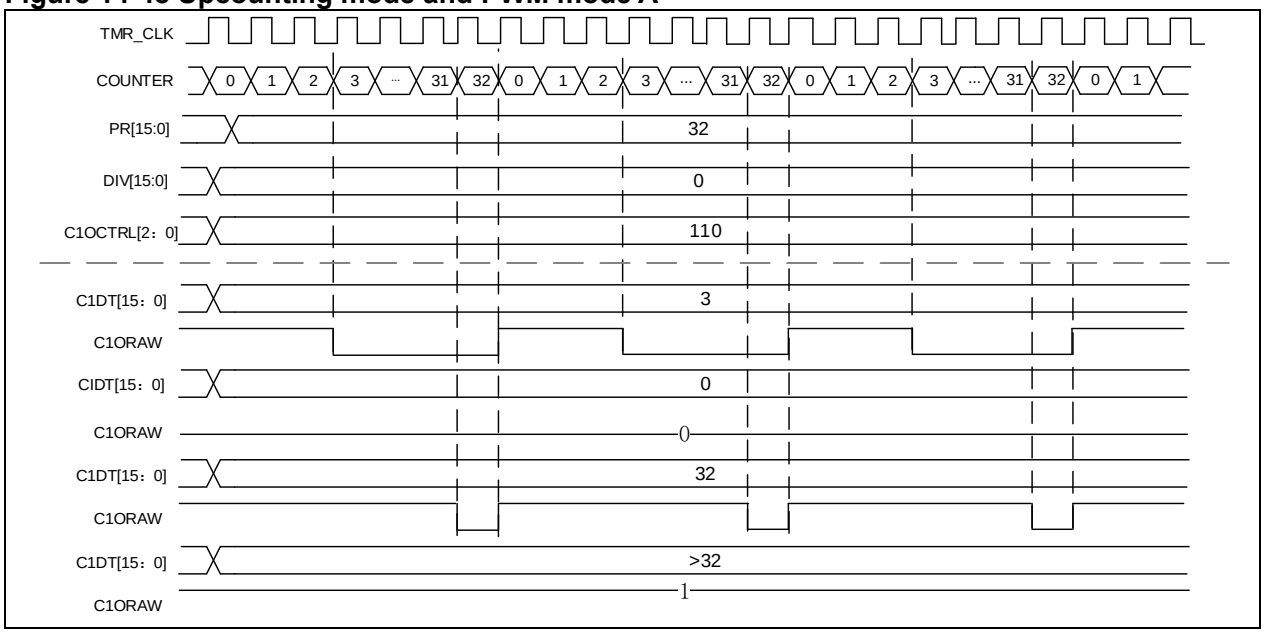
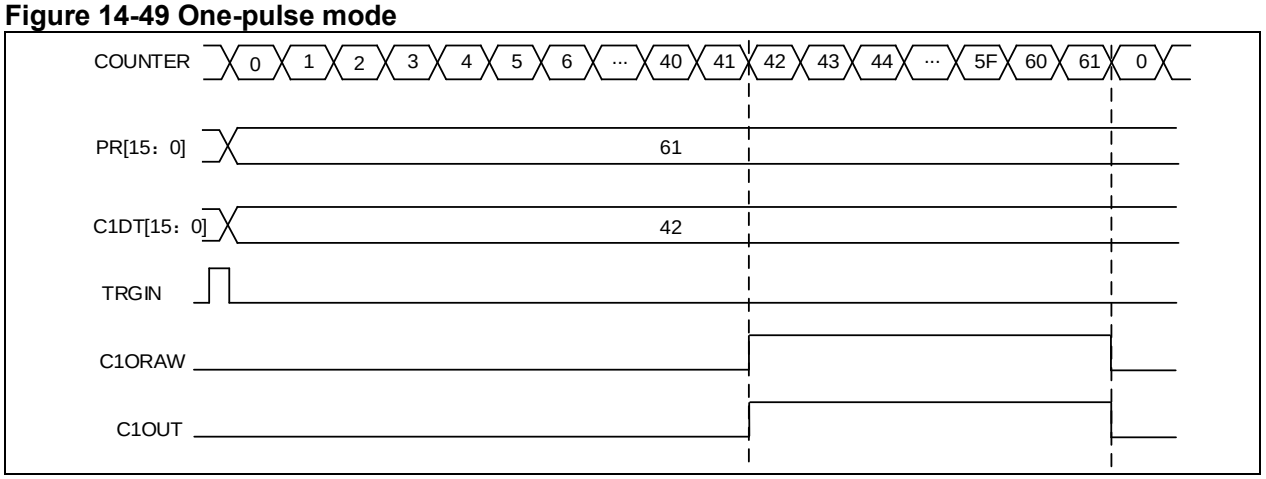


Figure 14-49 One-pulse mode



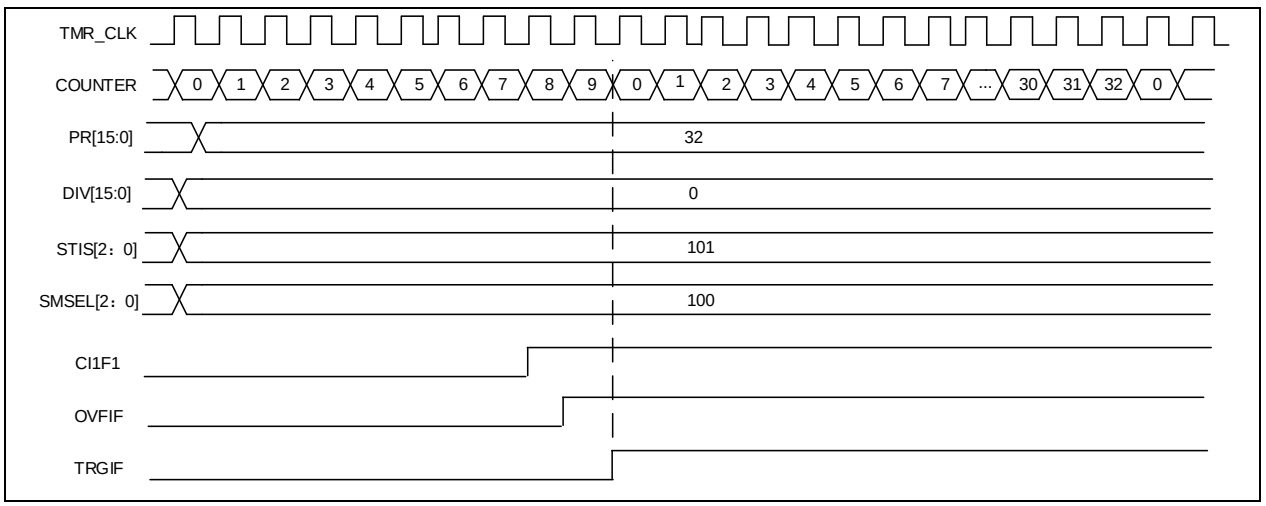
14.2.3.5TMR synchronization

TMR9 can be used as a slave timer to synchronize with master timer through internal signals. Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event is generated when OVFS=0.

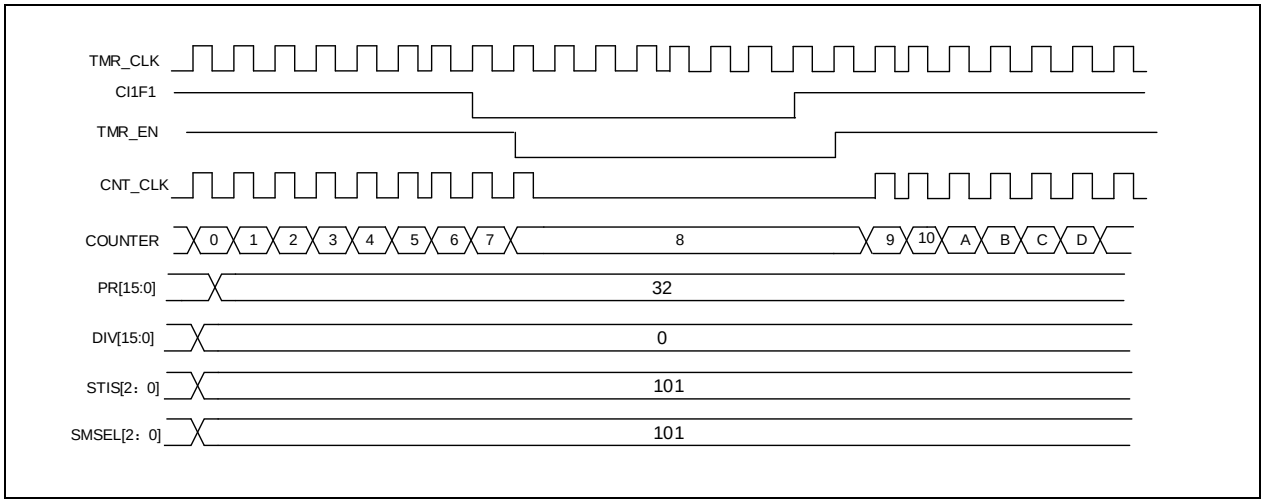
Figure 14-50 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

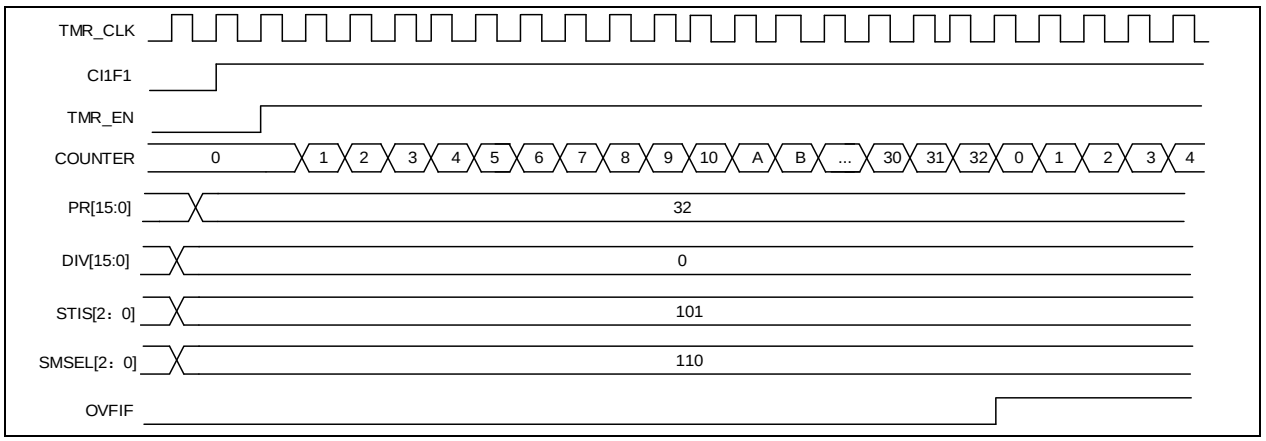
Figure 14-51 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-52 Example of trigger mode



Please refer to *Section 14.2.3.5* for more information on timer synchronization.

14.2.3.6 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module.

14.2.4 TMR9 registers

These peripheral registers must be accessed by word (32 bits).
TMR9 registers are mapped into a 16-bit addressable space.

Table 14-7 TMR9 register map and reset value

Register name	Register	Reset value
TMR9_CTRL1	0x00	0x0000
TMR9_STCTRL	0x08	0x0000
TMR9_IDEN	0x0C	0x0000
TMR9_ISTS	0x10	0x0000
TMR9_SWEVT	0x14	0x0000
TMR9_CM1	0x18	0x0000
TMR9_CCTRL	0x20	0x0000
TMR9_CVAL	0x24	0x0000
TMR9_DIV	0x28	0x0000
TMR9_PR	0x2C	0x0000
TMR9_C1DT	0x34	0x0000 0000
TMR9_C2DT	0x38	0x0000 0000

14.2.4.1 Control register1 (TMR9_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
Bit 9: 8	CLKDIV	0x0	rw	Clock divider This field is used to define the relationship between digital filter sampling frequency (f_{DTS}) and timer clock frequency (f_{CK_INT}). 00: No division, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 4	Reserved	0x0	resd	Kept at its default value
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Enabled 1: Disabled

14.2.4.2 Slave timer control register (TMR9_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x000	resd	Kept at its default value
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: Reserved Please refer to *Table 14-7* for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at its default value
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode — Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode — The counter starts counting when the TRGIN is high 110: Trigger mode — A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A — Rising edge of the TRGIN input clocks the counter Note: Please refer to count mode section for details on encoder mode A/B/C.

14.2.4.3 DMA/interrupt enable register (TMR9_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.2.4.4 Interrupt status register (TMR9_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated.

14.2.4.5 Software event register (TMR9_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5:3	Reserved	0x0	resd	Kept at its default value.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.2.4.6 Channel mode register1 (TMR9_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN=’0’: 00: Output 01: Input,C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A

				– OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; – OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B – OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; – OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6

				0001: $f_{SAMPLING}$=f_{CK_INT}, N=2 1001: $f_{SAMPLING}$=f_{DTS}/8, N=8 0010: $f_{SAMPLING}$=f_{CK_INT}, N=4 1010: $f_{SAMPLING}$=f_{DTS}/16, N=5 0011: $f_{SAMPLING}$=f_{CK_INT}, N=8 1011: $f_{SAMPLING}$=f_{DTS}/16, N=6 0100: $f_{SAMPLING}$=f_{DTS}/2, N=6 1100: $f_{SAMPLING}$=f_{DTS}/16, N=8 0101: $f_{SAMPLING}$=f_{DTS}/2, N=8 1101: $f_{SAMPLING}$=f_{DTS}/32, N=5 0110: $f_{SMPLING}$=f_{DTS}/4, N=6 1110: $f_{SAMPLING}$=f_{DTS}/32, N=6 0111: $f_{SAMPLING}$=f_{DTS}/4, N=8 1111: $f_{SAMPLING}$=f_{DTS}/32, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.2.4.7 Channel control register (TMR9_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x00	resd	Kept at its default value.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-8 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0)
1	CxOUT = CxORAW + polarity

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.2.4.8 Counter value (TMR9_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.2.4.9 Division value (TMR9_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.2.4.10 Period register (TMR9_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.2.4.11 Channel 1 data register (TMR9_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.2.4.12 Channel 2 data register (TMR9_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.2.5 TMR10 and TMR11 registers

These peripheral registers must be accessed by word (32 bits).

TMR10 and TMR11 registers are mapped into a 16-bit addressable space.

Table 14-9 TMR10 and TMR11 register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CM1	0x18	0x0000
TMRx_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_C1DT	0x34	0x0000

14.2.5.1 Control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value
Bit 9: 8	CLKDIV	0x0	rw	Clock divider This field is used to define the relationship between digital filter sampling frequency (f_{DTS}) and timer clock frequency (f_{CK_INT}). 00: No division, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 3	Reserved	0x0	resd	Default value
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Enabled 1: Disabled

14.2.5.2 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15:2	Reserved	0x0000	resd	Kept at its default value
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled

1: Enabled

14.2.5.3 Interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated.

14.2.5.4 Software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.2.5.5 Channel mode register1 (TMRx_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15:7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW. 000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A －OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; －OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B －OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; －OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved

				11: Reserved

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events. 0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Reserved 11: Reserved

14.2.5.6 Channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	C1P	0x0	rw	Channel 1 polarity When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable

	0: Input or output is disabled 1: Input or output is enabled

Table 14-10 Standard CxOUT channel output control bit

CxEN bit	CxOUT output state
0	Output disabled (CxOUT=0)
1	CxOUT = CxORAW + polarity

Note: The state of the external I/O pins connected to the standard CxOUT channel depends on the CxOUT channel state and the GPIO and IOMUX registers.

14.2.5.7 Counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.2.5.8 Division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK}$ /(DIV[15: 0]+1). DIV contains the value written at an overflow event.

14.2.5.9 Period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.2.5.10 Channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.3 Advanced-control timers (TMR1)

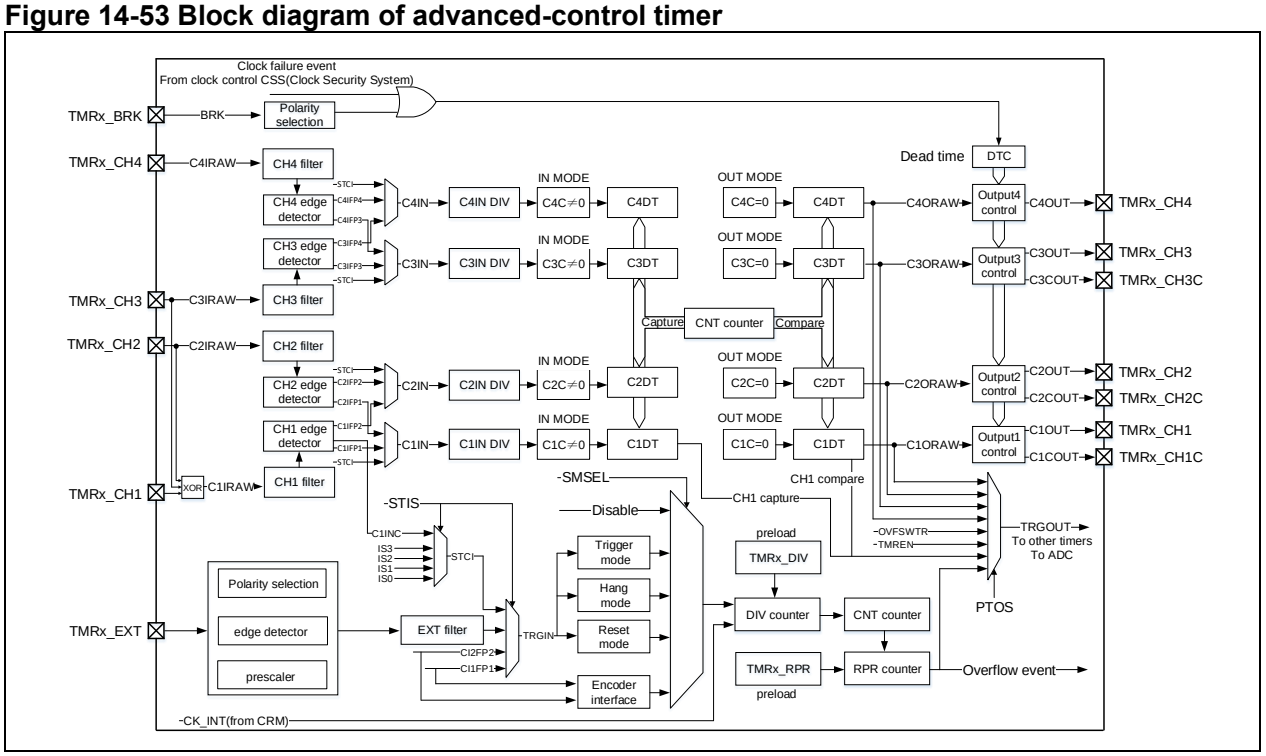
14.3.1 TMR1 introduction

The advanced-control timer TMR1 consists of a 16-bit counter supporting up and down counting modes, four capture/compare registers, and four independent channels to achieve embedded dead-time, input capture and programmable PWM output.

14.3.2 TMR1 main features

- Source of counter clock: internal clock, external clock an internal trigger input
- 16-bit up, down, up/down, repetition and encoder mode counter
- Four independent channels for input capture, output compare, PWM generation, one-pulse mode output and embedded dead-time
- Three independent channels for complementary output
- TMR break function
- Synchronization control between master and slave timers
- Interrupt/DMA is generated at overflow event, trigger event, break signal input and channel event
- Support TMR burst DMA transfer

Figure 14-53 Block diagram of advanced-control timer

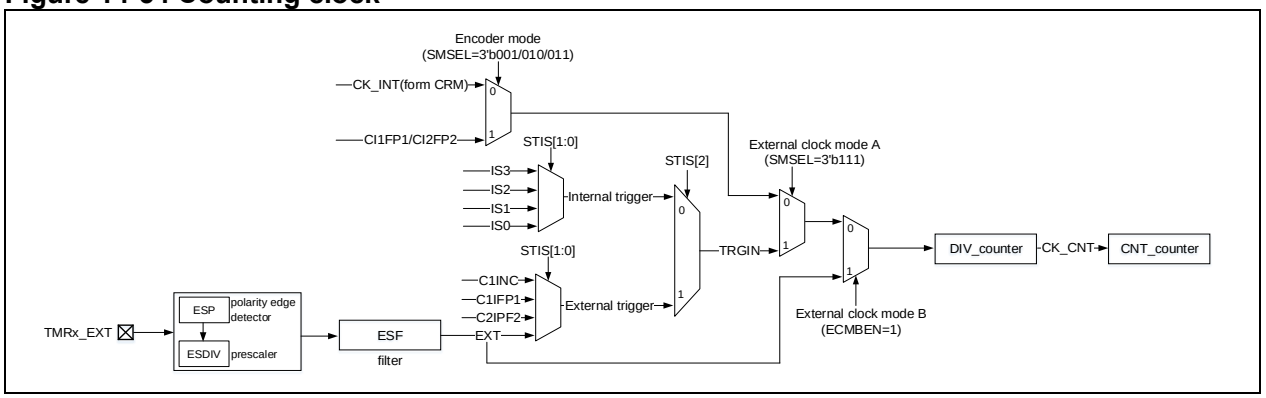


14.3.3 TMR1 functional overview

14.3.3.1 Counting clock

The count clock of TMR1 can be provided by the internal clock (CK_INT), external clock (external clock mode A and B) and internal trigger input (ISx).

Figure 14-54 Counting clock



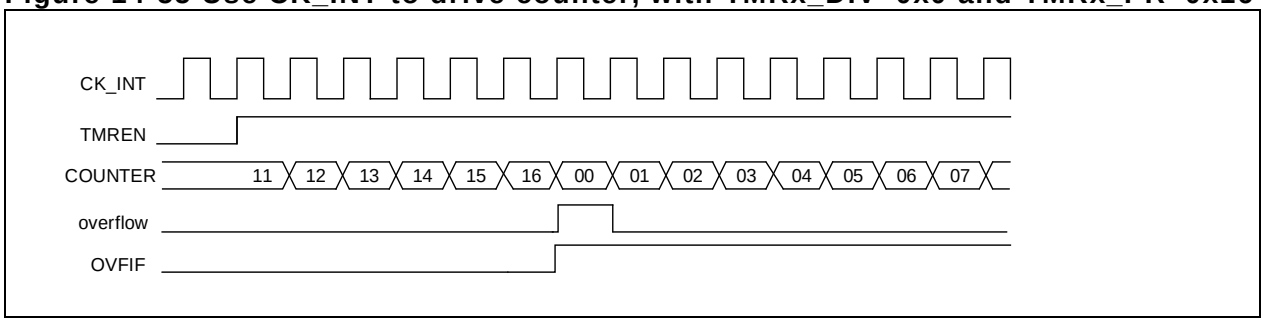
Internal clock (CK_INT)

By default, the CK_INT divided by the prescaler is used to drive the counter to start counting. When TMR's APB clock prescaler factor is 1, the CK_INT frequency is equal to that of APB, otherwise, it doubles the APB clock frequency.

The configuration process is as follows:

- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 to select count mode. If the one-way count direction is set, configure OWCDIR bit in the TMRx_CTRL1 register to select the specific direction.
- Set the TMRx_DIV register to set the counting frequency;
- Set the TMRx_PR register to set the counting period;
- Set the TMREN bit in the TMRx_CTRL1 register to enable the counter.

Figure 14-55 Use CK_INT to drive counter, with TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (TRGIN/EXT)

The counter clock can be provided by two external clock sources, namely, TRGIN and EXT signals.

SMSEL=3'b111: External clock mode A is selected. Select an external clock source TRGIN signal by setting the STIS[2: 0] bit to drive the counter to start counting.

The external clock sources include: C1INC (STIS=3'b100, channel 1 rising edge and falling edge), C1IFP1 (STIS=3'b101, the channel 1 signal with filtering and polarity selection), C2IFP2 (STIS=3'b110, a channel 2 signal with filtering and polarity selection) and EXT (STIS=3'b111, external input signal with polarity selection, frequency division and filtering).

ECMBEN=1: External clock mode B is selected. The counter is driven by external input that has gone through polarity selection, frequency division and filtering. The external clock mode B is equivalent to the external clock mode A which selects EXT signal as an external force TRGIN.

To use external clock mode A, follow the steps below:

-Set external source TRGIN parameters

If the TMRx_CH1 is used as a source of TRGIN, it is necessary to configure channel 1 input filter (C1DF[3:0] in TMRx_CM1 register) and channel 1 input polarity (C1P/C1CP in TMRx_CCTRL register);

If the TMRx_CH2 is used as source of TRGIN, it is necessary to configure channel 1 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in

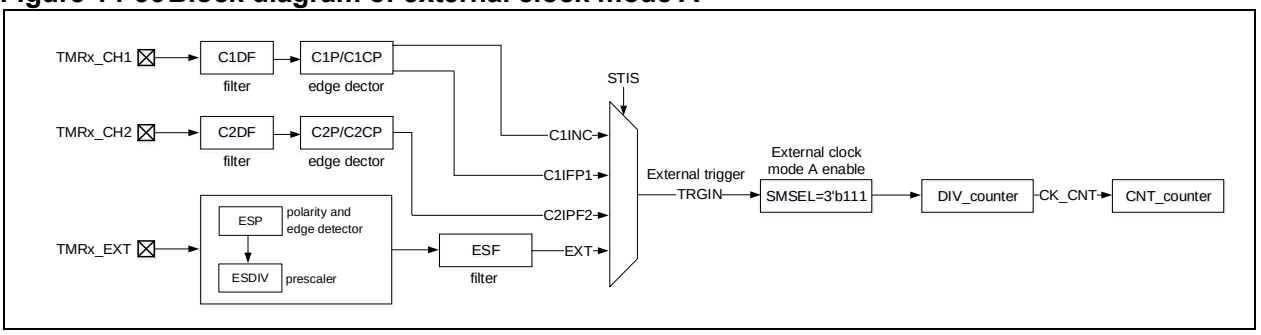
TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).

- Set TRGIN signal source using the STIS[1:0] bit in TMRx_STCTRL register
- Enable external clock mode A by setting SMSEL=3'b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register
- Enable counter through the TMREN bit in TMRx_CTRL1 register

To use external clock mode B, follow the steps below:

- Set external signal polarity through the ESP bit in TMRx_STCTRL register
- Set external signal frequency division through the ESDIV[1:0] bit in TMRx_STCTRL register
- Set external signal filter through the ESF[3:0] bit in TMRx_STCTRL register
- Enable external clock mode B through the ECMBEN bit in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] bit in TMRx_DIV register
- Set counting period through the PR[15:0] bit in TMRx_PR register
- Enable counter through the TMREN in TMRx_CTRL1 register

Figure 14-56 Block diagram of external clock mode A



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-57 Counting in external clock mode A, PR=0x32, DIV=0x0

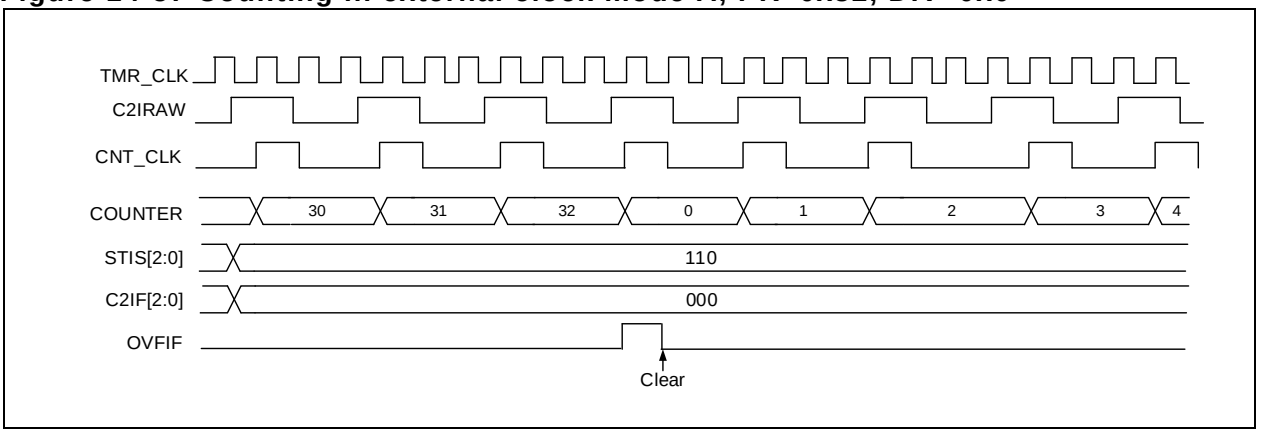
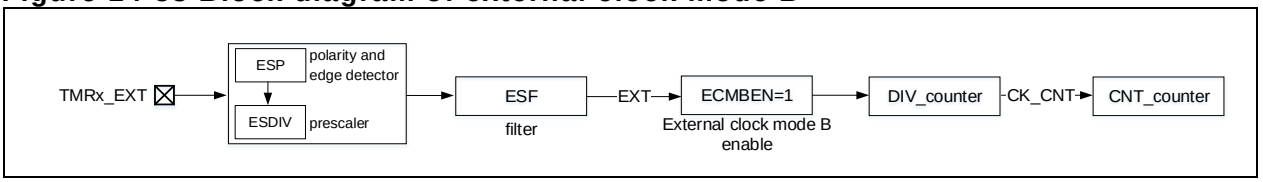
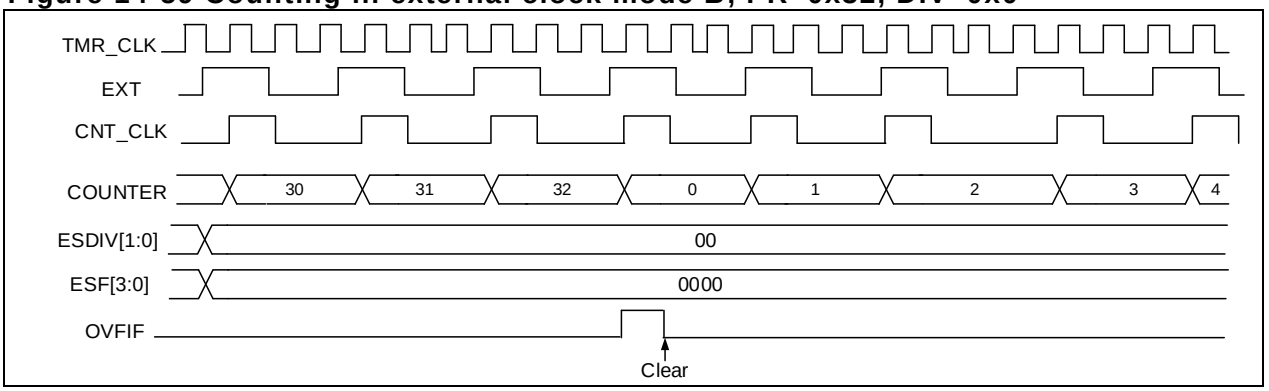


Figure 14-58 Block diagram of external clock mode B



Note: The delay between the signal on the input side and the actual clock of the counter is due to the synchronization circuit.

Figure 14-59 Counting in external clock mode B, PR=0x32, DIV=0x0



Internal trigger input (ISx)

Timer synchronization allows interconnection between several timers. The TMR_CLK of one timer can be provided by the TRGOUT signal output by another timer. Set the STIS[2: 0] bit to select internal trigger signal to enable counting.

Each timer consists of a 16-bit prescaler, which is used to generate the CK_CNT that enables the counter to count. The frequency division relationship between the CK_CNT and TMR_CLK can be adjusted by setting the value of the TMRx_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

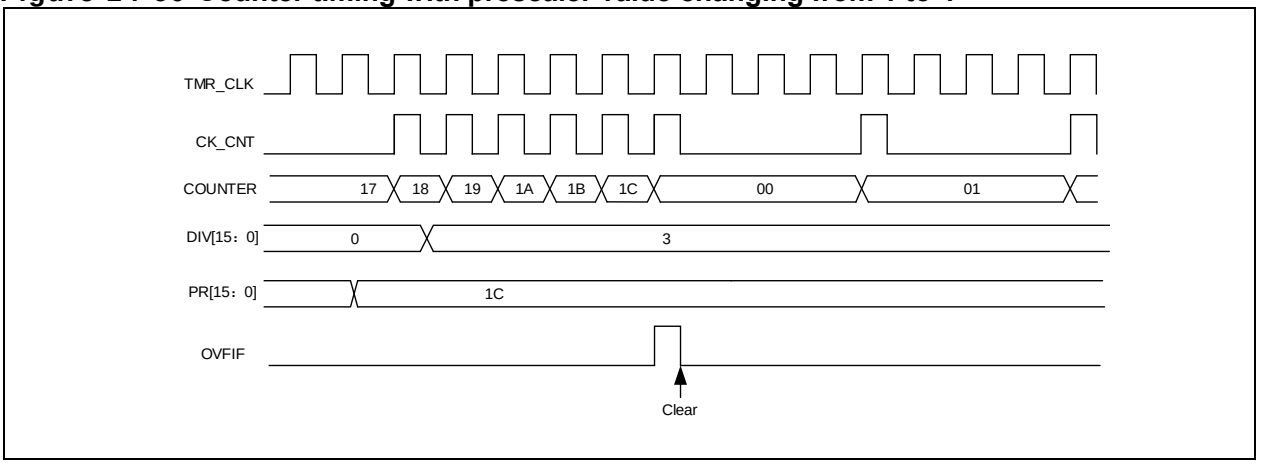
The internal trigger input is configured as follows:

- Set the TMRx_PR register to set the counting period;
- Set the TMRx_DIV register to set the counting frequency;
- Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 register to set the count mode;
- Set the STIS[2:0] bit (range: 3'b000~3'b011) in the TMRx_STCTRL register and select internal trigger;
- Set SMSEL[2:0]=3'b111 in the TMRx_STCTRL register and select external clock mode A;
- Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.

Table 14-11 TMRx internal trigger connection

Slave timer	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR1	TMR5	TMR2	TMR3	TMR4-

Figure 14-60 Counter timing with prescaler value changing from 1 to 4



14.3.3.2 Counting mode

The advanced-control timer supports several counting modes to meet different application scenarios, and it consists of a 16-bit counter supporting up, down, up/down counting modes.

The TMRx_PR register is used to configure the counting period. The value in the TMRx_PR is immediately moved to the shadow register by default. When the periodic buffer is enabled (PRBEN=1), the value in the TMRx_PR register is transferred to the shadow register at an overflow event.

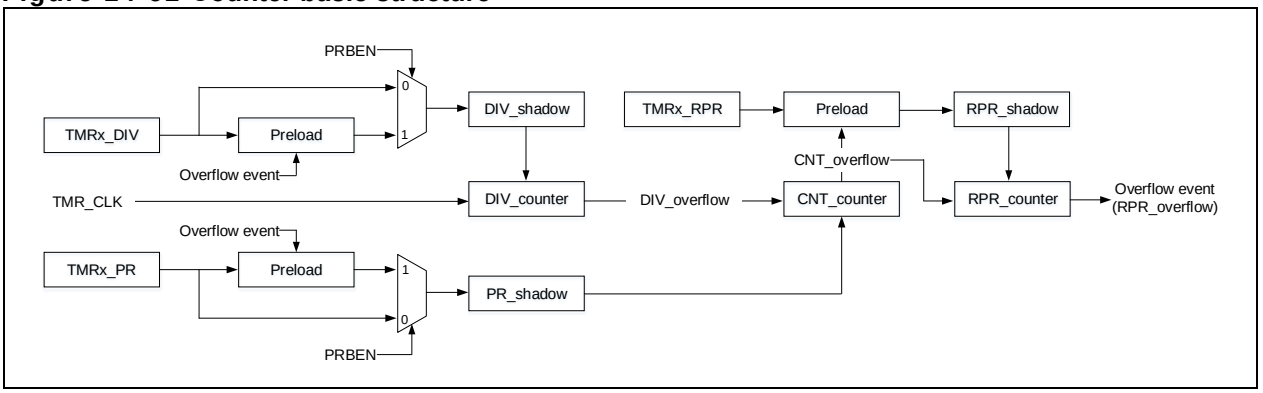
The TMRx_DIV register is used to configure the counting frequency. The counter counts once every count clock period (DIV[15:0]+1). Similar to the TMRx_PR register, when the periodic buffer is enabled, the value in the TMRx_DIV register is updated to the shadow register at an overflow event.

Reading the TMRx_CNT register returns to the current counter value, and writing to the TMRx_CNT register updates the current counter value to the value being written.

An overflow event is generated by default. Set OVFEN=1 in the TMRx_CTRL1 to disable generation of update events. The OVFS bit in the TMRx_CTRL1 register is used to select overflow event source. By default, counter overflow/underflow, setting OVFSWTR bit and the reset signal generated by the slave timer controller in reset mode trigger the generation of an overflow event. When the OVFS bit is set, only counter overflow/underflow triggers an overflow event.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-61 Counter basic structure



Upcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register to enable upcounting mode. In this mode, the counter counts from 0 to the value programmed in the TMRx_PR register, then restarts from 0, and generates a counter overflow event, with the OVFIF bit being set to 1. If the overflow event is disabled, the counter is no longer reloaded with the preload value and period value at a counter overflow event, otherwise, the counter is updated with the preload value and period value on an overflow event.

Figure 14-62 Overflow event when PRBEN=0

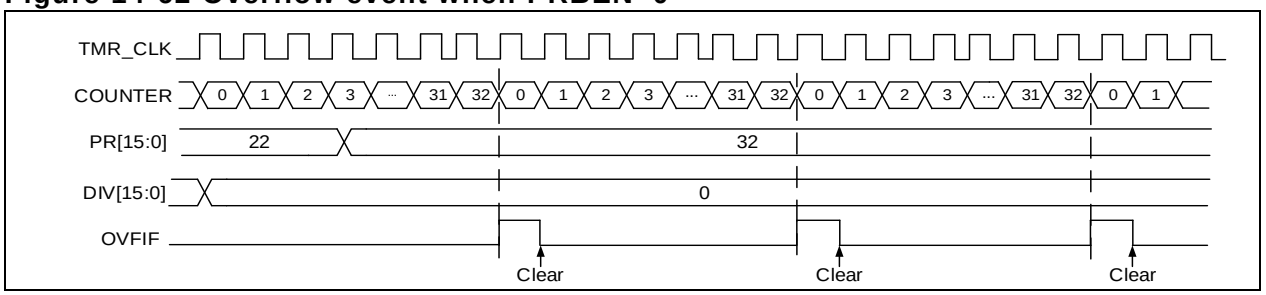
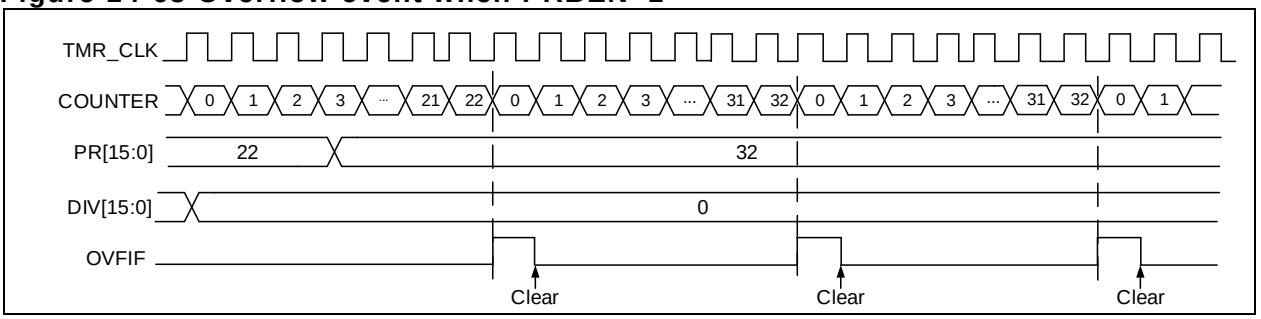


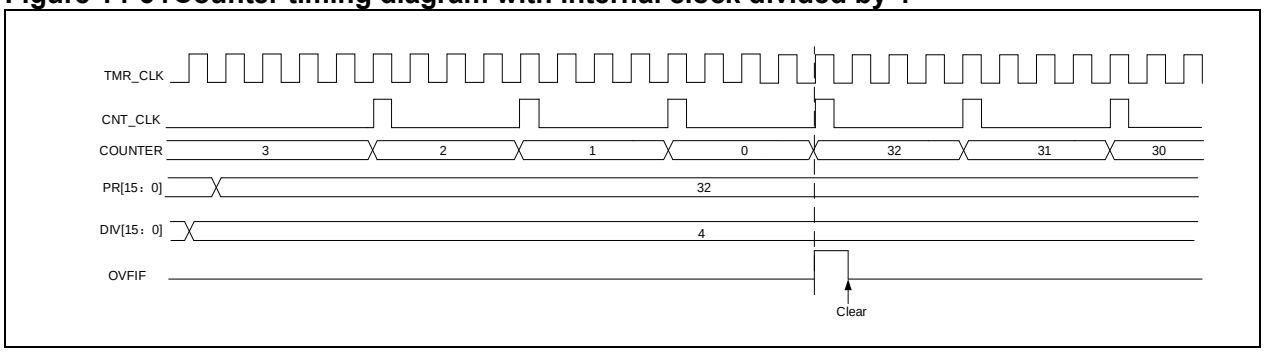
Figure 14-63 Overflow event when PRBEN=1



Downcounting mode

Set CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register to enable downcounting mode. In this mode, the counter counts from the value programmed in the TMRx_PR register down to 0, and restarts from the value programmed in the TMRx_PR register, and generates a counter underflow event.

Figure 14-64 Counter timing diagram with internal clock divided by 4



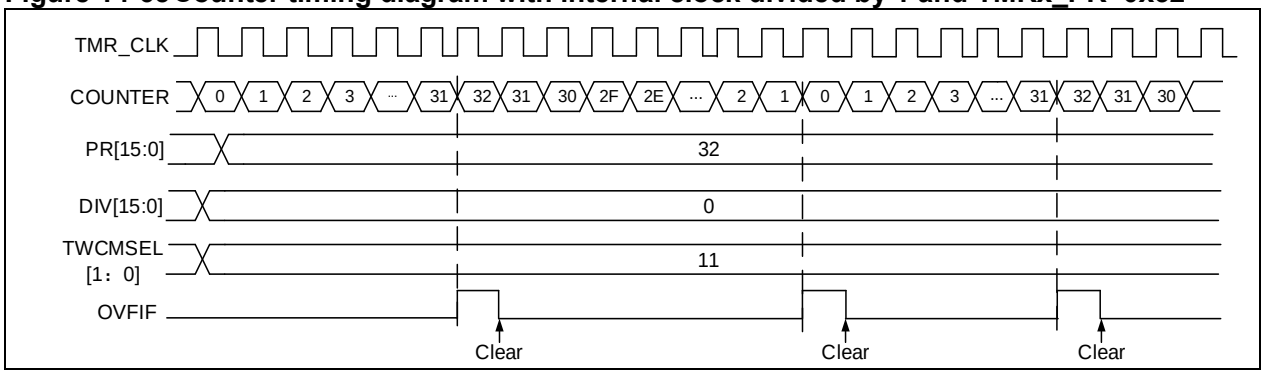
Up/down counting mode

Set CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register to enable up/down counting mode. In this mode, the counter counts up/down alternatively. When the counter counts from the value programmed in the TMRx_PR register down to 1, an underflow event is generated, and then restarts counting from 0; when the counter counts from 0 to the value of the TMRx_PR register -1, an overflow event is generated, and then restarts counting from the value of the TMRx_PR register. The OWCDIR bit indicates the current counting direction.

The TWCMSEL[1:0] bit in the TMRx_CTRL1 register is also used to select the CxIF flag setting method in up/down counting mode. In up/down counting mode 1 (TWCMSEL[1:0]=2'b01), CxIF flag can only be set when the counter counts down; in up/down counting mode 2 (TWCMSEL[1:0]=2'b10), CxIF flag can only be set when the counter counts up; in up/down counting mode 3 (TWCMSEL[1:0]=2'b11), CxIF flag can be set when the counter counts up/down.

Note: The OWCDIR is ready-only in up/down counting mode.

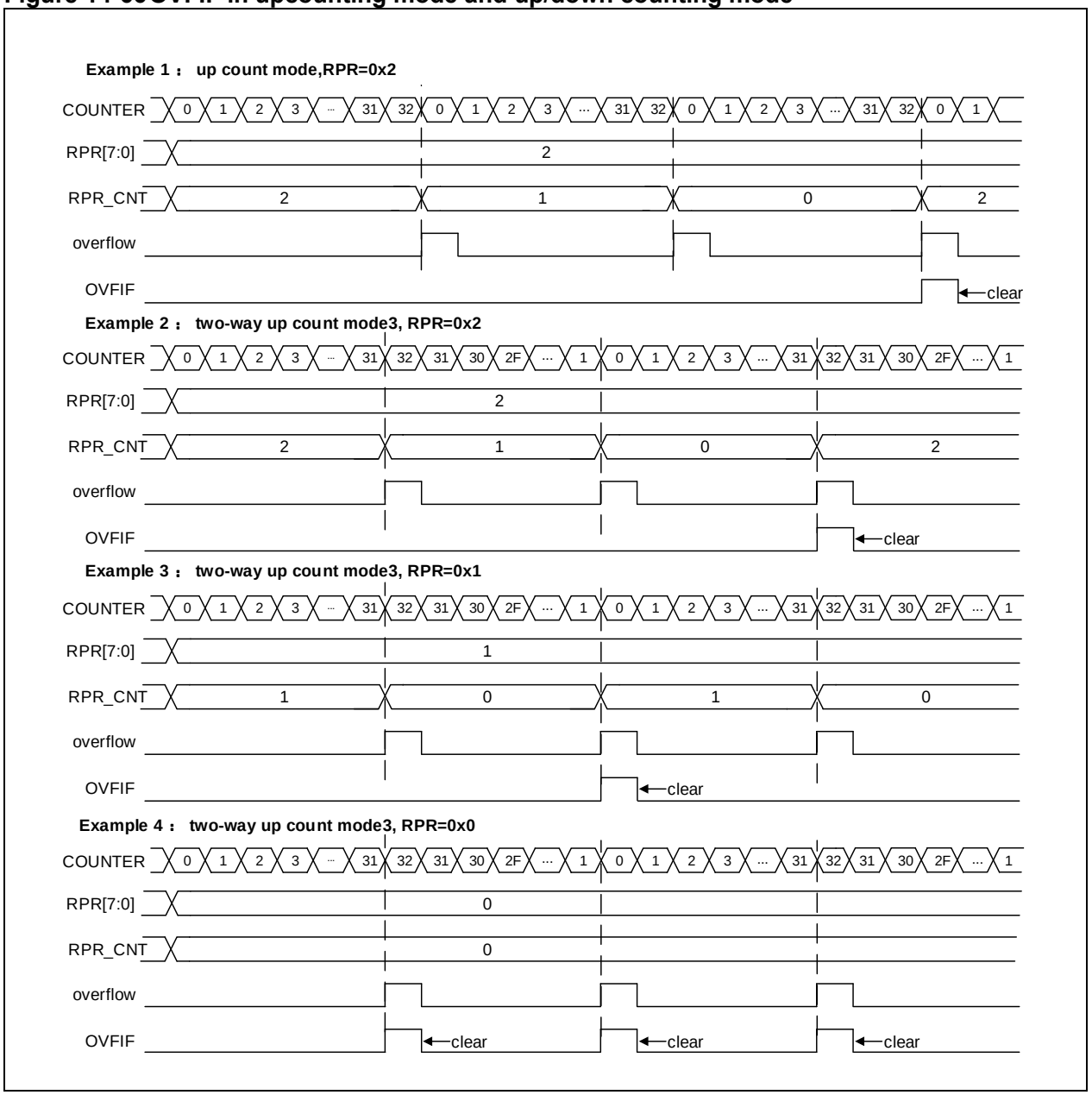
Figure 14-65 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Repetition counter mode:

The TMRx_RPR register is used to configure the counting period of repetition counter. The repletion counter mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event occurs once at every counter overflow (RPR[7:0]+1), and the repetition counter is decremented at each counter overflow. An overflow event is generated only when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

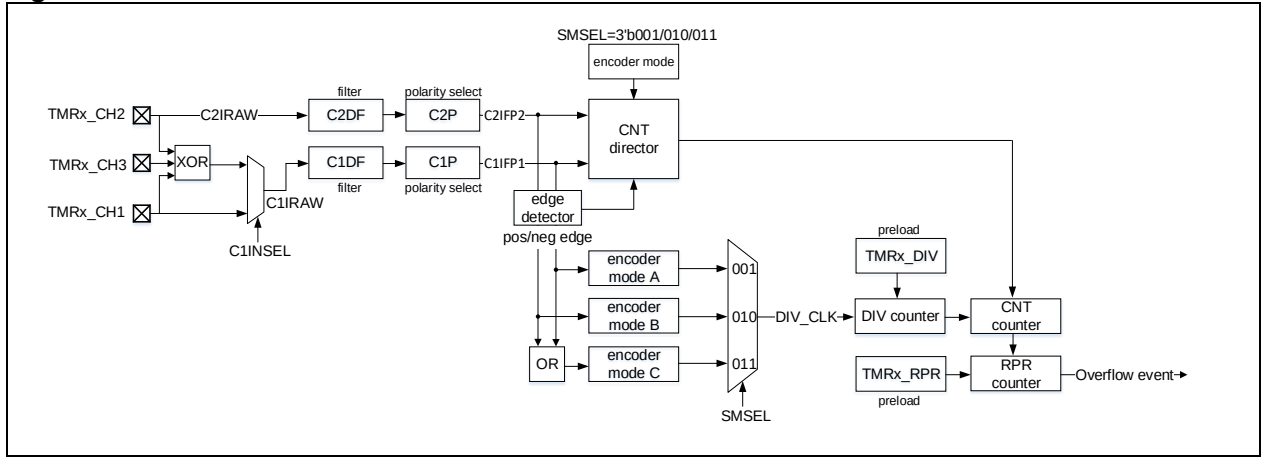
Figure 14-66 OVFIF in upcounting mode and up/down counting mode



Encoder interface mode

In this mode, the two input (TMRx_CH1 and TMRx_CH2) signals are required. Depending on the level on one input, the counter counts up or down on the edge of the other input signal. The OWCDIR bit indicates the direction of the counter.

Figure 14-67 Encoder mode structure



Encoder mode A: SMSEL=3'b001. The counter counts on the selected C1IFP1 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C1IFP1 and the level of C2IFP2.

Encoder mode B: SMSEL=3'b010. The counter counts on the selected C2IFP2 edge (rising and falling edges), and the counting direction is dependent on the edge direction of C2IFP2 and the level of C1IFP1.

Encoder mode C: SMSEL=3'b011. The counter counts on both C1IFP1 and C2IFP2 edges (rising and falling edges). The counting direction is dependent on the C1IFP1 edge direction and C2IFP2 level, and C2IFP2 edge direction and C1IFP1 level.

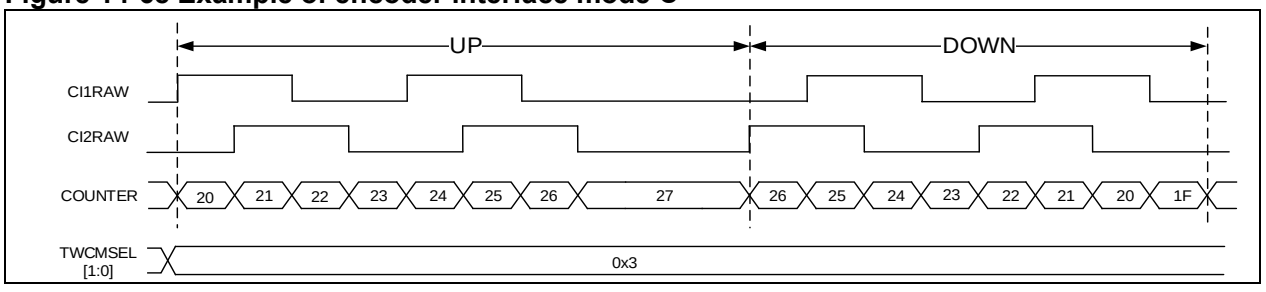
To use encoder mode, follow the procedures below:

- Set channel 1 input signal filtering through the C1DF[3:0] bit in the TMRx_CM1 register;
  Set channel 1 input signal active level through the C1P bit in the TMRx_CCTRL register
- Set channel 2 input signal filtering through the C2DF[3:0] bit in the TMRx_CM1 register;
  Set channel 2 input signal active level through the C2P bit in the TMRx_CCTRL register
- Set channel 1 as input mode through the C1C[1:0] bit in the TMRx_CM1 register;
  Set channel 2 as input mode through the C2C[1:0] bit in the TMRx_CM1 register
- Select encoder mode A (SMSEL=3'b001), encoder mode B (SMSEL=3'b010), or encoder mode C (SMSEL=3'b011) by setting the SMSEL[2:0] bit in the TMRx_STCTRL register
- Set counting cycles through the PR[15:0] bit in the TMRx_PR register
- Set counting frequency through the DIV[15:0] bit in the TMRx_DIV register
- Configure the corresponding IOs of TMRx_CH1 and TMRx_CH2 as multiplexed mode
- Enable counter through the TMREN bit in the TMRx_CTRL1 register

Table 14-12 Counting direction versus encoder signals

Active edge	Level on opposite signal (C1IFP1 to C2IFP2, C2IFP2 to C1IFP1)	C1IFP1 signal		C2IFP2 signal	
		Rising	Falling	Rising	Falling
Count on C1IFP1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Count on C2IFP2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Count on both C1IFP1 and C2IFP2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

Figure 14-68 Example of encoder interface mode C



14.3.3.3TMR input function

The TMR1 has four independent channels. Each channel can be configured as input or output. As input, the channel can be used for the filtering, selection, division and input capture of the input signals.

Figure 14-69 Input/output channel 1 main circuit

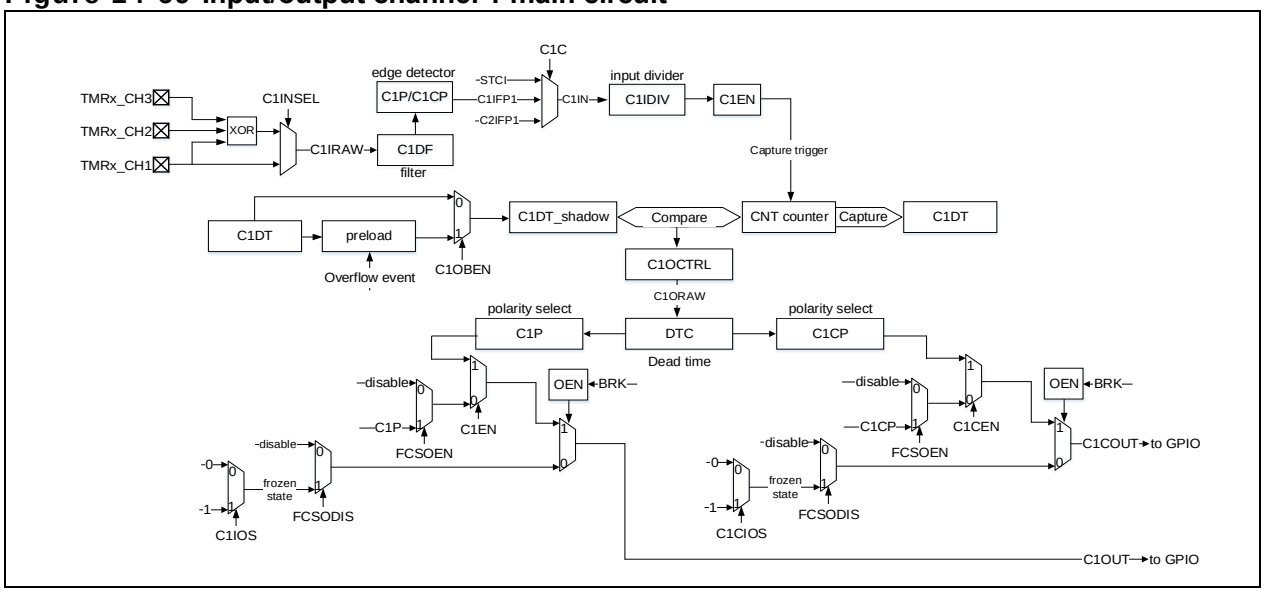
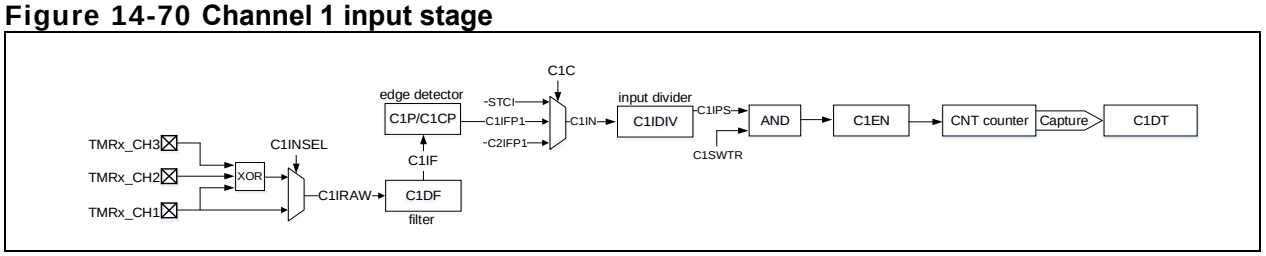


Figure 14-70 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latch the current counter values after the selected trigger signal is detected, and the capture compare interrupt flag bit (CxIF) is set to 1. An interrupt or a DMA request will be generated if the CxIEN or CxDEN bit is enabled. If the selected trigger signal is detected when the CxIF is set, a capture overflow event occurs. The TMRx_CxDT register overwrites the recorded value with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the configuration procedure mentioned below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set the filter bandwidth of C1IN signal (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR register
- Program the capture frequency division of C1IN signal (C1DIV[1: 0])
- Enable channel 1 input capture (C1EN=1)
- If needed, enable the relevant interrupt or DMA request by setting the C1IEN bit in the TMRx_IDEN register or the C1DEN bit in the TMRx_IDEN register

Timer Input XOR function

The timer input pins (TMRx_CH1, TMRx_CH2 and TMRx_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMRx_CTRL2 register) through an XOR gate.

The XOR gate can be used to connect Hall sensors. For example, connect the three XOR inputs to the three Hall sensors respectively so as to calculate the position and speed of the rotation by analyzing three Hall sensor signals.

PWM input

The PWM input mode applies to channel 1 and channel 2. To enable this mode, map the C1IN and C2IN to the same TMRx_CHx, and configure the CxIFPx of channel 1/2 to trigger slave timer controller reset.

The PWM input mode can be used to measure the period and duty cycle of input signal. The period and duty cycle of channel 1 can be measured as follows:

- Set C1C=2‘b01 to set C1IN as C1IFP1;
- Set C1P=1’b0 to set C1IFP1 rising edge active;
- Set C2C=2‘b10 to set C2IN as C1IFP2;
- Set C2P=1’b1 to set C1IFP2 falling edge active;
- Set STIS=3’b101 to set C1IFP1 as the slave timer trigger signal;
- Set SMSEL=3‘b100 to set the slave timer in reset mode;
- Set C1EN=1’b1 and C2EN=1’b1 to enable channel 1 and input capture.

In these configurations, the rising edge of channel 1 input signal triggers capture and saves captured values to the C1DT register, and channel 1 input signal rising edge resets the counter. The falling edge of channel 1 input signal triggers capture and saves captured values to the C2DT register. The period and duty of channel 1 input signal can be calculated through C1DT and C2DT respectively.

Figure 14-71 Example of PWM input mode configuration

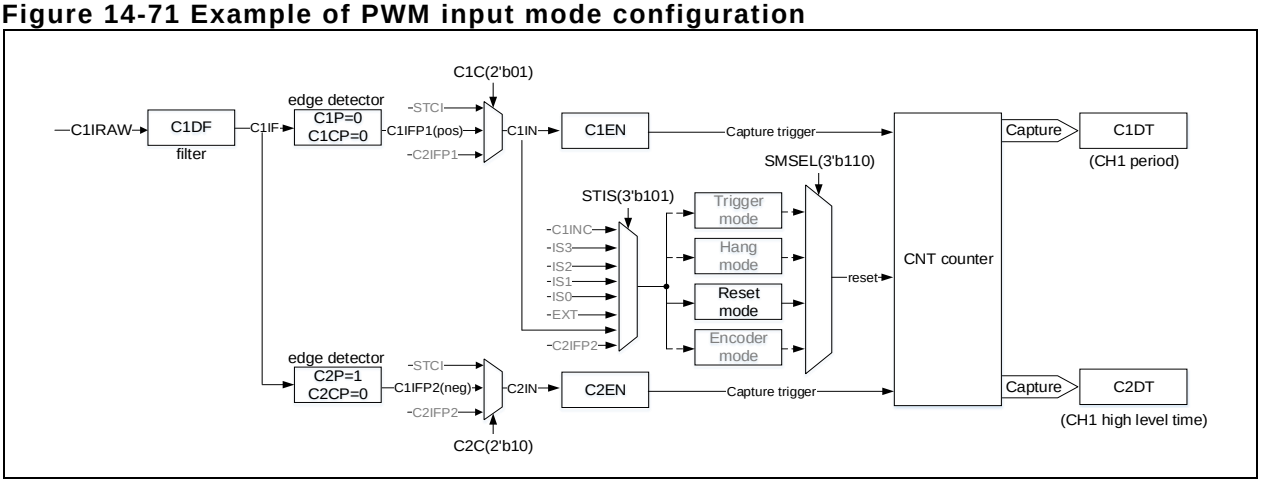
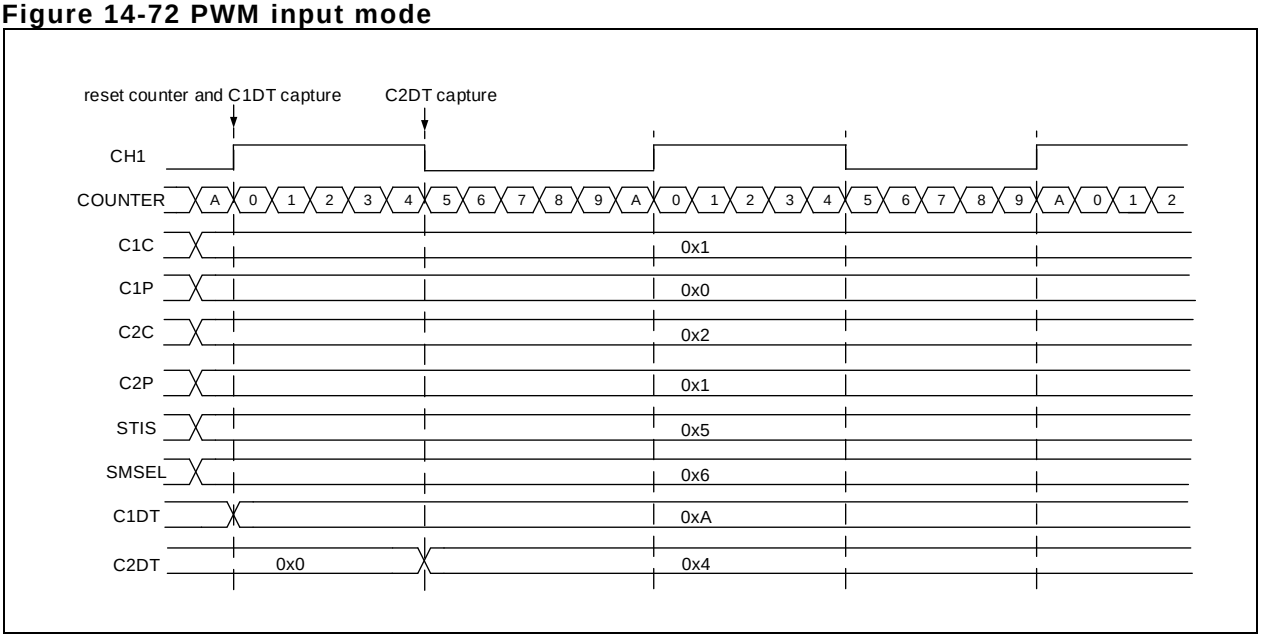


Figure 14-72 PWM input mode



14.3.3.4TMR output function

The TMR output consists of a comparator and an output controller. It is used to program the period, duty cycle and polarity of the output signal. The advanced-control timer output function varies from one channel to one channel.

Figure 14-73 Channel output stage (channel 1 to 3)

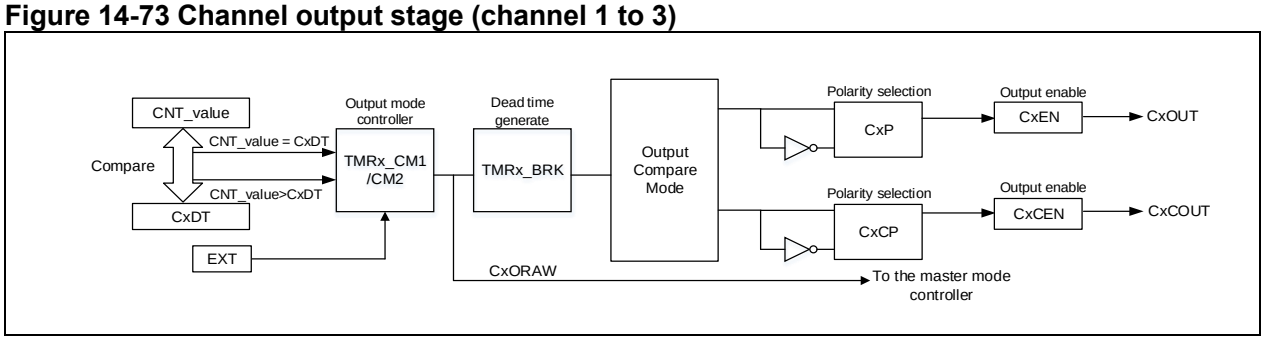
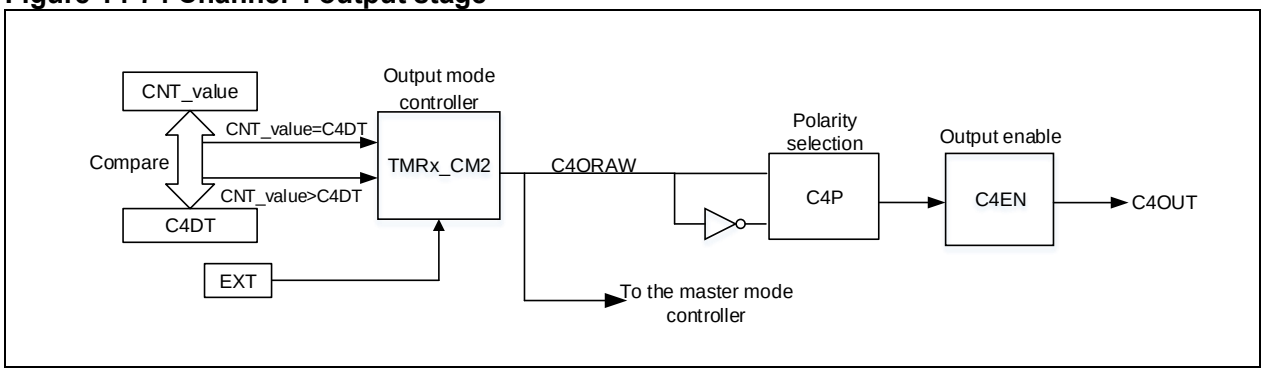


Figure 14-74 Channel 4 output stage



Output mode

Write CxC[2: 0]≠2'b00 to configure the channel as output to implement multiple output modes. In this case, the counter value is compared with the value in the TMRx_CxDT register, and the intermediate signal CxORAW is generated according to the output mode selected by CxOCTRL[2: 0], which is sent to IO after being processed by the output control circuit. The period of the output signal is configured by the TMRx_PR register, while the duty cycle by the TMRx_CxDT register.

Output compare modes include:

- **PWM mode A**: Set CxOCTRL=3'b110 to enable PWM mode A. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. To set PWM mode A, the following process is recommended:
  - Set the TMRx_PR register to set PWM period;
  - Set the TMRx_CxDT register to set PWM duty cycle;
  - Set CxOCTRL=3'b110 in TMRx_CM1/CM2 register and set output mode as PWM mode A;
  - Set the TMRx_DIV register to set the counting frequency;
  - Set the TWCMSEL[1:0] bit in the TMRx_CTRL1 to set the count mode;
  - Set the CxP and CxCP bits in the TMRx_CCTRL register to set the output polarity;
  - Set the CxEN and CxCEN bits in the TMRx_CCTRL register to enable channel output;
  - Set the OEN bit in the TMRx_BRK register to enable TMRx output;
  - Set the corresponding GPIO of TMR output channel as the multiplexed mode;
  - Set the TMREN bit in the TMRx_CTRL1 register to enable TMRx counter.
- **PWM mode B**: Set CxOCTRL=3'b111 to enable PWM mode B. In upcounting, when TMRx_C1DT>TMRx_CVAL, C1ORAW outputs low; otherwise, outputs high. In downcounting, when TMRx_C1DT<TMRx_CVAL, C1ORAW outputs high; otherwise, outputs low.
- **Forced output mode:** Set CxOCTRL=2'b100/101 to enable forced output mode. In this case, the CxORAW is forced to be the programmed level, irrespective of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.

- **Output compare mode:** Set CxOCTRL=3'b001/010/011 to enable output compare mode. In this case, when the counter value matches the value of the CxDT register, the CxORAW is forced high (CxOCTRL=3'b001), low (CxOCTRL=3'b010) or toggling (CxOCTRL=3'b011).
- **One-pulse mode:** This is a particular case of PWM mode. Set OCMEN=1 to enable one-pulse mode. In this mode, the comparison match is performed in the current counting period. The TMREN bit is cleared as soon as the current counting is completed. Therefore, only one pulse is output. When configured as in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.
- **Fast output mode:** Set CxOIEN=1 to enable this mode. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but at the beginning of the current counting period. In other words, the comparison result is advanced, so the comparison result between the counter value and the TMRx_CxDT register will determine the level of CxORAW in advance.

Figure 14-75 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-76 gives an example of the combination between upcounting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 14-77 gives an example of the combination between up/down counting mode and PWM mode A. The output signal behaves when PR=0x32 but CxDT is configured with a different value.

Figure 14-78 gives an example of the combination between upcounting mode and one-pulse PWM mode B. The counter only counts only one cycle, and the output signal sends only one pulse.

Figure 14-75 C1ORAW toggles when counter value matches the C1DT value

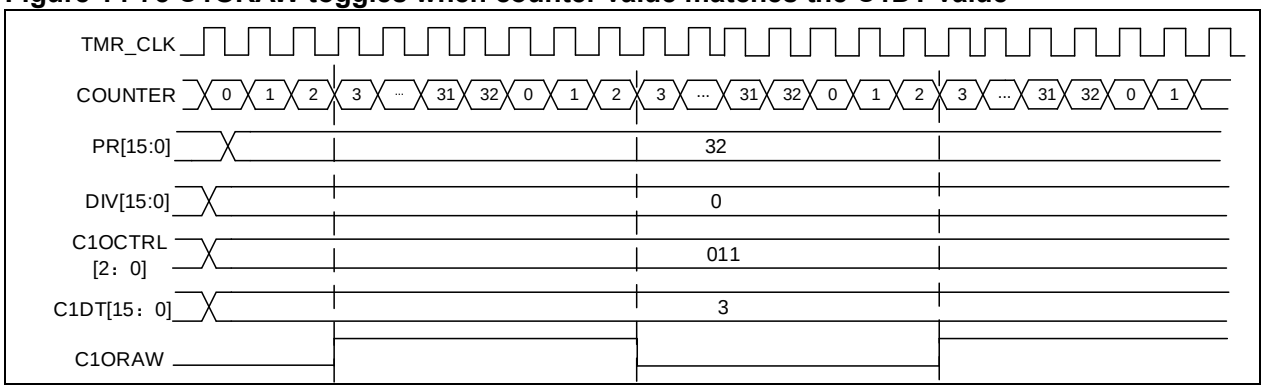


Figure 14-76 Upcounting mode and PWM mode A

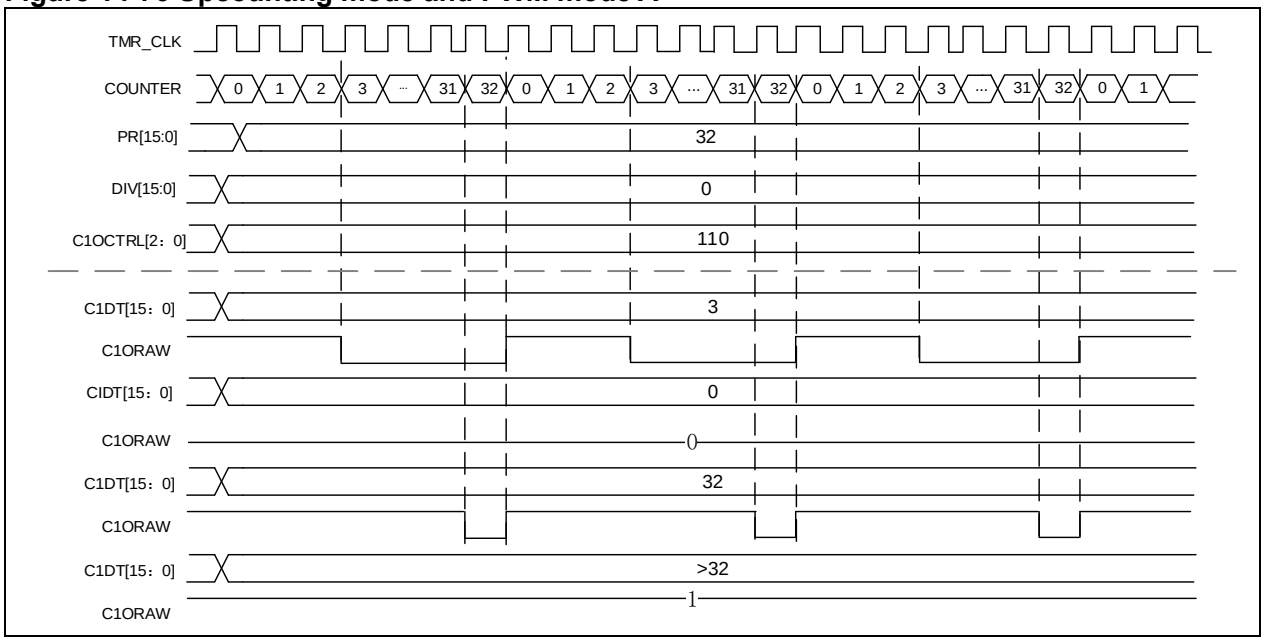


Figure 14-77 Up/down counting mode and PWM mode

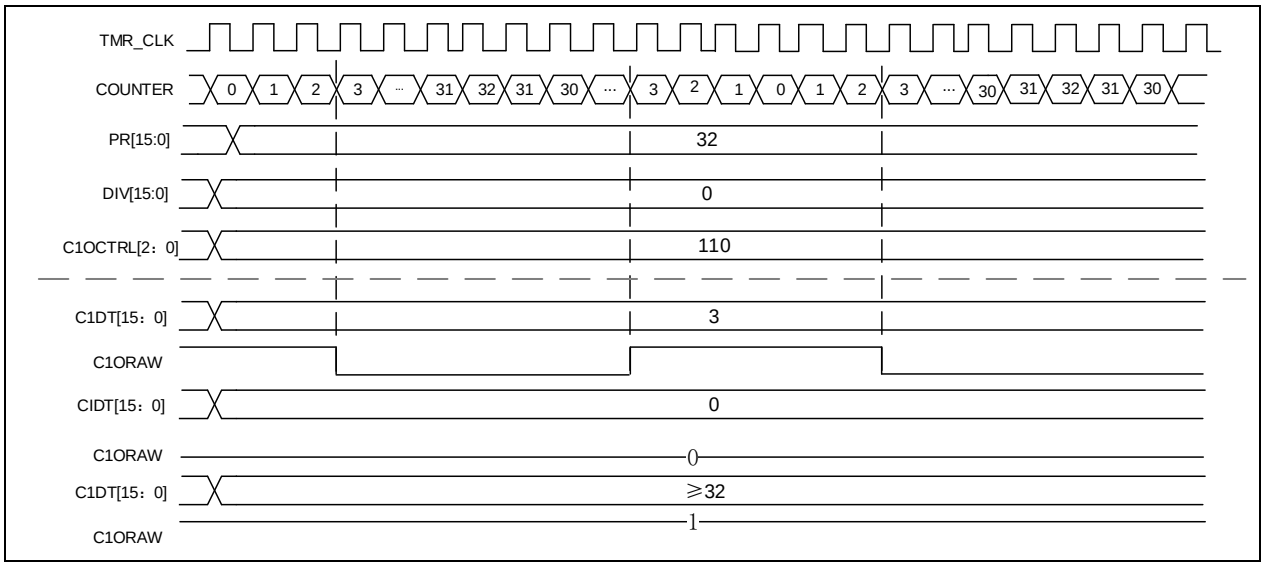
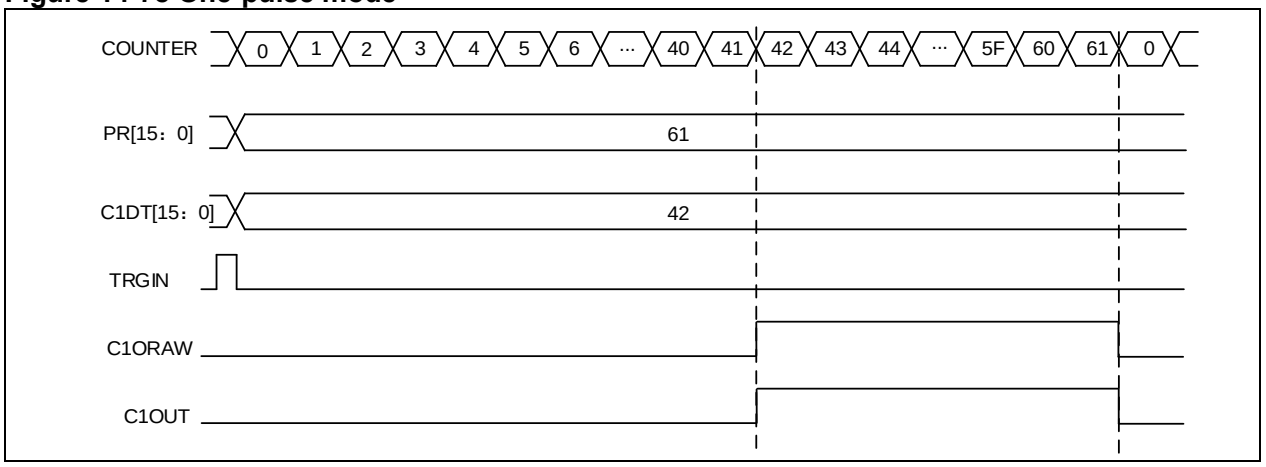


Figure 14-78 One-pulse mode



Master timer event output

When TMR is selected as the master timer, the following signal sources can be selected as TRGOUT signal to output to the slave timer, by setting the PTOS bit in the TMRxCTRL2 register.

-PTOS=3’b000, TRGOUT outputs software overflow event (OVFSWTR bit in the TMRx_SWEVT register);

-PTOS=3’b001, TRGOUT outputs counter enable signal;

-PTOS=3’b010, TRGOUT outputs counter overflow event;

-PTOS=3’b011, TRGOUT outputs capture and compare event;

-PTOS=3’b100, TRGOUT outputs C1ORAW signal;

-PTOS=3’b101, TRGOUT outputs C2ORAW signal;

-PTOS=3’b110, TRGOUT outputs C3ORAW signal;

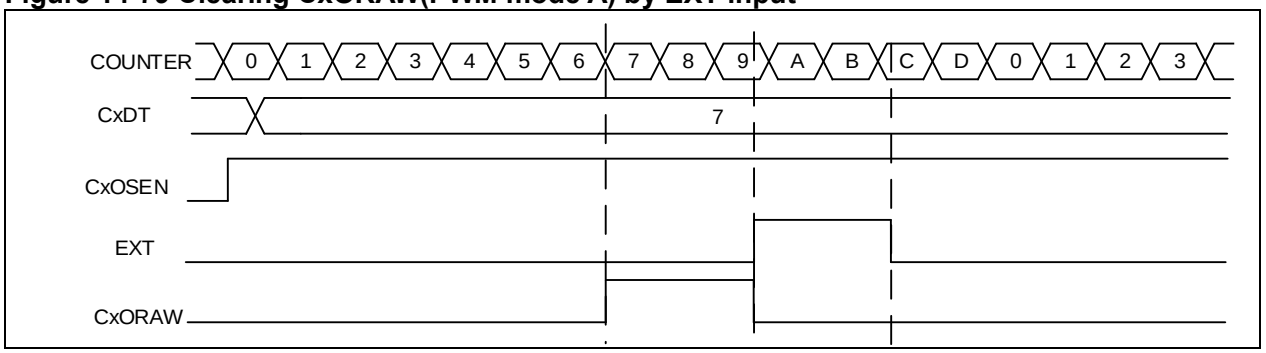
-PTOS=3’b111, TRGOUT outputs C4ORAW signal.

CxORAW clear

When the CxOSEN bit is set to 1, the CxORAW signal for a given channel is cleared by applying a high level to the EXT input. The CxORAW signal remains unchanged until the next overflow event.

This function can only be used in output capture or PWM modes, and does not work in forced mode. *Figure 79* shows the example of clearing CxORAW signal. When the EXT input is high, the CxORAW signal, which was originally high, is driven low; when the EXT is low, the CxORAW signal outputs the corresponding level according to the comparison result between the counter value and CxDT value.

Figure 14-79 Clearing CxORAW(PWM mode A) by EXT input



Dead-time insertion

The channel 1 to 3 of the advanced-control timers contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is defined by CxCP. Refer to *Table 14-14* for more information about the output state of CxOUT and CxCOUT.

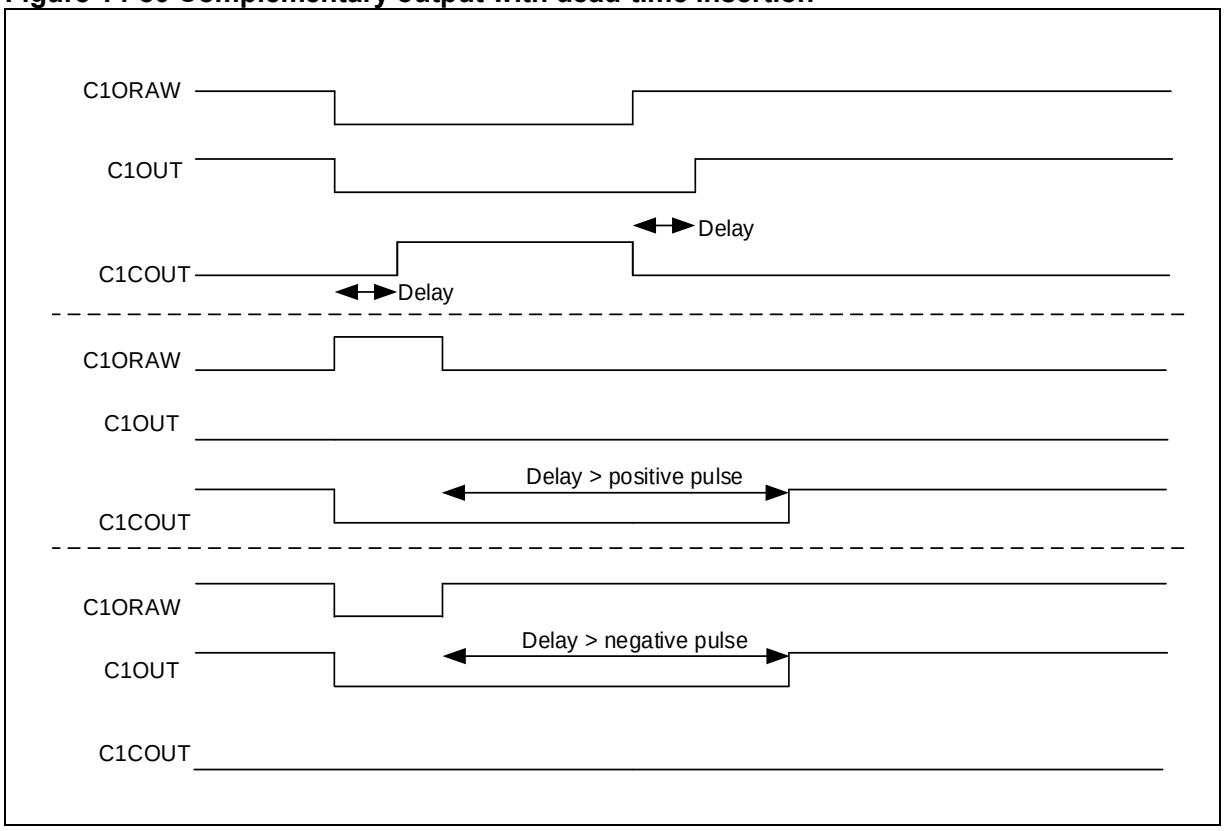
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

Setting both CxEN and CxCEN bits, and using DTC[7:0] bit to insert dead-time of different durations. After the dead-time insertion, the rising edge of the CxOUT is delayed compared to the rising edge of the reference signal; the rising edge of the CxCOU is delayed compared to the falling edge of the reference signal.

If the delay is greater than the width of the active output, C1OUT and C1COUT will not generate corresponding pulses. Therefore, the dead-time should be less than the width of the active output.

Figure 14-80 gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-80 Complementary output with dead-time insertion



14.3.3.5 TMR break function

When the break function is enabled (BRKEN=1), the CxOUT and CxCOUT are jointly controlled by OEN, FCSODIS, FCSOEN, CxIOS and CxCIOS. But, CxOUT and CxCOUT cannot be set both to active level at the same time. Please refer to *Table 14-14* for more details.

The break source can be the break input pin or a clock failure event. The polarity is controlled by the

BRKV bit.

When a break event occurs, there are the following actions:

- The OEN bit is cleared asynchronously, and the channel output state is selected by setting the FCSODIS bit. This function works even if the MCU oscillator is off.
- Once OEN=0, the channel output level is defined by the CxIOS bit. If FCSODIS=0, the timer output is disabled, otherwise, the output enable remains high.
- When complementary outputs are used:
  - The outputs are first put in reset state, that is, inactive state (depending on the polarity). This is done asynchronously so that it works even if no clock is provided to the timer.
  - If the timer clock is still active, then the dead-time generator is activated. The CxIOS and CxCIOS bits are used to program the level after dead-time. Even in this case, the CxIOS and CxCIOS cannot be driven to their active level at the same time. It should be note that because of synchronization on OEN, the dead-time duration is usually longer than usual (around 2 clk_tmr clock cycles)
  - If FCSODIS=0, the timer releases the enable output, otherwise, it keeps the enable output; the enable output becomes high as soon as one of the CxEN and CxCEN bits becomes high.
- If the break interrupt or DMA request is enabled, the break statue flag is set, and a break interrupt or DMA request can be generated.
- If AOEN=1, the OEN bit is automatically set again at the next overflow event.

Note: When the break input is active, the OEN cannot be set, nor the status flag, BRKIF can be cleared.

Figure 14-81 TMR output control

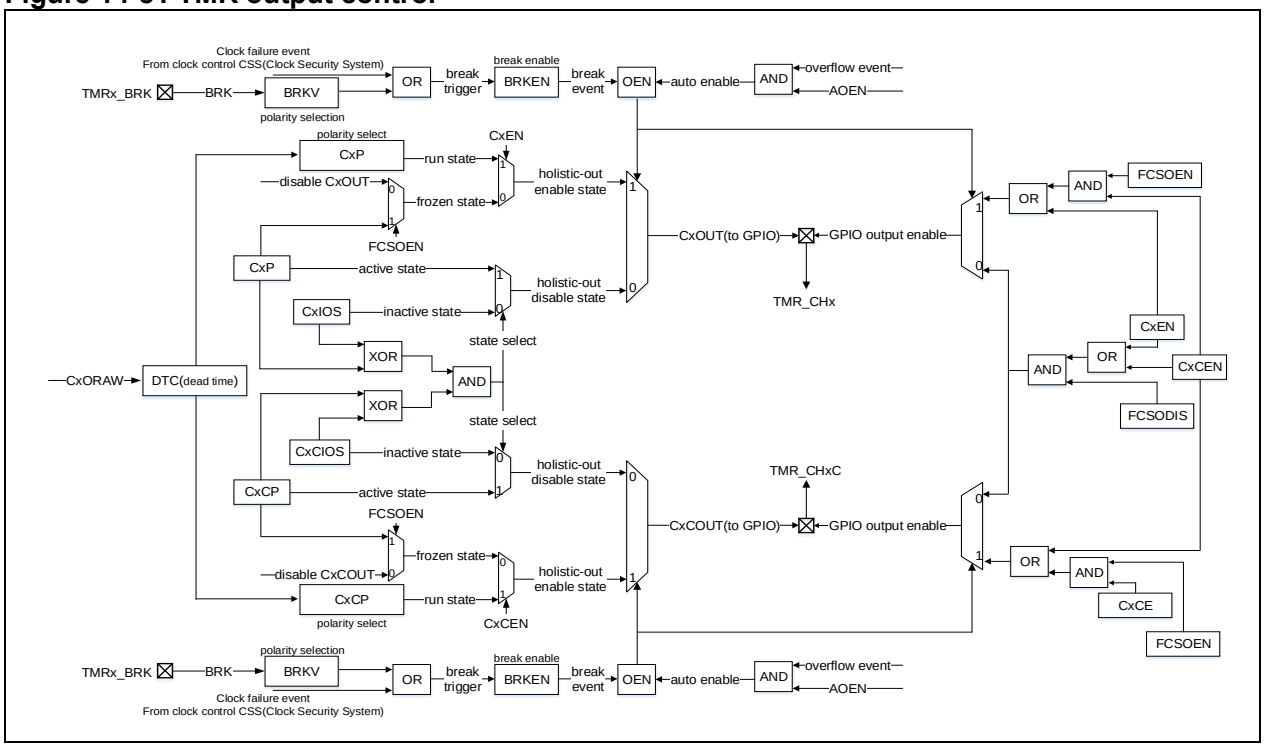
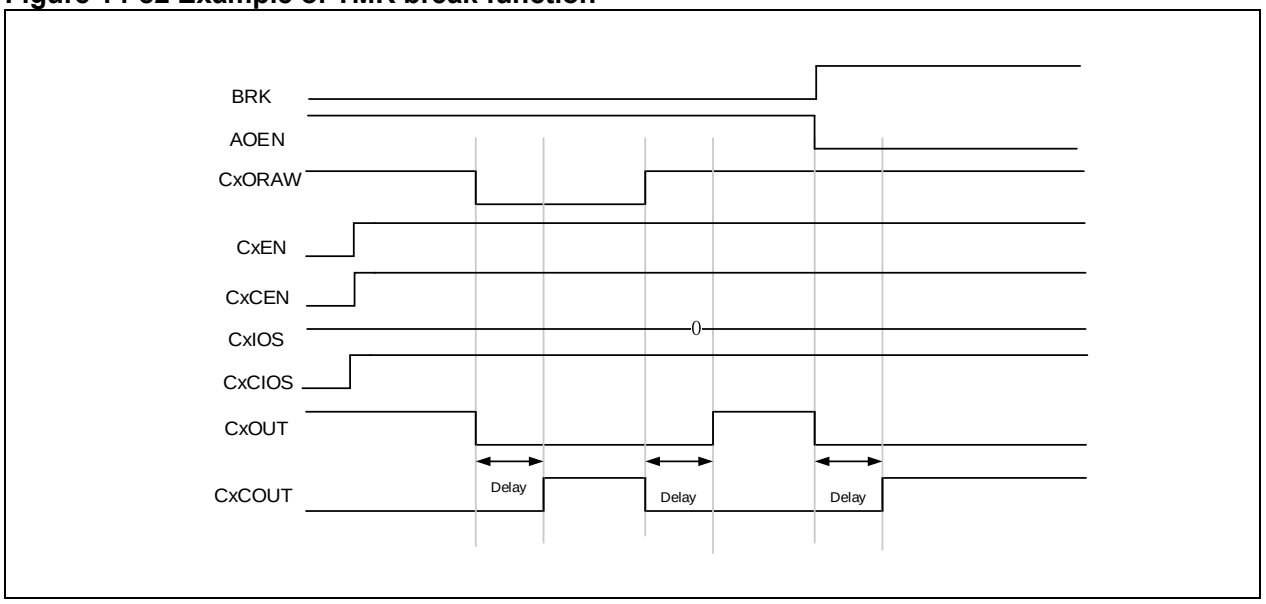


Figure 14-82 Example of TMR break function



14.3.3.6 TMR synchronization

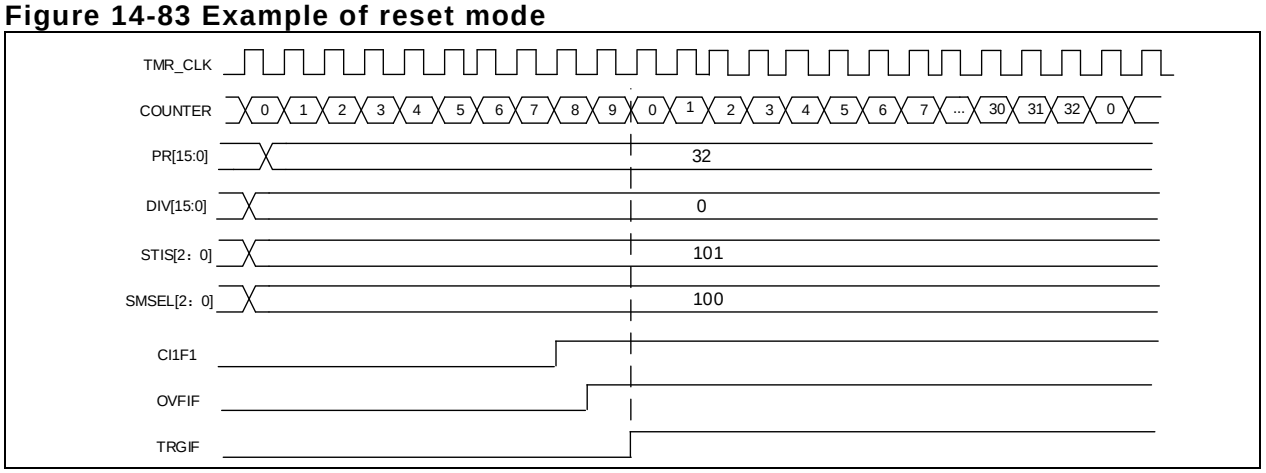
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave modes include:

Slave mode: Reset mode

The counter and its prescaler can be reset by a selected trigger signal. An overflow event can be generated when OVFS=0.

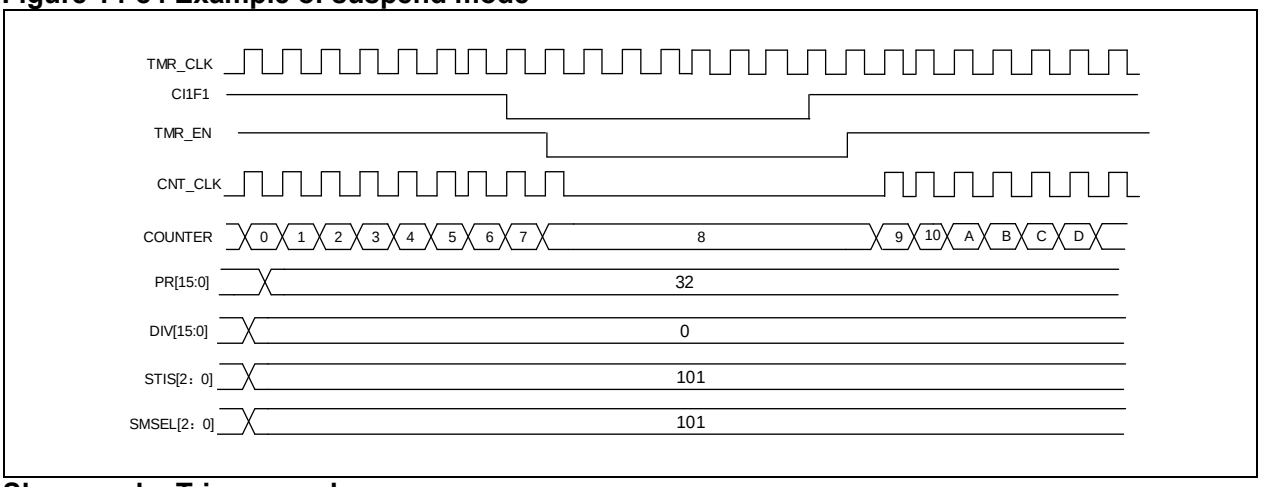
Figure 14-83 Example of reset mode



Slave mode: Suspend mode

In this mode, the counter is controlled by a selected trigger input. The counter starts counting when the trigger input is high and stops as soon as the trigger input is low.

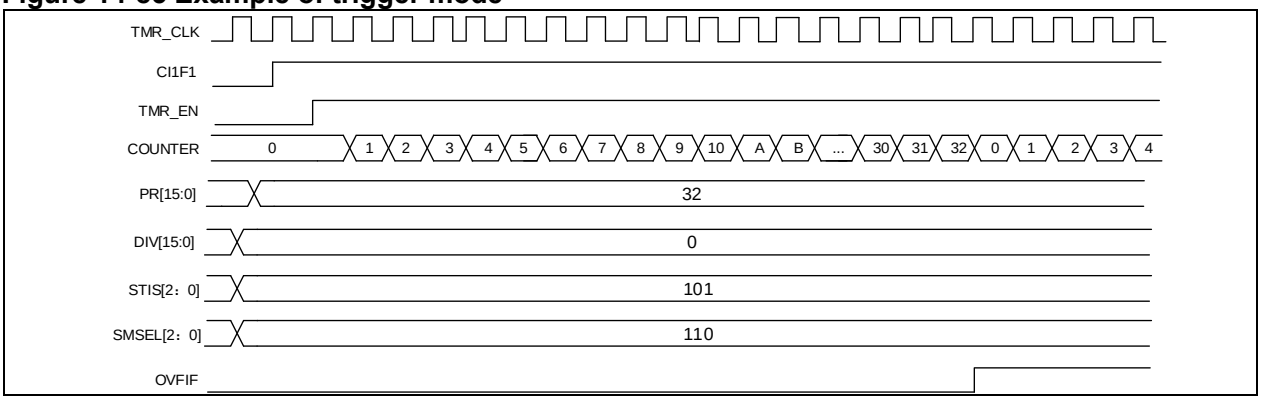
Figure 14-84 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-85 Example of trigger mode



Please refer to *Section 14.2.3.5* for more details.

14.3.3.7 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

14.3.4 TMR1 registers

These peripheral registers must be accessed by word (32 bits).

TMR1 and TMR8 register are mapped into a 16-bit addressable space.

Table 14-13 TMR1 register map and reset value

Register	Offset	Reset value
TMR1_CTRL1	0x00	0x0000
TMR1_CTRL2	0x04	0x0000
TMR1_STCTRL	0x08	0x0000
TMR1_IDEN	0x0C	0x0000
TMR1_ISTS	0x10	0x0000
TMR1_SWEVT	0x14	0x0000
TMR1_CM1	0x18	0x0000
TMR1_CM2	0x1C	0x0000
TMR1_CCTRL	0x20	0x0000
TMR1_CVAL	0x24	0x0000
TMR1_DIV	0x28	0x0000

TMR1_PR	0x2C	0x0000
TMR1_RPR	0x30	0x0000
TMR1_C1DT	0x34	0x0000
TMR1_C2DT	0x38	0x0000
TMR1_C3DT	0x3C	0x0000
TMR1_C4DT	0x40	0x0000
TMR1_BRK	0x44	0x0000
TMR1_DMACTRL	0x48	0x0000
TMR1_DMADT	0x4C	0x0000

14.3.4.1TMR1 control register1 (TMR1_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at its default value.
Bit 9: 8	CLKDIV	0x0	rw	Clock division This field is used to define the relationship between digital filter sampling frequency (f_{DTS}) and timer clock frequency (f_{CK_INT}). it is also used to set the ratio relationship between dead time base (T_{DTS}) and timer clock period (T_{CK_INT}) 00: No division, $f_{DTS}=f_{CK\_INT}$ 01: Divided by 2, $f_{DTS}=f_{CK\_INT}/2$ 10: Divided by 4, $f_{DTS}=f_{CK\_INT}/4$ 11: Reserved
Bit 7	PRBEN	0x0	rw	Period buffer enable 0: Period buffer is disabled 1: Period buffer is enabled
Bit 6: 5	TWCMSEL	0x0	rw	Two-way counting mode selection 00: One-way counting mode, depending on the OWCDIR bit 01: Two-way counting mode1, count up and down alternately, the CxIF bit is set only when the counter counts down 10: Two-way counting mode2, count up and down alternately, the CxIF bit is set only when the counter counts up 11: Two-way counting mode3, count up and down alternately, the CxIF bit is set when the counter counts up / down
Bit 4	OWCDIR	0x0	rw	One-way count direction 0: Up 1: Down
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is use to select whether to stop counting at an update event 0: The counter does not stop at an update event 1: The counter stops at an update event
Bit 2	OVFS	0x0	rw	Overflow event source This bit is used to select overflow event or DMA request sources. 0: Counter overflow, setting the OVFSWTR bit or overflow event generated by slave timer controller 1: Only counter overflow generates an overflow event
Bit 1	OVFEN	0x0	rw	Overflow event enable 0: Enabled 1: Disabled
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.3.4.2 TMR1 control register2 (TMR1_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	C4IOS	0x0	rw	Channel 4 idle output state
Bit 13	C3CIOS	0x0	rw	Channel 3 complementary idle output state
Bit 12	C3IOS	0x0	rw	Channel 3 idle output state
Bit 11	C2CIOS	0x0	rw	Channel 2 complementary idle output state
Bit 10	C2IOS	0x0	rw	Channel 2 idle output state
Bit 9	C1CIOS	0x0	rw	Channel 1 complementary idle output state OEN = 0 after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state OEN = 0 after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7	C1INSEL	0x0	rw	C1IN selection 0: CH1 pin is connected to C1IRAW input 1: The XOR result of CH1, CH2 and CH3 pins is connected to C1IRAW input
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the TMRx signal sent to the slave timer. 000: Reset 001: Enable 010: Update 011: Compare pulse 100: C1ORAW signal 101: C2ORAW signal 110: C3ORAW signal 111: C4ORAW signal
Bit 3	DRS	0x0	rw	DMA request source 0: Capture/compare event 1: Overflow event
Bit 2	CCFS	0x0	rw	Channel control bit flash selection This bit only acts on channels with complementary output. If the channel control bits are buffered: 0: Control bits are updated by setting the HALL bit 1: Control bits are updated by setting the HALL bit or a rising edge on TRGIN.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	CBCTRL	0x0	rw	Channel buffer control This bit acts on channels that have complementary output. 0: CxEN, CxCEN and CxOCTRL bits are not buffered. 1: CxEN, CxCEN and CxOCTRL bits are not buffered.

14.3.4.3 TMR1 slave timer control register (TMR1_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15	ESP	0x0	rw	External signal polarity 0: High or rising edge 1: Low or falling edge
Bit 14	ECMBEN	0x0	rw	External clock mode B enable This bit is used to enable external clock mode B 0: Disabled 1: Enabled
Bit 13: 12	ESDIV	0x0	rw	External signal divide This field is used to select the frequency division of an external trigger 00: Normal 01: Divided by 2 10: Divided by 4

				11: Divided by 8
Bit 11: 8	ESF	0x0	rw	External signal filter This field is used to filter an external signal. The external signal can be sampled only after it has been generated N times 0000: No filter, sampling by f_{DTS} 0001: $f_{SAMPLING}$ =f_{CK_INT}, N=2 0010: $f_{SAMPLING}$ =f_{CK_INT}, N=4 0011: $f_{SAMPLING}$ =f_{CK_INT}, N=8 0100: $f_{SAMPLING}$ =f_{DTS}/2, N=6 0101: $f_{SAMPLING}$ =f_{DTS}/2, N=8 0110: $f_{SAMPLING}$ =f_{DTS}/4, N=6 0111: $f_{SAMPLING}$=f_{DTS}/4, N=8 1000: $f_{SAMPLING}$=f_{DTS}/8, N=6 1001: $f_{SAMPLING}$=f_{DTS}/8, N=8 1010: $f_{SAMPLING}$=f_{DTS}/16, N=5 1011: $f_{SAMPLING}$=f_{DTS}/16, N=6 1100: $f_{SAMPLING}$=f_{DTS}/16, N=8 1101: $f_{SAMPLING}$=f_{DTS}/32, N=5 1110: $f_{SAMPLING}$=f_{DTS}/32, N=6 1111: $f_{SAMPLING}$=f_{DTS}/32, N=8
Bit 7	STS	0x0	rw	Subordinate TMR synchronization If enabled, master and slave timer can be synchronized. 0: Disabled 1: Enabled
Bit 6: 4	STIS	0x0	rw	Subordinate TMR input selection This field is used to select the subordinate TMR input. 000: Internal selection 0 (IS0) 001: Internal selection 1 (IS1) 010: Internal selection 2 (IS2) 011: Internal selection 3 (IS3) 100: C1IRAW input detector (C1INC) 101: Filtered input 1 (C1IF1) 110: Filtered input 2 (C1IF2) 111: External input (EXT) Refer to *Table 14-11* for details on ISx for each timer.
Bit 3	COSSEL	0x0	resd	Channel output switch selection This bit is used to select the switch source of CxORAW. 0: Select EXT as CxORAW switch 1: Select CxORAW_OFF as CxORAW switch
Bit 2: 0	SMSEL	0x0	rw	Subordinate TMR mode selection 000: Slave mode is disabled 001: Encoder mode A 010: Encoder mode B 011: Encoder mode C 100: Reset mode - Rising edge of the TRGIN input reinitializes the counter 101: Suspend mode - The counter starts counting when the TRGIN is high 110: Trigger mode - A trigger event is generated at the rising edge of the TRGIN input 111: External clock mode A - Rising edge of the TRGIN input clocks the counter Note: Refer to count mode section for details on encoder mode A/B/C.

14.3.4.4TMR1 DMA/interrupt enable register (TMR1_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	HALLDE	0x0	rw	HALL DMA request enable

				0: Disabled 1: Enabled
Bit 12	C4DEN	0x0	rw	Channel 4 DMA request enable 0: Disabled 1: Enabled
Bit 11	C3DEN	0x0	rw	Channel 3 DMA request enable 0: Disabled 1: Enabled
Bit 10	C2DEN	0x0	rw	Channel 2 DMA request enable 0: Disabled 1: Enabled
Bit 9	C1DEN	0x0	rw	Channel 1 DMA request enable 0: Disabled 1: Enabled
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7	BRKIE	0x0	rw	Break interrupt enable 0: Disabled 1: Enabled
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4	C4IEN	0x0	rw	Channel 4 interrupt enable 0: Disabled 1: Enabled
Bit 3	C3IEN	0x0	rw	Channel 3 interrupt enable 0: Disabled 1: Enabled
Bit 2	C2IEN	0x0	rw	Channel 2 interrupt enable 0: Disabled 1: Enabled
Bit 1	C1IEN	0x0	rw	Channel 1 interrupt enable 0: Disabled 1: Enabled
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.3.4.5 TMR1 interrupt status register (TMR1_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	C4RF	0x0	rw0c	Channel 4 recapture flag Please refer to C1RF description.
Bit 11	C3RF	0x0	rw0c	Channel 3 recapture flag Please refer to C1RF description.
Bit 10	C2RF	0x0	rw0c	Channel 2 recapture flag Please refer to C1RF description.
Bit 9	C1RF	0x0	rw0c	Channel 1 recapture flag This bit indicates whether a recapture is detected when C1IF=1. This bit is set by hardware, and cleared by writing “0”. 0: No capture is detected 1: Capture is detected.
Bit 8	Reserved	0x0	resd	Default value
Bit 7	BRKIF	0x0	rw0c	Break interrupt flag This bit indicates whether the break input is active or not. It is set by hardware and cleared by writing “0” 0: Inactive level 1: Active level

Bit 6	TRGIF	0x0	rw0c	Trigger interrupt flag This bit is set by hardware on a trigger event. It is cleared by writing “0”. 0: No trigger event occurs 1: Trigger event is generated. Trigger event: an active edge is detected on TRGIN input, or any edge in suspend mode.
Bit 5	HALLIF	0x0	rw0c	HALL interrupt flag This bit is set by hardware on HALL event. It is cleared by writing “0”. 0: No Hall event occurs. 1: Hall event is detected. HALL even: CxEN, CxCEN and CxOCTRL are updated.
Bit 4	C4IF	0x0	rw0c	Channel 4 interrupt flag Please refer to C1IF description.
Bit 3	C3IF	0x0	rw0c	Channel 3 interrupt flag Please refer to C1IF description.
Bit 2	C2IF	0x0	rw0c	Channel 2 interrupt flag Please refer to C1IF description.
Bit 1	C1IF	0x0	rw0c	Channel 1 interrupt flag If the channel 1 is configured as input mode: This bit is set by hardware on a capture event. It is cleared by software or read access to the TMRx_C1DT 0: No capture event occurs 1: Capture event is generated If the channel 1 is configured as output mode: This bit is set by hardware on a compare event. It is cleared by software. 0: No compare event occurs 1: Compare event is generated
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware on an overflow event. It is cleared by software. 0: No overflow event occurs 1: Overflow event is generated. If OVFEN=0 and OVFS=0 in the TMRx_CTRL1 register: − An overflow event is generated when OVFG= 1 in the TMRx_SWEVE register; − An overflow event is generated when the counter CVAL is reinitialized by a trigger event.

14.3.4.6 TMR1 software event register (TMR1_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x000	resd	Kept at its default value.
Bit 7	BRKSWTR	0x0	wo	Break event triggered by software This bit is set by software to generate a break event. 0: No effect 1: Generate a break event.
Bit 6	TRGSWTR	0x0	rw	Trigger event triggered by software This bit is set by software to generate a trigger event. 0: No effect 1: Generate a trigger event.
Bit 5	HALLSWTR	0x0	wo	HALL event triggered by software This bit is set by software to generate a HALL event. 0: No effect 1: Generate a HALL event. Note: This bit acts only on channels that have complementary output.
Bit 4	C4SWTR	0x0	wo	Channel 4 event triggered by software Please refer to C1M description.
Bit 3	C3SWTR	0x0	wo	Channel 3 event triggered by software Please refer to C1M description.
Bit 2	C2SWTR	0x0	wo	Channel 2 event triggered by software Please refer to C1M description
Bit 1	C1SWTR	0x0	wo	Channel 1 event triggered by software This bit is set by software to generate a channel 1 event. 0: No effect 1: Generate a channel 1 event.
Bit 0	OVFSWTR	0x0	wo	Overflow event triggered by software This bit is set by software to generate an overflow event. 0: No effect 1: Generate an overflow event.

14.3.4.7 TMR1 channel mode register1 (TMR1_CM1)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C2OSEN	0x0	rw	Channel 2 output switch enable
Bit 14: 12	C2OCTRL	0x0	rw	Channel 2 output control
Bit 11	C2OBEN	0x0	rw	Channel 2 output buffer enable
Bit 10	C2OIEN	0x0	rw	Channel 2 output enable immediately
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS register.
Bit 7	C1OSEN	0x0	rw	Channel 1 output switch enable 0: C1ORAW is not affected by EXT input. 1: Once a high level is detect on EXT input, clear C1ORAW.
Bit 6: 4	C1OCTRL	0x0	rw	Channel 1 output control This field defines the behavior of the original signal C1ORAW.

				000: Disconnected. C1ORAW is disconnected from C1OUT; 001: C1ORAW is high when TMRx_CVAL=TMRx_C1DT 010: C1ORAW is low when TMRx_CVAL=TMRx_C1DT 011: Switch C1ORAW level when TMRx_CVAL=TMRx_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A －OWCDIR=0, C1ORAW is high once TMRx_C1DT>TMRx_CVAL, else low; －OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVAL, else high; 111: PWM mode B － OWCDIR=0, C1ORAW is low once TMRx_ C1DT >TMRx_CVAL, else high; － OWCDIR=1, C1ORAW is high once TMRx_ C1DT <TMRx_CVAL, else low. *Note: In the configurations other than 000', the C1OUT is connected to C1ORAW. The C1OUT output level is not only subject to the changes of C1ORAW, but also the output polarity set by CCTRL.*
Bit 3	C1OBEN	0x0	rw	Channel 1 output buffer enable 0: Buffer function of TMRx_C1DT is disabled. The new value written to the TMRx_C1DT takes effect immediately. 1: Buffer function of TMRx_C1DT is enabled. The value to be written to the TMRx_C1DT is stored in the buffer register, and can be sent to the TMRx_C1DT register only on an overflow event.
Bit 2	C1OIEN	0x0	rw	Channel 1 output enable immediately In PWM mode A or B, this bit is used to accelerate the channel 1 output's response to the trigger event. 0: Need to compare the CVAL with C1DT before generating an output 1: No need to compare the CVAL and C1DT. An output is generated immediately when a trigger event occurs.
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C2DF	0x0	rw	Channel 2 digital filter
Bit 11: 10	C2IDIV	0x0	rw	Channel 2 input divider
Bit 9: 8	C2C	0x0	rw	Channel 2 configuration This field is used to define the direction of the channel 2 (input or output), and the selection of input pin when C2EN='0': 00: Output 01: Input, C2IN is mapped on C2IFP2 10: Input, C2IN is mapped on C1IFP2 11: Input, C2IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C1DF	0x0	rw	Channel 1 digital filter This field defines the digital filter of the channel 1. N stands for the number of filtering, indicating that the input edge can pass the filter only after N sampling events.

				0000: No filter, sampling is done at f_{DTS} 1000: $f_{SAMPLING}=f_{DTS}/8$, N=6 0001: $f_{SAMPLING}=f_{CK\_INT}$, N=2 1001: $f_{SAMPLING}=f_{DTS}/8$, N=8 0010: $f_{SAMPLING}=f_{CK\_INT}$, N=4 1010: $f_{SAMPLING}=f_{DTS}/16$, N=5 0011: $f_{SAMPLING}=f_{CK\_INT}$, N=8 1011: $f_{SAMPLING}=f_{DTS}/16$, N=6 0100: $f_{SAMPLING}=f_{DTS}/2$, N=6 1100: $f_{SAMPLING}=f_{DTS}/16$, N=8 0101: $f_{SAMPLING}=f_{DTS}/2$, N=8 1101: $f_{SAMPLING}=f_{DTS}/32$, N=5 0110: $f_{SMPLING}=f_{DTS}/4$, N=6 1110: $f_{SAMPLING}=f_{DTS}/32$, N=6 0111: $f_{SAMPLING}=f_{DTS}/4$, N=8 1111: $f_{SAMPLING}=f_{DTS}/32$, N=8
Bit 3: 2	C1IDIV	0x0	rw	Channel 1 input divider This field defines Channel 1 input divider. 00: No divider. An input capture is generated at each active edge. 01: An input compare is generated every 2 active edges 10: An input compare is generated every 4 active edges 11: An input compare is generated every 8 active edges Note: the divider is reset once C1EN='0'
Bit 1: 0	C1C	0x0	rw	Channel 1 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C1EN='0': 00: Output 01: Input, C1IN is mapped on C1IFP1 10: Input, C1IN is mapped on C2IFP1 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.3.4.8TMR1 channel mode register2 (TMR1_CM2)

The channel can be used in input (capture mode) or output (compare mode). The direction of a channel is defined by the corresponding CxC bits. All the other bits of this register have different functions in input and output modes. The CxOx describes its function in output mode when the channel is in output mode, while the CxIx describes its function in output mode when the channel is in input mode. Attention must be given to the fact that the same bit can have different functions in input mode and output mode.

Output compare mode:

Bit	Register	Reset value	Type	Description
Bit 15	C4OSEN	0x0	rw	Channel 4 output switch enable
Bit 14: 12	C4OCTRL	0x0	rw	Channel 4 output control
Bit 11	C4OBEN	0x0	rw	Channel 4 output buffer enable
Bit 10	C4OIEN	0x0	rw	Channel 4 output enable immediately
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7	C3OSEN	0x0	rw	Channel 3 output switch enable
Bit 6: 4	C3OCTRL	0x0	rw	Channel 3 output control
Bit 3	C3OBEN	0x0	rw	Channel 3 output buffer enable
Bit 2	C3OIEN	0x0	rw	Channel 3 output enable immediately
Bit 1: 0	C3C	0x0	rw	Channel 3 configuration

				This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

Input capture mode:

Bit	Register	Reset value	Type	Description
Bit 15: 12	C4DF	0x0	rw	Channel 4 digital filter
Bit 11: 10	C4IDIV	0x0	rw	Channel 4 input divider
Bit 9: 8	C4C	0x0	rw	Channel 4 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C4EN='0': 00: Output 01: Input, C4IN is mapped on C4IFP4 10: Input, C4IN is mapped on C3IFP4 11: Input, C4IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.
Bit 7: 4	C3DF	0x0	rw	Channel 3 digital filter
Bit 3: 2	C3IDIV	0x0	rw	Channel 3 input divider
Bit 1:0	C3C	0x0	rw	Channel 3 configuration This field is used to define the direction of the channel 1 (input or output), and the selection of input pin when C3EN='0': 00: Output 01: Input, C3IN is mapped on C3IFP3 10: Input, C3IN is mapped on C4IFP3 11: Input, C3IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.3.4.9TMR1 Channel control register (TMR1_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept its default value.
Bit 13	C4P	0x0	rw	Channel 4 polarity Please refer to C1P description.
Bit 12	C4EN	0x0	rw	Channel 4 enable Please refer to C1EN description.
Bit 11	C3CP	0x0	rw	Channel 3 complementary polarity Please refer to C1P description.
Bit 10	C3CEN	0x0	rw	Channel 3 complementary enable Please refer to C1EN description.
Bit 9	C3P	0x0	rw	Channel 3 polarity Please refer to C1P description.
Bit 8	C3EN	0x0	rw	Channel 3 enable Please refer to C1EN description.
Bit 7	C2CP	0x0	rw	Channel 2 complementary polarity Please refer to C1P description.
Bit 6	C2CEN	0x0	rw	Channel 2 complementary enable Please refer to C1EN description.
Bit 5	C2P	0x0	rw	Channel 2 polarity Please refer to C1P description.
Bit 4	C2EN	0x0	rw	Channel 2 enable Please refer to C1EN description.
Bit 3	C1CP	0x0	rw	Channel 1 complementary polarity 0: C1COUT is active high. 1: C1COUT is active low.
Bit 2	C1CEN	0x0	rw	Channel 1 complementary enable 0: Output is disabled. 1: Output is enabled.
Bit 1	C1P	0x0	rw	Channel 1 polarity

				When the channel 1 is configured as output mode: 0: C1OUT is active high 1: C1OUT is active low When the channel 1 is configured as input mode: 0: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 1: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-14 Complementary output channel CxOUT and CxCOUT control bits with break function

Control bit					Output state (1)	
OEN bit	FCSODIS bit	FCSOEN bit	CxEN bit	CxCEN bit	CxOUT output state	CxCOUT output state
1	X	0	0	0	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	0	1	Output disabled (no driven by the timer) CxOUT=0, Cx_EN=0	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+ polarity CxOUT= CxORAW xor CxP, Cx_EN=1	Output disabled (no driven by the timer) CxCOUT=0, CxCEN=0
		0	1	1	CxORAW+polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
		1	0	0	Output disabled (no driven by the timer) CxOUT=CxP, Cx_EN=0	Output disabled (no driven by the timer) CxCOUT=CxCP, CxCEN=0
		1	0	1	Off-state (Output enabled with inactive level) CxOUT=CxP, Cx_EN=1	CxORAW + polarity, CxCOUT= CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + polarity, CxOUT= CxORAW xor CxP, Cx_EN=1	Off-state (Output enabled with inactive level) CxCOUT=CxCP, CxCEN=1
		1	1	1	CxORAW+ polarity+dead-time, Cx_EN=1	CxORAW inverted+polarity+dead-time, CxCEN=1
0	0	X	0	0	Output disabled (no driven by the timer) Asynchronously: CxOUT=CxP, Cx_EN=0, CxCOUT=CxCP, CxCEN=0; If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	Off-state (Output enabled with inactive level) Asynchronously: CxOUT =CxP, Cx_EN=1, CxCOUT=CxCP, CxCEN=1; If the clock is present: after a dead-time, CxOUT=CxIOS, CxCOUT=CxCIOS, assuming that CxIOS and CxCIOS do not correspond to CxOUT and CxCOUT active level.	
	1		0	1		
	1		1	0		
	1		1	1		

Note: If the two outputs of a channel are not used (CxEN = CxCEN = 0), CxIOS, CxCIOS, CxP and CxCP must be cleared.

Note: The state of the external I/O pins connected to the complementary CxOUT and CxCOUT channels depends on the CxOUT and CxCOUT channel state and the GPIO and the IOMUX registers.

14.3.4.10 TMR1 counter value (TMR1_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.3.4.11 TMR1 division value (TMR1_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK}$ / (DIV[15: 0]+1). The value of this register is transferred to the actual prescaler register when an overflow event occurs.

14.3.4.12 TMR1 period register (TMR1_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This defines the period value of the TMRx counter. The timer stops working when the period value is 0.

14.3.4.13 TMR1 repetition period register (TMR1_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	rw	Kept at its default value.
Bit 7: 0	RPR	0x00	rw	Repetition of period value This field is used to reduce the generation rate of overflow events. An overflow event is generated when the repetition counter reaches 0.

14.3.4.14 TMR1 channel 1 data register (TMR1_C1DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C1DT	0x0000	rw	Channel 1 data register When the channel 1 is configured as input mode: The C1DT is the CVAL value stored by the last channel 1 input event (C1IN) When the channel 1 is configured as output mode: C1DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C1OBEN bit, and the corresponding output is generated on C1OUT as configured.

14.3.4.15 TMR1 channel 2 data register (TMR1_C2DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C2DT	0x0000	rw	Channel 2 data register When the channel 2 is configured as input mode: The C2DT is the CVAL value stored by the last channel 2 input event (C1IN) When the channel 2 is configured as output mode: C2DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C2OBEN bit, and the corresponding output is generated on C2OUT as configured.

14.3.4.16 TMR1 channel 3 data register (TMR1_C3DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C3DT	0x0000	rw	Channel 3 data register When the channel 3 is configured as input mode: The C3DT is the CVAL value stored by the last channel 3 input event (C1IN) When the channel 3 is configured as output mode: C3DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C3OBEN bit, and the corresponding output is generated on C3OUT as configured.

14.3.4.17 TMR1 channel 4 data register (TMRx_C4DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C4DT	0x0000	rw	Channel 4 data register When the channel 4 is configured as input mode: The C4DT is the CVAL value stored by the last channel 4 input event (C1IN) When the channel 3 is configured as output mode: C4DT is the value to be compared with the CVAL value. Whether the written value takes effective immediately depends on the C4OBEN bit, and the corresponding output is generated on C4OUT as configured.

14.3.4.18 TMR1 break register (TMR1_BRK)

Bit	Register	Reset value	Type	Description
Bit 15	OEN	0x0	rw	Output enable This bit acts on the channels as output. It is used to enable CxOUT and CxCOUT outputs. 0: Disabled 1: Enabled
Bit 14	AOEN	0x0	rw	Automatic output enable OEN is set automatically at an overflow event. 0: Disabled 1: Enabled
Bit 13	BRKV	0x0	rw	Break input validity This bit is used to select the active level of a break input. 0: Break input is active low. 1 Break input is active high.
Bit 12	BRKEN	0x0	rw	Break enable This bit is used to enable break input. 0: Break input is disabled. 1: Break input is enabled.
Bit 11	FCSOEN	0x0	rw	Frozen channel status when holistic output enable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and OEN=1. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output inactive level.
Bit 10	FCSODIS	0x0	rw	Frozen channel status when holistic output disable This bit acts on the channels that have complementary output. It is used to set the channel state when the timer is inactive and OEN=0. 0: CxOUT/CxCOUT outputs are disabled. 1: CxOUT/CxCOUT outputs are enabled. Output idle level.
Bit 9: 8	WPC	0x0	rw	Write protection configuration This field is used to enable write protection. 00: Write protection is OFF. 01: Write protection level 3, and the following bits are write protected:

				TMRx_BRK: DTC, BRKEN, BRKV and AOEN TMRx_CTRL2: CxIOS and CxCIOS 10: Write protection level 2. The following bits and all bits in level 3 are write protected: TMRx_CCTRL: CxP and CxCP TMRx_BRK: FCSODIS and FCSOEN 11: Write protection level 1. The following bits and all bits in level 2 are write protected: TMRx_CMx: C2OCTRL and C2OBEN Note: Once WPC>0, its content remains frozen until the next system reset.
Bit 7: 0	DTC	0x00	rw	Dead-time configuration This field defines the duration of the dead-time insertion. The 3-bit MSB of DTC[7: 0] is used for function selection: 0xx: DT = DTC [7: 0] * TDTS 10x: DT = (64+ DTC [5: 0]) * TDTS * 2 110: DT = (32+ DTC [4: 0]) * TDTS * 8 111: DT = (32+ DTC [4: 0]) * TDTS * 16

Note: Based on lock configuration, AOEN, BRKV, BRKEN, FCSODIS, FCSOEN and DTC[7:0] can all be write protected. Thus it is necessary to configure write protection when writing to the TMRx_BRK register for the first time.

14.3.4.19 TMR1 DMA control register (TMR1_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at its default value.
Bit 12:8	DTB	0x00	rw	DMA transfer bytes This field defines the number of DMA transfers: 00000: 1 byte 00001: 2 bytes 00010: 3 bytes 00011: 4 bytes .... 10000: 17 bytes 10001: 18 bytes
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	ADDR	0x00	rw	DMA transfer address offset ADDR is defined as an offset starting from the address of the TMRx_CTRL1 register: 00000: TMRx_CTRL1 00001: TMRx_CTRL2 00010: TMRx_STCTRL .....

14.3.4.20 TMR1 DMA data register (TMR1_DMADT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DMADT	0x0000	rw	DMA data register A write/read operation to the DMADT register accesses any TMR register located at the following address: TMRx peripheral address + ADDR*4 to TMRx peripheral address + ADDR*4 + DTB*4

15 Window watchdog timer (WWDT)

15.1 WWDT introduction

The window watchdog downcounter must be reloaded in a limited time window to prevent the watchdog circuits from generating a system reset. The window watch dog is used to detect the occurrence of system malfunctions.

The window watchdog timer is clocked by a divided APB1_CLK. The precision of the APB1_CLK enables the window watchdog to take accurate control of the limited window.

15.2 WWDT main features

- 7-bit downcounter
- If the watchdog is enabled, a system reset is generated when the value of the downcounter is less than 0x40 or when the downcounter is reloaded outside the window.
- The downcounter can be reloaded by enabling the counter interrupt.

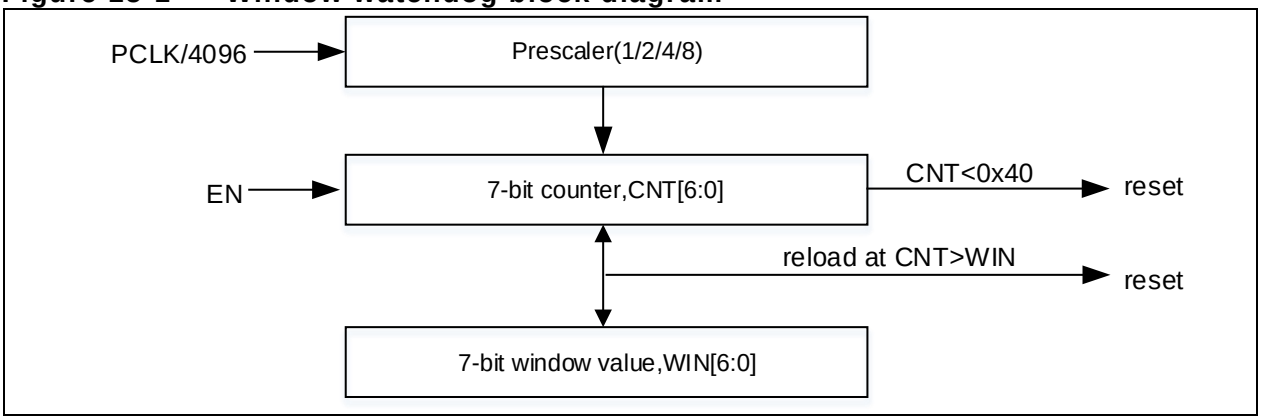
15.3 WWDT functional overview

If the watchdog is enabled, a system reset is generated at the following conditions:

When the 7-bit downcounter scrolls from 0x40 to 0x3F;

When the counter is reloaded while the 7-bit downcounter is greater than the value programmed in the window register.

Figure 15-1 Window watchdog block diagram



To prevent system reset, the counter must be reloaded only when its value is less than the value stored in the window register and greater than 0x40.

The WWDT counter is clocked by a divided APB1_CLK, with the division factor being defined by the DIV[1: 0] bit in the WWDT_CFG register. The counter value determines the maximum counter period before the watchdog generates a reset. The WIN[6: 0] bit can be used to configure the window value.

WWDT offers reload counter interrupt feature. If enabled, the WWDT will set the RLDF flag when the counter value reaches 0x40h, and an interrupt is generated accordingly. The interrupt service routine (ISTS) can be used to reload the counter to prevent a system reset. Note that if CNT[6]=0, setting the WWDTEN bit will generate a system reset, so the CNT[6] bit must be always set (CNT[6]=1) while writing to the WWDT_CTRL register to prevent the occurrence of an immediate reset once the window watchdog is enabled.

The formula to calculate the window watchdog time out:

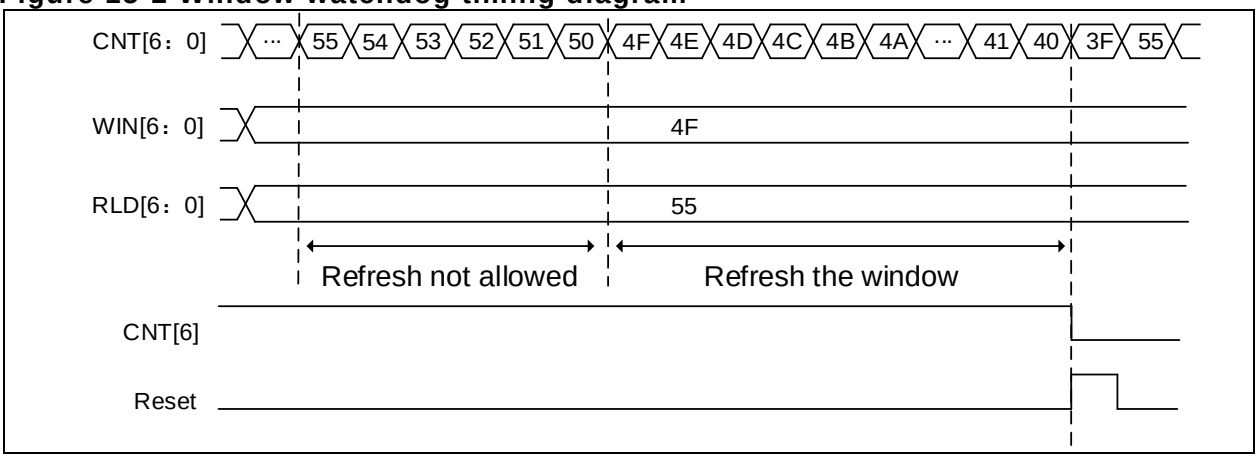
$$T_{WWDT} = T_{PCLK1} \times 4096 \times 2^{DIV[1:0]} \times (CNT[5:0] + 1); (ms)$$

Where: T_{PCLK1} refers to APB1 clock period, in ms.

Table 15-1 Minimum and maximum timeout value when PCLK1=72 MHz

Prescaler	Min. Timeout value	Max. Timeout value
0	56.5μs	3.64ms
1	113.5μs	7.28ms
2	227.5μs	14.56ms
3	455μs	29.12ms

Figure 15-2 Window watchdog timing diagram



15.4 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the WWDT counter stops counting by setting the WWDT_PAUSE in the DEBUG module. Refer to Chapter 30.2 for more information.

15.5 WWDT registers

These peripheral registers must be accessed by word (32 bits).

Table 15-2 WWDT register map and reset value

Register name	Offset	Reset value
WWDT_CTRL	0x00	0x7F
WWDT_CFG	0x04	0x7F
WWDT_STS	0x08	0x00

15.5.1 Control register (WWDT_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	WWDTEN	0x0	rw1s	Window watchdog enable 0: Disabled 1: Enabled This bit is set by software, but can be cleared only after reset.
Bit 6: 0	CNT	0x7F	rw	Downcounter When the counter counts down to 0x3F, a reset is generated.

15.5.2 Configuration register (WWDT_CFG)

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	RLDIEN	0x0	rw	Reload counter interrupt 0: Disabled 1: Enabled
Bit 8: 7	DIV	0x0	rw	Clock division value 00: PCLK1 divided by 4096 01: PCLK1 divided by 8192 10: PCLK1 divided by 16384 11: PCLK1 divided by 32768
Bit 6: 0	WIN	0x7F	rw	Window value If the counter is reloaded while its value is greater than the window register value, a reset is generated. The counter must be reloaded between 0x40 and WIN[6: 0].

15.5.3 Status register (WWDT_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	RLDF	0x0	rw0c	Reload counter interrupt flag This flag is set when the downcounter reaches 0x40. 'This bit is set by hardware and cleared by software.

16 Watchdog timer (WDT)

16.1 WDT introduction

The WDT is driven by a dedicated low-speed clock (LICK). Due to the lower clock accuracy of LICK, the WDT is best suited to the applications that have lower timing accuracy and can run independently outside the main application.

16.2 WDT main features

- 12-bit downcounter
- The counter is clocked by LICK (can work in Stop and Standby modes)
- A system reset is generated when the counter value is decremented to 0

16.3 WDT functional overview

WDT enable:

Both software and hardware operations can be used to enable WDT. In other words, the WDT can be enabled by writing 0xCCCC to the WDT_CMD register; or when the user enables the hardware watchdog through user system data area, the WDT will be automatically enabled after power-on reset.

WDT reset:

When the counter value of the WDT counts down to 0, a WDT reset be generated. Thus the WDT_CMD register must be written with the value 0xAAAA at regular intervals to reload the counter value to avoid the WDT reset.

WDT write-protected:

The WDT_DIV and WDT_RLD registers are write-protected. Writing the value 0x5555 to the WDT_CMD register will unlock write protection. The update status of these two registers are indicated by the DIVF and RLDF bits in the WDT_STS register. If a different value is written to the WDT_CMD register, these two registers will be re-protected. Writing the value 0xAAAA to the WDT_CMD register also enables write protection.

WDT clock:

The WDT counter is clocked by the LICK. The LICK is an internal RC clock. The timeout period is also within a certain range, so a margin should be taken into account when configuring timeout period. The LICK can be calibrated to obtain the WDT timeout with a relatively accuracy. Refer to *Section 4.1.1* for details.

Figure 16-1 WDT block diagram

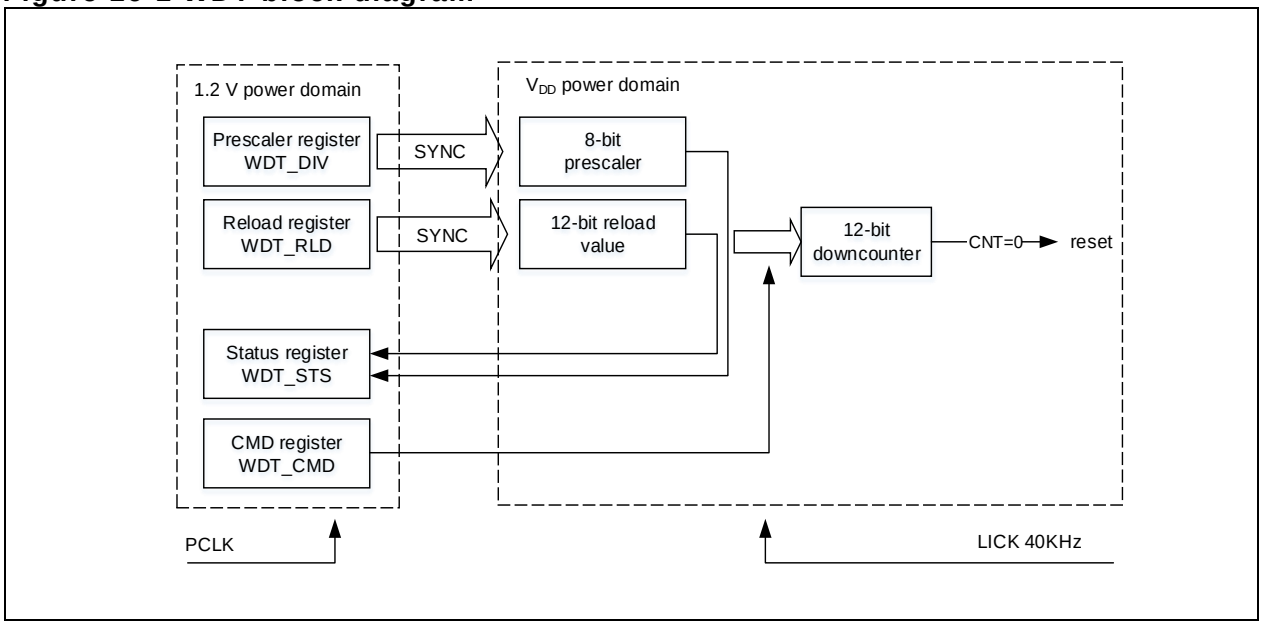


Table 16-1 WDT timeout period (LICK=40kHz)

Prescaler divider	DIV[2: 0] bits	Min.timeout (ms) RLD[11: 0] = 0x000	Max. timeout (ms) RLD[11: 0] = 0xFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 or 7)	6.4	26214.4

16.4 Debug mode

When the microcontroller enters debug mode (Cortex®-M4 core halted), the WDT counter stops counting by setting the WDT_PAUSE in the DEBUG module.

16.5 WDT registers

These peripheral registers must be accessed by words (32 bits).

Table 16-2 WDT register and reset value

Register name	Offset	Reset value
WDT_CMD	0x00	0x0000 0000
WDT_DIV	0x04	0x0000 0000
WDT_RLD	0x08	0x0000 0FFF
WDT_STS	0x0C	0x0000 0000

16.5.1 Command register (WDT_CMD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	CMD	0x0000	wo	Command register 0xAAAA: Reload counter 0x5555: Unlock write-protected WDT_DIV and WDT_RLD 0xCCCC: Enable WDT. If the hardware watchdog has been enabled, ignore this operation.

16.5.2 Divider register (WDT_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2: 0	DIV	0x0	rw	Clock division value 000: LICK divided by 4 001: LICK divided by 8 010: LICK divided by 16 011: LICK divided by 32 100: LICK divided by 64 101: LICK divided by 128 110: LICK divided by 256 111: LICK divided by 256 The write protection must be unlocked in order to enable write access to the register. The register can be read only when DIVF=0.

16.5.3 Reload register (WDT_RLD)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11: 0	RLD	0xFFF	rw	Reload value The write protection must be unlocked in order to enable write access to the register. The register can be read only when RLDF=0.

16.5.4 Status register (WDT_STS)

(Reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1	RLDF	0x0	ro	Reload value update complete flag 0: Reload value update complete 1: Reload value update is in process. The reload register WDT_RLD can be written only when RLDF=0
Bit 0	DIVF	0x0	ro	Division value update complete flag 0: Division value update complete 1: Division value update is in process. The divider register WDT_DIV can be written only when DIVF=0

17 Enhanced real-time clock (ERTC)

17.1 ERTC introduction

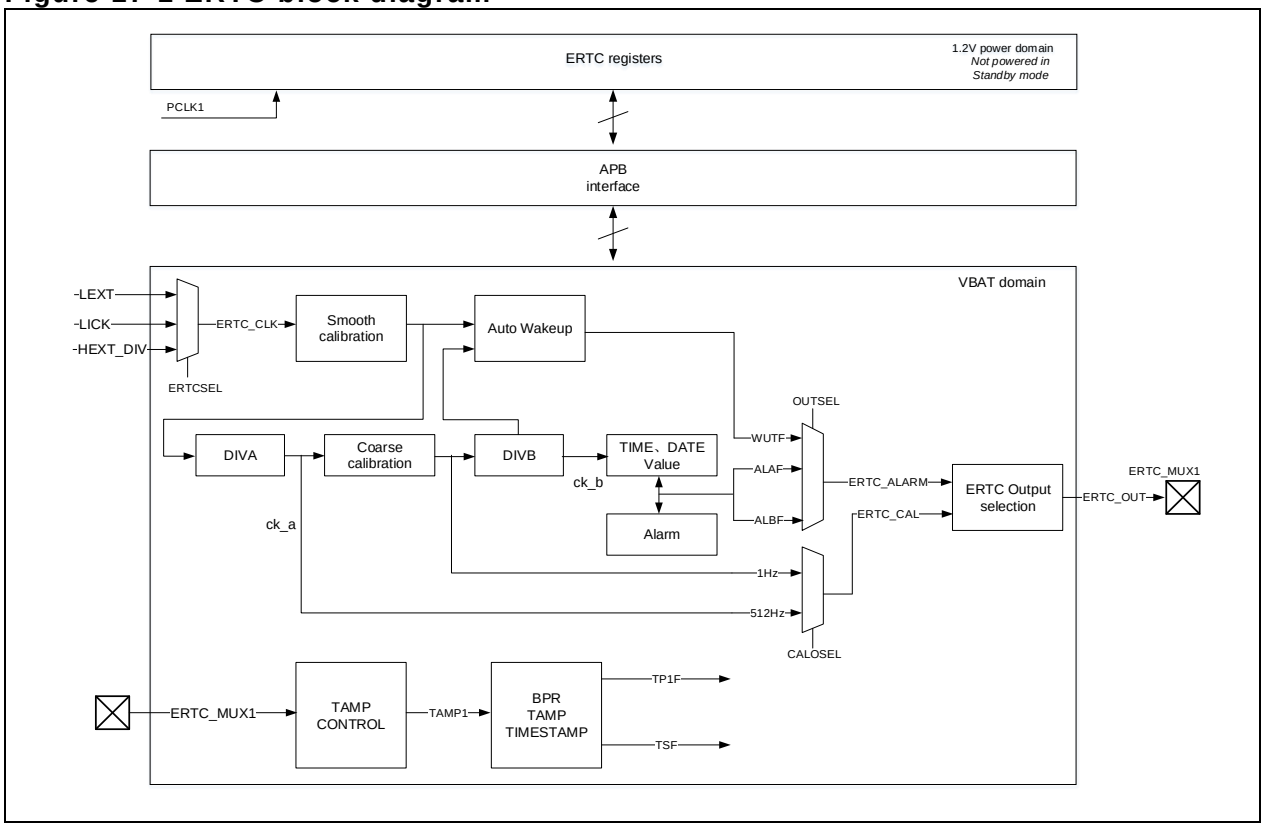
The real-time clock provides a calendar clock function. The time and date can be modified by modifying the ERTC_TIME and ERTC_DATE register.

The ERTC module is in battery powered domain, which means that it keeps running and free from the influence of system reset and VDD power off as long as VBAT is powered.

17.2 ERTC main features

- Real-time calendar (automatic processing of month days, including 28 (February in a common year), 29 (February in a leap year), 30 (a lunar month of 30 days) and 31 (a solar month of 31 days), where the current register being a multiple of 4 indicates a leap year), two programmable alarms
- Periodic auto-wakeup
- Reference clock detection
- 2x programmable tamper detection, supporting time stamp feature
- Supports fine and coarse calibration
- 20 x battery powered registers
- 5 x interrupts: alarm A/B, periodic auto-wakeup, tamper detection and time stamp
- Multiplexed function output, calibration clock output, alarm events or wakeup events
- Multiplexed function input, reference clock input, one-channel tamper detection and time stamp

Figure 17-1 ERTC block diagram



17.3 ERTC function overview

17.3.1 ERTC clock

ERTC clock source (ERTC_CLK) is selected via clock controller from a LEXT, LICK, and divided HEXT (selected through the ERTCSEL[1: 0] bit in the CRM_BPDC register).

The ERTC embeds two dividers: A and B, programmed by the DIVA[6: 0] and DIVB[14: 0] respectively. It is recommended that the DIVA is configured to a higher value in order to minimum power consumption. After being divided by prescaler A and B, the ERTC_CLK generates ck_a and ck_b clocks, respectively. The ck_a is used for subsecond update, while the ck_b is used for calendar update and periodic autowakeup. The clock frequency of ck_a and ck_b can be obtained from the following equation:

$$F_{ck\_a} = \frac{f_{ERTC\_CLK}}{DIVA + 1}$$

$$F_{ck\_b} = \frac{f_{ERTC\_CLK}}{(DIVB + 1) \times (DIVA + 1)}$$

To obtain ck_b with frequency of 1 Hz, DIVA=127, DIVB=255, and 32.768 kH LEXT should be used. This ck_b is then used for calendar update.

17.3.2 ERTC initialization

ERTC register write protection

After a power-on reset, all ERTC registers are write protected. Such protection mechanism is not affected by the system reset. Write access to the ERTC registers (except the ERTC_STS[14: 8], ERTC_TAMP and ERTC_BPRx registers) can be enabled by unlocking it.

To unlock the write protection of ERTC registers, the steps below should be respected:

1. Enable power interface clock by setting PWCEN=1 in the CRM_APB1EN register
2. Unlock write protection of the battery powered domain by setting BPWEN=1 in the PWC_CTRL register
3. Write 0xCA and 0x53 to the ERTC_WP register in sequence. Writing an incorrect key will activate the write protection again.

Table 17-1 lists the ERTC registers that can be configured only after the write protection is unlocked and when the initialization mode is entered.

Table 17-1 ERTC register map and reset values

Register	ERTC_WP enabled	Whether to enter initialization mode	Others
ERTC_TIME	Y	Y	-
ERTC_DATE	Y	Y	-
ERTC_CTRL	Y	Bit 7, 6 and 4 only	-
ERTC_STS	Y, except [14: 8]	-	-
ERTC_DIV	Y	Y	-
ERTC_WAT	Y	N	Configurable when WATWF=1
ERTC_ALA	Y	N	Configurable when ALAWF =1
ERTC_ALB	Y	N	Configurable when ALBWF =1
ERTC_WP			
ERTC_SBS	-	-	-
ERTC_TADJ	Y	N	Configurable when TADJF=0

ERTC_TSTM	-	-	-
ERTC_TSDT	-	-	-
ERTC_TSSBS	-	-	-
ERTC_SCAL	Y	N	Configurable when CALUPDF=0
ERTC_TAMP	N	N	-
ERTC_ALASBS	Y	N	Configurable when ALAWF =1
ERTC_ALASBS	Y	N	Configurable when ALBWF =1
ERTC_BPRx	N	N	-

Clock and calendar initialization

After the register write protection is unlocked, follow the procedure below for clock and calendar initialization:

1. Set the IMEN bit to enter initialization mode
2. Wait until the initialization flag INITF bit is set
3. Configure DIVB and DIVA.
4. Configure the clock and calendar values.
5. Leave the initialization mode by clearing the IMEN bit. Wait until the UPDF bit is set, indicating the completion of the calendar update. The calendar starts counting.

The ERTC also allows the fine-tuning for daylight saving time and clock.

Daylight saving time feature: It is used to increase (ADD1H=1) or decrease (DEC1H=1) one hour in the calendar, without completing the whole initialization process.

Clock calibration: It is used for the fine calibration of the current clock. If only DECSBS[14: 0] is configured, the value will be added to the DIVB counter and a clock latency will be generated. If only ADD1S bit is set, the current clock will increase by one second. If both DECSBS[14: 0] and ADDIS bit are configured, the clock will increase by a fraction of a second.

Time latency (ADD1S=0): DECSBS/(DIVB+1)

Time advance (ADD1S=1): 1-(DECSBS/(DIVB+1)

Note: To avoid subsecond overflow, SBS[15]=0 must be asserted before setting the ERTC_TADJ register. Reference clock detection and coarse digital calibration cannot be used at the same time. Thus when RCDEN=1, coarse digital calibration is not supported.

Reading the calendar

The ERTC offers two different ways to read the calendar, that is, synchronous read (DREN=0) and asynchronous read (DREN=1).

In the case of DREN=0, the clock and calendar values can be obtained by reading a synchronous shadow register via the PCLK1. The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK. The shadow register is reset by a system reset. To ensure consistency between the 3 values (ERTC_SBS, ERTC_TIME and ERTC_DATE registers), reading lower-order registers will lock the values in the higher-order registers until the ERTC_DATE register is read. For example, reading the ERTC_SBS register will lock the values in the ERTC_TIME and ERTC_DATE registers.

In the case of DREN=1, the ERTC will perform direct read access to the ERTC clock and calendar located in the battery powered domain with the PCLK1, avoiding the occurrence of errors caused by time synchronization. In this mode, the UPDF flag is cleared by hardware. To ensure the data is correct when reading clock and calendar, the software must read the clock and calendar registers twice, and compare the results of two read operations. If the result is not aligned, read again until that the results of two read accesses are consistent. Besides, it is also possible to compare the least significant bits of the two read operations to determine their consistency.

Note: In Standby and Deepsleep modes, the current calendar values are not copied into the shadow registers. When waking up from these two modes, UPDF=0 must be asserted, and then wait until UPDF=1, to ensure that the latest calendar value can be read. In synchronous read (DREN=0) mode, the frequency of the PCLK1 must be at least seven times the ERTC_CLK frequency. In asynchronous read (DREN=1), an additional APB cycle is required to complete the read operations of the calendar register.

Alarm clock initialization

The ERTC contains two programmable alarm clocks: alarm clock A and alarm clock B, and their respective interrupts.

The alarm clock value is programmed with the ERTC_ALASBS/ERTC_ALA (ERTC_ALBSBS/ERTC_ALB). When the programmed alarm value matches the calendar value, an alarm event is generated if an alarm clock is enabled. The MASKx bit can be used to selectively mask calendar fields. The calendar fields, which are masked, are not allocated with an alarm clock.

To configure the alarm clocks, the following steps should be respected:

1. Disable alarm clock A or alarm clock B (by setting ALAEN=0 or ALBEN=0)
2. Wait until the ALAWF or ALBWF bit is set to enable write access to the alarm clock A or B
3. Configure alarm clock A or B registers (ERTC_ALA/ERTC_ALASBS and ERTC_ALB/ERTC_ALBSBS)
4. Enable alarm clock A or B by setting ALAEN=1 or ALBEN=1

Note: If MASK1=0 in the ERTC_ALA or ERTC_ALB, the alarm clock can work normally only when the DIVB value is at least equal to 3.

17.3.3 Periodic automatic wakeup

Periodical automatic wakeup unit is used to wake up ERTC from low power consumption modes automatically. The period is programmed with the VAL[15: 0] bit (When WATCLK[2]=1, it is extended to 17 bits, and the wakeup counter value is VAL+2^{16}). When the wakeup counter value drops from the VAL to zero, the WATF bit is set, and a wakeup event is generated, with the wakeup counter being reloaded with the VAL value. An interrupt is also generated if a periodic wakeup interrupt is enabled.

The WATCLK[2: 0] bit can be used to select a wakeup timer clock, including ERTC_CLK/16, ERTC_CLK/8, ERTC_CLK/4, ERTC_CLK/2 and ck_b (usually 1Hz). The cooperation between wakeup timer clocks and wakeup counter values enable users to adjust the wakeup period freely.

To enable a periodic automatic wakeup, the following steps should be respected:

1. Disable a periodic automatic wakeup by setting WATEN=0
2. Wait until WATWF=1 to enable write access to the wakeup reload timer and WATCLK[2: 0]
3. Configure the wakeup timer counter value and wakeup timer through VAL[15: 0] and WATCLK[2: 0] bits
4. Enable a timer by setting WATEN=1

Note:

A wakeup timer is not affected by a system reset and low power consumption modes (Sleep, Deepsleep and Standby modes)

In debug mode, if the ERTC)CLK is selected as a wakeup clock, the counter for periodic wakeup will work normally.

17.3.4 ERTC calibration

Two calibration methods are available: coarse and fine calibration. But the two calibration methods cannot be used together.

Coarse digital calibration:

Coarse digital calibration can be used to advance or delay the calendar updates by increasing or decreasing ck_a cycles.

When positive calibration is enabled (CALDIR=0), 2 ck_a cycles are added every minute (around 15360 ck_a cycles) for the first 2xCALVAL minutes of the 64-minute cycle. This causes the calendar to be updated sooner.

When negative calibration is enabled (CALDIR=1), 1 ck_a cycle is ignored every minute (around ck_a cycles) for the first 2xCALVAL minutes of the 64-minute cycle. This causes the calendar to be updated later.

Note: Coarse digital calibration can work correctly only when the DIVA is 6 or above.

Smooth digital calibration:

Smooth digital calibration has a higher and well-distributed performance than the coarse digital calibration. The calibration is performed by increasing or decreasing ERTC_CLK in an evenly manner.

The smooth digital calibration period is around 2^{20} ERTC_CLK (32 seconds) when the ERTC_CLK is 32.768 kHz. The DEC[8: 0] bit specifies the number of pulses to be masked during the 2^{20} ERTC_CLK cycles. A maximum of 511 pulses can be removed. When the ADD is set, 512 pulses can be inserted during the 2^{20} ERTC_CLK cycles. When DEC[8: 0] and ADD are sued together, a deviation ranging from -511 to +512 ERTC_CLK cycles can be added during the 2^{20} ERTC_CLK cycles.

The effective calibrated frequency (F_{SCAL}):

$$F_{SCAL} = F_{ERTC\_CLK} \times [\,1 + \frac{ADD \times 512 - DEC}{2^{20} + DEC - ADD \times 512}\,]$$

When the divider A is less than 3, the calibration operates as if ADD was equal to 0. The divider B value should be reduced so that each second is accelerated by 8 ERTC_CLK cycles, which means that 256 ERTC_CLK cycles are added every 32 seconds. When DEC[8: 0] and ADD are sued together, a deviation ranging from -255 to +256 ERTC_CLK cycles can be added during the 2^{20} ERTC_CLK cycles.

At this point, the effective calibrated frequency (F_{SCAL})

$$F_{SCAL} = F_{ERTC\_CLK} \times [\,1 + \frac{256 - DEC}{2^{20} + DEC - 256}\,]$$

It is also possible to select 8 or 16-second digital calibration period through the CAL8 and CAL16 bits. The 8-second period takes priority over 16-second. In other words, when both 8-second and 16-second are enabled, 8-second calibration period prevails.

The CALUPDF flag in the ERTC indicates the calibration status. During the configuration of ERTC_SCAL registers, the CALUPDF bit is set, indicating that the calibration value is being updated; Once the calibration value is successfully applied, this bit is cleared automatically, indicating the completion of the calibration value update.

17.3.5 Reference clock detection

The calendar update can be synchronized (not used in low-power modes) to a reference clock (usually the mains 50 or 60 Hz) with a higher precision. This reference clock is used to calibrate the precision of the calendar update frequency (1 Hz)

When it is enabled, the reference clock edge detection is performed during the first 7 ck_a periods around each of the calendar updates. When detected, the edge is used to update calendar values, and 3 ck_a periods are used for subsequent calendar updates. Each time the reference clock edge is detected, the divider A value is forced to reload, making the reference clock and the 1 Hz clock are aligned. If the 1 Hz clock has a slight shift, a more accurate reference clock can be used to fine-tune the 1 Hz clock so that it is aligned with the reference clock. If no reference clock edge is detected, the calendar is updated based on ERTC's original clock source.

Note: Once the reference clock detection is enabled, the DIVA and DIVB must be kept at its respective reset value (0x7F and 0xFF respectively). The clock synchronization cannot be used in conjunction with the coarse digital calibration.

17.3.6 Time stamp

When time stamp event is detected on the tamper pin (valid edge is detected), the current calendar value will be stored to the time stamp register.

When a time stamp event occurs, the time stamp flag bit (TSF) in the ERTC_STS register will be set. If a new time stamp event is detected when time stamp flag (TSF) is already set, then the time stamp overflow flag (TSOF) will be set, but the time stamp registers will remain the result of the last event. By setting the TSIEN bit, an interrupt can be generated when a time stamp event occurs.

Usage of time stamp:

1. How to enable time stamp when a valid edge is detected on a tamper pin
- Select a time stamp in by setting the TSPIN bit
- Select a rising edge or falling edge to trigger time stamp by setting the TSEDG bit
- Enable time stamp by setting TSEN=1
2. How to save time stamp on a tamper event
- Configure tamper detection registers
- Enable tamper detection time stamp by setting TPTSEN=1

Note: The TSF bit will be set after two ck_a cycles following a time stamp event. It is suggested that users poll TSOF bit when the TSF is set.

17.3.7 Tamper detection

The ERTC has one tamper detection mode: TAMP1. It can be configured as a level detection with filter or edge detection. TAMP1 can select either ERTC_MUX.

The TP1F will be set when a valid tamper event is detected. An interrupt will also be generated if a tamper detection interrupt is enabled. If the TPTSEN bit is already set, a time stamp event will be generated accordingly. Once a tamper event occurs, the battery powered registers will be reset so as to ensure data security in the battery powered domain.

How to configure edge detection

1. Select edge detection by setting TPFLT=00, and select a valid edge (TP1EDG)
2. According to your needs, configure whether to activate a time stamp on a tamer event (TPTSEN=1)
3. According to your needs, enable a tamper detection interrupt (TPIEN=1)
4. Enable TAMP1 (setting TP1EN=1)

How to configure level detection with filter

1. Select level detection with filter, and valid level sampling times (TPFLT≠00)
2. Select tamper detection valid level (TP1EDG)
3. Select tamper detection sampling frequency (through the TPFREQ bit)
4. According to your needs, enable tamper detection pull-up (setting TPPU=1). When TPPU=1 is asserted, tamper detection pre-charge time must be configured through the TPPR bit
5. According to your needs, configure whether to activate a time stamp on a tamper event (TPTSEN=1)
6. According to your needs, enable a tamper interrupt (TPIEN=1)
7. Enable TMAP1 by setting TP1EN=1.

In the case of edge detection mode, the following two points deserve our attention:

1. If a rising edge is configured to enable tamper detection, and the tamper detection pin turns to high level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled;
2. If a falling edge is configured to enable tamper detection, and the tamper detection pin turns to low level before tamper detection is enabled, then a tamper event will be detected right after the tamper detection is enabled.

Note: The TSF bit will be set after two ck_a cycles following a time stamp event. It is suggested that users poll TSOF bit when the TSF is set.

17.3.8 Multiplexed function output

ERTC provides a set of multiplexed function output for the following events:

1. Clocks calibrated (OUTSEL=0 and CALOEN=1)
- Output 512Hz (CALOSEL=0)
- Output 1Hz (CALOSEL=1)
2. Alarm clock A (OUTSEL=1)
3. Alarm clock B (OUTSEL=2)
4. Wakeup events (OUTSEL=3)

When alarm clock or wakeup events are selected (OUTSEL≠0), it is possible to select output type (open-drain or push-pull) with the OUTTYPE bit, and output polarity with the OUTP bit.

17.3.9 ERTC wakeup

ERTC can be woken up by alarm clocks, periodic auto wakeup, time stamps or tamper events. To enable an ERTC interrupt, configure as follows:

1. Configure the EXINT line corresponding to ERTC interrupts as an interrupt mode and enable it, and select a rising edge
2. Enable a NVIC channel corresponding to ERTC interrupts
3. Enable an ERTC interrupt

Table 17-2 lists the ERTC clock sources, events and interrupts that are able to wakeup low-power modes.

Table 17-2 ERTC low-power mode wakeup

Clock sources	Events	Wake up Sleep	Wake up Deepsleep	Wakeup Standby
HEXT	Alarm clock A	√	×	×
	Alarm clock B	√	×	×
	Periodic automatic wakeup	√	×	×
	Time stamp	√	×	×
	Tamper event	√	×	×
LICK	Alarm clock A	√	√	√
	Alarm clock B	√	√	√
	Periodic automatic wakeup	√	√	√
	Time stamp	√	√	√
	Tamper event	√	√	√
LEXT	Alarm clock A	√	√	√
	Alarm clock B	√	√	√
	Periodic automatic wakeup	√	√	√
	Time stamp	√	√	√
	Tamper event	√	√	√

Table 17-3 Interrupt control bits

Interrupt events	Event flag	Interrupt enable bit	EXINT line
Alarm clock A	ALAF	ALAIEN	17
Alarm clock B	ALBF	ALBIEN	17
Periodic automatic wakeup	WATF	WATIEN	22
Time stamp	TSF	TSIEN	21
Tamper event	TP1F/TP2F	TPIEN	21

17.4 ERTC registers

These peripheral registers must be accessed by words (32 bits).
ERTC registers are 16-bit addressable registers.

Table 17-4 ERTC register map and reset values

Register name	Offset	Reset value
ERTC_TIME	0x00	0x0000 0000
ERTC_DATE	0x04	0x0000 2101
ERTC_CTRL	0x08	0x0000 0000
ERTC_STS	0x0C	0x0000 0007
ERTC_DIV	0x10	0x007F 00FF
ERTC_WAT	0x14	0x0000 FFFF
ERTC_CCAL	0x18	0x0000 0000

ERTC_ALA	0x1C	0x0000 0000
ERTC_ALB	0x20	0x0000 0000
ERTC_WP	0x24	0x0000 0000
ERTC_SBS	0x28	0x0000 0000
ERTC_TADJ	0x2C	0x0000 0000
ERTC_TSTM	0x30	0x0000 0000
ERTC_TSDT	0x34	0x0000 000D
ERTC_TSSBS	0x38	0x0000 0000
ERTC_SCAL	0x3C	0x0000 0000
ERTC_TAMP	0x40	0x0000 0000
ERTC_ALASBS	0x44	0x0000 0000
ERTC_ALBSBS	0x48	0x0000 0000
ERTC_BPRx	0x50-0x9C	0x0000 0000

17.4.1 ERTC time register (ERTC_TIME)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM Note: This bit is applicable for 12-hr format only. It is 0 for 24-hr format instead.
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.2 ERTC date register (ERTC_DATE)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 20	YT	0x0	rw	Year tens
Bit 19: 16	YU	0x0	rw	Year units
Bit 15: 13	WK	0x1	rw	Week day 0: Forbidden 1: Monday 2: Tuesday 3: Wednesday 4: Thursday 5: Friday 6: Saturday 7: Sunday
Bit 12	MT	0x0	rw	Month tens
Bit 11: 8	MU	0x1	rw	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value.
Bit 5: 4	DT	0x0	rw	Date tens

Bit 3: 0	DU	0x1	rw	Date units

17.4.3 ERTC control register (ERTC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	CALOEN	0x0	rw	Calibration output enable 0: Calibration output disabled 1: Calibration output enabled
Bit 22: 21	OUTSEL	0x0	rw	Output source selection 00: Output source disabled 01: Alarm clock A 10: Alarm clock B 11: Wakeup events
Bit 20	OUTP	0x0	rw	Output polarity 0: High 1: Low
Bit 19	CALOSEL	0x0	rw	Calibration output selection 0: 512Hz 1: 1Hz
Bit 18	BPR	0x0	rw	Battery powered domain data register This bit in the battery powered domain is not affected by a system reset. It is used to store the daylight saving time change or others that need to be saved permanently.
Bit 17	DEC1H	0x0	wo	Decrease 1 hour 0: No effect 1: Subtract 1 hour Note: This bit is applicable only when the current hour is not 0. The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 16	ADD1H	0x0	wo	Add 1 hour 0: No effect 1: Add 1 hour Note: The next second takes effect when this bit is set (don't set this bit when the hour is being incremented)
Bit 15	TSIEN	0x0	rw	Timestamp interrupt enable 0: Timestamp interrupt disabled 1: Timestamp interrupt enabled
Bit 14	WATIEN	0x0	rw	Wakeup timer interrupt enable 0: Wakeup timer interrupt disable 1: Wakeup timer interrupt enabled
Bit 13	ALBIEN	0x0	rw	Alarm B interrupt enable 0: Alarm B interrupt disabled 1: Alarm B interrupt enabled
Bit 12	ALAIEN	0x0	rw	Alarm A interrupt enable 0: Alarm A interrupt disabled 1: Alarm A interrupt enabled
Bit 11	TSEN	0x0	rw	Timestamp enable 0: Timestamp disabled 1: Timestamp enabled
Bit 10	WATEN	0x0	rw	Wakeup timer enable 0: Wakeup timer disabled 1: Wakeup timer enabled
Bit 9	ALBEN	0x0	rw	Alarm B enable 0: Alarm B disabled 1: Alarm B enabled
Bit 8	ALAEN	0x0	rw	Alarm A enable 0: Alarm A disabled 1: Alarm A enabled
Bit 7	CCALEN	0x0	rw	Coarse calibration enable 0: Coarse calibration disabled 1: Coarse calibration enabled
Bit 6	HM	0x0	rw	Hour mode 0: 24-hour format 1: 12-hour format
Bit 5	DREN	0x0	rw	Date/time register direct read enable

				0: Date/time register direct read disabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the synchronized registers, which are updated once every two ERTC_CLK cycles 1: Date/time register direct read enabled. ERTC_TIME, ERTC_DATE and ERTC_SBS values are taken from the battery powered domain.
Bit 4	RCDEN	0x0	rw	Reference clock detection enable 0: Reference clock detection disabled 1: Reference clock detection enabled
Bit 3	TSEDG	0x0	rw	Timestamp trigger edge 0: Rising edge 1: Falling edge
Bit 2: 0	WATCLK	0x0	rw	Wakeup timer clock selection 000: ERTC_CLK/16 001: ERTC_CLK/8 010: ERTC_CLK/4 011: ERTC_CLK/2 10x: ck_b 11x: ck_b is selected. 2^{16} is added to the wakeup counter value, and wakeup time =ERTC_WAT+2^{16}. *Note: The write access to this field is supported when WATEN=0 and WATWF=1.*

17.4.4 ERTC initialization and status register (ERTC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CALUPDF	0x0	ro	Calibration value update complete flag 0: Calibration value update is complete 1: Calibration value update is in progress This bit is automatically set when software writes to the ERTC_SCAL register. It is automatically cleared when a new calibration value is taking into account. When this bit is set, the write access to the ERTC_SCAL register is not allowed.
Bit 15: 14	Reserved	0x0	resd	Kept at its default value.
Bit 13	TP1F	0x0	rw0c	Tamper detection 1 flag 0: No tamper event 1: Tamper event occurs
Bit 12	TSOF	0x0	rw0c	Timestamp overflow flag 0: No timestamp overflow 1: Timestamp overflow occurs If a new time stamp event is detected when time stamp flag (TSF) is already set, this bit will be set by hardware.
Bit 11	TSF	0x0	rw0c	Timestamp flag 0: No timestamp event 1: Timestamp event occurs It is recommended to double check the TSOF flag after reading a timestamp and clearing the TSF. Otherwise, a new timestamp event may be detected while clearing the TSF. *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 10	WATF	0x0	rw0c	Wakeup timer flag 0: No wakeup timer event 1: Wakeup timer event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 9	ALBF	0x0	rw0c	Alarm clock B flag 0: No alarm clock event 1: Alarm clock event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 8	ALAF	0x0	rw0c	Alarm clock A flag

				0: No alarm clock event 1: Alarm clock event occurs *Note: The clearing operation of this bit takes effect after two APB_CLK cycles.*
Bit 7	IMEN	0x0	rw	Initialization mode enable 0: Initialization mode disabled 1: Initialization mode enabled When an initialization mode is entered, the calendar stops running.
Bit 6	IMF	0x0	ro	Enter initialization mode flag 0: Initialization mode is not entered 1: Initialization mode is entered The ERTC_TIME, ERTC_DATE and ERTC_DIV registers can be modified only when an initialization mode is enabled (INITEN=1) and entered (INITEF=1).
Bit 5	UPDF	0x0	rw0c	Calendar update flag 0: Calendar update is in progress 1: Calendar update is complete The UPDF bit is set each time the shadow register is synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK.
Bit 4	INITF	0x0	ro	Calendar initialization flag 0: Calendar has not been initialized 1: Calendar has been initialized This bit is set when the calendar year filed (ERTC_DATE) is different from 0. It is cleared when the year is 0.
Bit 3	TADJF	0x0	ro	Time adjustment flag 0: No time adjustment 1: Time adjustment is in progress This bit is automatically set when a write access to the ERTC_TADJ register is performed. It is automatically cleared at the end of time adjustment.
Bit 2	WATWF	0x1	ro	Wakeup timer register allows write flag 0: Wakeup timer register configuration not allowed 1: Wakeup timer register configuration allowed
Bit 1	ALBWF	0x1	ro	Alarm B register allows write flag 0: Alarm B register write operation not allowed 1: Alarm B register write operation allowed
Bit 0	ALAWF	0x1	ro	Alarm A register allows write flag 0: Alarm A register write operation not allowed 1: Alarm A register write operation allowed

17.4.5 ERTC divider register (ERTC_DIV)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22: 16	DIVA	0x7F	rw	Divider A
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14: 0	DIVB	0x00FF	rw	Divider B Calendar clock = ERTC_CLK/((DIVA+1)x(DIVB+1))

17.4.6 ERTC wakeup timer register (ERTC_WAT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	VAL	0xFFFF	rw	Wakeup timer reload value

17.4.7 ERTC coarse calibration register (ERTC_CCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	CALDIR	0x0	rw	Calibration direction 0: Positive calibration 1: Negative calibration
Bit 6: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4: 0	CALVAL	0x00	rw	Calibration value Positive calibration 00000: +0 ppm 00001: +4 ppm 00010: +8 ppm 11111: +126 ppm Negative calibration 00000: -0 ppm 00001: -2 ppm 00010: -4 ppm … 11111: - 63 ppm

17.4.8 ERTC alarm clock A register (ERTC_ALA)

Bit	Register	Reset value	Type	Description
Bit 31	MASK4	0x0	rw	Date/week day mask 0: Date/week day is not masked 1: Alarm clock doesn't care about date/week day
Bit 30	WKSEL	0x0	rw	Date/week day select 0: Date 1: Week day (DT[1: 0] is not used)
Bit 29: 28	DT	0x0	rw	Date tens
Bit 27: 24	DU	0x0	rw	Date/week day units
Bit 23	MASK3	0x0	rw	Hour mask 0: No hour mask 1: Alarm clock doesn't care about hours
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	MASK2	0x0	rw	Minute mask 0: No minute mask 1: Alarm clock doesn't care about minutes
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	MASK1	0x0	rw	Second mask 0: No second mask 1: Alarm clock doesn't care about seconds
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.9 ERTC alarm clock B register (ERTC_ALB)

Bit	Register	Reset value	Type	Description
Bit 31	MASK4	0x0	rw	Date/week day mask 0: Date/week day is not masked 1: Alarm clock doesn't care about date/week day
Bit 30	WKSEL	0x0	rw	Date/week day select 0: Date 1: Week day (DT[1: 0] is not used)
Bit 29: 28	DT	0x0	rw	Date tens
Bit 27: 24	DU	0x0	rw	Date/week day units
Bit 23	MASK3	0x0	rw	Hour mask 0: No hour mask 1: Alarm clock doesn't care about hours
Bit 22	AMPM	0x0	rw	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	rw	Hour tens
Bit 19: 16	HU	0x0	rw	Hour units
Bit 15	MASK2	0x0	rw	Minute mask 0: No minute mask 1: Alarm clock doesn't care about minutes
Bit 14: 12	MT	0x0	rw	Minute tens
Bit 11: 8	MU	0x0	rw	Minute units
Bit 7	MASK1	0x0	rw	Second mask 0: No second mask 1: Alarm clock doesn't care about seconds
Bit 6: 4	ST	0x0	rw	Second tens
Bit 3: 0	SU	0x0	rw	Second units

17.4.10 ERTC write protection register (ERTC_WP)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value
Bit 7: 0	CMD	0x00	wo	Command register All ERTC register write protection is unlocked by writing 0xCA and 0x53. Writing any other value will re-activate write protection.

17.4.11 ERTC subsecond register (ERTC_SBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value Subsecond is the value in the DIVB counter. Clock frequency = ERTC_CLK/(DIVA+1)

17.4.12 ERTC time adjustment register (ERTC_TADJ)

Bit	Register	Reset value	Type	Description
Bit 31	ADD1S	0x0	wo	Add 1 second 0: No effect 1: Add one second This bit can be written only when TADJF=0. It is intended to be used with DECSBS in order to fine-tune the time.
Bit 30: 15	Reserved	0x0000	resd	Kept at its default value
Bit 14: 0	DECSBS	0x0000	wo	DECSBS[14: 0]: Decrease sub-second value Delay (ADD1S=0): Delay = DECSBS/(DIVB+1) Advance (ADD1S=1) : Advance =1-(DECSBS/(DIVB+1))

17.4.13 ERTC time stamp time register (ERTC_TSTM)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value
Bit 22	AMPM	0x0	ro	AM/PM 0: AM 1: PM *Note: This bit is applicable for 12-hour format only. It is 0 for 24-hour format.*
Bit 21: 20	HT	0x0	ro	Hour tens
Bit 19: 16	HU	0x0	ro	Hour units
Bit 15	Reserved	0x0	resd	Kept at its default value
Bit 14: 12	MT	0x0	ro	Minute tens
Bit 11 : 8	MU	0x0	ro	Minute units
Bit 7	Reserved	0x0	resd	Kept at its default value
Bit 6: 4	ST	0x0	ro	Second tens
Bit 3: 0	SU	0x0	ro	Second units

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.14 ERTC time stamp date register (ERTC_TSDT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 13	WK	0x0	ro	Week day
Bit 12	MT	0x0	ro	Month tens
Bit 11: 8	MU	0x0	ro	Month units
Bit 7: 6	Reserved	0x0	resd	Kept at its default value
Bit 5: 4	DT	0x0	ro	Date tens
Bit 3: 0	DU	0x0	ro	Date units

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.15 ERTC time stamp subsecond register (ERTC_TSSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	SBS	0x0000	ro	Sub-second value

Note: The content of this register is valid only when the TSF is set in the ERTC_STS register. It is cleared when TSF bit is reset.

17.4.16 ERTC smooth calibration register (ERTC_SCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15	ADD	0x0	rw	Add ERTC clock 0: No ERTC clock added 1: One ERTC_CLK is inserted every 211 ERTC_CLK cycles
Bit 14	CAL8	0x0	rw	8-second calibration period 0: No effect 1: 8-second calibration
Bit 13	CAL16	0x0	rw	16 second calibration period 0: No effect 1: 16-second calibration
Bit 12: 9	Reserved	0x0	resd	Kept at its default value
Bit 8: 0	DEC	0x000	rw	Decrease ERTC clock DEC out of ERTC_CLK cycles are masked during the 2^{20}

				ERTC_CLK periods. This bit is usually used with ADD. When the ADD is set, the actual number of ERTC_CLK is equal to 2^{20}+512-DEC during the 2^{20} ERTC_CLK periods.

17.4.17 ERTC tamper configuration register (ERTC_TAMP)

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value
Bit 18	OUTTYPE	0x0	rw	Output type 0: Open-drain output 1: Push-pull output
Bit 17: 16	Reserved	0x0	resd	Kept at its default value
Bit 15	TPPU	0x0	rw	Tamper detection pull-up 0: Tamper detection pull-up enabled 1: Tamper detection pull-up disabled
Bit 14: 13	TPPR	0x0	rw	Tamper detection pre-charge time 0: 1 ERTC_CLK cycle 1: 2 ERTC_CLK cycles 2: 4 ERTC_CLK cycles 3: 8 ERTC_CLK cycles
Bit 12: 11	TPFLT	0x0	rw	Tamper detection filter time 0: No filter 1: Tamper is detected after 2 consecutive samples 2: Tamper is detected after 4 consecutive samples 3: Tamper is detected after 8 consecutive samples
Bit 10: 8	TPFREQ	0x0	rw	Tamper detection frequency 0: ERTC_CLK/32768 1: ERTC_CLK/16384 2: ERTC_CLK/8192 3: ERTC_CLK/4096 4: ERTC_CLK/2048 5: ERTC_CLK/1024 6: ERTC_CLK/512 7: ERTC_CLK/256
Bit 7	TPTSEN	0x0	rw	Tamper detection timestamp enable 0: Tamper detection timestamp disabled 1: Tamper detection timestamp enabled. Save timestamp on a tamper event.
Bit 6: 3	Reserved	0x0	resd	Kept at its default value
Bit 2	TPIEN	0x0	rw	Tamper detection interrupt enable 0: Tamper detection interrupt disabled 1: Tamper detection interrupt enabled
Bit 1	TP1EDG	0x0	rw	Tamper detection 1 valid edge If TPFLT=0: 0: Rising edge 1: Falling edge If TPFLT>0: 0: Low 1: High
Bit 0	TP1EN	0x0	rw	Tamper detection 1 enable 0: Tamper detection 1 disabled 1: Tamper detection 1 enabled

17.4.18 ERTC alarm clock A subsecond register (ERTC_ALASBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
Bit 27: 24	SBSMSK	0x0	rw	Sub-second mask 0: No comparison. Alarm A doesn't care about subseconds. 1: SBS[0] is compared 2: SBS[1: 0] are compared 3: SBS[2: 0] are compared ... 14: SBS[13: 0] are compared 15: SBS[14: 0] are compared
Bit 23: 15	Reserved	0x000	rw	Kept at its default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

17.4.19 ERTC alarm clock B subsecond register (ERTC_ALBSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at its default value
Bit 27: 24	SBSMSK	0x0	rw	Sub-second mask 0: No comparison. Alarm B doesn't care about subseconds. 1: SBS[0] is compared 2: SBS[1: 0] are compared 3: SBS[2: 0] are compared ... 14: SBS[13: 0] are compared 15: SBS[14: 0] are compared
Bit 23: 15	Reserved	0x000	rw	Kept at its default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

17.4.20 ERTC battery powered domain data register (ERTC_BPRx)

Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Battery powered domain data BPR_DTx registers are powered on by V_{BAT} so that they are not reset by a system reset. They are reset on a tamper event or when a battery powered domain is reset.

18 Analog-to-digital converter (ADC)

18.1 ADC introduction

The ADC is a peripheral that converts an analog input signal into a 12-bit digital signal. Its sampling rate is as high as 2 MSPS. It has up to 18 channels for sampling and conversion.

18.2 ADC main features

In terms of analog:

- 12-bit configurable resolution
- Self-calibration time: 172 ADC clock cycles
- ADC conversion time
- ADC conversion time is 0.5 μs at 28 MHz
- ADC power supply: 2.6 V ~ 3.6 V
- ADC input range: $V_{REF-} \leqslant V_{IN} \leqslant V_{REF+}$

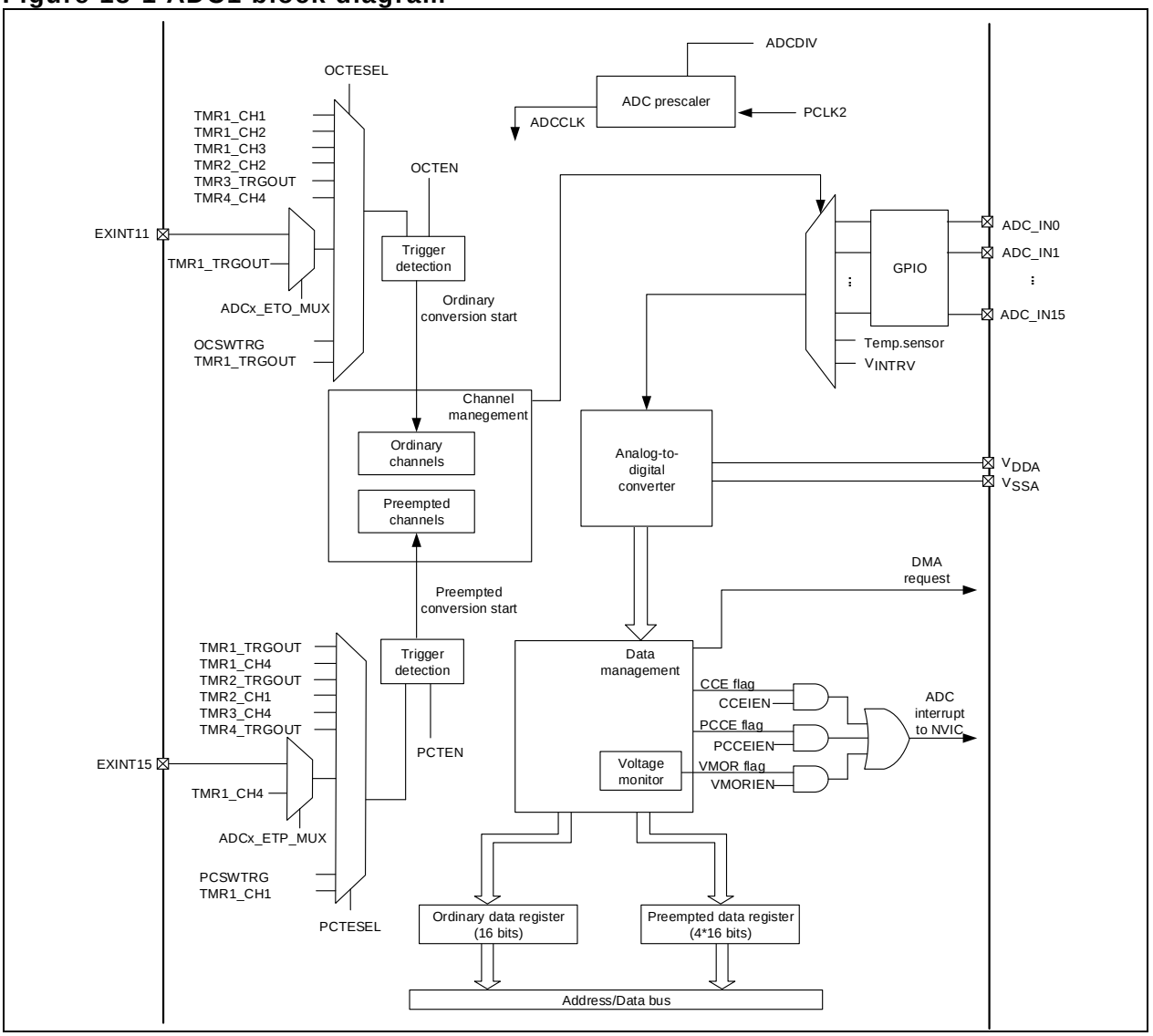
In terms of digital control:

- Regular channels and preempted channels with different priority
- Regular channels and preempted channels both have their own trigger detection circuit
- Each channel can independently define its own sampling time
- Conversion sequence supports various conversion modes
- Optional data alignment mode
- Programmable voltage monitor threshold
- Regular channels with DMA transfers
- Interrupt generation at one of the following events:
  - End of the conversion of preempted channels
  - End of the conversion of regular channels
  - Voltage outside the threshold programmed

18.3 ADC structure

Figure 18-1 shows the block diagram of ADC.

Figure 18-1 ADC1 block diagram



Input pin description:

- V_{DDA}: Analog supply, ADC analog supply
- V_{SSA}: Analog supply ground, ADC analog supply ground
- ADCx_IN: Analog input signal channels

18.4 ADC functional overview

18.4.1 Channel management

Analog signal channel input:

There are 18 analog signal channel inputs for each of the ADCs, expressed by ADC_Inx (x=0 to 17).

- ADC1_IN0 to ADC1_IN15 are referred to as the external analog input, ADC1_IN16 as an internal temperature sensor, ADC1_IN17 as an internal reference voltage;

Channel conversion

The conversions are divided into two groups: ordinary and preempted channels. The preempted group has priority over the ordinary group.

If the preempted channel trigger occurs during the ordinary channel conversion, then the ordinary channel conversion is interrupted, giving the priority to the preempted channel, and the ordinary channel continues its conversion at the end of the preempted channel conversion. If the ordinary channel trigger occurs during the preempted channel conversion, the ordinary channel conversion won't start until the end of the preempted channel conversion.

Program the ADC_Inx into the ordinary channel sequence (ADC_OSQx) and the preempted channel

sequence (ADC_PSQ), and the same channel can be repeated, the total number of sequences is determined by OCLEN and PCLEN, then it is ready to enable the ordinary channel or preempted channel conversion.

18.4.1.1 Internal temperature sensor

The temperature sensor is connected to ADC1_IN16. Before the temperature sensor channel conversion, it is required to enable the ITSRVEN bit in the ADC_CTRL2 register and wait after power-on time.

The converted data of such channel, along with the voltage value at 25°C and Avg_Slope ,can be used to calculate the temperature.

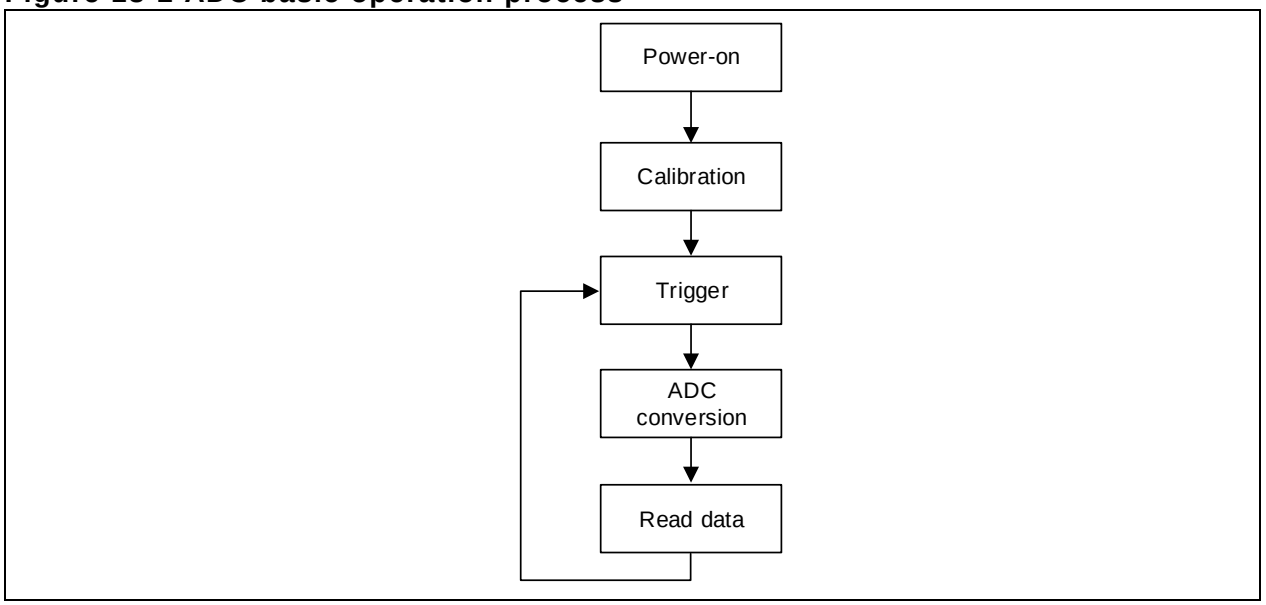
18.4.1.2 Internal reference voltage

The internal reference voltage of the typical value 1.2 V is connected to ADC1_IN17. It is required to enable the ITSRVEN bit in the ADC_CTRL2 register before the internal reference channel conversion. The converted data of such channel can be used to calculate the external reference voltage.

18.4.2 ADC operation process

Figure 18-2 shows the basic operation process of the ADC. It is recommended to do the calibration after the initial power-on in order to improve the accuracy of sampling and conversion. After the calibration, trigger is used to enable ADC sampling and conversion. Read data at the end of the conversion.

Figure 18-2 ADC basic operation process



18.4.2.1 Power-on and calibration

Power-on

Set the ADC1EN bit in the CRM_APB2EN register to enable ADC clocks: PCLK2 and ADCCLK.

Program the desired ADCCLK frequency by setting the ADCDIV bit in the CRM_CFG register. The ADCCLK is derived from PCLK2 frequency division.

Note: ADCCLK must be less than 28 MHz.

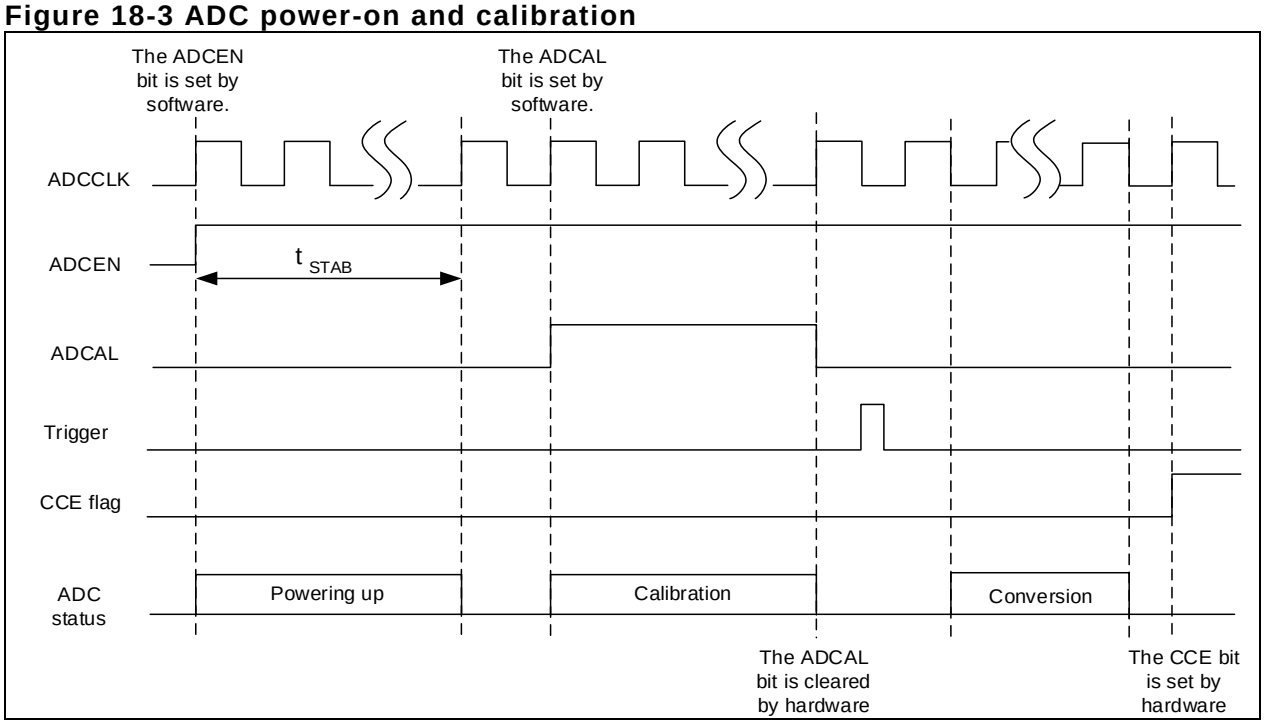
Then set the ADCEN bit in the ADC_CTRL2 register to supply the ADC, and wait until the RDF flab is set before starting ADC conversion. Clear the ADCEN bit will stop the ADC conversion and result in a reset. In the meantime, the ADC is switched off to save power.

Calibration

After power-on, the calibration is enabled by setting the ADCAL bit in the ADC_CTRL2 register. When the calibration is complete, the ADCAL bit is cleared by hardware and the conversion is performed by software trigger.

After each calibration, the calibration value is stored in ADC_ODT register, and then value is automatically sent back to the ADC so as to eliminate capacitance errors. The storage of the calibration value will not set the OCCE flag, or generate interrupts or DMA requests.

Figure 18-3 ADC power-on and calibration



18.4.2.2 Trigger

The ADC triggers contain ordinary channel trigger and preempted channel trigger. The ordinary channel conversion is triggered by ordinary channel triggers while the preempted channel conversion is triggered by preempted ones. After the OCTEN or PCTEN bit is set in the ADC_CTRL2 register, the ADC starts conversion after a trigger source is detected.

The conversion can be triggered by software write operation to the OCSWTRG and PCSWTRG bits in the ADC_CTRL2 register, or by an external event. The external events include timer and pin triggers. The OCTESEL and PCTESEL bits in the ADC_CTRL2 register are used to select specific trigger sources, as shown in *Table 18-1*. Besides, the ordinary channel has a special trigger source, which repeatedly enables ADCEN to trigger conversion, without the need of enabling the OCTEN bit in the ADC_CTRL2 register.

Table 18-1 Trigger sources for ADC

OCTESEL		Trigger source	PCTESEL		Trigger source
0000		TMR1_CH1 event	0000		TMR1_TRGOUT event
0001		TMR1_CH2 event	0001		TMR1_CH4 event
0010		TMR1_CH3 event	0010		TMR2_TRGOUT event
0011		TMR2_CH2 event	0011		TMR2_CH1 event
0100		TMR3_TRGOUT event	0100		TMR3_CH4 event
0101		TMR4_CH4 event	0101		TMR4_TRGOUT event
0110	ADC1_ETO_MUX=0	EXINT line11 external pin	0110	ADC1_ETP_MUX=0	EXINT line15 external pin
	ADC1_ETO_MUX=1	TMR1_TRGOUT event		ADC1_ETP_MUX=1	TMR1_CH4 event
0111		OCSWTRG bit	0111		PCSWTRG bit
1000		Reserved	1000		Reserved
1001		Reserved	1001		Reserved
1010		Reserved	1010		Reserved
1011		Reserved	1011		Reserved
1100		Reserved	1100		Reserved
1101		TMR1_TRGOUT event	1101		TMR1_CH1 event
1110		Reserved	1110		Reserved
1111		Reserved	1111		Reserved

18.4.2.3 Sampling and conversion sequence

The sampling period can be configured by setting the CSPTx bit in the ADC_SPT1 and ADC_SPT2 registers. A single one conversion time is calculated with the following formula:

$$\text{A single one conversion time (ADCCLK period)} = \text{sampling time} + \text{resolution bits} + 12.5$$

Example:

If the CSPTx selects 1.5 period, then one conversion needs 1.5+12.5=14 ADCCLK periods

If the CSPTx selects 7.5 period, then one conversion needs 7.5+12.5=20 ADCCLK periods.

18.4.3 Conversion sequence management

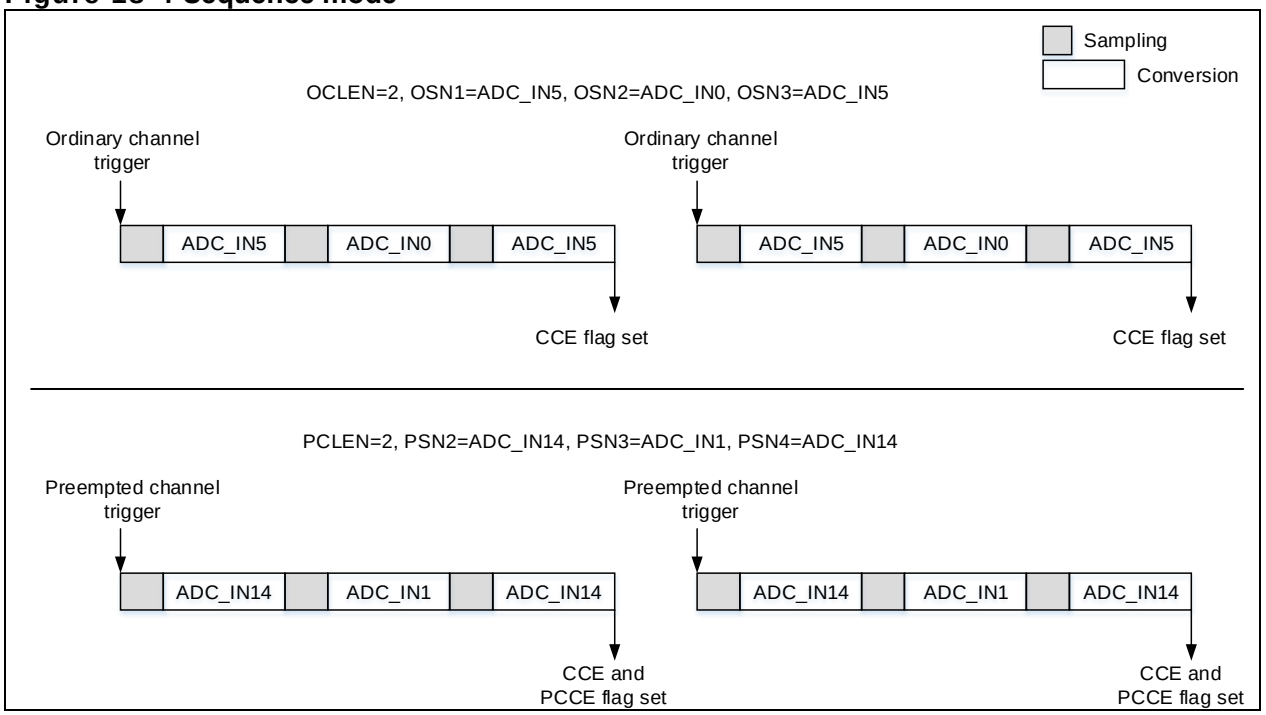
Only one channel is converted at each trigger event by default, that is, OSN1-defined channel or PSN4-defined channel.

The detailed conversion sequence modes are described in the following sections. With this, the channels can be converted in a specific order.

18.4.3.1 Sequence mode

The sequence mode is enabled by setting the SQEN bit in the ADC_CTRL1 register. The ADC_OSQx registers are used to configure the sequence and total number of the ordinary channels while the ADC_PSQ register is used to define the sequence and total number of the preempted channels. When the sequence mode is enabled, a single trigger event enables the conversion of a group of channels in order. The ordinary channels start converting from the QSN1 while the preempted channels starts from the PSNx, where x=4-PCLEN. Figure 18-4 shows an example of the behavior in sequence mode.

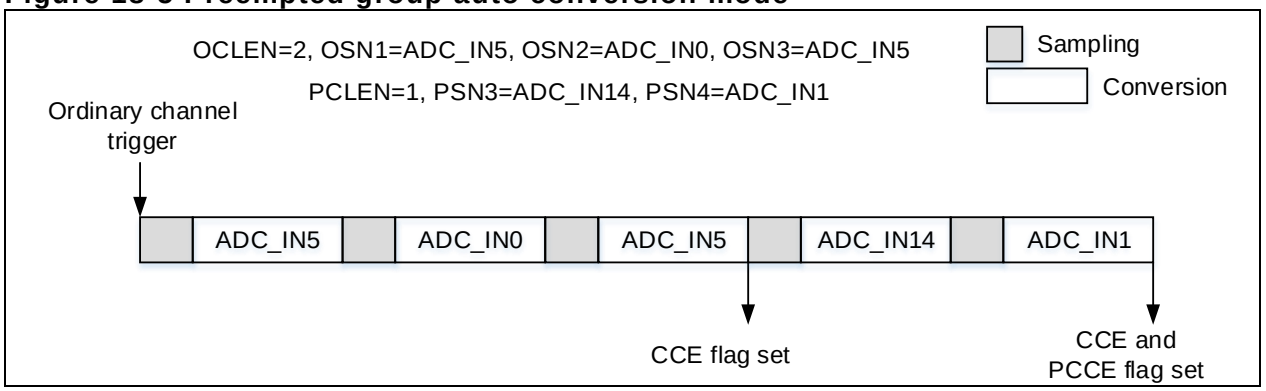
Figure 18-4 Sequence mode



18.4.3.2 Automatic preempted group conversion mode

The automatic preempted group conversion mode is enabled by setting the PCAUTOEN bit in the ADC_CTRL1 register. Once the ordinary channel conversion is over, the preempted group will automatically continue its conversion. This mode can work with the sequence mode. The preempted group conversion starts automatically at the end of the conversion of the ordinary group. *Figure 18-5* shows an example of the behavior when the automatic preempted group conversion mode works with the ordinary group.

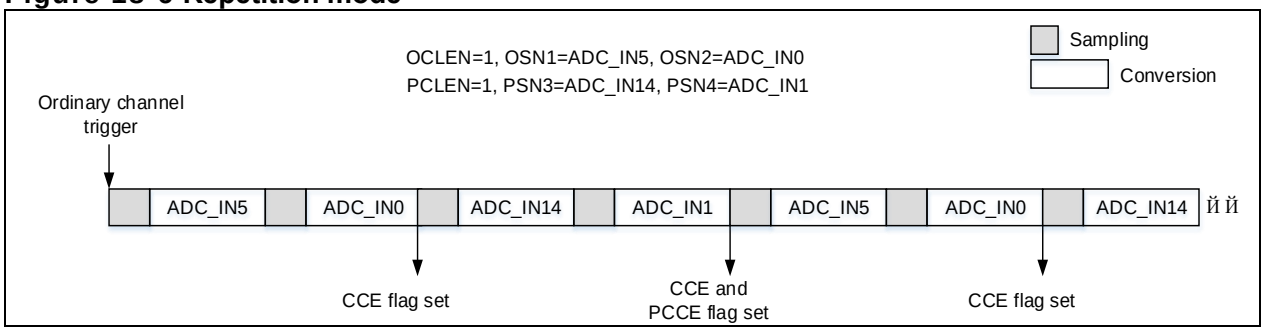
Figure 18-5 Preempted group auto conversion mode



18.4.3.3 Repetition mode

The repetition mode is enabled by setting the RPEN bit in the ADC_CTRL2 register. When a trigger signal is detected, the ordinary channels will be converted repeatedly. This mode can work with the ordinary channel conversion in the sequence mode to enable the repeated conversion of the ordinary group. Such mode can also work with the preempted group auto conversion mode to repeatedly convert the ordinary group and preempted group in sequence. Figure 18-6 shows an example of the behavior when the repetition mode works with the sequence mode and preempted group auto conversion mode.

Figure 18-6 Repetition mode



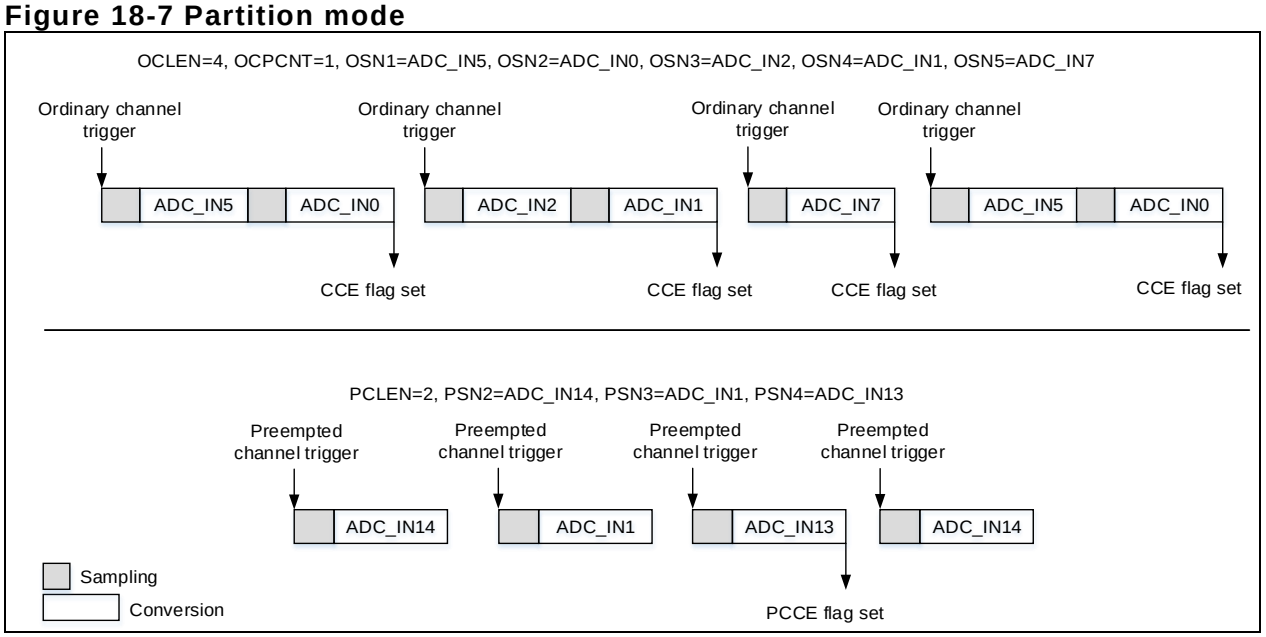
18.4.3.4 Partition mode

The partition mode of the ordinary group can be enabled by setting the OCPEN bit in the ADC_CTRL1 register. In this mode, the ordinary group conversion sequence length (OCLEN bit in the ADC_OSQ1 register) is divided into a smaller sub-group, in which the number of the channels is programmed with the OCPCNT bit in the ADC_CTRL1 register. A single trigger event will enable the conversion of all the channels in the sub-group. Each trigger event selects different sub-group in order.

Set the PCPEN bit in the ADC_CTRL1 register will enable the partition mode of the preempted group. In this mode, the preempted group conversion sequence length (OCLEN bit in the ADC_OSQ1 register) is divided into a sub-group with only one channel. A single one trigger event will enable the conversion of all the channels in the sub-group. Each trigger event selects different sub-group in order.

The partition mode cannot be used with the repetition mode at the same time. *Figure 18-7* shows an example of the behavior in partition mode for ordinary group and preempted group.

Figure 18-7 Partition mode



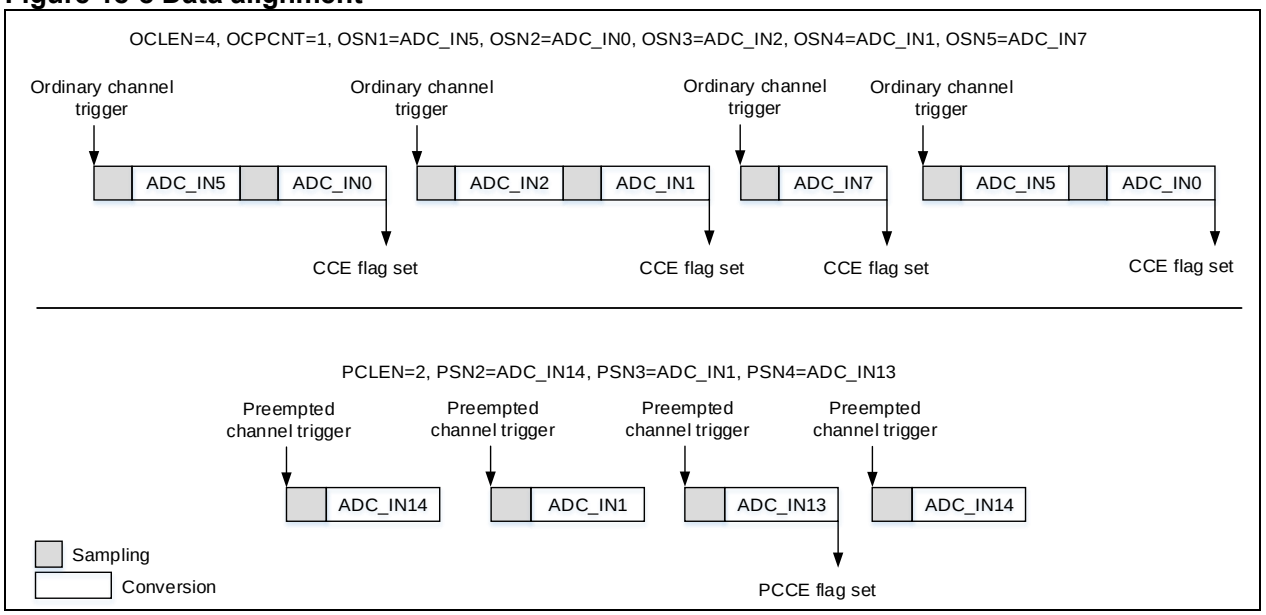
18.4.4 Data management

At the end of the conversion of the ordinary group, the converted value is stored in the ADC_ODT register. Once the preempted group conversion ends, the converted data of the preempted group is stored in the ADC_PDTx register.

18.4.4.1 Data alignment

DTALIGN bit in the ADC_CTRL2 register selects the alignment of data (right-aligned or left-aligned). Apart from this, the converted data of the preempted group is decreased by the offset written in the ADC_PCDTOx register. Thus the result may be a negative value, marked by SIGN, as shown in *Figure 18-8*.

Figure 18-8 Data alignment



18.4.4.2 Data read

Read access to the ADC_ODT register using CPU or DMA gets the converted data of the ordinary group. Read access to the ADC_PDTx register using CPU gets the converted data of the preempted group.

When the OCDMAEN bit is set in the ADC_CTRL2 register, the ADC will issue DMA requests each time the ADC_OTD register is updated.

18.4.5 Voltage monitoring

The OCVMEN bit or PCVMEN bit in the ADC_CTRL1 register is used to enable voltage monitoring based on the converted data.

The VMOR bit will be set if the converted result is outside the high threshold (ADC_VMHB register) or less than the low threshold (ADC_VMLB register).

The VMSGEN bit in the ADC_CTRL1 register is used to enable voltage monitor on either a single channel or all the channels. The VMCSEL bit is used to select a specific channel that requires voltage monitoring.

Voltage monitoring is based on the comparison result between the original converted data and the 12-bit voltage monitor boundary register, irrespective of the PCDTOx and DTALIGN bits.

18.4.6 Status flag and interrupts

Each of the ADCs has its dedicated ADCx_STS registers, that is, OCCS (ordinary channel conversion start flag), PCCS (preempted channel conversion start flag), PCCE (preempted channel conversion end flag), OCCE (ordinary channel conversion end flag) and VMOR (voltage monitor out of range).

PCCE, CCE and VMOR have their respective interrupt enable bits. Once the interrupt bits are enabled, the corresponding flag is set and an interrupt is sent to CPU.

18.5 ADC registers

Table 18-2 lists ADC register map and their reset values.

These peripheral registers must be accessed by word (32 bits).

Table 18-2 ADC register map and reset values

Register name	Offset	Reset value
ADC_STS	0x000	0x0000 0000
ADC_CTRL1	0x004	0x0000 0000
ADC_CTRL2	0x008	0x0000 0000
ADC_SPT1	0x00C	0x0000 0000
ADC_SPT2	0x010	0x0000 0000
ADC_PCDTO1	0x014	0x0000 0000
ADC_PCDTO2	0x018	0x0000 0000
ADC_PCDTO3	0x01C	0x0000 0000
ADC_PCDTO4	0x020	0x0000 0000
ADC_VMHB	0x024	0x0000 0FFF
ADC_VMLB	0x028	0x0000 0000
ADC_OSQ1	0x02C	0x0000 0000
ADC_OSQ2	0x030	0x0000 0000
ADC_OSQ3	0x034	0x0000 0000
ADC_PSQ	0x038	0x0000 0000
ADC_PDT1	0x03C	0x0000 0000
ADC_PDT2	0x040	0x0000 0000
ADC_PDT3	0x044	0x0000 0000
ADC_PDT4	0x048	0x0000 0000
ADC_ODT	0x04C	0x0000 0000

18.5.1 ADC status register (ADC_STS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	OCCS	0x0	rw0c	Ordinary channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No ordinary channel conversion started 1: Ordinary channel conversion has started
Bit 3	PCCS	0x0	rw0c	Preempted channel conversion start flag This bit is set by hardware and cleared by software (writing 0). 0: No preempted channel conversion started 1: Preempted channel conversion has started
Bit 2	PCCE	0x0	rw0c	Preempted channel end of conversion flag This bit is set by hardware and cleared by software (writing 0). 0: Conversion is not complete 1: Conversion is complete
Bit 1	OCCE	0x0	rw0c	End of conversion flag This bit is set by hardware. It is cleared by software (writing 0) or by reading the ADC_ODT register. 0: Conversion is not complete 1: Conversion is complete Note: This bit is set at the end of the ordinary or preempted group.
Bit 0	VMOR	0x0	rw0c	Voltage monitoring out of range flag This bit is set by hardware and cleared by software (writing 0). 0: Voltage is within the value programmed 1: Voltage is outside the value programmed

18.5.2 ADC control register1 (ADC_CTRL1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23	OCVMEN	0x0	rw	Voltage monitoring enable on ordinary channels 0: Voltage monitoring disabled on ordinary channels 1: Voltage monitoring enabled on ordinary channels
Bit 22	PCVMEN	0x0	rw	Voltage monitoring enable on preempted channels 0: Voltage monitoring disabled on preempted channels 1: Voltage monitoring enabled on preempted channels
Bit 21: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15: 13	OCPCNT	0x0	rw	Partitioned mode conversion count of ordinary channels 000: 1 channel 001: 2 channels …… 111: 8 channels Note: In this mode, the preempted group converts only one channel at each trigger.
Bit 12	PCPEN	0x0	rw	Partitioned mode enable on preempted channels 0: Partitioned mode disabled on preempted channels 1: Partitioned mode enabled on preempted channels
Bit 11	OCPEN	0x0	rw	Partitioned mode enable on ordinary channels This is set and cleared by software to enable or disable

				partitioned mode on ordinary channels. 0: Partitioned mode disabled on ordinary channels 1: Partitioned mode enabled on ordinary channels
Bit 10	PCAUTOEN	0x0	rw	Preempted group automatic conversion enable after ordinary group 0: Preempted group automatic conversion disabled 1: Preempted group automatic conversion enabled
Bit 9	VMSGEN	0x0	rw	Voltage monitoring enable on a single channel 0: Disabled (Voltage monitoring enabled on all channels) 1: Enabled (Voltage monitoring enabled a single channel)
Bit 8	SQEN	0x0	rw	Sequence mode enable 0: Sequence mode disabled (a single channel is converted) 1: Sequence mode enabled (the selected multiple channels are converted) Note: If this mode is enabled and the CCEIEN/PCCEIEN is set, a CCE or PCCE interrupt is generated only at the end of conversion of the last channel.
Bit 7	PCCEIEN	0x0	rw	Conversion end interrupt enable on Preempted channels 0: Conversion end interrupt disabled on Preempted channels 1: Conversion end interrupt enabled on Preempted channels
Bit 6	VMORIEN	0x0	rw	Voltage monitoring out of range interrupt enable 0: Voltage monitoring out of range interrupt disabled 1: Voltage monitoring out of range interrupt enabled
Bit 5	CCEIEN	0x0	rw	Channel conversion end interrupt enable 0: Channel conversion end interrupt disabled 1: Channel conversion end interrupt enabled
Bit 4: 0	VMCSEL	0x00	rw	Voltage monitoring channel select This filed is valid only when the VMSGEN is enabled. 00000: ADC_IN0 channel 00001: ADC_IN1 channel …… 01111: ADC_IN15 channel 10000: ADC_IN16 channel 10001: ADC_IN17 channel 10010~11111: Unused, configuration is not allowed.

18.5.3 ADC control register2 (ADC_CTRL2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 30: 26	Reserved	0x00	resd	Kept at its default value
Bit 23	ITSRVEN	0x0	rw	Internal V_{INTRV} enable 0: Internal V_{INTRV} disabled 1: Internal V_{INTRV} enabled
Bit 22	OCSWTRG	0x0	rw	Conversion of ordinary channels triggered by software 0: Conversion of ordinary channels not triggered 1: Conversion of ordinary channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 21	PCSWTRG	0x0	rw	Conversion of preempted channels triggered by software 0: Conversion of preempted channels not triggered 1: Conversion of preempted channels triggered (This bit is cleared by software or by hardware as soon as the conversion starts)
Bit 20	OCTEN	0x0	rw	Trigger mode enable for ordinary channel conversion 0: Disabled

				1: Enabled
Bit 25 Bit 19: 17	OCTESEL	0x0	rw	Trigger event select for ordinary channels conversion 0000: Timer 1 CH1 event 0001: Timer 1 CH2 event 0010: Timer 1 CH3 event 0011: Timer 2 CH2 event 0100: Timer 3 TRGOUT event 0101: Timer 4 CH4 event 0110: EXINT line 11/ TMR1_TRGOUT event 0111: OCSWTRG 1000~1100: Unused. Configuration is not allowed. 1101: Timer 1 TRGOUT event 1110~1111: Unused.
Bit 16	Reserved	0x0	resd	Kept at its default value
Bit 15	PCTEN	0x0	rw	Trigger mode enable for preempted channels conversion 0: Disabled 1: Enabled
Bit 24 Bit 14: 12	PCTESEL	0x0	rw	Trigger event select for preempted channels conversion For ADC1 and ADC2, the trigger events are configured as follows: 0000: Timer 1 TRGOUT event 0001: Timer 1 CH4 event 0010: Timer 2 TRGOUT event 0011: Timer 2 CH1 event 0100: Timer 3 CH4 event 0101: Timer 4 TRGOUT event 0110: EXINT line 15/TMR1_CH4 event 0111: PCSWTRG 1000~1100: Unused. Configuration is not allowed. 1101: Timer 1 CH1 event 1110~1111: Unused.
Bit 11	DTALIGN	0x0	rw	Data alignment 0: Right alignment 1: Left alignment
Bit 10: 9	Reserved	0x0	resd	Kept at its default value
Bit 8	OCDMAEN	0x0	rw	DMA transfer enable of ordinary channels 0: Disabled 1: Enabled
Bit 7: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	ADCALINIT	0x0	rw	Initialize A/D calibration This bit is set by software and cleared by hardware. It is cleared after the calibration registers are initialized. 0: No initialization occurred or initialization completed 1: Enable initialization or initializations is ongoing
Bit 2	ADCAL	0x0	rw	A/D Calibration 0: No calibration occurred or calibration completed 1: Enable calibration or calibration is in process
Bit 1	RPEN	0x0	rw	Repetition mode enable 0: Repetition mode disabled When SQEN=0, a single conversion is done each time when a trigger event arrives; when SQEN=1, a group of conversion is done each timer when a trigger event arrives. 1: Repetition mode enabled

				When SQEN =0, continuous conversion mode on a single channel is enabled at each trigger event; when SQEN =1, continuous conversion mode on a group of channels is enabled at each trigger event.
Bit 0	ADCEN	0x0	rw	A/D converter enable 0: A/D converter disabled (ADC goes to power-down mode) 1: A/D converter enabled Note: When this bit is in OFF state, write an ON command can wake up The ADC from power-down mode. When this bit in ON state, write another ON command can start a regular group conversion. The application should pay attention to the fact that there is a delay of t_{STAB} between power on and start of conversion.

18.5.4 ADC sampling time register 1 (ADC_SPT1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 21	CSPT17	0x0	rw	Sample time selection of channel ADC_IN17 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 20: 18	CSPT16	0x0	rw	Sample time selection of channel ADC_IN16 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 17: 15	CSPT15	0x0	rw	Sample time selection of channel ADC_IN15 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 14: 12	CSPT14-	0x0	rw	Sample time selection of channel ADC_IN14 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 11: 9	CSPT13	0x0	rw	Sample time selection of channel ADC_IN13 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 8: 6	CSPT12	0x0	rw	Sample time selection of channel ADC_IN12 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles

				011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 5: 3	CSPT11	0x0	rw	Sample time selection of channel ADC_IN11 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 2: 0	CSPT10	0x0	rw	Sample time selection of channel ADC_IN10 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles

18.5.5 ADC sampling time register 2 (ADC_SPT2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value.
Bit 29: 27	CSPT9	0x0	rw	Sample time selection of channel ADC_IN9 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 26: 24	CSPT8	0x0	rw	Sample time selection of channel ADC_IN8 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 23: 21	CSPT7	0x0	rw	Sample time selection of channel ADC_IN7 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 20: 18	CSPT6	0x0	rw	Sample time selection of channel ADC_IN6 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 17: 15	CSPT5	0x0	rw	Sample time selection of channel ADC_IN5 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 14: 12	CSPT4	0x0	rw	Sample time selection of channel ADC_IN4 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles

				011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 11: 9	CSPT3	0x0	rw	Sample time selection of channel ADC_IN3 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 8: 6	CSPT2	0x0	rw	Sample time selection of channel ADC_IN2 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 5: 3	CSPT1	0x0	rw	Sample time selection of channel ADC_IN1 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles
Bit 2: 0	CSPT0	0x0	rw	Sample time selection of channel ADC_IN0 000: 1.5 cycles 001: 7.5 cycles 010: 13.5 cycles 011: 28.5 cycles 100: 41.5 cycles 101: 55.5 cycles 110: 71.5 cycles 111: 239.5 cycles

18.5.6 ADC preempted channel data offset register x (ADC_PCDTOx) (x=1..4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	PCDTOx	0x000	rw	Data offset for Preempted channel x Converted data stored in the ADC_PDTx = Raw converted data – ADC_PCDTOx

18.5.7 ADC voltage monitor high threshold register (ADC_VWHB)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	VMHB	0xFFF	rw	Voltage monitoring high boundary

18.5.8 ADC voltage monitor low threshold register (ADC_ VWLB)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value
Bit 11: 0	VMLB	0x000	rw	Voltage monitoring low boundary

18.5.9 ADC ordinary sequence register 1 (ADC_ OSQ1)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value
Bit 23: 20	OCLEN	0x0	rw	Ordinary conversion sequence length 0000: 1 conversion 0001: 2 conversions …… 1111: 16 conversions
Bit 19: 15	OSN16	0x00	rw	Number of 16th conversion in ordinary sequence
Bit 14: 10	OSN15	0x00	rw	Number of 15th conversion in ordinary sequence
Bit 9: 5	OSN14	0x00	rw	Number of 14th conversion in ordinary sequence
Bit 4: 0	OSN13	0x00	rw	Number of 13th conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 3, it means that the 13th conversion is ADC_IN3 channel.

18.5.10 ADC ordinary sequence register 2 (ADC_ OSQ2)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN12	0x00	rw	Number of 12th conversion in ordinary sequence
Bit 24: 20	OSN11	0x00	rw	Number of 11th conversion in ordinary sequence
Bit 19: 15	OSN10	0x00	rw	Number of 10th conversion in ordinary sequence
Bit 14: 10	OSN9	0x00	rw	Number of 9th conversion in ordinary sequence
Bit 9: 5	OSN8	0x00	rw	Number of 8th conversion in ordinary sequence
Bit 4: 0	OSN7	0x00	rw	Number of 7th conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 7th conversion is ADC_IN8 channel.

18.5.11 ADC ordinary sequence register 3 (ADC_ OSQ3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 29: 25	OSN6	0x00	rw	Number of 6th conversion in ordinary sequence
Bit 24: 20	OSN5	0x00	rw	Number of 5th conversion in ordinary sequence
Bit 19: 15	OSN4	0x00	rw	Number of 4th conversion in ordinary sequence
Bit 14: 10	OSN3	0x00	rw	Number of 3rd conversion in ordinary sequence
Bit 9: 5	OSN2	0x00	rw	Number of 2nd conversion in ordinary sequence
Bit 4: 0	OSN1	0x00	rw	Number of 1st conversion in ordinary sequence Note: The number can be from 0 to 17. For example, if the number is set to 8, it means that the 1st conversion is ADC_IN17 channel.

18.5.12 ADC preempted sequence register (ADC_ PSQ)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at its default value
Bit 21: 20	PCLEN	0x0	rw	Preempted conversion sequence length 00: 1 conversion 01: 2 conversions 10: 3 conversions 11: 4 conversions
Bit 19: 15	PSN4	0x00	rw	Number of 4th conversion in preempted sequence
Bit 14: 10	PSN3	0x00	rw	Number of 3rd conversion in preempted sequence
Bit 9: 5	PSN2	0x00	rw	Number of 2nd conversion in preempted sequence
Bit 4: 0	PSN1	0x00	rw	Number of 1st conversion in preempted sequence Note: The number can be from 0 to 17. For example, if the number is set to 3, it refers to the ADC_IN3 channel. If PCLEN is less than 4, the conversion sequence starts from 4-PCLEN. For example, when ADC_PSQ ([21: 0]) =10 00110 00101 00100 00011, it indicates that the scan conversion follows the sequence: 4, 5, 6, not 3, 4,5.

18.5.13 ADC preempted data register x (ADC_ PDTx) (x=1..4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value
Bit 15: 0	PDTx	0x0000	ro	Conversion data from preempted channel

18.5.14 ADC ordinary data register (ADC_ ODT)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	ADC2ODT	0x0000	ro	ADC2 conversion data of ordinary channel Note: These bits are reserved in ADC2 and ADC3. In ADC1, these bits are valid only in master/slave mode, and they contain the conversion result from the ADC2 ordinary channels.
Bit 15: 0	ODT	0x0000	ro	Conversion data of ordinary channel

19 Controller area network (CAN)

19.1 CAN introduction

CAN (Controller Area Network) is a distributed serial communication protocol for real-time and reliable data communication among various nodes. It supports the CAN protocol version 2.0A and 2.0B.

19.2 CAN main features

- Baud rates up to 1M bit/s
- Supports the time triggered communication
- Interrupt enable and mask
- Configurable automatic retransmission mode

Transmission

- Three transmit mailboxes
- Configurable transmit priority
- Supports the time stamp on transmission

Reception

- Two FIFOs with three-level depth
- 14 filter banks
- Supports the identifier list mode
- Supports the identifier mask mode
- FIFO overrun management

Time triggered communication mode

- 16-bit timers
- Time stamp on transmission

19.3 Baud rate

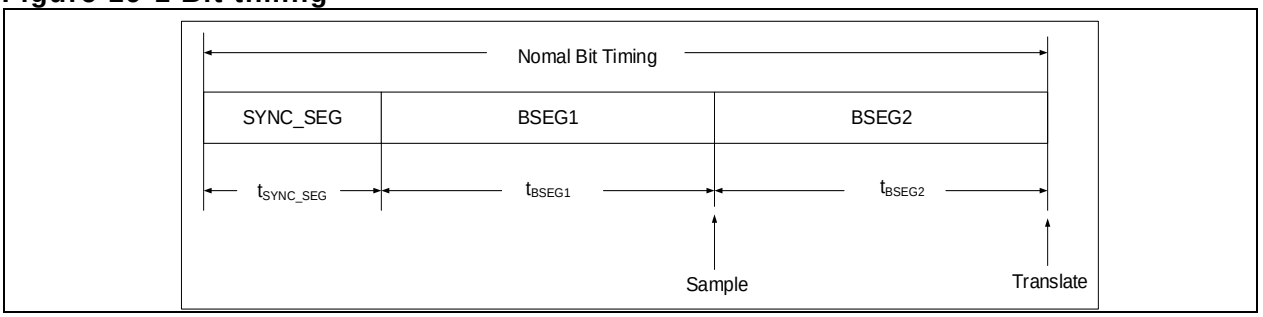
The nominal bit time of the CAN bus consists of three parts as follows:

Synchronization segment (SYNC_SEG): This segment has one time unit, and its time duration is defined by the BRDIV[11: 0] bit in the CAN_BTMG register.

Bit segment 1 (BIT SEGMENT 1): It is referred to as BSEG1 including the PROP_SEG and PHASE_SEG1 of the CAN standard. Its duration is between 1 and 16 time units, defined by the BTS1[3: 0] bit.

Big segment 2 (BIT SEGMENT 2): It is referred to as BSEG2 including the PHASE_SEG2 of the CAN standard. Its duration is between 1 and 8 time units, defined by the BTS2[2: 0] bit.

Figure 19-1 Bit timing



Baud rate formula:

$$BaudRate = \frac{1}{\text{Nomal Bit Timimg}}$$

$$Nomal\ \ Bit\ \ Timing = t_{SYNC\_SEG} + t_{BSEG1} + t_{BSEG2}$$

where

$$t_{SYNC\_SEG} = 1\ \text{x}\ t_q$$

$$t_{BSEG1} = (1 + \text{BTS1}[3:0])\ \text{x}\ t_q$$

$$t_{BSEG2} = (1 + \text{BTS2}[2:0])\ \text{x}\ t_q$$

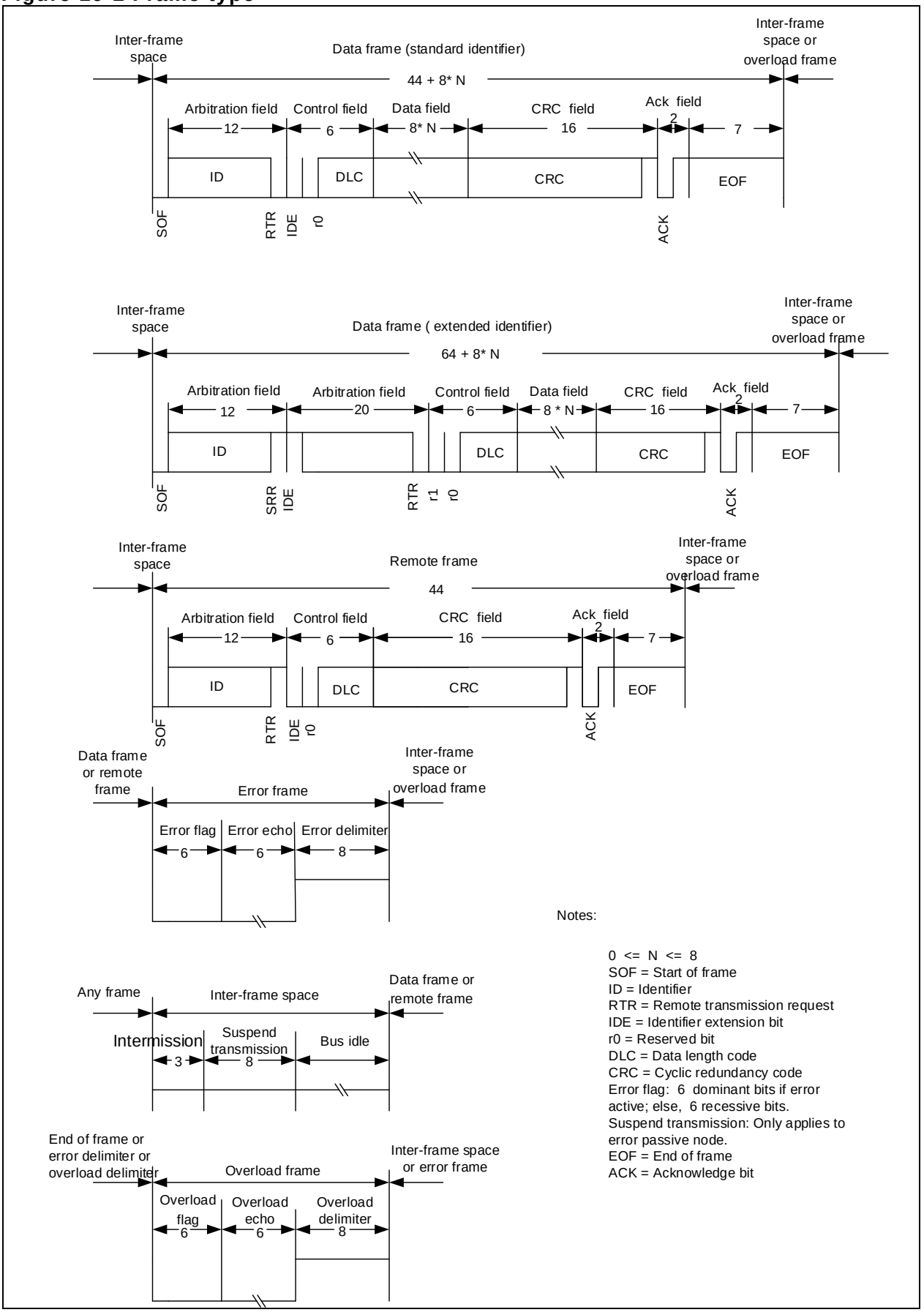
$$t_q = (1 + \text{BRDIV}[11:0])\ \text{x}\ t_{pclk}$$

Hard synchronization and resynchronization

The start location of each bit in CAN nodes is always in synchronization segment by default, and the sampling is performed at the edge location of bit segment 1 and big segment 2 simultaneously.

During the actual transmission, each bit of the CAN nodes has certain phase error due to the oscillator drift, transmission delay among the network nodes and noise interference. To avoid the impact on the communication, the start-bit edge and its subsequent falling edge can be synchronized or resynchronized. The time length of the synchronization compensation cannot be greater than the resynchronization width (1 to 4 time units, defined by the RSAW[1: 0] bit).

Figure 19-2 Frame type



Notes:

0 <= N <= 8
SOF = Start of frame
ID = Identifier
RTR = Remote transmission request
IDE = Identifier extension bit
r0 = Reserved bit
DLC = Data length code
CRC = Cyclic redundancy code
Error flag: 6 dominant bits if error active; else, 6 recessive bits.
Suspend transmission: Only applies to error passive node.
EOF = End of frame
ACK = Acknowledge bit

19.4 Interrupt management

The CAN controller contains four interrupt vectors that can be used to enable or disable interrupts by setting the CAN_INTEN register.

Figure 19-3 Transmit interrupt generation

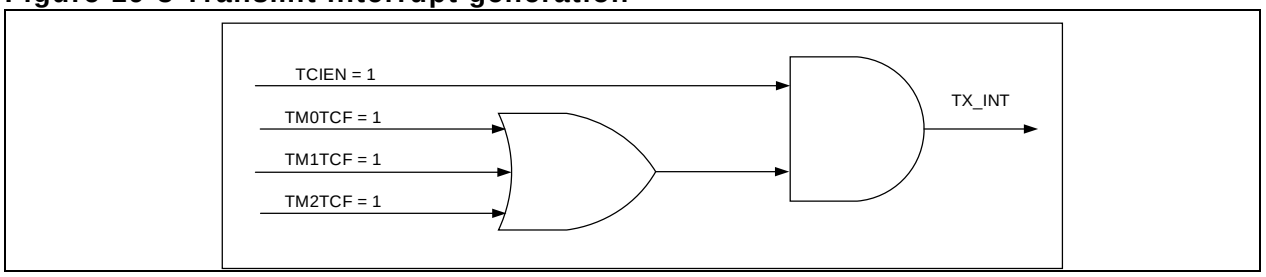


Figure 19-4 Receive interrupt 0 generation

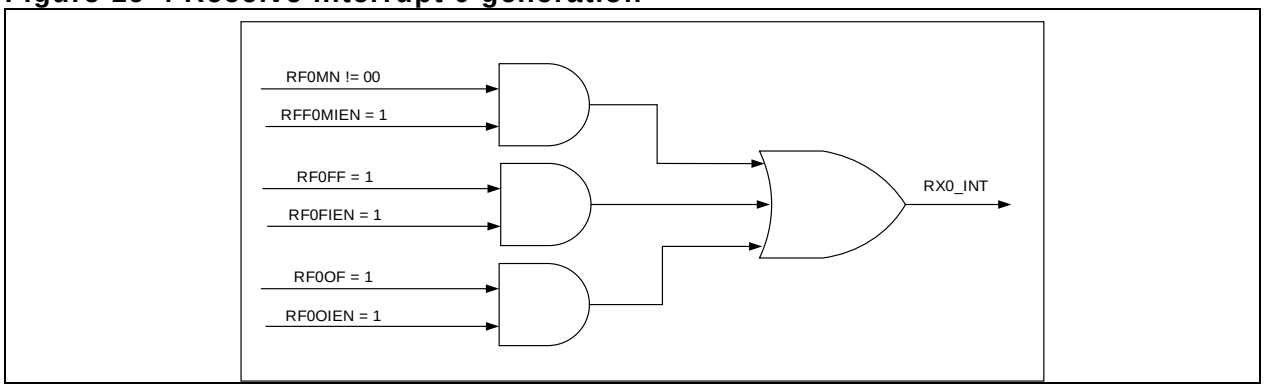


Figure 19-5 Receive interrupt 1 generation

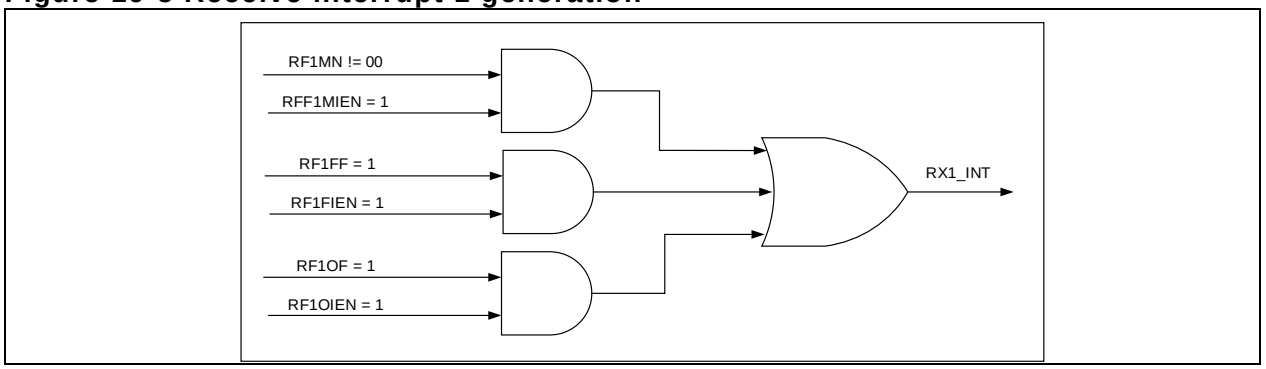
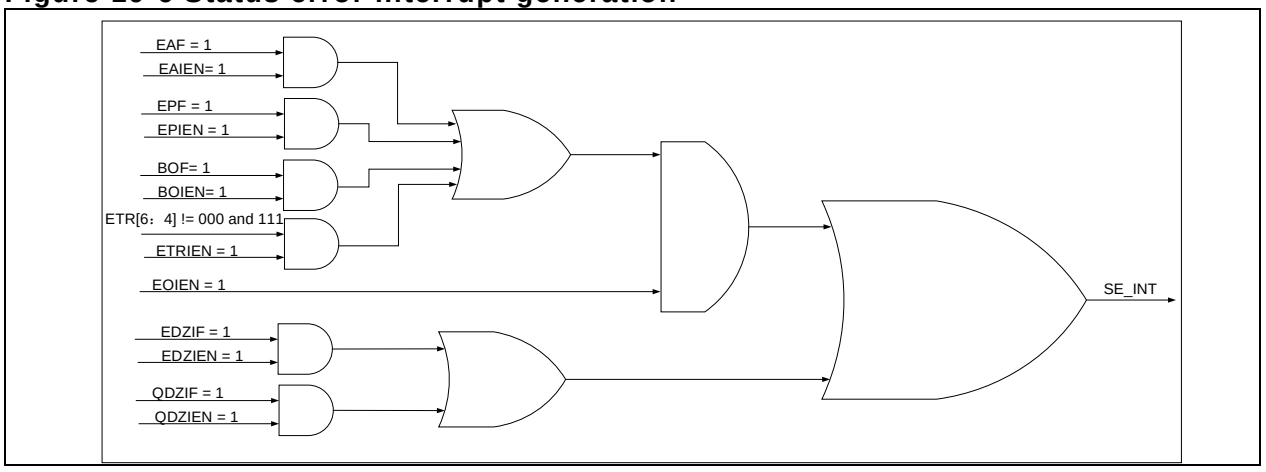


Figure 19-6 Status error interrupt generation



19.5 Design tips

The following information can be used as reference for CAN application development:

- Debug control
  When the system enters the debug mode, the CAN controller stops or continues to work normally, depending on the CANx_PAUSE bit in the DEBUG_CTRL register or the PTD bit in the CAN_MCTRL register.
- Time triggered communication
  The timer triggered communication is used to improve the real-time performance so as to avoid bus competition. It is activated by setting TTCEN=1 in the CAN_MCTRL register. The internal 16-bit timer is incremented each CAN bit time, and is sampled on the Start Of Frame bit to generate the time stamp value, which is stored in the CAN_RFCx and CAN_TMCx register.
- Register access protection
  The CAN_BTMG register can be modified only when the CAN is in frozen mode.
  Although the transmission of incorrect data will not cause problems at the network level, it can have severe impact on the application. Thus a transmit mailbox can be modified only when it is in empty state.
  The filter configuration in the CAN_FMCFG, CAN_FBWCFG and CAN_FRF registers can be modified only when FCS=1. The CAN_FiFBx register can be modified only when FCS=1 or FAENx=0.

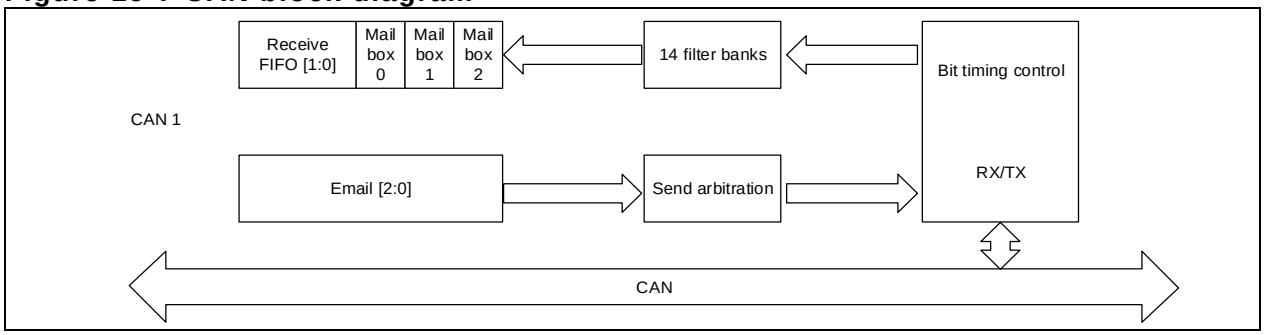
19.6 Functional overview

19.6.1 General description

As the number of nodes in the CAN network and the number of messages grows, an enhanced filtering mechanism is required to handle all types of messages in order to reduce the processing time of message reception. One FIFO scheme is used to ensure that the CPU can concentrate on application tasks for a long period of time without the loss of messages. In the meantime, the priority order of the messages to be transmitted is configured by hardware. Standard identifiers (11-bit) and extended identifiers (29-bit) are fully supported by hardware.

Based on the above mentioned conditions, the CAN controller provides 14 scalable/configurable identifier filter banks, 2 receive FIFOs with storing 3 complete messages each and being totally managed by hardware, and 3 transmit mailboxes with their transmit priority order defined by the transmit scheduler.

Figure 19-7 CAN block diagram



19.6.2 Operating modes

The CAN controller has three operating modes:

- Sleep mode

  After a system reset, the CAN controller is in Sleep mode. In this mode, the CAN clock is stopped to reduce power consumption and an internal pull-up resistance is disabled. However, the software can still access to the mailbox registers.

  The software request the CAN controller to enter Sleep mode by setting the DZEN bit in the CAN_MCTRL register. The hardware confirms the request by setting the DZC bit in the CAN_MSTS register.

  Exit Sleep mode in two ways: The CAN controller can be woke up by hardware clearing the DZEN bit when the AEDEN bit in the CAN_MCTRL register and the CAN bus activity is detected. It can also be woke up by software clearing the DZEN bit.

  Switch to Frozen mode: The CAN controller switches from Sleep mode to Frozen mode when the FZEN bit is set in the CAN_MCTRL register and the DZEN bit is cleared. Such switch operation is confirmed by hardware setting the FZC bit in the CAN_MSTS register.

  Switch to Communication mode: The CAN controller enters Communication mode when the FZEN and DZEN bits are both cleared and the CAN controller has synchronized with the bus. In other words, it must wait for 11 consecutive recessive bits to be detected on the CANRX pin.

- Frozen mode

  The software initialization can be done only in Frozen mode, including the CAN_BTMG and CAN_MCTRL registers. But the initialization of the 14 CAN filter banks (mode, scale, FIFO association, activation and filter values) can be done in non-Frozen mode. When the CAN controller is in Frozen mode, message reception and transmission are both disabled.

  Switch to Communication mode: The CAN controller leaves Frozen mode when the FZEN bit is cleared in the CAN_MCTRL register. This switch operation is confirmed by hardware clearing the FZC bit in the CAN_MSTS register. The CAN controller must be synchronized with the bus.

  Switch to Sleep mode: The CAN controller enters Sleep mode if DZEN=1 and FZEN=0 in the CAN_MCTRL register. This switch operation is confirmed by hardware setting the DZC bit in the CAN_MSTS register.

- Communication mode

  After the CAN_BTMG and CAN_MCTRL registers are configured in Frozen mode, the CAN controller enters Communication mode and is ready for message reception and transmission.

  Switch to Sleep mode: The CAN controller switches to Sleep mode when the DZEN bit is set in the CAN_MCTRL register and the current CAN bus transmission is complete.

  Switch to Frozen mode: The CAN controller enters Frozen mode when the FZEN bit is set in the CAN_MCTRL register and the current CAN bus transmission is complete.

19.6.3 Test modes

The CAN controller defines three test modes, including Listen-only mode, Loop back mode and combined Listen-only and Loop back mode. Test mode can be selected by setting the LOEN and LBEN bits in the CAN_BTMG register.

- Listen-only mode is selected when the LOEN bit is set in the CAN_BTMG register. In this mode, the CAN is able to receive data, but only recessive bits are output on the CANTX pin. In the meantime, the dominant bits output on the CANTX can be monitored by the receive side but without affecting the CAN bus.
- Loop back mode is selected by setting the LBEN bit in the CAN_BTMG register. In this mode, The CAN only receives the level signal on its CANTX pin. Meanwhile, the CAN can also send data to the external bus. The Loop back mode is mainly used for self-test functions.
- It is possible to combine the Listen-only and Loop back mode by setting the LOEN and LBEN bits in the CAN_BTMG register. In this case, the CAN is disconnected from the bus network, the CANTX pin remains in recessive state, and the transmit side is connected to the receive side.

19.6.4 Message filtering

The received message has to go through filtering by its identifier. If passed, the message will be stored in the corresponding FIFOs. If not, the message will be discarded. The whole operation is done by hardware without using CPU resources.

Filter bit width

The CAN controller provides 14 configurable and scalable filter banks (0~13). Each filter bank has two 32-bit registers, CAN_FiFB1 and CAN_FiFB2. The filter bit width can be configured as two 16 bits or one 32 bits, depending on the corresponding bits in the CAN_FBWCFG register.

32-bit filter register CAN_FiFBx includes the SID[10: 0], EID[17: 0], IDT and RTR bits.

CAN_FiFB1[31: 21]	CAN_FiFB1[20: 3]	CAN_FiFB1[2: 0]		
CAN_FiFB2[31: 21]	CAN_FiFB2[20: 3]	CAN_FiFB2[2: 0]		
SID[10: 0]/EID[28: 18]	EID[17: 0]	IDT	RTR	0

Two 16-bit filter register CAN_FiFBx includes SID[10: 0], IDT, RTR and EID[17: 15] bits

CAN_FiFB1[31: 21]	CAN_FiFB1 [20: 19]	CAN_FiFB1 [18: 16]	CAN_FiFB1[15: 5]	CAN_FiFB1 [4: 3]	CAN_FiFB1 [2: 0]	
CAN_FiFB2[31: 21]	CAN_FiFB2 [20: 19]	CAN_FiFB2 [18: 16]	CAN_FiFB2[15: 5]	AN_FiFB2 [4: 3]	CAN_FiFB2 [2: 0]	
SID[10: 0]	IDT	RTR	EID[17: 15]	SID[10: 0]	IDT	RTR	EID[17: 15]

Filtering mode

The filter can be configured in identifier mask mode or in identifier list mode by setting the FMSELx bit in the CAN_FMCFG register. The mask mode is used to specify which bits must match the pre-programmed identifiers, and which bits do not need. In identifier list mode, the identifier must match the pre-programmed identifier. The two modes can be used in conjunction with filter width to deliver four filtering modes below:

Figure 19-8 32-bit identifier mask mode

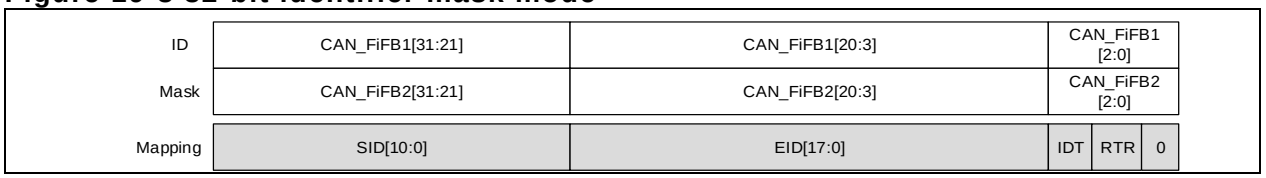


Figure 19-9 32-bit identifier list mode

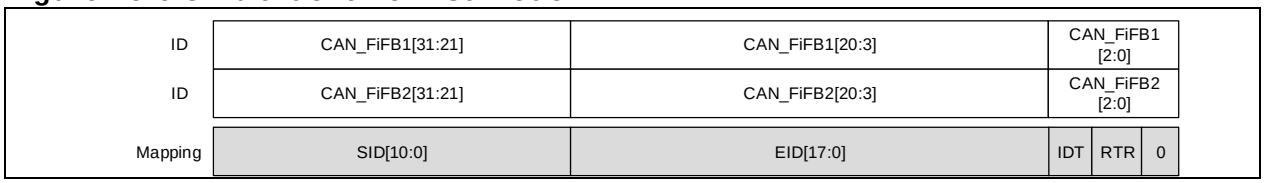


Figure 19-10 16-bit identifier mask mode

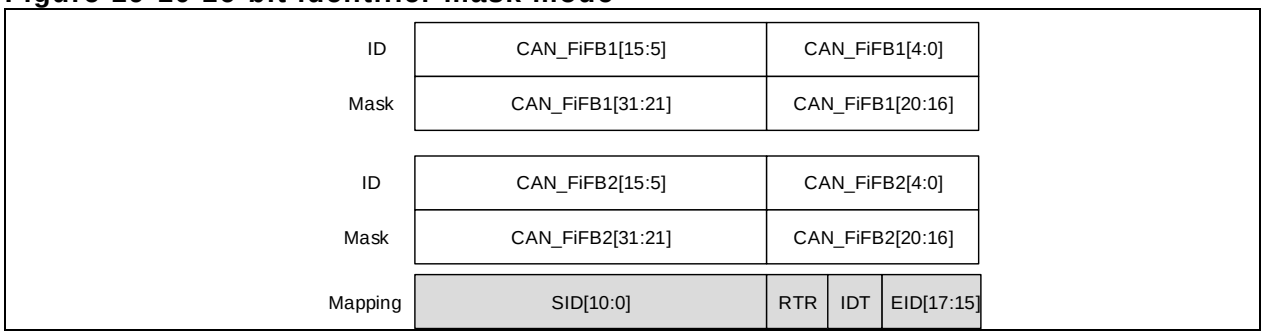
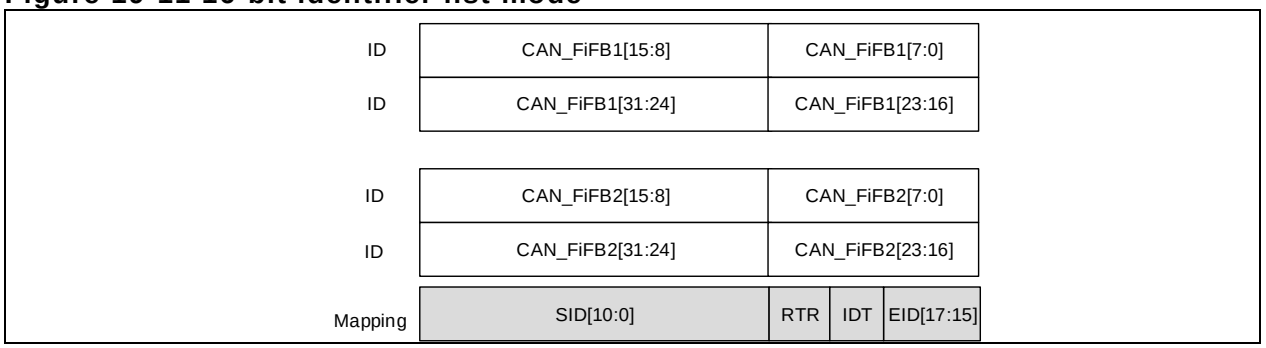


Figure 19-11 16-bit identifier list mode



Filter match number

14 filter banks have different filtering effects dependent on the bit width mode. For example, 32-bit identifier mask mode contains the filters numbered n while 16-bit identifier list mode contains the filters numbered n, n+1, n+2 and n+3. When a frame of message passes through the numbered-N filter, the number N is stored in the RFFMN[7: 0] bit in the CAN_RFCx register. The distribution of the filter number does not take into account the activation state of the filter banks.

Filter bank	FIFO0	Active	Filter number
0	CAN_F0FB1[31:0]-ID	Yes	0
	CAN_F0FB2[31:0]-ID		1
1	CAN_F1FB1[15:0]-ID	Yes	2
	CAN_F1FB1[31:16]-ID		3
	CAN_F1FB2[15: 0]-ID		4
	CAN_F1FB2[31: 16]-ID		5
2	CAN_F2FB1[31: 0]-ID	Yes	6
	CAN_F2FB2[31: 0]-Mask		
6	CAN_F6FB1[15: 0]-ID	No	7
	CAN_F6FB1[31: 16]-Mask		
	CAN_F6FB2[15: 0]-ID		8
	CAN_F6FB2[31: 16]-Mask		
9	CAN_F9FB1[31: 0]-ID	No	9
	CAN_F9FB2[31: 0]-ID		10
10	CAN_F10FB1[15: 0]-ID	Yes	11
	CAN_F10FB1[31:16]-Mask		
	CAN_F10FB2[15: 0]-ID		12
	CAN_F10FB2[31:16]-Mask		
12	CAN_F12FB1[15: 0]-ID	No	13
	CAN_F12FB1[31: 16]-ID		14
	CAN_F12FB2[15: 0]-ID		15
	CAN_F12FB2[31: 16]-ID		16

Filter bank	FIFO1	Active	Filter number
3	CAN_F3FB1[15:0]-ID	Yes	0
	CAN_F3FB1[31:16]-ID		1
	CAN_F3FB2[15:0]-ID		2
	CAN_F3FB2[31:16]-ID		3
4	CAN_F4FB1[31: 0]-ID	Yes	4
	CAN_F4FB2[31:0]-Mask		
5	CAN_F5FB1[15: 0]-ID	No	5
	CAN_F5FB1[31:16]-Mask		
	CAN_F5FB2[15: 0]-ID		6
	CAN_F5FB2[31:16]-Mask		
7	CAN_F7FB1[15: 0]-ID	No	7
	CAN_F7FB1[31:16]-ID		8
	CAN_F7FB2[15: 0]-ID		9
	CAN_F7FB2[31:16]-ID		10
8	CAN_F8FB1[31: 0]-ID	Yes	11
	CAN_F8FB2[31:0]-Mask		
11	CAN_F11FB1[31:0]-ID	Yes	12
	CAN_F11FB2[31:0]-ID		13
13	CAN_F13FB1[15:0]-ID	Yes	14
	CAN_F13FB1[31:16]-ID		15
	CAN_F13FB2[15:0]-ID		16
	CAN_F13FB2[31:16]-ID		17

Priority rules

When the CAN controller receives a frame of message, the message may pass through several filters. In this case, the filter match number stored in the receive mailbox is determined according to the following priority rules:

- A 32-bit filter has priority over a 16-bit filter
- For filters with identical bit width, the identifier list mode has priority over the identifier mask mode
- For filter with identical bit width and identifier mode, the lower number has priority over the higher number.

Filter configuration

- The CAN filters are configured by setting the FCS bit in the CAN_FCTRL register.
- Identifier mask mode or identifier list mode can be selected by setting the FMSELx bit in the CAN_FMCFG register.
- The filter bit width can be configured as two 16 bits or one 32 bits by setting the FBWSELx bit in the CAN_FBWCFG register.
- The filter x is associated with FIFO0 or FIFO1 by setting the FRFSELx bit in the CAN_FRF register.
- The filter banks x are activated by setting FAENx=1 in the CAN_FACFG register.
- Configure 0~13 filter banks by writing to the CAN_FiFBx register (i=0…13; x=1,2).
- Complete the CAN filter configuration by setting FCS=0 in the CAN_FCTRL register.

19.6.5 Message transmission

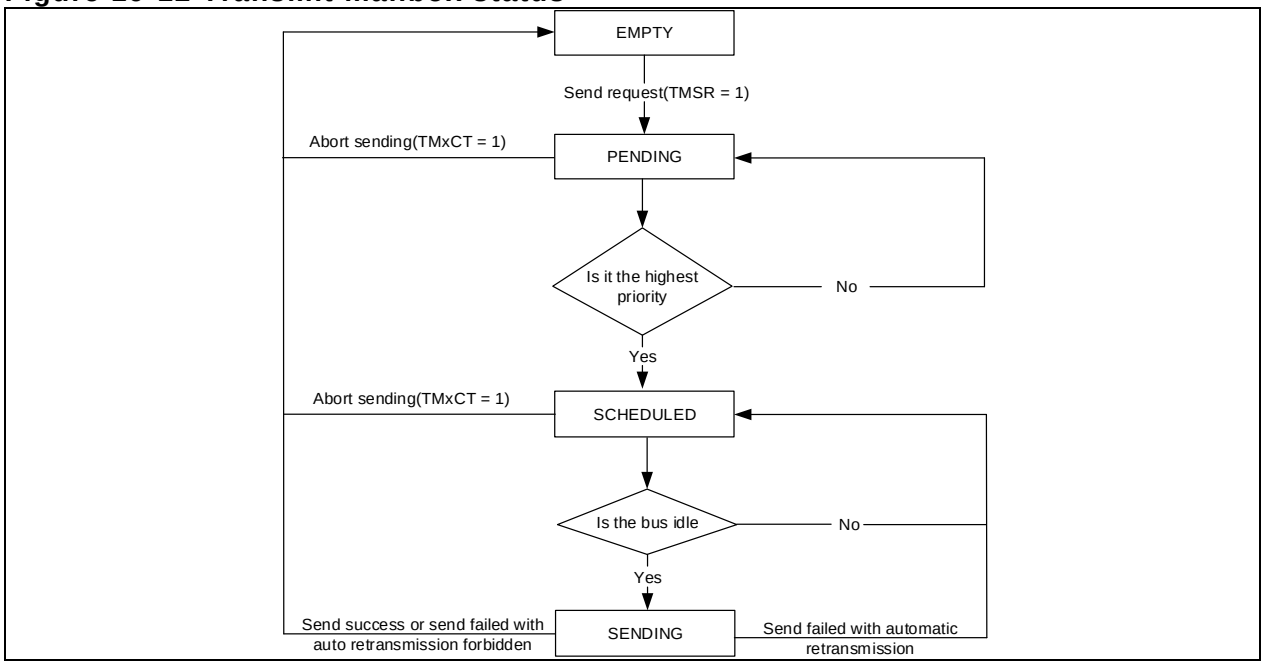
Register configuration

To transmit a message, it is necessary to select one transmit mailbox and configure it through the CAN_TMIx, CAN_TMCx, CAN_TMDTLx and CAN_TMDTHx registers. Once the mailbox configuration is complete, setting the TMSR bit in the CAN_TMIx register can initiate CAN transmission.

Message transmission

The mailbox enters pending state immediately after the mailbox is configured and the CAN controller receives the transmit request. At this point, the CAN controller will confirm whether the mailbox is given the highest priority or not. If yes, it will enter SCHEDULED STATE, otherwise, it will wait to get the highest priority. The mailbox in SCHEDULED state will monitor the CAN bus state so that the messages in SCHEDULED mailbox can be transmitted as soon as the CAN bus becomes idle. The mailbox will enter EMPTY state at the end of the message transmission.

Figure 19-12 Transmit mailbox status



Transmit priority configuration

When two or more transmit boxes are in PENDING state, their transmit priority must be given.

By identifier: When MMSSR=0 in the CAN_MCTRL register, the transmit order is defined by the identifier of the message in the mailbox. The message with lower identifier value has the highest priority. If the identifier values are the same, the message with lower mailbox number will be transmitted first.

By transmit request order: When MMSSR=1 in the CAN_MCTRL register, the transmit priority is given by the transmit request order of mailboxes.

Transmit status and error status

The TMxTCF, TMxTSF, TMxALF, TMxTEF and TMxEF bits in the CAN_TSTS register are used to indicate transmit status and error status.

TMxTCF bit: Transmission complete flag, indicating that the data transmission is complete when TMxTCF=1.

TMxTSF bit: Transmission success flag, indicating that the data has been transmitted successfully when TMxTSF =1.

TMxALF bit: Transmission arbitration lost flag, indicating that the data transmission arbitration is lost when TMxALF=1.

TMxTEF bit: Transmission error flag, indicating that the data transmission failed due to bus error, and an error frame is sent when TMxTEF=1.

TMxEF bit: Mailbox empty flag, indicating that the data transmission is complete and the mailbox becomes empty when TMxEF=1.

Transmit abort

The TMxCT bit is set in the CAN_TSTS register to abort the transmission of the current mailbox, detailed as follows:

When the current transmission fails or arbitration is lost, if the automatic retransmission mode is disabled, the transmit mailbox become EMPTY; if the automatic retransmission mode is enabled, the transmit mailbox becomes SCHEDULED, the mailbox transmission then is aborted and becomes EMPTY.

When the current transmission is complete successfully, the mailbox becomes EMPTY.

19.6.6 Message reception

Register configuration

The CAN_RFIx, CAN_RFCx, CAN_RFDTLx and CAN_RFDTHx registers can be used by user applications to obtain valid messages.

Message reception

The CAN controller boasts two FIFO with three levels to receive messages. FIFO rule is adopted. When the message is received correctly and has passed the identifier filtering, it is regarded as a valid message and is stored in the corresponding FIFO. The number of the received messages RFxMN[1: 0] will be incremented by one whenever the receive FIFO receives a valid message. If a valid message is received when RFxMN[1: 0]=3, the controller will select either to overwrite the previous messages or discard the new incoming message through the MDRSEL bit in the CAN_MCTRL register.

In the meantime, when the user reads a frame of message and the RFxR is set in the CAN_RFx register, one FIFO mailbox is released, and RFxMN[1: 0] bit is decremented by one in the CAN_RFx register.

Receive FIFO status

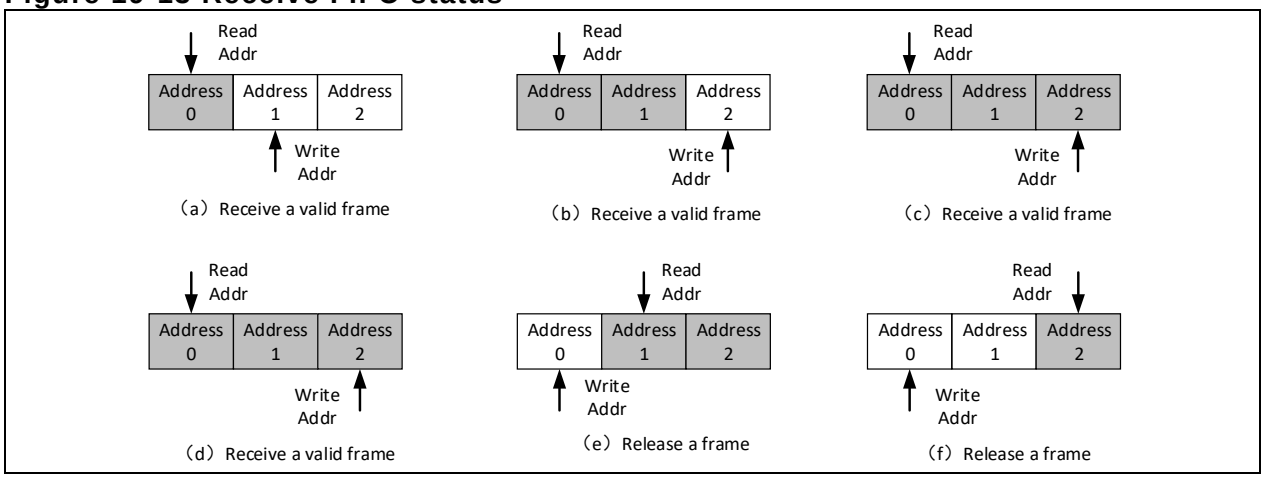
RFxMN[1: 0], RFxFF and RFxOF bits in the RFx register are used to indicate receive FIFO status.

RFxMN[1: 0]: indicates the number of valid messages stored in the FIFOx.

RFxFF: indicates that three valid messages are stored in the FIFOx (i.e. the three mailboxes are full), as shown in (c) of *Figure 19-13*.

RFxOF: indicates that a new valid message has been received while the FIFOx is full, as shown in (d) of *Figure 19-13*.

Figure 19-13 Receive FIFO status



19.6.7 Error management

The status of CAN nodes is indicated by the receive error counter (TEC) and transmit error counter (REC) bits in the CAN_ESTS register. In the meantime, the ETR[2: 0] bit in the CAN_ESTS register is used to record the last error source, and the corresponding interrupts will be generated when the CAN_INTEN register is enabled.

- Error active flag: When both TEC and REC are lower than 128, the system is in the error active state. An error active flag is set when an error is detected.
- Error passive flag: When either TEC or REC is greater than 127, the system is in the error passive state. An error passive flag is set when an error is detected.
- Bus-off state: The bus-off state is entered when TEC is greater than 255. In this state, it is impossible to transmit and receive messages. The CAN resumes from bus-off state in two ways:

  Option 1: When AEBOEN=0 in the CAN_MCTRL register, in communication mode, the software requests to enter Frozen mode and exit Frozen mode, and CAN will then resume from bus-off state after 128 occurrences of 11 consecutive recessive bits have been detected on the CAN RX pin.

  Option 2: When AEBOEN=1 in the CAN_MCTRL register, the CAN will resume from bus-off state automatically after 128 occurrences of 11 consecutive recessive bits have been detected on the CAN RX pin.

19.7 CAN registers

These peripheral registers must be accessed by words (32 bits).

Table 19-1 CAN register map and reset values

Register name	Offset	Reset value
MCTRL	000h	0x0001 0002
MSTS	004h	0x0000 0C02
TSTS	008h	0x1C00 0000
RF0	00Ch	0x0000 0000
FR1	010h	0x0000 0000
INTEN	014h	0x0000 0000
ESTS	018h	0x0000 0000
BTMG	01Ch	0x0123 0000
Reserved	020h~17Fh	xx
TMI0	180h	0xXXXX XXXX
TMC0	184h	0xXXXX XXXX

TMDTL0	188h	0xXXXX XXXX
TMDTH0	18Ch	0xXXXX XXXX
TMI1	190h	0xXXXX XXXX
TMC1	194h	0xXXXX XXXX
TMDTL1	198h	0xXXXX XXXX
TMDTH1	19Ch	0xXXXX XXXX
TMI2	1A0h	0xXXXX XXXX
TMC2	1A4h	0xXXXX XXXX
TMDTL2	1A8h	0xXXXX XXXX
TMDTH2	1ACh	0xXXXX XXXX
RFI0	1B0h	0xXXXX XXXX
RFC0	1B4h	0xXXXX XXXX
RFDTL0	1B8h	0xXXXX XXXX
RFDTH0	1BCh	0xXXXX XXXX
RFI1	1C0h	0xXXXX XXXX
RFC1	1C4h	0xXXXX XXXX
RFDTL1	1C8h	0xXXXX XXXX
RFDTH1	1CCh	0xXXXX XXXX
Reserved	1D0h~1FFh	xx
FCTRL	200h	0x2A1C 0E01
FMCFG	204h	0x0000 0000
Reserved	208h	xx
FBWCFG	20Ch	0x0000 0000
Reserved	210h	xx
FRF	214h	0x0000 0000
Reserved	218h	xx
FACFG	21Ch	0x0000 0000
Reserved	220h~23Fh	xx
F0FB1	240h	0xXXXX XXXX
F0FB2	244h	0xXXXX XXXX
F1FB1	248h	0xXXXX XXXX
F1FB2	24Ch	0xXXXX XXXX
…	…	…
F13FB1	2A8h	0xXXXX XXXX
F13FB2	2ACh	0xXXXX XXXX

19.7.1 CAN control and status registers

19.7.1.1 CAN master control register (CAN_MCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	PTD	0x1	rw	Prohibit trans when debug 0: Transmission works during debug 1: Transmission is prohibited during debug. Receive FIFO can be still accessible normally. Note: Transmission can be disabled only when PTD and CANx_PAUSE bits in the DEBUG_CTRL register are set simultaneously.
Bit 15	SPRST	0x0	rw1s	Software partial reset 0: Normal 1: Software partial reset Note: SPRST only reset receive FIFO and MCTRL register. The CAN enters Sleep mode after reset. Then this bit is automatically cleared by hardware.
Bit 14: 8	Reserved	0x00	resd	Kept at its default value.
Bit 7	TTCEN	0x0	rw	Time triggered communication mode enable 0: Time triggered communication mode disabled 1: Time triggered communication mode enabled
Bit 6	AEBOEN	0x0	rw	Automatic exit bus-off enable 0: Automatic exit bus-off disabled 1: Automatic exit bus-off enabled Note: When Automatic exit bus-off mode is enabled, the hardware will automatically leave bus-off mode as soon as an exit timing is detected on the CAN bus. When Automatic exit bus-off mode is disabled, the software must enter/leave the freeze mode once more, and then the bus-off state is left only when an exit timing is detected on the CAN bus.
Bit 5	AEDEN	0x0	rw	Automatic exit doze mode enable 0: Automatic exit sleep mode disabled 1: Automatic exit sleep mode enabled Note: When Automatic exit Sleep mode is disabled, the Sleep mode is left by software clearing the sleep request command. When Automatic exit sleep mode is enabled, the Sleep mode is left without the need of software intervention as soon as a message is monitored on the CAN bus.
Bit 4	PRSFEN	0x0	rw	Prohibit retransmission enable when sending fails enable 0: Retransmission is enabled. 1: Retransmission is disabled.
Bit 3	MDRSEL	0x0	rw	Message discard rule select when overflow 0: The previous message is discarded. 1: The new incoming message is discarded.
Bit 2	MMSSR	0x0	rw	Multiple message transmit sequence rule 0: The message with the smallest identifier is first transmitted. 1: The message with the first request order is first transmitted.
Bit 1	DZEN	0x1	rw	Doze mode enable 0: Sleep mode is disabled. 1: Sleep mode is enabled. Note: The hardware will automatically leave sleep mode when the AEDEN is set and a message is monitored on the CAN bus. After CAN reset or partial software reset, this bit is forced

				to be set by hardware, that is, the CAN will keep in sleep mode, by default.
Bit 0	FZEN	0x0	rw	Freeze mode enable 0: Freeze mode disabled 1: Freeze mode enabled Note: The CAN leaves Freeze mode once 11 consecutive recessive bits have been detected on the RX pin. For this reason, the software acknowledges the entry of Freeze mode after the FZC bit is cleared by hardware. The Freeze mode is entered only when the current CAN activity (transmission or reception) is completed. Thus the software acknowledges the exit of Freeze mode after the FZC bit is cleared by hardware.

19.7.1.2 CAN master status register (CAN_MSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	REALRX	0x1	ro	Real time level on RX pin 0: Low 1: High
Bit 10	LSAMPRX	0x1	ro	Last sample level on RX pin 0: Low 1: High Note: This value keeps updating with the REALRX.
Bit 9	CURS	0x0	ro	Current receive status 0: No reception occurs 1: Reception is in progress Note: This bit is set by hardware when the CAN reception starts, and it is cleared by hardware at the end of reception.
Bit 8	CUSS	0x0	ro	Current transmit status 0: No transmit occurs 1: Transmit is in progress Note: This bit is set by hardware when the CAN transmission starts, and it is cleared by hardware at the end of transmission.
Bit 7: 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	EDZIF	0x0	rw1c	Enter doze mode interrupt flag 0: Sleep mode is not entered or no condition for flag set. 1: Sleep mode is entered. Note: This bit is set by hardware only when EDZIEN=1 and the CAN enters Sleep mode. When set, this bit will generate a status change interrupt. This bit is cleared by software (writing 1 to itself) or by hardware when DZC is cleared.
Bit 3	QDZIF	0x0	rw1c	Exit doze mode interrupt flag 0: Sleep mode is not left or no condition for exit. 1: Sleep mode has been left or exit condition has generated. Note: This bit is cleared by software (writing 1 to itself) Sleep mode is left when a SOF is detected on the bus. When QDZIEN=1, this bit will generate a status change interrupt.
Bit 2	EOIF	0x0	rw1c	Error occur interrupt flag 0: No error interrupt or no condition for error interrupt flag 1: Error interrupt is generated. Note: This bit is cleared by software (writing 1 to itself). This bit is set by hardware only when the corresponding bit is set in the CAN_ESTS register and the corresponding interrupt enable bit in the CAN_INTEN register is enabled. When EOIEN=1, setting this bit will generate a status

				change interrupt.
Bit 1	DZC	0x1	ro	Doze mode acknowledge 0: The CAN is not in Sleep mode. 1: CAN is in Sleep mode. Note: This bit is used to decide whether the CAN is in Sleep mode or not. This bit acknowledges the Sleep mode request generated by software. The Sleep mode can be entered only when the current CAN activity (transmission or reception) is completed. For this reason, the software acknowledges the entry of Sleep mode after this bit is set by hardware. The Sleep mode is left only once 11 consecutive recessive bits have been detect on the CAN RX pin. For this reason, the software acknowledges the exit of Sleep mode after this bit is cleared by hardware.
Bit 0	FZC	0x0	ro	Freeze mode confirm 0: The CAN is not in Freeze mode. 1: The CAN is in Freeze mode. Note: This bit is used to decide whether the CAN is in Freeze mode or not. This bit acknowledges the Freeze mode request generated by software. The Freeze mode can be entered only when the current CAN activity (transmission or reception) is completed. For this reason, the software acknowledges the entry of Freeze mode after this bit is set by hardware. The Freeze mode is left only once 11 consecutive recessive bits have been detect on the CAN RX pin. For this reason, the software acknowledges the exit of Freeze mode after this bit is cleared by hardware.

19.7.1.3CAN transmit status register (CAN_TSTS)

Bit	Register	Reset value	Type	Description
Bit 31	TM2LPF	0x0	ro	Transmit mailbox 2 lowest priority flag 0: Mailbox 2 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 2 has the lowest priority.)
Bit 30	TM1LPF	0x0	ro	Transmit mailbox 1 lowest priority flag 0: Mailbox 1 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 1 has the lowest priority.)
Bit 29	TM0LPF	0x0	ro	Transmit mailbox 0 lowest priority flag 0: Mailbox 0 is not given the lowest priority. 1: Lowest priority (This indicates that more than one mailboxes are pending for transmission, the mailbox 0 has the lowest priority.)
Bit 28	TM2EF	0x1	ro	Transmit mailbox 2 empty flag This bit is set by hardware when no transmission is pending in the mailbox 2.
Bit 27	TM1EF	0x1	ro	Transmit mailbox 1 empty flag This bit is set by hardware when no transmission is pending in the mailbox 1.
Bit 26	TM0EF	0x1	ro	Transmit mailbox 0 empty flag This bit is set by hardware when no transmission is pending in the mailbox 0.
Bit 25: 24	TMNR	0x0	ro	Transmit Mailbox number record Note: If the transmit mailbox is free, these two bits refer to the number of the next transmit mailbox free. For example, in case of free CAN, the value of these two bit becomes 01 after a message transmit request is

				written. If the transmit box is full, these two bits refer to the number of the transmit mailbox with the lowest priority. For example, when there are three messages are pending for transmission, the identifiers of mailbox 0, mailbox 1 and mailbox 2 are 0x400, 0x433 and 0x411 respectively, and the value of these two bits becomes 01.
Bit 23	TM2CT	0x0	rw1c	Transmit mailbox 2 cancel transmit 0: No effect 1: Transmission is cancelled. Note: Software sets this bit to abort the transmission of mailbox 2. This bit is cleared by hardware when the transmit message in the mailbox 2 is cleared. Setting this bit has no effect if the mailbox 2 is free.
Bit 22: 20	Reserved	0x0	resd	Kept at its default value.
Bit 19	TM2TEF	0x0	rw1c	Transmit mailbox 2 transmission error flag 0: No error 1: Mailbox 2 transmission error Note: This bit is set when the mailbox 2 transmission error occurred. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 18	TM2ALF	0x0	rw1c	Transmit mailbox 2 arbitration lost flag 0: No arbitration lost 1: Transmit mailbox 2 arbitration lost Note: This bit is set when the mailbox 2 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 17	TM2TSF	0x0	rw1c	Transmit mailbox 2 transmission success flag 0: Transmission failed 1: Transmission was successful. Note: This bit indicates whether the mailbox 2 transmission is successful or not. It is cleared by software writing 1.
Bit 16	TM2TCF	0x0	rw1c	Transmit mailbox 2 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 2 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM1TSF, TM1ALF and TM1TEF bits of mailbox 2.
Bit 15	TM1CT	0x0	rw1s	Transmit mailbox 1 cancel transmit 0: No effect 1: Mailbox 1 cancel transmit Note: This bit is set by software to abort the transmission request on mailbox 1. Clearing the message transmission on mailbox 1 will clear this bit. Setting by this software has no effect when the mailbox 1 is free.
Bit 14: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	TM1TEF	0x0	rw1c	Transmit mailbox 1 transmission error flag 0: No error 1: Mailbox 1 transmission error Note: This bit is set when the mailbox 1 transmission error occurred. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 10	TM1ALF	0x0	rw1c	Transmit mailbox 1 arbitration lost flag

				0: No arbitration lost 1: Transmit mailbox 1 arbitration lost Note: This bit is set when the mailbox 1 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 9	TM1TSF	0x0	rw1c	Transmit mailbox 1 transmission success flag 0: Transmission failed 1: Transmission was successful. Note: This bit indicates whether the mailbox 1 transmission is successful or not. It is cleared by software writing 1.
Bit 8	TM1TCF	0x0	rw1c	Transmit mailbox 1 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 1 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM1TSF, TM1ALF and TM1TEF bits of mailbox 1.
Bit 7	TM0CT	0x0	rw1s	Transmit mailbox 0 cancel transmit 0: No effect 1: Mailbox 0 cancel transmit Note: This bit is set by software to abort the transmission request on mailbox 0. Clearing the message transmission on mailbox 0 will clear this bit. Setting by this software has no effect when the mailbox 0 is free.
Bit 6: 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	TM0TEF	0x0	rw1c	Transmit mailbox 0 transmission error flag 0: No error 1: Mailbox 0 transmission error Note: This bit is set when the mailbox 0 transmission error occurred. It is cleared by software writing 0 or by hardware at the start of the next transmission
Bit 2	TM0ALF	0x0	rw1c	Transmit mailbox 0 arbitration lost flag 0: No arbitration lost 1: Transmit mailbox 0 arbitration lost Note: This bit is set when the mailbox 0 transmission failed due to an arbitration lost. It is cleared by software writing 1 or by hardware at the start of the next transmission
Bit 1	TM0TSF	0x0	rw1c	Transmit mailbox 0 transmission success flag 0: Transmission failed 1: Transmission was successful. Note: This bit indicates whether the mailbox 0 transmission is successful or not. It is cleared by software writing 1.
Bit 0	TM0TCF	0x0	rw1c	Transmit mailbox 0 transmission completed flag 0: Transmission is in progress 1: Transmission is completed Note: This bit is set by hardware when the transmission/abort request on mailbox 0 has been completed. It is cleared by software writing 1 or by hardware when a new transmission request is received. Clearing this bit will clear the TM1TSF, TM1ALF and TM1TEF bits of mailbox 0.

19.7.1.4CAN receive FIFO 0 register (CAN_RF0)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5	RF0R	0x0	rw1s	Receive FIFO 0 release 0: No effect 1: Release FIFO Note: This bit is set by software to release FIFO 0. It is cleared by hardware when the FIFO 0 is released. Setting this bit by software has no effect when the FIFO 0 is empty. If there are more than two messages pending in the FIFO 0, the software has to release the FIFO 0 to access the second message.
Bit 4	RF0OF	0x0	rw1c	Receive FIFO 0 overflow flag 0: No overflow 1: Receive FIFO 0 overflow Note: This bit is set by hardware when a new message has been received and passed the filter while the FIFO 0 is full. It is cleared by software by writing 1.
Bit 3	RF0FF	0x0	rw1c	Receive FIFO 0 full flag 0: Receive FIFO 0 is not full 1: Receive FIFO 0 is full Note: This bit is set by hardware when three messages are pending in the FIFO 0. It is cleared by software by writing 1.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1: 0	RF0MN	0x0	ro	Receive FIFO 0 message num Note: These two bits indicate how many messages are pending in the FIFO 0. RF0ML bit is incremented by one each time a new message has been received and passed the filter while the FIFO 0 is not full. RF0ML bit is decremented by one each time the software releases the receive FIFO 0 by writing 1 to the RF0R bit.

19.7.1.5CAN receive FIFO 1 register (CAN_RF1)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5	RF1R	0x0	rw1s	Receive FIFO 1 release 0: No effect 1: Release FIFO Note: This bit is set by software to release FIFO 1. It is cleared by hardware when the FIFO 1 is released. Setting this bit by software has no effect when the FIFO 1 is empty. If there are more than two messages pending in the FIFO 0, the software has to release the FIFO 1 to access the second message.
Bit 4	RF1OF	0x0	rw1c	Receive FIFO 1 overflow flag 0: No overflow 1: Receive FIFO 1 overflow Note: This bit is set by hardware when a new message has been received and passed the filter while the FIFO 1 is full. It is cleared by software by writing 1.
Bit 3	RF1FF	0x0	rw1c	Receive FIFO 1 full flag 0: Receive FIFO 1 is not full 1: Receive FIFO 1 is full

				Note: This bit is set by hardware when three messages are pending in the FIFO 1. It is cleared by software by writing 1.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1: 0	RF1MN	0x0	ro	Receive FIFO 1 message num Note: These two bits indicate how many messages are pending in the FIFO 1. RF1ML bit is incremented by one each time a new message has been received and passed the filter while the FIFO 1 is not full. RF1ML bit is decremented by one each time the software releases the receive FIFO 1 by writing 1 to the RF1R bit.

19.7.1.6 CAN interrupt enable register (CAN_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at its default value.
Bit 17	EDZIEN	0x0	rw	Enter doze mode interrupt enable 0: Enter sleep mode interrupt disabled 1: Enter sleep mode interrupt enabled Note: EDZIF flag bit corresponds to this interrupt. An interrupt is generated when both this bit and EDZIF bit are set.
Bit 16	QDZIEN	0x0	rw	Quit doze mode interrupt enable 0: Quit sleep mode interrupt disabled 1: Quit sleep mode interrupt enabled Note: The flag bit of this interrupt is the QDZIF bit. An interrupt is generated when both this bit and QDZIF bit are set.
Bit 15	EOIEN	0x0	rw	Error occur interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled Note: The flag bit of this interrupt is the EOIF bit. An interrupt is generated when both this bit and EOIF bit are set.
Bit 14: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11	ETRIEN	0x0	rw	Error type record interrupt enable 0: Error type record interrupt disabled 1: Error type record interrupt enabled Note: EOIF is set only when this interrupt is enabled and the ETR[2: 0] is set by hardware.
Bit 10	BOIEN	0x0	rw	Bus-off interrupt enable 0: Bus-off interrupt disabled 1: Bus-off interrupt enabled Note: EOIF is set only when this interrupt is enabled and the BOF is set by hardware.
Bit 9	EPIEN	0x0	rw	Error passive interrupt enable 0: Error passive interrupt disabled 1: Error passive interrupt enabled Note: EOIF is set only when this interrupt is enabled and the EPF is set by hardware.
Bit 8	EAIEN	0x0	rw	Error active interrupt enable 0: Error warning interrupt disabled 1: Error warning interrupt enabled Note: EOIF is set only when this interrupt is enabled and the EAF is set by hardware.
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	RF1OIEN	0x0	rw	Receive FIFO 1 overflow interrupt enable 0: Receive FIFO 1 overflow interrupt disabled 1: Receive FIFO 1 overflow interrupt enabled Note: The flag bit of this interrupt is the RF1OF bit. An interrupt is generated when this bit and RF1OF bit are set.
Bit 5	RF1FIEN	0x0	rw	Receive FIFO 1 full interrupt enable

				0: Receive FIFO 1 full interrupt disabled 1: Receive FIFO 1 full interrupt enabled Note: The flag bit of this interrupt is the RF1FF bit. An interrupt is generated when this bit and RF1FF bit are set.
Bit 4	RF1MIEN	0x0	rw	FIFO 1 receive message interrupt enable 0: FIFO 1 receive message interrupt disabled 1: FIFO 1 receive message interrupt enabled Note: The flag bit of this interrupt is RF1MN bit, so an interrupt is generated when this bit and RF1MN bit are set.
Bit 3	RF0OIEN	0x0	rw	Receive FIFO 0 overflow interrupt enable 0: Receive FIFO 0 overflow interrupt disabled 1: Receive FIFO 0 overflow interrupt enabled Note: The flag bit of this interrupt is RF0OF bit, so an interrupt is generated when this bit and RF0OF bit are set.
Bit 2	RF0FIEN	0x0	rw	Receive FIFO 0 full interrupt enable 0: Receive FIFO 0 full interrupt disabled 1: Receive FIFO 0 full interrupt enabled Note: The flag bit of this interrupt is the RF0FF bit. An interrupt is generated when this bit and RF0FF bit are set
Bit 1	RF0MIEN	0x0	rw	FIFO 0 receive message interrupt enable 0: FIFO 0 receive message interrupt disabled 1: FIFO 0 receive message interrupt enabled Note: The flag bit of this interrupt is the RF0MN bit. An interrupt is generated when this bit and RF0MN bit are set
Bit 0	TCIEN	0x0	rw	Transmit mailbox empty interrupt enable 0: Transmit mailbox empty interrupt disabled 1: Transmit mailbox empty interrupt enabled Note: The flag bit of this interrupt is the TMxTCF bit. An interrupt is generated when this bit and TMxTCF bit are set

19.7.1.7 CAN error status register (CAN_ESTS)

Bit	Register	Reset value	Type	Description
Bit 31: 24	REC	0x00	ro	Receive error counter This counter is implemented in accordance with the receive part of the fault confinement mechanism of the CAN protocol.
Bit 23: 16	TEC	0x00	ro	Transmit error counter This counter is implemented in accordance with the transmit part of the fault confinement mechanism of the CAN protocol.
Bit 15: 7	Reserved	0x00	resd	Kept at its default value.
Bit 6: 4	ETR	0x0	rw	Error type record 000: No error 001: Bit stuffing error 010: Format error 011: Acknowledgement error 100: Recessive bit error 101: Dominant bit error 110: CRC error 111: Set by software Note: This field is used to indicate the current error type. It is set by hardware according to the error condition detected on the CAN bus. It is cleared by hardware when a message has been transmitted or received successfully. If the error code 7 is not used by hardware, this field can be set by software to monitor the code update.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	BOF	0x0	ro	Bus-off flag 0: Bus-off state is not entered. 1: Bus-off state is entered. Note: When the TEC is greater than 255, the bus-off state is entered, and this bit is set by hardware.

Bit 1	EPF	0x0	ro	Error passive flag 0: Error passive state is not entered 1: Error passive state is entered Note: This bit is set by hardware when the current error times has reached the Error passive state limit (Receive Error Counter or Transmit Error Counter >127)
Bit 0	EAF	0x0	ro	Error active flag 0: Error active state is not entered 1: Error active state is entered Note: This bit is set by hardware when the current error times has reached the Error active state limit (Receive Error Counter or Transmit Error Counter ≥96)

19.7.1.8 CAN bit timing register (CAN_BTMG)

Bit	Register	Reset value	Type	Description
Bit 31	LOEN	0x0	rw	Listen-Only mode 0: Listen-Only mode disabled 1: Listen-Only mode enabled
Bit 30	LBEN	0x0	rw	Loop back mode 0: Loop back mode disabled 1: Loop back mode enabled
Bit 29: 26	Reserved	0x0	resd	Kept at its default value.
Bit 25: 24	RSAW	0x1	rw	Resynchronization width tRSAW = tCAN x (RSAW[1: 0] + 1) Note: This field defines the maximum of time unit that the CAN hardware is allowed to lengthen or shorten in a bit.
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22: 20	BTS2	0x2	rw	Bit time segment 2 tBTS2 = tCAN x (BTS2[2: 0] + 1) Note: This field defines the number of time unit in Bit time segment 2.
Bit 19: 16	BTS1	0x3	rw	Bit time segment 1 tBTS1 = tCAN x (BTS1[3: 0] + 1) Note: This field defines the number of time unit in Bit time segment 1.
Bit 15: 12	Reserved	0x0	resd	Kept at its default value.
Bit 11: 0	BRDIV	0x000	rw	Baud rate division tq = (BRDIV[11: 0]+1) x tPCLK Note: This field defines the length of a time unit (tq).

19.7.2 CAN mailbox registers

This section describes the registers of the transmit and receive mailboxes. Refer to *section 19.6.5* for more information on register map.

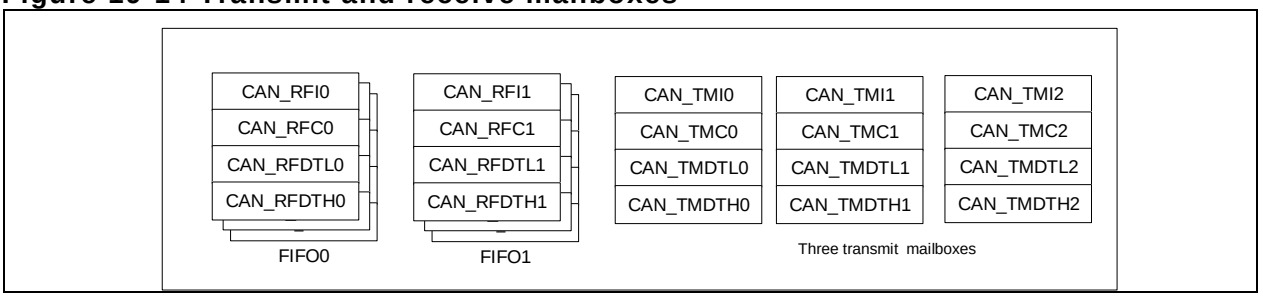
Transmit and receive mailboxes are the same except:

- RFFMN field in the CAN_RFCx register
- A receive mailbox is read only
- A transmit mailbox can be written only when empty. TMxEF=1 in the CAN_TSTS register indicates that the mailbox is empty.

There are three transmit mailboxes and two receive mailboxes. Each receive mailbox has 3-level depth of FIFO, and can only access to the first received message in the FIFO.

Each mailbox contains four registers.

Figure 19-14 Transmit and receive mailboxes



19.7.2.1 Transmit mailbox identifier register (CAN_TMIx) (x=0..2)

Note: 1. This register is write protected when its mailboxes are pending for transmission.
2.This register implements the Transmit Request control (bit 0) — reset value 0.

Bit	Register	Reset value	Type	Description
Bit 31: 21	TMSID/ TMEID	0xXXX	rw	Transmit mailbox standard identifier or extended identifier high bytes Note: This field defines the 11-bit high bytes of the standard identifier or extended identifier.
Bit 20: 3	TMEID	0xXXXXX	rw	Transmit mailbox extended identifier Note: This field defines the 18-bit low bytes of the extended identifier.
Bit 2	TMIDSEL	0xX	rw	Transmit mailbox identifier type select 0: Standard identifier 1: Extended identifier
Bit 1	TMFRSEL	0xX	rw	Transmit mailbox frame type select 0: Data frame 1: Remote frame
Bit 0	TMSR	0x0	rw	Transmit mailbox send request 0: No effect 1: Transmit request Note: This bit is cleared by hardware when the transmission has been completed (The mailbox becomes empty)

19.7.2.2 Transmit mailbox data length and time stamp register (CAN_TMCx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 16	TMTS	0xXXXX	rw	Transmit mailbox time stamp Note: This field contains the value of the CAN timer sampled at the SOF transmission.
Bit 15: 9	Reserved	0xXX	resd	Kept at its default value
Bit 8	TMTSTEN	0xX	rw	Transmit mailbox time stamp transmit enable 0: Time stamp is not sent 1: Time stamp is sent Note: This bit is valid only when the time-triggered communication mode is enabled. In the time stamp MTS[15: 0], the MTS[7: 0] is stored in the TMDT7, and MTS[15: 8] in the TMDT6. The data length must be programmed as 8 to send time stamp.
Bit 7: 4	Reserved	0xX	resd	Kept at its default value
Bit 3: 0	TMDTBL	0xX	rw	Transmit mailbox data byte length Note: This field defines the data length of a transmit message. A transmit message can contain from 0 to 8 data bytes.

19.7.2.3 Transmit mailbox data low register (CAN_TMDTLx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 24	TMDT3	0xXX	rw	Transmit mailbox data byte 3
Bit 23: 16	TMDT2	0xXX	rw	Transmit mailbox data byte 2
Bit 15: 8	TMDT1	0xXX	rw	Transmit mailbox data byte 1
Bit 7: 0	TMDT0	0xXX	rw	Transmit mailbox data byte 0

19.7.2.4 Transmit mailbox data high register (CAN_TMDTHx) (x=0..2)

All the bits in the register are write protected when the mailbox is not in empty state.

Bit	Register	Reset value	Type	Description
Bit 31: 24	TMDT7	0xXX	rw	Transmit mailbox data byte 7
Bit 23: 16	TMDT6	0xXX	rw	Transmit mailbox data byte 6 Note: This field will be replaced with MTS[15: 8] when the time-triggered communication mode is enabled and the corresponding time stamp transmit is enabled.
Bit 15: 8	TMDT5	0xXX	rw	Transmit mailbox data byte 5
Bit 7: 0	TMDT4	0xXX	rw	Transmit mailbox data byte 4

19.7.2.5 Receive FIFO mailbox identifier register (CAN_RFIx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 21	RFSID/RFEID	0xXXX	ro	Receive FIFO standard identifier or receive FIFO extended identifier Note: This field defines the 11-bit high bytes of the standard identifier or extended identifier.
Bit 20: 3	RFEID	0xXXXXX	ro	Receive FIFO extended identifier Note: This field defines the 18-bit low bytes of the extended identifier.
Bit 2	RFIDI	0xX	ro	Receive FIFO identifier type indication 0: Standard identifier 1: Extended identifier
Bit 1	RFFRI	0xX	Ro	Receive FIFO frame type indication 0: Data frame 1: Remote frame
Bit 0	Reserved	0x0	resd	Kept at its default value

19.7.2.6 Receive FIFO mailbox data length and time stamp register (CAN_RFCx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 16	RFTS	0xXXXX	ro	Receive FIFO time stamp Note: This field contains the value of the CAN timer sampled at the start of a receive frame.
Bit 15: 8	RFFMN	0xXX	ro	Receive FIFO filter match number Note: This field contains the filter number that a message has passed through.
Bit 7: 4	Reserved	0xX	resd	Kept at its default value
Bit 3: 0	RFDTL	0xX	ro	Receive FIFO data length Note: This field defines the data length of a receive message. A transmit message can contain from 0 to 8 data bytes. For a remote frame, its data length RFDTl is fixed 0.

19.7.2.7 Receive FIFO mailbox data low register (CAN_RFDTLx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description

Bit 31: 24	RFDT3	0xXX	ro	Receive FIFO data byte 3
Bit 23: 16	RFDT2	0xXX	ro	Receive FIFO data byte 2
Bit 15: 8	RFDT1	0xXX	ro	Receive FIFO data byte 1
Bit 7: 0	RFDT0	0xXX	ro	Receive FIFO data byte 0

19.7.2.8 Receive FIFO mailbox data high register (CAN_RFDTHx) (x=0..1)

Note: All the receive mailbox registers are read only.

Bit	Register	Reset value	Type	Description
Bit 31: 24	RFDT7	0xXX	ro	Receive FIFO data byte 7
Bit 23: 16	RFDT6	0xXX	ro	Receive FIFO data byte 6
Bit 15: 8	RFDT5	0xXX	ro	Receive FIFO data byte 5
Bit 7: 0	RFDT4	0xXX	ro	Receive FIFO data byte 4

19.7.3 CAN filter registers

19.7.3.1 CAN filter control register (CAN_FCTRL)

Note: All the non-reserved bits of this register are controlled by software completely.

Bit	Register	Reset value	Type	Description
Bit 31: 1	Reserved	0x160E0700	resd	Kept at its default value
Bit 0	FCS	0x1	rw	Filter configuration switch 0: Disabled (Filter bank is active) 1: Enabled (Filter bank is in configuration mode) Note: The initialization of the filter bank can be configured only when it is in configuration mode.

19.7.3.2 CAN filter mode configuration register (CAN_FMCFG)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at its default value
Bit 13: 0	FMSELx	0x0000	rw	Filter mode select Each bit corresponds to a filter bank. 0: Identifier mask mode 1: Identifier list mode

19.7.3.3 CAN filter bit width configuration register (CAN_ FBWCFG)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at its default value
Bit 13: 0	FBWSELx	0x0000	rw	Filter bit width select Each bit corresponds to a filter bank. 0: Dual 16-bit 1: Single 32-bit

19.7.3.4 CAN filter FIFO association register (CAN_ FRF)

Note: This register can be written only when FCS=1 in the CAN_FCTRL register (The filter is in configuration mode)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at its default value
Bit 13: 0	FRFSELx	0x0000	rw	Filter relation FIFO select Each bit corresponds to a filter bank. 0: Associated with FIFO0 1: Associated with FIFO1

19.7.3.5 CAN filter activation control register (CAN_ FACFG)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at its default value
Bit 13: 0	FAENx	0x0000	rw	Filter active enable Each bit corresponds to a filter bank. 0: Disabled 1: Enabled

19.7.3.6 CAN filter bank i filter bit register (CAN_ FiFBx) (i=0..13; x=1..2)

Note: There are 14 filter banks (i=0..13). Each filter bank consists of two 32-bit registers, CAN_FiFB[2: 1]. This register can be modified only when the FAENx bit of the CAN_FACFG register is cleared or the FCS bit of the CAN_FCTRL register is set.

Bit	Register	Reset value	Type	Description
Bit 31: 0	FFDB	0xXXXX XXXX	rw	Filters filter data bit Identifier list mode: The configuration value of the register matches with the level of the corresponding bit of the data received on the bus (If it is a standard frame, the value of the corresponding bit of the extended frame is neglected.) Identifier mark mode: Only the bit with its register configuration value 1 can match with the level of the corresponding bit of the data received on the bus. It don't care when the register value is 0.

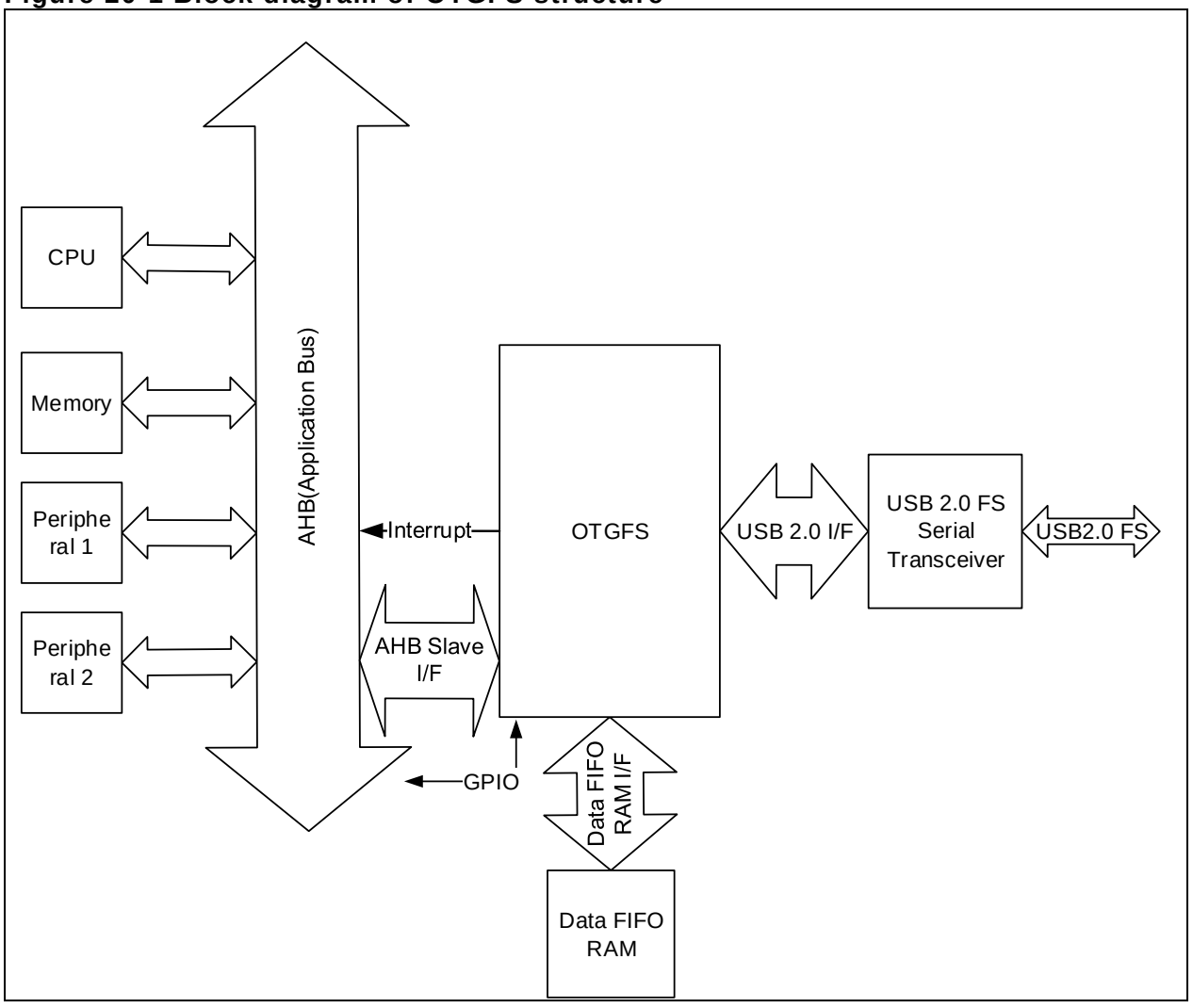
20 Universal serial bus full-seed device interface (OTGFS)

As a full-speed dual-role device, the OTGFS is fully compliant with the Universal Serial Bus Specification Revision2.0.

20.1 USBFS structure

Figure 20-1 shows the block diagram of the OTGFS structure. The OTGFS module is connected to the AHB and has a dedicated SRAM of 1280 bytes.

Figure 20-1 Block diagram of OTGFS structure



20.2 OTGFS functional description

The OTGFS module consists of an OTGFS controller, PHY and 1280-byte SRAM.

The OTGFS supports control transfer, bulk transfer, interrupt transfer and synchronous transfer.

The OTGFS is a USB full-speed dual role device controller (DRD). The device mode is selected by the FDEVMODE bit in the OTGFS_GUSBCFG register, while the host mode is selected by the FHSTMODE bit in the OTGF_GUSBCFG register. The internal 1.5KΩ pull-up resistor and 1.5KΩ pull-down resistor are available in the OTG PHY for the sake of dual role device.

In device mode, the OTGFS supports one bidirectional control endpoint, 3 IN endpoints, and 3 OUT endpoints, in which the endpoint 3 can be configured as endpoint 4(1); in hose mode, the OTGFS supports 8 host channels.

The OTGFS supports SOF pulse feature: a SOF pulse generates at a SOF packet, the pulse can output to the timer 2;

Suspend mode is supported. The OTGFS goes into power-saving mode after Suspend mode is entered.

As a device, a unified FIFO buffer is allocated for all OUT endpoints, and a separate FIFO buffer is provided to each of IN endpoints. As a host, a unified receive FIFO is allocated for all receive channels, a unified transmit FIFO for all non-periodic transmit channels, and a unified transmit FIFO for all periodic transmit channels.

OTGFS supports suspend mode. When the STOPPCLK bit in the OTGFS_PCGCCTL register is set, OTGFS goes into suspend mode if the bus signal is not received in continuous 3ms. Besides, power consumption reduction can be achieved by setting the LP_MODE bit in the OTGFS_GCCFG register to disable PHY receiving function.

Note 1: This feature is available in Revision C products.

20.3 OTGFS clock and pin configuration

20.3.1 OTGFS clock configuration

The OTGFS interface has two clocks: USB control clock and APB bus clock. The USB full-speed device bus speed standard is 12Mb/s±0.25%, so it is necessary to supply 48MHz±0.25% for the OTGFS to perform USB bus sampling.

OTGFS 48M clock source:

- Divided by PLL

  The PLL output frequency must ensure that the USBDIV (see CRM_CFG register) can be divided to 48MHz.

Note: The APB clock frequency must be greater than 30 MHz when OTGFS is enabled.

20.3.2 OTGFS pin configuration

The OTGFS input/output pins are multiplexed with GPIOs. The GPIOs are used as OTGFS in one of the following conditions:

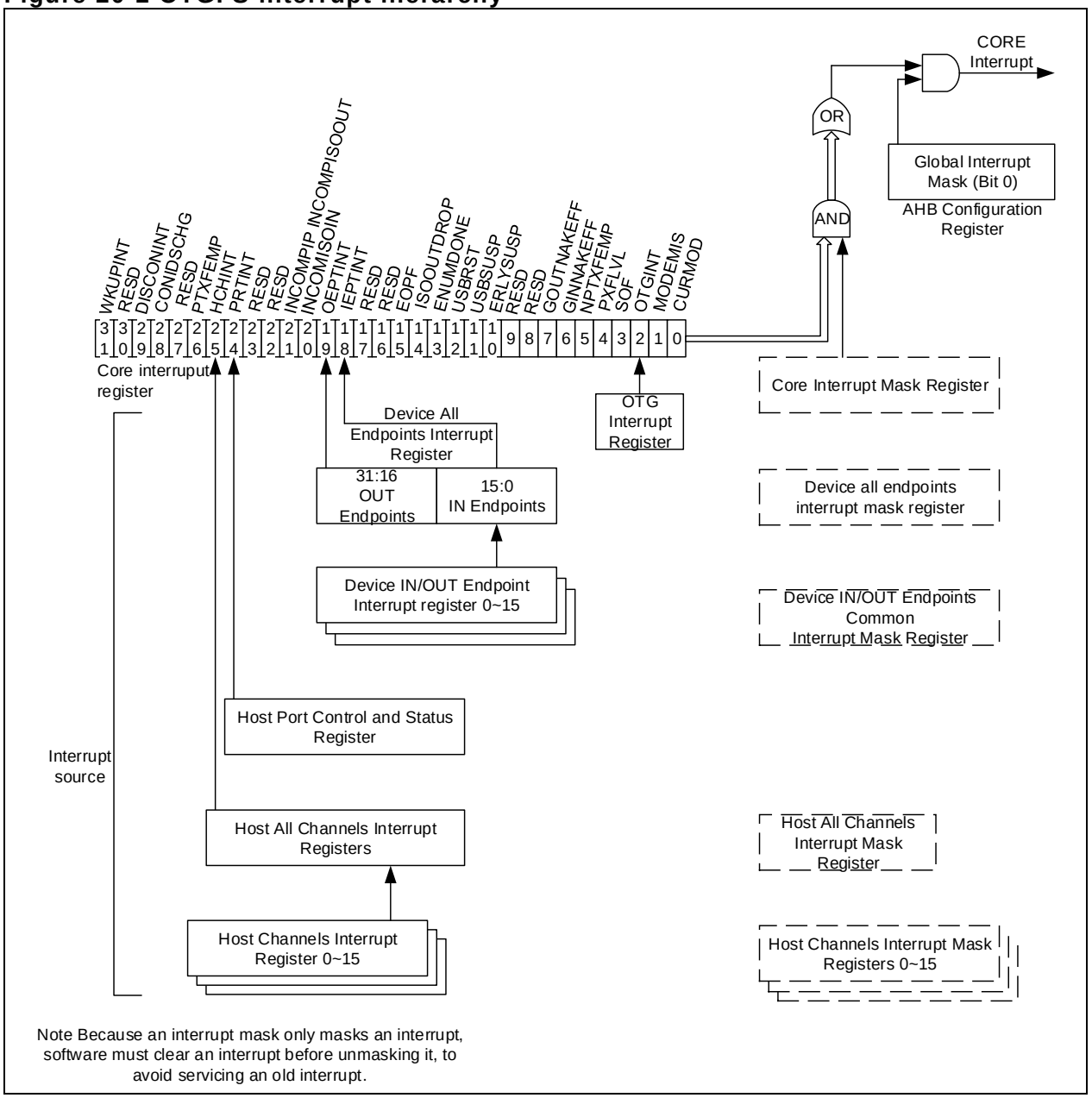
Table 20-1 OTGFS input/output pins

Pin	GPIO	Description
OTGFS_SOF	PA8	Enable OTG in CRM, and SOFOUTEN=1
OTGFS_VBUS	PA9	PA9 is configured as floating input
OTGFS_ID	PA10	Enable OTG in CRM
OTGFS_D-	PA11	Enable OTG in CRM, and PWRDOWN=1
OTGFS_D+	PA12	Enable OTG in CRM, and PWRDOWN=1

20.4 OTGFS interrupts

Figure 20-2 shows the OTGFS interrupt hierarchy. Refer to the OTGFS interrupt register (OTGFS_GINTSTS) and OTGFS interrupt mask register (OTGFS_GINTMSK).

Figure 20-2 OTGFS interrupt hierarchy



20.5 OTGFS functional description

20.5.1 OTGFS initialization

If the cable is connected during power-on, the current operation mode bit (CURMOD bit) in the controller interrupt register indicates the current mode. The OTGFS controller enters host mode when A-type plug is connected or device mode when a B-type plug is connected.

This section explains the initialization of the OTGFS controller after power-on. The application must follow the initialization sequence, however in host or device mode. All controller global registers are initialized according to the controller configuration.

1. Program the following fields in the global AHB configuration register:

- Global interrupt mask bit = 0x1
- Non-periodic transmit FIFO empty level
- Periodic transmit FIFO empty level

2. Program the following fields in the global AHB configuration register:

- OTGFS_GINTMSK.RXFLVLMSK = 0x0

3. Program the following fields in the OTGFS_GUSBCFG register:

- Full-speed timeout standard bit
- USB turnaround time bit

4. The software must unmask the following bits in the OTGFS_GINTMSK register:

- OTG interrupt mask
- Mode mismatch interrupt mask

5. The software can read the CURMOD bit in the OTGFS_GINTSTS register to determine whether the OTGFS controller is operating in host or device mode.

20.5.2 OTGFS FIFO configuration

20.5.2.1 Device mode

A dynamic FIFO allocation is required during power-on or USB reset. In device mode, the application must meet the following conditions before modifying FIFO SRAM allocation.

- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.EPENA = 0x0
- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.NAKSTS = 0x1

The TXFNUM bit in the OTGFS_GRSTCTL register is used to refresh the controller transmit FIFO. Refer to Section Refresh controller transmit FIFO for more information.

Attention should be paid to the following information during FIFO SRAM allocation:

(1) Receive FIFO SRAM allocation

- SRAM for SETUP Packets: 13 WORDs must be reserved in the receive FIFO to receive one SETUP Packet on control endpoint. The controller does not use these locations, which are reserved for SETUP packets.
- One WORD is to be reserved for global OUT NAK
- Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (largest packet size/4) + 1 must be allocated to receive data packets. If several synchronous endpoints are enabled, at least two (largest packet size/4) + 1 spaces are needed to receive data packets. In most cases, two (largest packet size/4) + 1 spaces are recommended so that the USB can receive the subsequent packet while the previous packet is being transferred to the AHB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.
- Transfer complete status information, along with the last packet for each endpoint, is also pushed to the FIFO
- One location must be reserved for the disable status bit of each endpoint
- Typically, two WORDs for each OUT endpoint are recommended.

(2) Transmit FIFO SRAM allocation

The minimum SRAM space required for each IN endpoint transmit FIFO is the maximum data packet size for that particular IN endpoint. The more the space allocated to the transmit IN endpoint FIFO, the better the USB performance, and this helps to avoid latency on the AHB line.

Table 20-2 OTGFS transmit FIFO SRAM allocation

FIFO name	SRAM size
Receive FIFO	rx_fifo_size, including setup packets, OUT endpoint control information and OUT data packets.
Transmit FIFO 0	tx_fifo_size[0]
Transmit FIFO 1	tx_fifo_size[1]
Transmit FIFO 2	tx_fifo_size[2]
……	……
Transmit FIFO i	tx_fifo_size[i]

Configure the following registers according to the above mentioned:

1. OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

- OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size

2. Endpoint 0 TX FIFO size register (OTGFS_DIEPTXF0)

- OTGFS_DIEPTXF0.INEPT0TXDEP = tx_fifo_size[0]
- OTGFS_DIEPTXF0.INEPT0TXSTADDR = rx_fifo_size

3. Device IN endpoint transmit FIFO#1 size register (OTGFS_DIEPTXF1)

- OTGFS_DIEPTXF1.INEPTXFSTADDR = OTGFS_DIEPTXF0.INEPT0TXSTADDR + tx_fifo_size[0]

4. Device IN endpoint transmit FIFO#2 size register (OTGFS_DIEPTXF2)

- OTGFS_DIEPTXF2.INEPTXFSTADDR=OTGFS_DIEPTXF1.INEPTXFSTADDR + tx_fifo_size[1]

5. Device IN endpoint transmit FIFO#i size register (OTGFS_DIEPTXFi)

- OTGFS_DIEPTXFi.INEPTXFSTADDR=OTGFS_DIEPTXFi-1.INEPTXFSTADDR +tx_fifo_size[i-1]

6. After SRAM allocation, refresh transmit FIFO and receive FIFO to ensure normal FIFO running.

- OTGFS_GRSTCTL.TXFNUM = 0x10
- OTGFS_GRSTCTL.TXFFLSH = 0x1
- OTGFS_GRSTCTL.RXFFLSH = 0x1

The application cannot perform other operations on the controller until the TXFFLSH and RXFFLSH bits are cleared.

20.5.2.2 Host mode

In host mode, the application must confirm the following status before changing FIFO SRAM allocation:

- All channels have been disabled
- All FIFOs are empty

After FIFO SRAM allocation is complete, the application must refresh all FIFOs in the controller through the TXFNUM bit in the OTGFS_GRSTCTL register.

After allocation, the FIFO pointers must be reset by refreshing operation to ensure normal FIFO running. Refer to Section Refresh controller transmit FIFO for more information.

(1) Receive FIFO SRAM allocation

Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (largest packet size/4) + 2 must be allocated to receive data packets. If more synchronous endpoints are enabled, then at least two (largest packet size/4) + 2 spaces must be allocated to receive back-to-back packets. In most cases, two (largest packet size/4) + 2 spaces are recommended so that the USB can receive the subsequent packet while the previous packet is being transferred to the AHB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.

Transfer complete status information and channel abort information, along with the last packet in the host channel is also pushed to the FIFO. Thus, two WORDs must be allocated for this.

(2) Transmit FIFO SRAM allocation

The minimum SRAM space required for the host non-periodic transmit FIFO is the largest packet size of all non-periodic OUT channels. The more the space allocated to the non-periodic FIFO, the better the USB performance, and this helps to avoid latency on the AHB line. Typically, two largest packet sizes of space is recommended so that the AHB can get the next data packet while the current packet is being transferred to the USB. If there is a longer latency on AHB, sufficient spaces must be reserved to receive multiple packets in order to prevent synchronous data packet loss.

The minimum SRAM space required for the host periodic transmit FIFO is the largest packet size of all periodic OUT channels.

(3) Internal storage space allocation

Table 20-3 OTGFS internal storage space allocation

FIFO Name	Data SRAM Size
Receive FIFO	rx_fifo_size
Non-periodic transmit FIFO	tx_fifo_size[0]
Periodic transmit FIFO	tx_fifo_size[1]

Configure the following registers according to the above mentioned:

1. OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

- OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size

2. OTGFS Non-periodic TX FIFO size register (OTGFS_GNPTXFSIZ)

- OTGFS_GNPTXFSIZ.NPTXFDEP = tx_fifo_size[0]
- OTGFS_GNPTXFSIZ. NPTXFSTADDR = rx_fifo_size

3. OTGFS host periodic transmit FIFO size register (OTGFS_HPTXFSIZ)

- OTGFS_HPTXFSIZ.PTXFSIZE = tx_fifo_size[1]
- OTGFS_HPTXFSIZ.PTXFSTADDR = OTGFS_GNPTXFSIZ.NPTXFSTADDR + tx_fifo_size[0]

4. After SRAM allocation, refresh transmit FIFO and receive FIFO to ensure normal FIFO running.

- OTGFS_GRSTCTL.TXFNUM = 0x10
- OTGFS_GRSTCTL.TXFFLSH = 0x1
- OTGFS_GRSTCTL.RXFFLSH = 0x1
- The application cannot perform other operations on the controller until the TXFFLSH and RXFFLSH bits are cleared.

20.5.2.3 Refresh controller transmit FIFO

The application refreshes all transmit FIFOs through the TXFFLSH bit in the OTGFS_GRSTCTL register:

- Check whether GINNAKEFF=0 or not in the OTGFS_GINTSTS register. If this bit has been cleared, write 0x1 to the OTGFS_DCTL.SGNPINNAK register. When the NACK valid interrupt is set, it means that the controller does not read FIFO.
- Wait until GINNAKEFF = 0x1 in the OTGFS_GINTSTS register, indicating that the NAK configuration has taken effect for all IN endpoints.
- Poll the OTGFS_GRSTCTL register and wait until AHBIDLE=1. AHBIDLE = H indicates that the controller does not write the FIFO.
- Confirm whether TXFFLSH = 0x0 or not in the OTGFS_GRSTCTL register. If TXFFLSH is cleared, write the transmit FIFO number to be refreshed into the OTGFS_GRSTCTL.TXFNUM register.
- Set TXFFLSH = 0x1 in the OTGFS_GRSTCTL register, and wait until it is cleared.
- Set the CGNPINNAK bit in the OTGFS_DCTL register.

20.5.3 OTGFS host mode

In host mode, an external voltage pump is required to supply VBUS continuously, for the controller is unable to provide 5V for VBUS internally.

20.5.3.1 Host initialization

The following steps must be respected to initialize the controller:

1. Unmask interrupt through the PRTINTMSK bit in the OTGFS_GINTMSK register
2. Program the OTGFS_HCFG register
3. Set PRTPWR = 0x1 in the OTGFS_HPRT register to drive VBUS supply on the USB
4. Wait until that the PRTCONDETbit is set in the OTGFS_HPRT0 register, indicating that the device is connected to the port
5. Set PRTRST = 0x1 in the OTGFS_HPRT register to issue a reset operation
6. Wait for at least 10 ms to ensure the completion of the reset
7. Set PRTRST = 0x0 in the OTGFS_HPRT register
8. Wait for the interrupt (PRTENCHNG bit in the OTGFS_HPRT register)
9. Read the PRTSPD bit in the OTGFS_HPRT register to get the enumeration speed
10. Configure the HFIR register according to the selected PHY clock value
11. Select the size of the receive FIFO by setting the OTGFS_GRXFSIZ register
12. Select the start address and size of the non-periodic transmit FIFO by setting the OTGFS_GNPTXFSIZ register
13. Select the start address and size of the periodic transmit FIFO by setting the OTGFS_HPTXFSIZ register

To communicate with the device, the application must enable and initialize at least one channel according to OTGFS channel initialization requirements.

20.5.3.2 OTGFS channel initialization

To communicate with the device, the application must enable and initialize at least one channel according to the following steps:

1. Unmask the following interrupts by setting the OTGFS_GINTMSK register:

- Non-periodic transmit FIFO empty for OUT transfers
- Non-periodic transmit FIFO half empty for OUT transfers

2. Unmask the interrupts of the selected channels by setting the OTGFS_HAINTMSK register
3. Unmask the transfer-related interrupts in the host channel interrupt register by setting the OTGFS_HCINTMSKx register
4. Configure the total transfer size (in bytes), and the expected number of the packets (including short packets) for the OTGFS_HCTSIZx register of the selected channel. The application must configure the PID bit according to the initial data PID (it is the PID on the first OUT transfer, or to be received from the first IN transfer)
5. Configure the transfer size to ensure that the transfer size of the channel is a multiple of the largest packet size
6. Configure the OTGFS_HCCHARx register of the selected channel according to the device endpoint characteristics such as type, speed and direction (the channel cannot be enabled by setting the enable bit until the application is ready for packet transfer or reception)

20.5.3.3 Halting a channel

The application can disable a channel by writing 0x1 to the CHDIS and CHENA bits in the OTGFS_HCCHARx register. This enables the host to refresh the summited requests (if any) and generates a channel halted interrupt. The application cannot re-allocate channels for other transactions until an interrupt is generated in the OTGFS_HCINTx register (CHHLTD bit). Those transactions that have already been started on the USB line are not interrupted by the host.

Before disabling a channel, the application must ensure that there is at least one free space available in the non-periodic request queue (when disabling a non-period channel) or the periodic request queue (when disabling a periodic channel). The application can refresh the submitted requests when the request queue is full (before disabling the channel) by setting CHDIS=0x1, and CHENA=0 in the OTGFS_HCCHARx register.

When there is a transaction input in the request queue, the controller will trigger a RXFLVL interrupt. The application must generate a channel halted interrupt through the OTGFS_GRXSTSP register.

The application is expected to abort a channel on any of the following conditions:

- When an interrupt (XFERC bit) is received in the OTGFS_HCINTx register during a non-periodic IN transfer
- When an STALL , XACTERR , BBLERR or DTGLERR interrupt in the OTGFS_HCINTx register is received for an IN or OUT channel
- When a DISCONINT (device disconnected) interrupt event is received in the OTGFS_GINTSTS register, the application must check the PRTCONSTS bit in the OTGFS_HPRT register. This is because when the device is disconnected with the host, the PRTCONSTS bit will be reset in the OTGFS_HPRT register. The application must initiate a software reset to ensure that all channels have been cleared. Once the device is reconnected, the host must start a USB reset.
- When the application needs to abort a transfer before normal completion

20.5.3.4 Queue depth

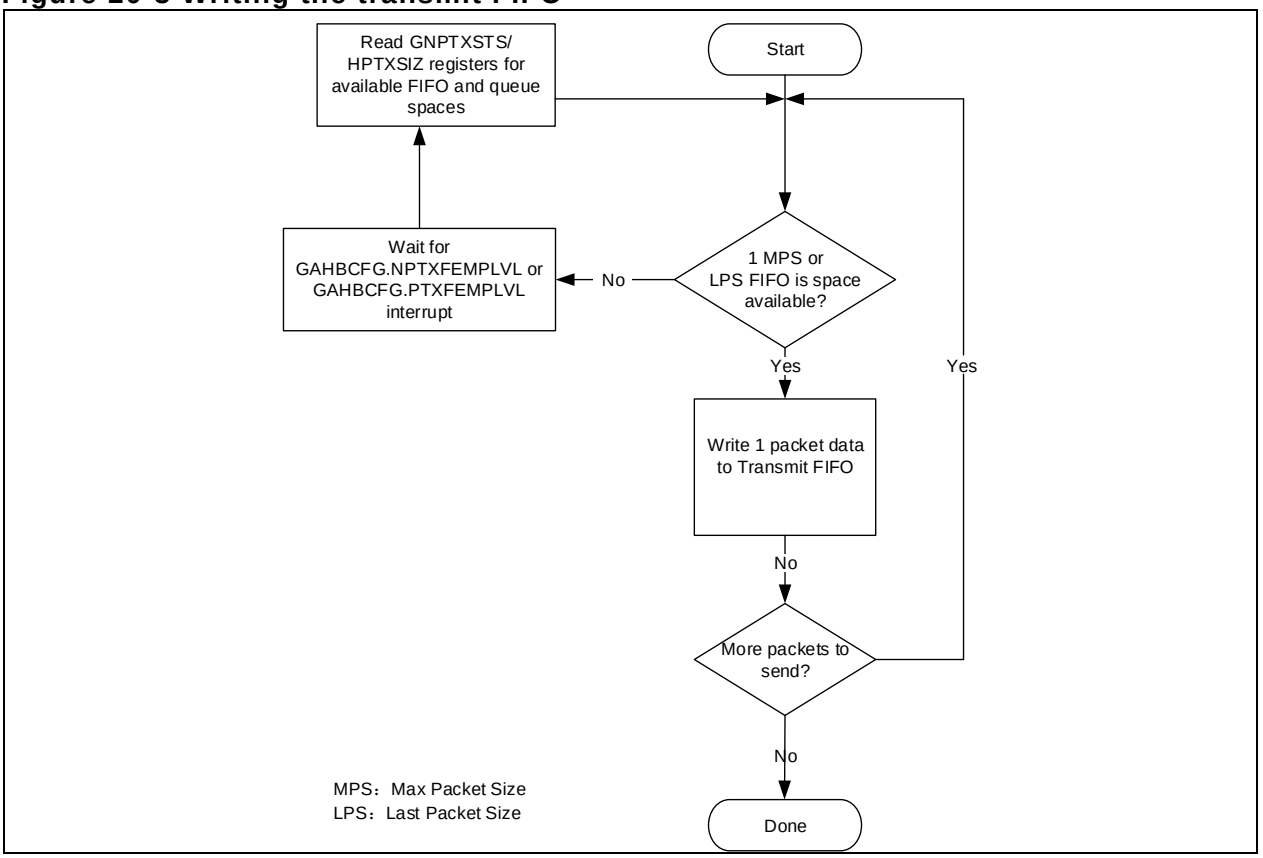
Up to 8 interrupt and synchronous transfer requests are supported in the periodic hardware transfer request queue; while up to 8 control and bulk transfer requests are allowed in the non-periodic hardware transfer request queue.

- Writing the transmit FIFO

Figure 20-3 shows the flow chart of writing the transmit FIFO. The OTGFS host automatically writes a request (OUT request) to the periodic/non-periodic request queue when writing the last one WORD packet. The application must ensure that at least one free space is available in the periodic/non-periodic

request queue before starting to write to the transmit FIFO. The application must always write to the transmit FIFO in WORDs. If the packet size is not aligned with WORD, the application must use padding. The OTGFS host determines the actual packet size according to the programmed maximum packet size and transfer size.

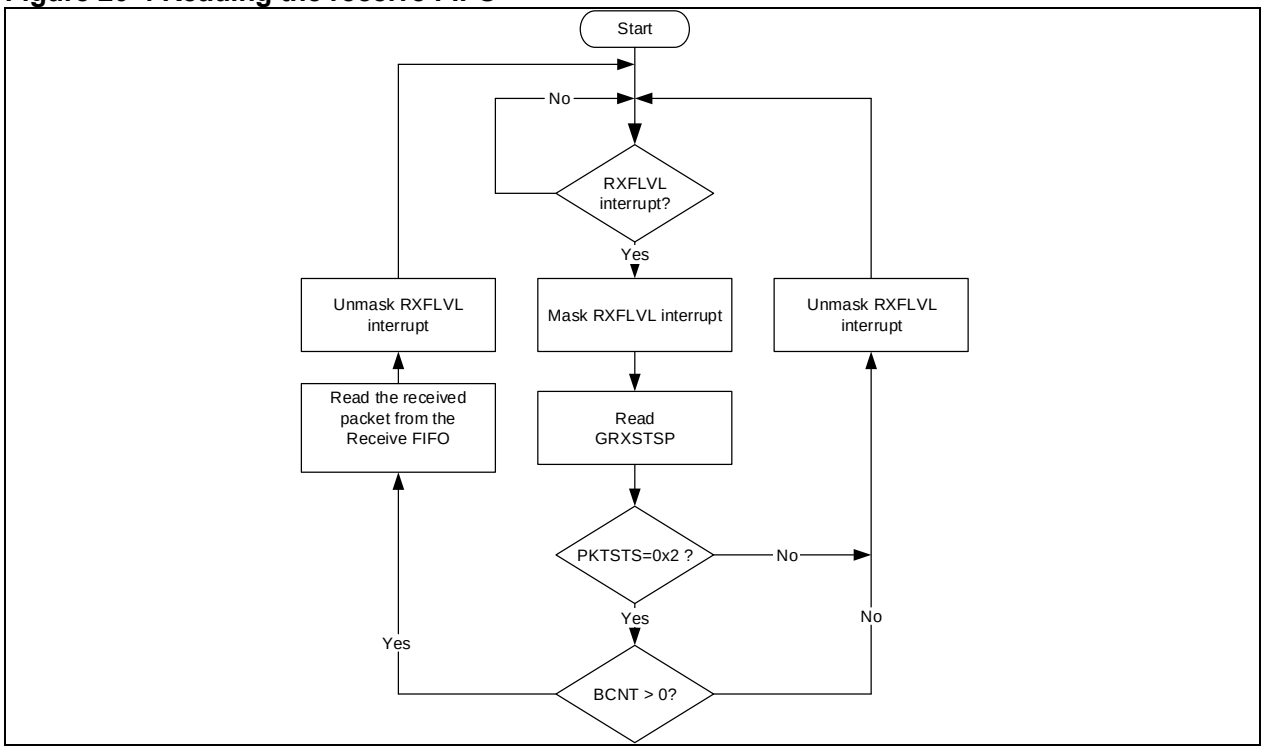
Figure 20-3 Writing the transmit FIFO



- Reading the receive FIFO

Figure 20-4 shows the flow chart of reading the receive FIFO. The application must ignore all packet statuses other than IN data packet (0x0010)

Figure 20-4 Reading the receive FIFO



20.5.3.5 Special cases

(1) Handling babble conditions

The OTGFS controller handles two cases of babble: packet babble and port babble. Packet babble occurs if the device sends more than the largest packet size for the channel. Port babble occurs if the controller continues to receive data from the device at EOF2 (the end of frame 2, which is very close to SOF)

When the OTGFS controller detects a packet babble, it stops writing data to the receiver buffer and waits for the completion of packet. When it detects the end of packet, the OTGFS flushes the data already written in the receiver buffer and generates a babble interrupt.

When the OTGFS controller detects a port babble, it flushes the receive FIFO and disables the port. Then the controller generates a Port disable interrupt. Once receiving the interrupt, the application must determine that this is not caused by an overcurrent condition (another cause of the port disable interrupt) by checking the PRTOVRCACT bit in the OTGFS_HPRT register, then perform a software reset. The controller does not send any more tokens if a port babble signal is detected.

(2) Handling device disconnected conditions

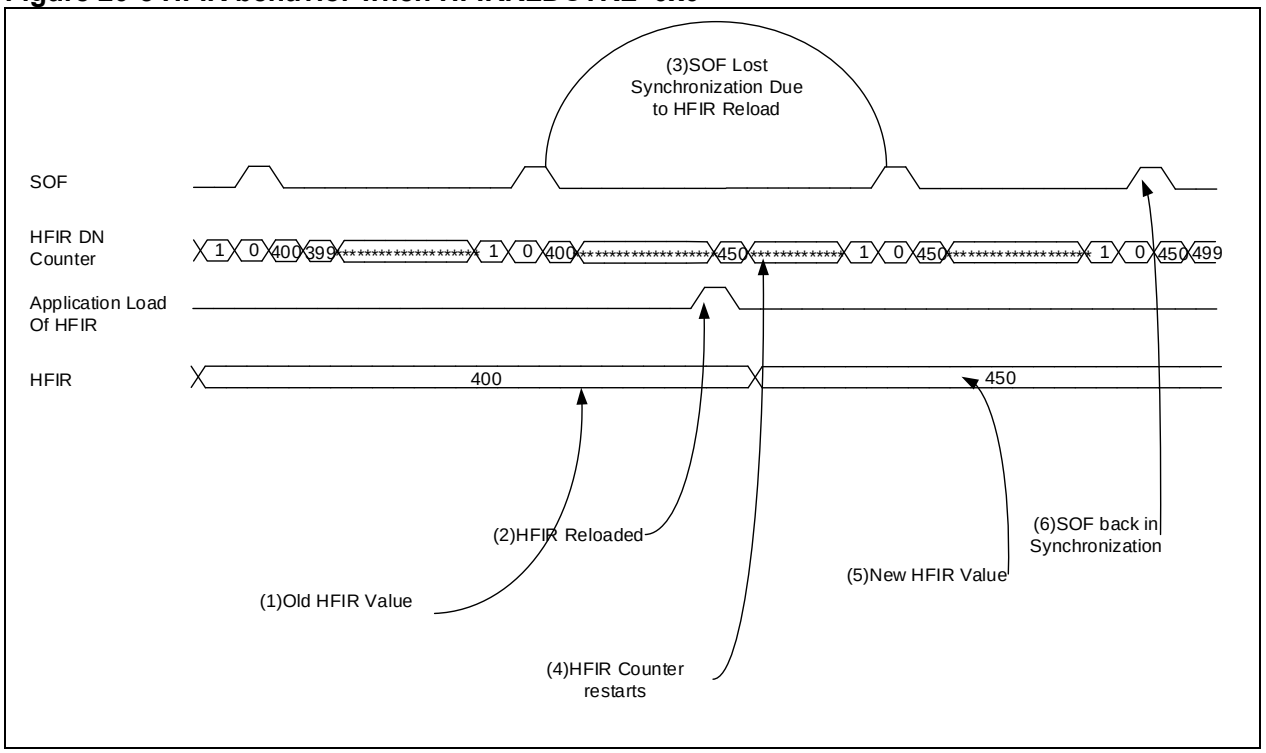
If the device is suddenly disconnected, an interrupt is generated on a disconnect event (DISCONINT bit in the OTGFS_GINTSTS register). Upon receiving this interrupt, the application must start a software reset through the CSFTRST in the OTGFS_GRSTCTL register.

20.5.3.6 Host HFIR feature

The host frame interval register (HFIR) defines an interval between two consecutive SOFs (full-speed) or Keep-Alive tokens. This field contains the number of PHY clock for the required frame interval. This is mainly used to adjust the SOF duration based on PHY clock frequencies.

Figure 20-5 shows the HFIR behavior when the HFIRRLDCTRL = 0x0 in the OTGFS_HFIR register.

Figure 20-5 HFIR behavior when HFIRRLDCTRL=0x0



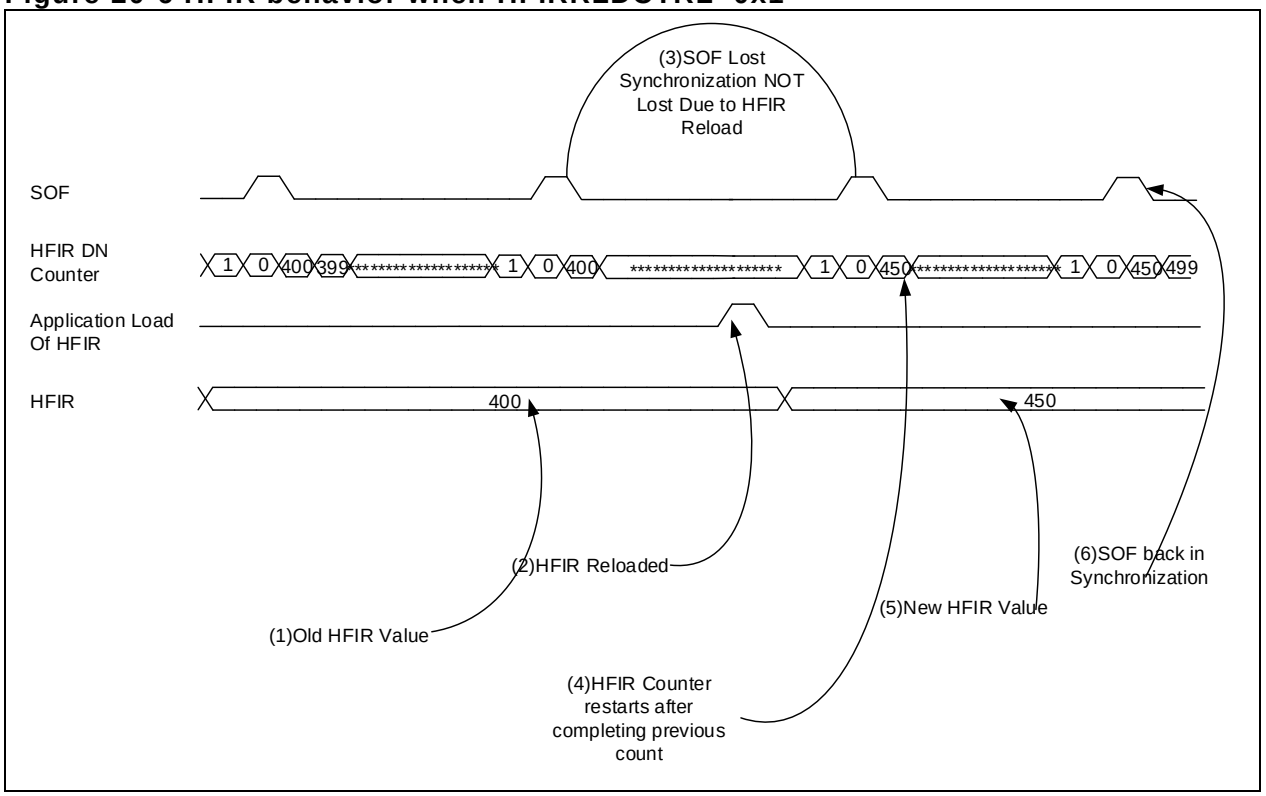
The sequence of operation is as follows:

1. After power-on reset, the current HFIR value set by the application is shown
2. The application loads a new value into the HFIR register
3. The HFIR downcounter is reloaded, so it will immediately restart counting to cause SOF synchronization loss
4. Restart HFIR counter

5. The HFIR register receives a new programmed value
6. Obtain SOF synchronization again after the first SOF is generated using the HFIR new feature

Figure 20-6 shows the HFIR behavior when HFIRRLDCTRL=0x1 in the OTGFS_HFIR register.

Figure 20-6 HFIR behavior when HFIRRLDCTRL=0x1



The sequence of operation is as follows:

1. After power-on reset, the current HFIR value set by the application is shown
2. The application loads a new HFIR value; the HFIR counter does not apply this new value, but continues counting until it reaches 0
3. The counter generates a SOF when it reaches 0 using the old HRIF value
4. The HFIR counter applies a new value
5. New HFIR value takes effect

The SOF synchronization resumes after going through above-mentioned steps.

20.5.3.7 Initialize bulk and control IN transfers

Figure 20-7 shows a typical bulk or control IN transfer operation. Refer to channel 2 (ch_2) for more information. The assumptions are as follows:

- The application is attempting to receive two largest-packet-size packets (transfer size is 64 bytes)
- The receive FIFO contains at least one largest-packet-size packet and two status WORDs per each packet (72 bytes for full-speed transfer)
- The non-periodic request queue depth is 4

(1) Operation process for common bulk and control IN transfers

The sequence of operations shown in *Figure 20-7* is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements)
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the non-periodic request queue
3. The controller issues an IN token after completing the current OUT transfer
4. The controller generates a RXFLVL interrupt as soon as the receive packet is written into the receive FIFO
5. To handle the RXFLVL interrupt, mask the RXFLVL interrupt and read the received packet status to

determine the number of bytes received, and then read the receive FIFO. Following this step to unmask the RXFLVL interrupt

6. The controller generates the RXFLVL interrupt when the transfer complete status is written into the receive FIFO
7. The application must read the receive packet status, and ignore it when the receive packet status is not an IN data packet
8. The controller generates an XFERC interrupt as soon as the receive packet is read
9. To handle an XFERC interrupt, disable the channel (see Halting a channel) and stop writing the OTGFS_HCCHAR2 register. The controller writes a channel halted request to the non-periodic request queue once the OTGFS_HCCHAR2 register is written
10. The controller generates an RXFLVL interrupt as soon as the halt status is written to the receive FIFO
11. Read and ignore the receive packet status
12. The controller generates a CHHLTD interrupt as soon as the halt status is read from the receive FIFO
13. In response to the CHHLTD interrupt, the processor does not allocate the channel for other transfers.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during bulk and control IN transfers

```
Unmask (XACTERR/XFERC/BBLERR/STALL/DATATGLERR)
if (XFERC)
    {
    Reset Error Count
    Unmask CHHLTD
    Disable Channel
    Reset Error Count
    Mask ACK
    }
else if (XACTERR or BBLERR or STALL)
    {
    Unmask CHHLTD
    Disable Channel
    if (XACTERR)
        {
        Increment Error Count
        Unmask ACK
        }
    }
else if (ChHltd)
    {
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
        {
        De-allocate Channel
        }
    else
        {
        Re-initialize Channel
        }
    }
```

```
else if (ACK)
    {
    Reset Error Count
    Mask ACK
    }
else if (DATATGLERR)
    {
    Reset Error Count
}
```

20.5.3.8 Initialize bulk and control OUT/SETUP transfers

Figure 20-7 shows a typical bulk or control transfer OUT/SETUP transfer operation. Refer to channel 1 (ch_1) for more information. It is necessary to send two bulk transfer OUT packets. The control transfer SETUP operation is the same, just the fact that it has only one packet. The assumptions are as follows:

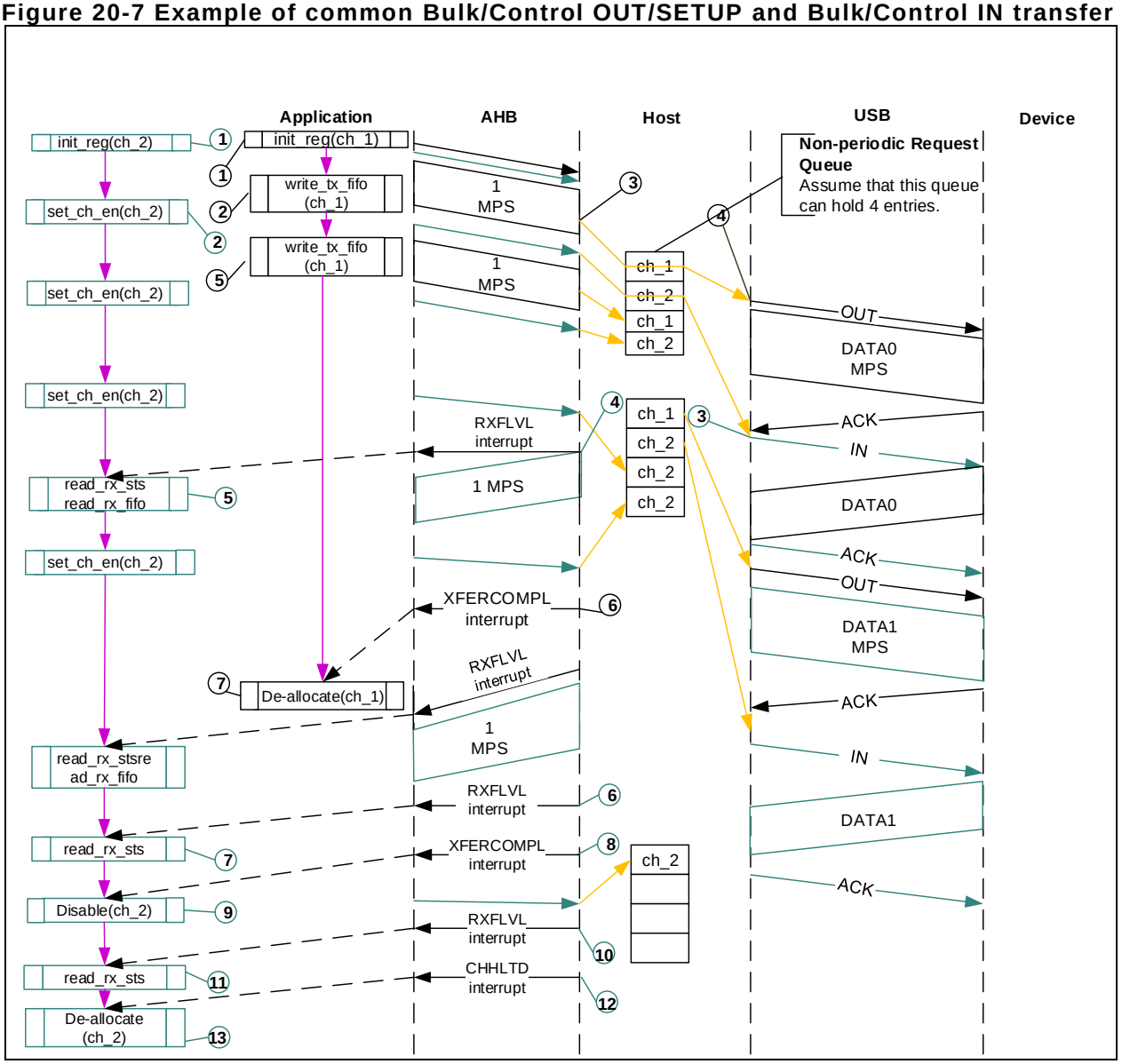
- The application is attempting to send two largest-packet-size packets (transfer size is 64 bytes)
- The non-periodic transmit FIFO can store two packets (128 bytes for full-speed transfer)
- The non-periodic request queue depth is 4

(1) OUT/SETUP operation process for common bulk and control transfer

The sequence of operations shown in *Figure 20-7* is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements)
2. Write the first packet for channel 1
3. Along with the last WORD write, the controller writes a request to the non-periodic request queue
4. The controller sends an OUT token in the current frame as soon as the non-periodic queue becomes empty
5. Write the second packet (the last one) to the channel 1
6. The controller generate an XFERC interrupt as soon as the previous transfer is completed successfully
7. In response to the XFERC interrupt, the processor does not allocate the channel for other transfers.

Figure 20-7 Example of common Bulk/Control OUT/SETUP and Bulk/Control IN transfer



(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during bulk and control transfer OUT/SETUP operation

```
Unmask (NAK/XACTERR/NYET/STALL/XFERC)
if (XFERC)
    {
    Reset Error Count
    Mask ACK
    De-allocate Channel
    }
else if (STALL)
    {
    Transfer Done = 1
    Unmask CHHLTD
    Disable Channel
    }
else if (NAK or XACTERR or NYET)
    {
    Rewind Buffer Pointers
```

```
    Unmask CHHLTD
    Disable Channel
    if (XactErr)
        {
        Increment Error Count
        Unmask ACK
        }
    else
        {
        Reset Error Count
        }
    }
else if (CHHLTD)
    {
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
        {
        De-allocate Channel
        }
    else
        {
        Re-initialize Channel (Do ping protocol for HS)
        }
    }
else if (ACK)
    {
    Reset Error Count
    Mask ACK
    }
```

Notes:

- The application can only write the transmit FIFO when the transmit FIFO and request queue has free spaces. The application must check whether there is a free space in the transmit FIFO through the NPTXFEMP bit in the OTGFS_GINTSTS register
- The application can only write a request when the request queue has free spaces and wait until an XFERC interrupt is received

20.5.3.9 Initialize interrupt IN transfers

Figure 20-8 shows the operation process of a typical interrupt IN transfer. Refer to channel 2 (ch_2). The assumptions are as follows:

- The application is attempting to receive one largest-packet-size packet (transfer size is 64 bytes) from an odd frame
- The receive FIFO can store at least one largest-packet-size packet and two status WORDs per packet (1031 bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

The sequence of operations shown in *Figure 20-8* (channel 2) is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the periodic request queue
3. The OTGFS host writes an IN request to the periodic request queue each time the CHENA is set in

the OTGFS_HCCHAR2 register

4. The OTGFS host attempts to send an IN token in the next frame (odd)
5. The OTGFS host generates a RXFLVL interrupt as soon as an IN packet is received and written to the receive FIFO
6. To handle the RXFLVL interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO. The application must mask the RXFLVL interrupt before reading the receive FIFO, and unmask the interrupt after reading the entire packet
7. The controller generates the RXFLVL interrupt when the transfer complete status is written to the receive FIFO. The application must read and ignore the receive packet when the receive packet is not an IN packet
8. The controller generates an XFERC interrupt as soon as the receive packet is read
9. To handle the XFERC interrupt, read the PKTCN bit in the OTGFS_HCTSIZ2 register. If the PKTCNT bit in the OTGFS_HCTSIZ2 is not equal to 0, disable the channel before re-initializing the channel for the next transfer. If PKTCNT == 0 in the OTGFS_HCTSIZ2 register, re-initialize the channel for the next transfer. In this case, the application must reset the ODDFRM bit in the OTGFS_HCCHAR2 register.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during interrupt IN transfer

```
Unmask (NAK/XACTERR/XFERC/BBLERR/STALL/FRMOVRUN/DATATGLERR)
if (XFERC)
  {
  Reset Error Count
  Mask ACK
  if (HCTSIZx.PKTCNT == 0)
    {
    De-allocate Channel
    }
  else
    {
    Transfer Done = 1
    Unmask CHHLTD
    Disable Channel
    }
  }
else if (STALL or FRMOVRUN or NAK or DATATGLERR or BBLERR)
  {
  Mask ACK
  Unmask CHHLTD
  Disable Channel
  if (STALL or BBLERR)
    {
    Reset Error Count
    Transfer Done = 1
    }
  else if (!FRMOVRUN)
    {
    Reset Error Count
    }
  }
else if (XACTERR)
```

```
{
Increment Error Count
Unmask ACK
Unmask CHHLTD
Disable Channel
}
else if (CHHLTD)
{
Mask CHHLTD
if (Transfer Done or (Error_count == 3))
  {
  De-allocate Channel
  }
else Re-initialize Channel (in next b_interval - 1 uF/F)
  }
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
```

The application can only write a request to the same channel when the remaining space in the request queue reaches the number defined in the MC field, before switching to other channels (if any).

20.5.3.10 Initialize interrupt OUT transfers

Figure 20-8 shows a typical interrupt OUT transfer operation. Refer to channel 1 (ch_1). The assumptions are as follows:

- The application is attempting to send one largest-packet-size pack (transfer size is 64 bytes) starting from an odd frame
- The periodic transmit FIFO can store one packet (1KB bytes for full-speed transfer)
- The periodic request queue depth is 4

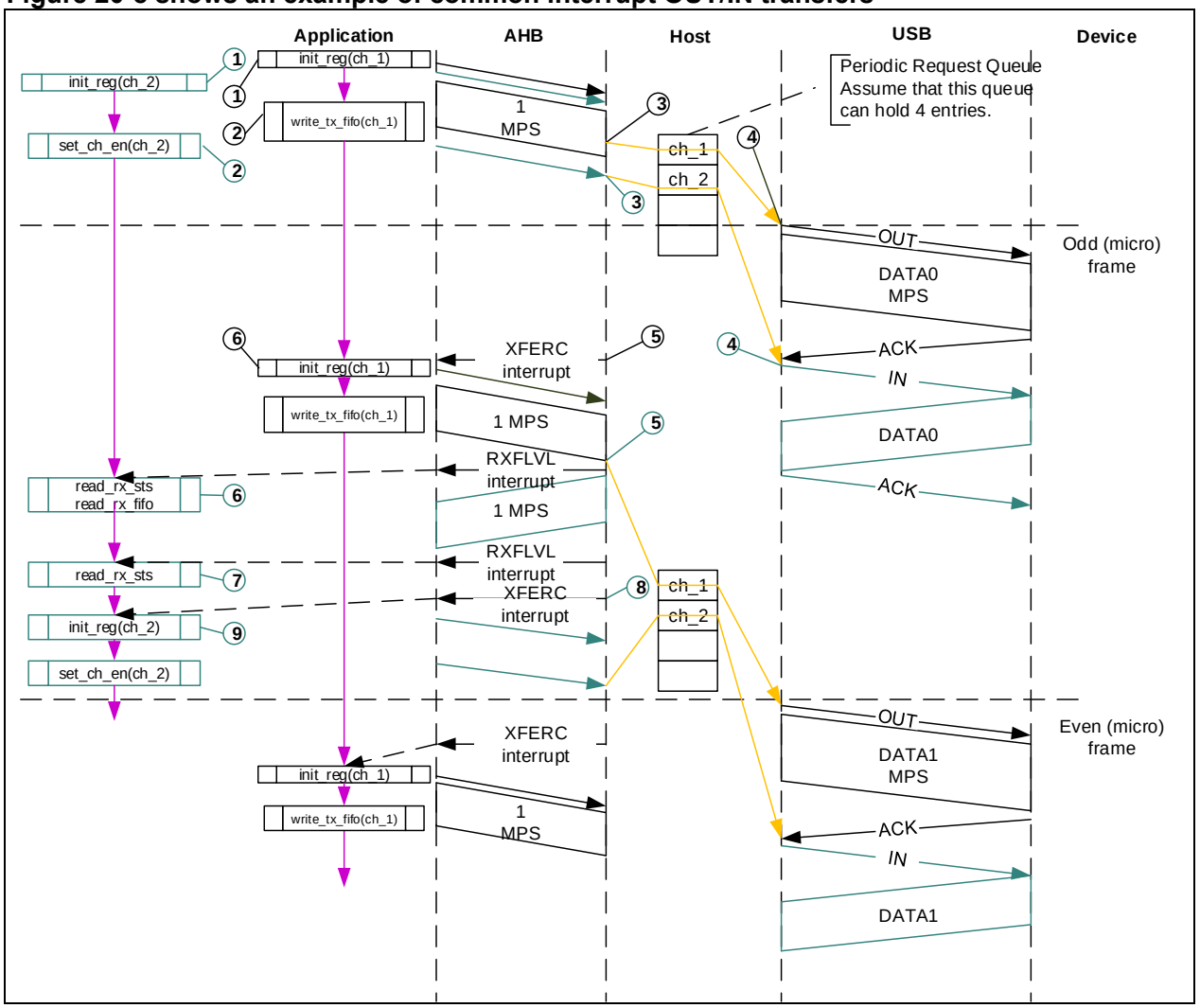
(1) Common interrupt IN operation process

The sequence of operations shown in *Figure 20-8* (channel 1) is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Write the first packet to the channel 1
3. Along with the last WORD write of each packet, the host writes a request to the periodic request queue
4. The host sends an OUT token in the next frame (odd)
5. The host generates an XFERC interrupt after the last packet is transmitted successfully
6. In response to the XFERC interrupt, re-initialize the channel for the next transfer.

(2) Handling interrupts

Figure 20-8 shows an example of common interrupt OUT/IN transfers



The following code describes the interrupt service routine related to the channel during interrupt OUT transfers

```
Unmask (NAK/XACTERR/STALL/XFERC/FRMOVRUN)
if (XFERC)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL or FRMOVRUN)
{
Mask ACK
Unmask CHHLTD
Disable Channel
if (STALL)
{
Transfer Done = 1
}
}
else if (NAK or XACTERR)
{
Rewind Buffer Pointers
```

```
Reset Error Count
Mask ACK
Unmask CHHLTD
Disable Channel
}
else if (CHHLTD)
{
Mask CHHLTD
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
```

Before switching to other channels (if any), the application can only write packets based on the number defined in the MC filed to the transmit FIFO and request queue when the transmit FIFO has free spaces. The application can determine whether the transmit FIFO has free spaces through the NPTXFEMP bit in the OTGFS_GINTSTS register.

20.5.3.11 Initialize synchronous IN transfers

Figure 20-9 shows the operation process of a typical synchronous IN transfer. Refer to channel 2 (ch_2). The assumptions are as follows:

- The application is attempting to receive one largest-packet-size packet (transfer size is 1023 bytes), starting from the next odd frame
- The receive FIFO can store at least one largest-packet-size packet and two status WORDs per packet (1031 bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

The sequence of operations shown in *Figure 20-9* (channel 2) is as follows:

1. Initialize channel 2 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Set the CHENA bit in the OTGFS_HCCHAR2 register to write an IN request to the periodic request queue
3. The OTGFS host writes an IN request to the periodic request queue each time the CHENA is set in the OTGFS_HCCHAR2 register
4. The OTGFS host attempts to send an IN token in the next frame (odd)
5. The OTGFS host generates a RXFLVL interrupt as soon as an IN packet is received and written to the receive FIFO
6. To handle the RXFLVL interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO. The application must mask the RXFLVL interrupt before reading the receive FIFO, and unmask the interrupt after reading the entire packet
7. The controller generates the RXFLVL interrupt when the transfer complete status is written to the receive FIFO. The application must read and ignore the receive packet when the receive packet is

not an IN packet (GRXSTSR.PKTSTS!= 0x0010)

8. The controller generates an XFERC interrupt as soon as the receive packet is read
9. To handle the XFERC interrupt, read the PKTCN bit in the OTGFS_HCTSIZ2 register. If the PKTCNT bit in the OTGFS_HCTSIZ2 is not equal to 0, disable the channel before re-initializing the channel for the next transfer. If PKTCNT == 0 in the OTGFS_HCTSIZ2 register, re-initialize the channel for the next transfer. In this case, the application must reset the ODDFRM bit in the OTGFS_HCCHAR2 register.

(2) Handling interrupts

The following code describes the interrupt service routine related to the channel during synchronous IN transfers

```
Unmask (XACTERR/XFERC/FRMOVRUN/BBLERR)
if (XFERC or FRMOVRUN)
  {
  if (XFERC and (HCTSIZx.PKTCNT == 0))
    {
    Reset Error Count
    De-allocate Channel
    }
  else
    {
    Unmask CHHLTD
    Disable Channel
    }
  }
else if (XACTERR or BBLERR)
  {
  Increment Error Count
  Unmask CHHLTD
  Disable Channel
  }
else if (CHHLTD)
  {
  Mask CHHLTD
  if (Transfer Done or (Error_count == 3))
    {
    De-allocate Channel
    }
  else
    {
    Re-initialize Channel
    }
  }
```

20.5.3.12 Initialize synchronous OUT transfers

Figure 20-9 shows a typical synchronous OUT transfer operation. Refer to channel 1 (ch_1). The assumptions are as follows:

- The application is attempting to send one largest-packet-size packet (transfer size is 1023 bytes) to every frame from the next odd frame
- The periodic transmit FIFO can store one packet (1KB bytes for full-speed transfer)
- The periodic request queue depth is 4

(1) Common interrupt IN operation process

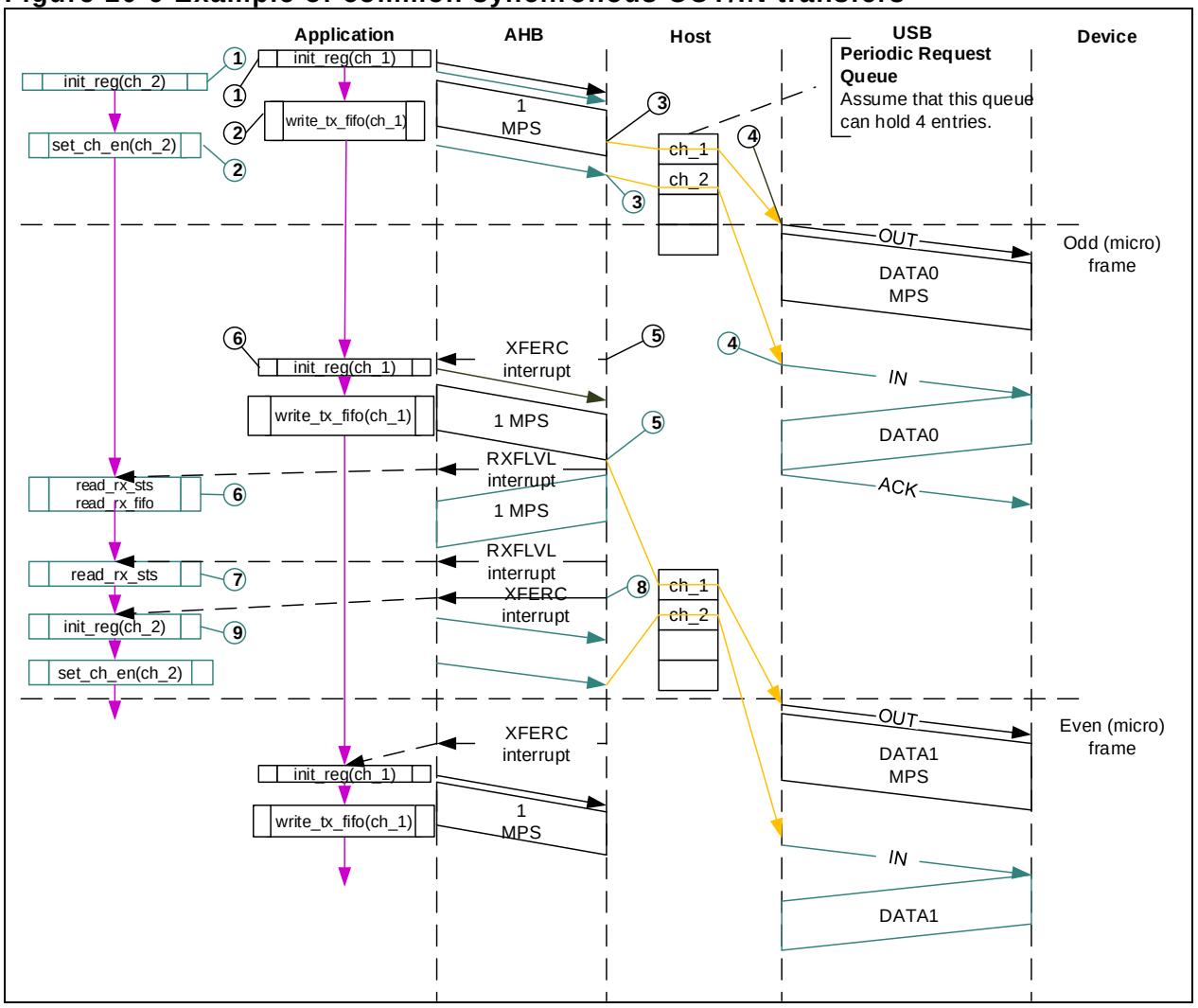
The sequence of operations shown in *Figure 20-9* (channel 2) is as follows:

1. Initialize channel 1 (according to OTGFS channel initialization requirements). The application must set the ODDFRM bit in the OTGFS_HCCHAR2 register
2. Write the first packet to the channel 1
3. Along with the last WORD write of each packet, the host writes a request to the periodic request queue
4. The OTGFS host sends an OUT token in the next frame (odd)
5. The host generates an XFERC interrupt after the last packet is transmitted successfully
6. In response to the XFERC interrupt, re-initialize the channel for the next transfer.

(2) Handling interrupts

Figure 20-9 shows an example of common synchronous OUT transfers

Figure 20-9 Example of common synchronous OUT/IN transfers



The following code describes the interrupt service routine related to the channel during synchronous OUT transfers

```
Unmask (FRMOVRUN/XFERC)
if (XFERC)
  {
  De-allocate Channel
  }
else if (FRMOVRUN)
  {
  Unmask CHHLTD
  Disable Channel
```

```
  }
else if (CHHLTD)
  {
  Mask CHHLTD
  De-allocate Channel
  }
```

20.5.4 OTGFS device mode

20.5.4.1 Device initialization

The application must perform the following steps to initialize the controller during power-on or after switching a mode from host to device:

1. Program the following fields in the OTGFS_DCFG register
- Device speed
- Non-zero-length status OUT handshake
- Periodic frame interval
2. Clear the SFTDISCON bit in the OTGFS_DCTL register. The controller will start connection after clearing this bit
3. Program the OTGFS_GINTMSK register to unmask the following interrupts:
- USB reset
- Enumeration done
- Early suspend
- USB suspend
- SOF
4. Wait for the USBRESET interrupt in the OTGFS_GINTSTS register. It indicates that a reset signal has been detected on the USB (lasting for about 10ms). Upon receiving this interrupt, the application must follow the steps defined in USB initialization on USB reset.
5. Wait for the ENUMDONE interrupt in the OTGFS_GINTSTS register. It indicates the end of USB reset. Upon receiving this interrupt, the application must read the OTGFS_DSTS register to determine the enumeration speed and perform the steps defined in Endpoint initialization on enumeration completion. At this time, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

20.5.4.2 Endpoint initialization on USB reset

This section describes the operations required for the application to perform when a USB reset signal is detected:

1. Set the NAK bit for all OUT endpoints
- OTGFS_DOEPCTLx.SNAK = 0x1(for all OUT endpoints)
2. Unmask the following interrupt bits
- OTGFS_DAINTMSK.INEP0 = 0x1(control IN endpoint 0)
- OTGFS_DAINTMSK.OUTEP0 = 0x1(control OUT endpoint 0)
- OTGFS_DOEPMSK.SETUP = 0x1
- OTGFS_DOEPMSK.XFERC = 0x1
- OTGFS_DIEPMSK.XFERC = 0x1
- OTGFS_DIEPMSK.TIMEOUT = 0x1
3. To receive/transmit data, the device must perform Device initialization steps to initialize registers
4. Allocate SRAM for each endpoint
- Program the OTGFS_GRXFSIZ register to be able to receive control OUT data and SETUP data. If the allocated SRAM is equal to at least 1 largest-packet-size of control endpoint 0 + 2 WORDs (for the status of the control OUT data packet) +10 WORDs (for setup packets)

- Program the OTGFS_DIEPTXF0 register to be able to transmit control IN data. The allocated SRAM is equal to at least 1 largest-packet-size of control endpoint 0

5. Reset the device address in the device configuration register

6. Program the following fields in the endpoint-specific registers to ensure that control OUT endpoint 0 is able to receive a SETUP packet

- OTGFS_DOEPTSIZ0.SUPCNT = 0x3(to receive up to 3 consecutive SETUP packets)

At this point, all initialization required to receive SETUP packets is done.

20.5.4.3 Endpoint initialization on enumeration completion

This section describes the operations required for the application to perform when an enumeration completion interrupt signal is detected:

- Upon detecting the enumeration completion interrupt signal, read the OTGFS_DSTS register to get the enumeration speed
- Program the MPS bit in the OTGFS_DIEPCTL0 register to set the maximum packet size. This operation is used to configure control endpoint 0. The maximum packet size for a control endpoint depends on the enumeration speed
- Unmask SOF interrupts.

At this point, the device is ready to receive SOF packets and has been configured to perform control transfers on control endpoint 0.

20.5.4.4 Endpoint initialization on SetAddress command

This section describes the operations required for the application to perform when the application receives a SetAddress command in a SETUP packet

- Program the OTGFS_DCFG register with the device address received in the SetAddress command
- Program the controller to send an IN packet

20.5.4.5 Endpoint initialization on SetConfiguration/SetInterface command

This section describes the operations required for the application to perform when the application receives a SetConfiguration / SetInterface command in a SETUP packet.

- When a SetConfiguration command is received, the application must program the endpoint registers according to the characteristics of the valid endpoints defined in the new configuration
- When a SetInterface command is received, the application must program the endpoint registers of the endpoints affected by this command
- Some endpoints that were valid in the previous configuration are not valid in the new configuration. These invalid endpoints must be disabled
- Refer to Endpoint activation and USB endpoint deactivation for more information on how to activate or disable a certain endpoint
- Unmask the interrupt for each valid endpoint and mask the interrupts for all invalid endpoints in the DAINTMSK register
- Refer to OTGFS FIFO configuration for more information on how to program SRAM for each FIFO
- After all required endpoints are configured, the application must program the controller to send a status IN packet

At this point, the device controller has been ready to receive and transmit any type of data packet.

20.5.4.6 Endpoint activation

This section describes how to activate a device endpoint or configure an existing device endpoint to a new type.

1. Program the following bits in the OTGFS_DIEPCTLx register (for IN or bidirectional endpoints) or the OTGFS_DOEPCTLx register (for OUT or bidirectional endpoints)

- Largest packet size
- USB valid endpoint = 0x1
- Endpoint start data toggle (for interrupt and bulk endpoints
- Endpoint type
- Transmit FIFO number

2. Once the endpoint is activated, the controller starts decoding the tokens issued to this endpoint and sends out a valid handshake for each valid token received for the endpoint

20.5.4.7 USB endpoint deactivation

This section describes how to deactivate an existing endpoint. Disable the suspended transfer before performing endpoint deactivation.

- Clear the USB valid endpoint bit in the OTGFS_DIEPCTLx register (for IN or bidirectional endpoints) or the OTGFS_DOEPCTLx register (for OUT or bidirectional endpoints)
- Once the endpoint is deactivated, the controller will ignore the tokens issued to this endpoint, which causes a USB timeout.

20.5.4.8 Control write transfers (SETUP/Data OUT/Status IN)

This section describes the steps required for control write transfers.

The application programming process is as follows:

1. When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet
2. If the last SETUP packet received before the generation of the SETUP interrupt indicates data OUT stage, program the controller to perform OUT transfers based on Asynchronous OUT data transfer operation
3. The application can receive up to 64-byte data for a single OUT data transfer of control endpoint 0. If the application expects to receive more than 64-byte data during data OUT stage, it must re-enable the endpoint to receive another 64-byte data, and it must continue this operation until the completion of all data reception in data stage
4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register during the last OUT transfer, it indicates the end of data OUT stage of control transfer
5. Once the completion of data OUT stage, the application must perform the following steps:

- If the application needs to transfer a new SETUP packet, it must re-enable control OUT endpoints (refer to OUT data transfers)

  OTGFS_DOEPCTLx.EPENA = 0x1
- To execute the received SETUP commands, the application must configure the corresponding registers in the controller. This is optional, depending on the received SETUP command type

6. During status IN stage, the application must follow the requirements of Non-periodic (for bulk and control) IN data transfers to program registers to perform data IN transfers
7. When the XFERC interrupt is set in the OTGFS_DOEPINTx register is set, it indicates that the status stage of control transfers is started. As soon as Data transfer complete mode and Status stage start bit are set in the receive FIFO packet status register, the controller generates an interrupt. The Transfer complete interrupt can be cleared through the XFERC bit in the OTGFS_DOEPINTx register

   Repeat above-mentioned steps until an interrupt (XFERC bit in the OTGFS_DIEPINTx register) is generated on the endpoint, which indicates the end of control write transfers.

20.5.4.9 Control read transfers (SETUP/Data IN/Status OUT)

This section describes the steps required for control read transfers.

The application programming process is as follows:

- When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the

end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet

- If the last SETUP packet received before the generation of the SETUP interrupt indicates data IN stage, program the controller to perform IN transfers based on Non-periodic IN data transfer operation
- The application can receive up to 64-byte data for a single IN data transfer of control endpoint 0. If the application expects to receive more than 64-byte data during data IN stage, it must re-enable the endpoint to receive another 64-byte data, and it must continue this operation until the completion of all data transfers in data stage
- Repeat above-mentioned steps until the XFERC interrupt is generated in the OTGFS_DIEPINTx register for each IN transfer on the endpoint
- When the XFERC interrupt is set in the OTGFS_DOEPINTx register during the last IN transfer, it indicates the end of data OUT stage of control transfer
- To execute data OUT transfer at status OUT stage, the application must configure the controller. This is optional.

The application must program the NZSTSOUTHSHK bit in the OTGFS_DCFG register, and then send data OUT transfer at status stage

The XFERC interrupt bit is set in the OTGFS_DOEPINTx register to indicate the end of status OUT stage of control transfer, marking the completion of control read transfers.

20.5.4.10 Control transfers (SETUP/Status IN)

This section describes the two-phase control transfer operation..

The application programming process is as follows:

1. When the SETUP bit is set in the OTGFS_DOEPINTx register, it indicates that a valid SETUP packet has been sent to the application, and data stage is initiated, see OUT data transfers. At the end of the SETUP stage, the application must rewrite 3 to the SUPCNT bit in the OTGFS_DOEPTSIZx register to receive the subsequent SETUP packet
2. The application decodes the last SETUP packet received before the generation of the SETUP interrupt. If the SETUP packet indicates two-level control commands, the application must perform the following steps:

- Set OTGFS_DOEPCTLx.EPENA = 0x1
- The application must program the registers in the controller to perform the received SETUP commands

3. For status IN stage, the application must program the registers based on Non-periodic (bulk and control) IN data transfers to perform data IN transfers
4. The XFERC interrupt bit is set in the OTGFS_DIEPINTx register to indicate the end of status IN stage of control transfers.

20.5.4.11 Read FIFO packets

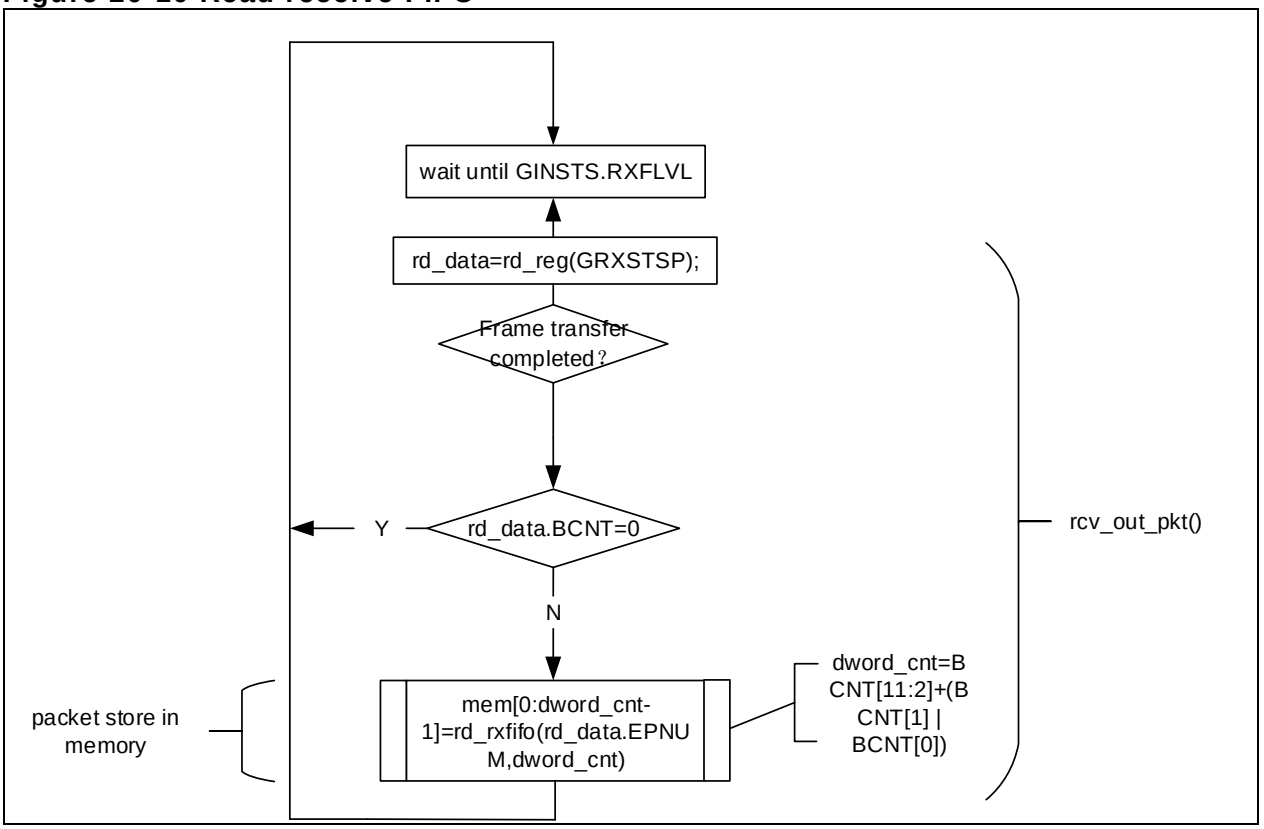
This section describes how to read FIFO packets (OUT data and SETUP packets)

1. The application must read the OTGFS_GRXSTSP register as soon as the RXFLVL interrupt bit is detected in the OTGFS_GINTSTS register
2. The application can mask the RXFLVL interrupt bit in the OTGFS_GINTSTS register by setting RXFLVL = 0x0 in the OTGFS_GINTMSK register, until it has read the data packets from the receive FIFO
3. If the received packet byte is not 0, the byte count amount of data is popped from the receive data FIFO and stored in memory. If the received packet byte count is 0, no data is read from the receive data FIFO
4. The receive FIFO packet status reading indicates one of the following conditions:
5. Global OUT NAK mode: PKTSTS = Global OUT NAK, BCNT = 0x000, EPNUM = Don’t Care (0x0) and DPID = Don’t Care (0x00), indicating that the global OUT NAK bit has taken effect

- SETUP packet mode: PKTSTS = SETUP, BCnt = 0x008, EPNUM = Control EP Num and DPID = D0, indicating that a SETUP packet for the specified endpoint is now available for reading from the receive FIFO
- Setup stage done mode: PKTSTS = Setup Stage Done, BCNT = 0x0, EPNUM = Control EP Num and DPID = Don't Care (0x00), indicating the completion of the Setup stage for the specified endpoint, and the start of the data stage. After this request is popped from the receive FIFO, the controller triggers a Setup interrupt on the specified control OUT endpoint
- Data OUT packet mode: PKTSTS = DataOUT, BCnt =size of the received data OUT packet (0 ≤ BCNT ≤ 1024), EPNUM =Endpoint number on which the data packet was received, DPID =Actual data PID
- Data transfer complete mode: PKTSTS = Data OUT transfer done, BCNT = 0x0, EPNUM =OUT endpoint number on which the data transfer is complete, DPID = Don't Care (0x00). These data indicate that an OUT data transfer for the specified OUT endpoint has been complete. After this request is popped from the receive FIFO, the controller triggers a Transfer Completed interrupt on the specified OUT endpoint. PKTSTS code can be found in the OTGFS_GRXSTSR / OTGFS_GRXSTSP register

6. After the valid data is popped from the receive FIFO, the RXFLVL interrupt bit in the OTGFS_GINTSTS register must be unmasked
7. Step 1-5 must be repeated each time the application detects the interrupt line due to the RXFLVL bit in the OTGFS_GINTSTS register. Reading an empty receive FIFO will result in unexpected behavior. *Figure 20-10* shows a flowchart.

Figure 20-10 Read receive FIFO



20.5.4.12 OUT data transfers

This section describes the internal data flow during data OUT and SETUP transfers, and how the application handles SETUP transfers.

(1) Setup transfers

This section describes how to handle SETUP data packets and the application's operating sequence of handling SETUP transfers. After power-on reset, the application must follow the OTGFS Initialization process to initialize the controller. Before communicating with the host, the application must initialize the endpoints based on Device Initialization, and refer to Read FIFO packets for more information.

[Application requirements]

1. To receive a SETUP packet, the SUPCNT bit (OTGFS_DOEPTSIZx) on a control OUT endpoint must be programmed to be a non-zero value. When the application sets the SUPCNT bit to a non-zero value, the controller receives SETUP packets and writes them to the receive FIFO, irrespective of the NAK status bit and EPENA bit in the OTGFS_DOEPCTLx register. The SUPCNT bit is decremented each time the control endpoint receives a SETUP packet. If the SUPCNT bit is not programmed to a proper value before receiving a SETUP packet, the controller still receives the SETUP packet and decrements the SUPCNT bit, but the application may not be able to determine the exact number of SETUP packets received in the SETUP stage of a control transfer.

- OTGFS_DOEPTSIZx.SUPCNT = 0x3

2. The application must allocate some extra space for the receive data FIFO to ensure that up to three SETUP packets can be received on a control endpoint

- The space to be reserved is 13 WORDs. Four WORDs are required for one SETUP packet, one WORD is required for the Setup stage and 8 WORDs are required to store two extra SETUP packets among all control endpoints
- Four WORDs per SETUP packet are required to store 8-byte SETUP data and 4-byte Transfer completed status and 4-byte SETUP status (SETUP packet mode). The controller must reserve this space to receive data
- FIFO is used to write SETUP data only, and never for data packets

3. The application must read 2-WORDs SETUP packet from the receive data
4. The application must read and discard the Transfer Completed status WORD from the receive FIFO, and ignore the Transfer Completed interrupt due to this read operation.

[Internal data flow]

1. When a SETUP packet is received, the controller writes the received data to the receive FIFO, without checking whether there is available space in the receive FIFO, irrespective of the NAK and Stall bits on the control endpoints.

- The controller sets the IN NAK and OUT NAK bits for the control IN/OUT endpoints on which the SETUP packet was received.

2. For every SETUP packet received on the USB line, 3 WORDs of data are written to the receive FIFO, and the SUPCNT bit is decremented by 1 automatically.

- The first WORD contains control information used internally by the controller
- The second WORD contains the first 4 bytes of the SETUP command
- The third WORD contains the last 4 bytes of the SETUP command

3. When the SETUP stage switches to data IN/OUT stage, the controller writes a SETUP status done WORD to the receive FIFO, indicating the end of the SETUP stage.
4. The application reads the SETUP packages through the AHB bus.
5. When the application pops the Setup stage done WORD from the receive FIFO, the controller interrupts the application through the SETUP interrupt bit in the OTGFS_DOEPINTx register, indicating that the application can start processing the SETUP packet received.
6. The controller clears the endpoint enable bit for control OUT endpoints.

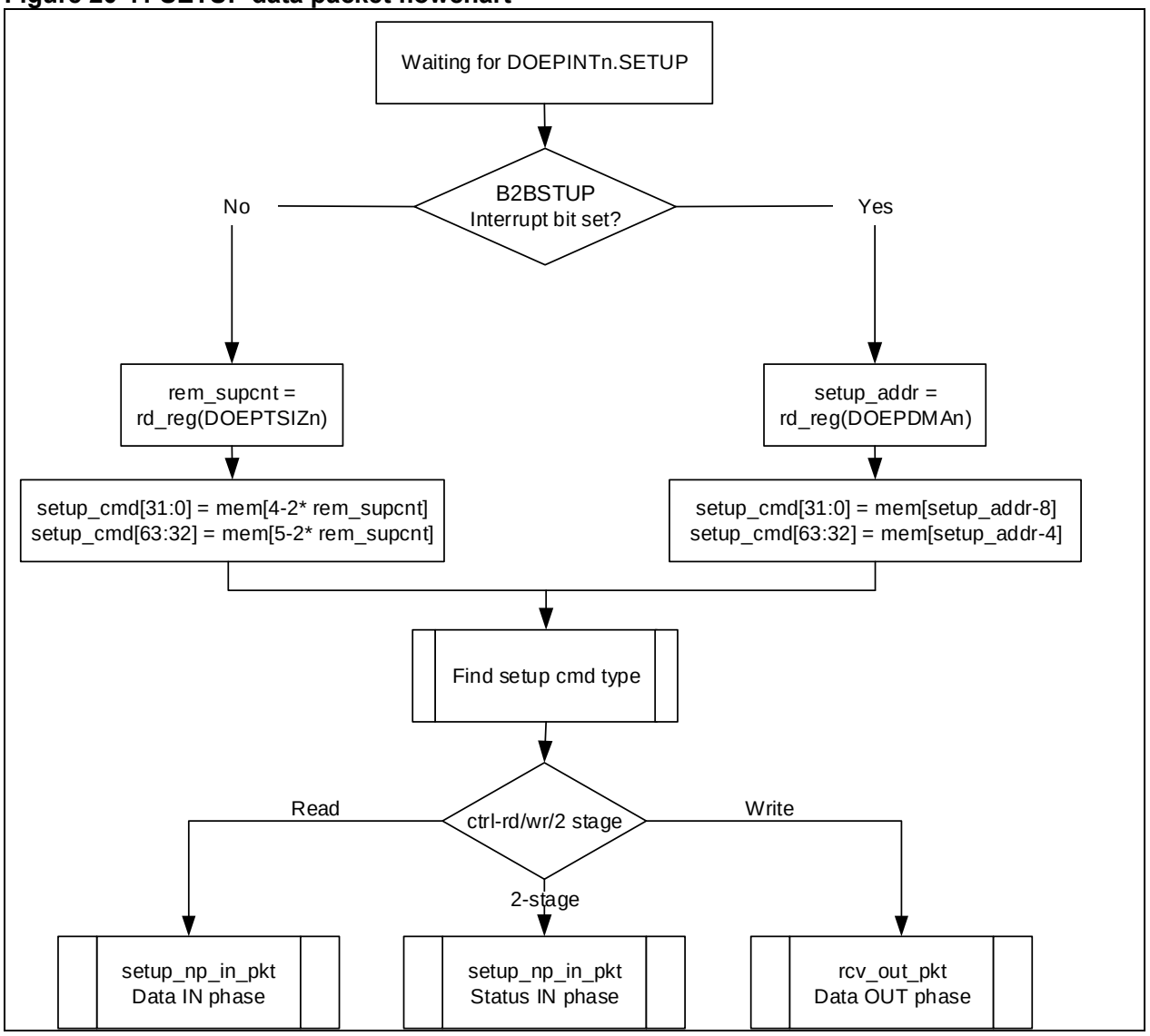
[Application programming process]

1. Program the OTGFS_DOEPTSIZx register

- OTGFS_DOEPTSIZx.SUPCNT = 0x3

2. Wait for the RXFLVL interrupt bit in the OTGFS_GINTSTS register and read and empty the data packets from the receive FIFO (Refer to Read FIFO packets for details). This operation can be repeated several times.
3. When the SETUP interrupt bit is set in the OTGFS_DOEPINTx register, it indicates that the SETUP data transfer has been completed successfully. Upon this interrupt, the application must read the OTGFS_DOEPTSIZx register to determine the number of SETUP packets received, and process the last received SETUP packet.

Figure 20-11 SETUP data packet flowchart



(2) Handling more than three consecutive SETUP packets

Per the USB 2.0 specification, typically, a host does not send more than three consecutive SETUP packets to the same endpoint during a SETUP packet error. However, the USB2.0 specification does not limit the number of consecutive SETUP packets a host can send to the same endpoint. If this condition occurs, the OTGFS controller generates an interrupt (B2BSTUP bit in the OTGFS_DOEPINTx register).

20.5.4.13 IN data transfers

This section describes the internal data flow during IN data transfers and how the application handles IN data transfers.

1. The application can either select a polling or an interrupt mode.

- In polling mode, the application monitors the status of the endpoint transmit data FIFO by reading the OTGFS_DTXFSTSx register to determine whether there is enough space in the data FIFO.
- In interrupt mode, the application must wait for the TXFEMP interrupt bit in the OTGFS_DIEPINTx register, and then read the OTGFS_DTXFSTSx register to determine whether there is enough space in the data FIFO.
- To write a single non-zero-length data packet, there must be enough space to write the entire data packet in the data FIFO.
- To write zero-length data packet, the application does not need to check the FIFO space.

2. Either way, when the application determines that there is enough space to write a transmit packet, the

application can first write into the endpoint control register before writing the data into the data FIFO. Normally, except for setting the endpoint enable bit, the application must do a read modify write on the OTGFS_DIEPCTLx register to avoid modifying the contents of the register. If the space is enough, the application can write multiple data packets for the same endpoint into the transmit FIFO. For the periodic IN endpoints, the application must write packets for only one frame. It can write packets for the next periodic transfer only after the previous transfer has been completed.

20.5.4.14 Non-periodic (bulk and control) IN data transfers

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS initialization. Before communicating with a host, the controller must follow the steps defined in Device initialization to initialize endpoints.

[Application requirements]

1. For IN transfers, the Transfer Size bit in the Endpoint Transfer Size register indicates a payload that contains multiple largest-packet-size packets and a short packet. This short packet is transmitted at the end of the transfer.

- To transmit several largest-packet-size packets and a short packet:

  Transfer size [epnum] = n * mps[epnum] + sp (n is an integer ≥ 0 and 0 ≤ sp < mps[epnum])

  If (sp > 0), then packet count [epnum] = n + 1. Otherwise, packet count [epnum] = n
- To transmit a single zero-length data packet:

  Transfer size [epnum] = 0x0

  Packet count [epnum] = 0x1
- To transmit several largest-packet-size packets and a zero-length data packet (at the end of the transfer), the application must split the transfer into two parts. First send the largest-packet-size packets and then the zero-length data packet alone.

  First transfer: Transfer size [epnum] = n * mps[epnum]; Packet count = n;

  Second transfer: Transfer size [epnum] = 0x0; Packet count = 0x1;

2. If an endpoint is enabled for data transfers, the controller updates the Transfer size register. At the end of the IN transfer (indicated by endpoint disable interrupt bit), the application must read the Transfer size register to determine how much data in the transmit FIFO have already been sent on the USB line.
3. Data fetched in the transmit FIFO = Application-programmed initial transfer size – Controller-updated final transfer size

- Data transmitted on USB = (Application-programmed initial packet count – Controller-updated final packet count) * mps[epnum]
- Data to be transmitted on USB = Application-programmed initial transfer size – Data transmitted on USB

[Internal data flow]

1. The application must set the transfer size and packet count bits in the endpoint control registers and enable the endpoint to transmit the data.
2. The application must also write the required data to the transmit FIFO of the endpoint.
3. Each time a data packet is sent to the transmit FIFO by the application the transfer size for this endpoint is decremented with the packet size. The application must continue to write data until the transfer size of the endpoint becomes 0. After writing data to the FIFO, the “packet count in the FIFO” is incremented (this is a 3-bit count for each IN endpoint transmit FIFO data packet, which is internally maintained by the controller. For an IN endpoint FIFO, the maximum number of packets maintained by the controller at any time is 8). For non-zero-length packets, a separate flag is set for each FIFO, without any data in the FIFO.
4. After the data is written to the transmit FIFO, the controller reads them upon receiving an IN token. For each non-synchronous IN data packet transmitted with an ACK handshake signal, the number of packets for the endpoint is decremented by 1, until the packet count becomes 0. The packet count is not decremented due to a timeout.
5. For zero-length data packets (indicated by an internal zero-length flag), the controller sends zero-

length packets according to the IN token, and the packet count is decremented automatically.

6. If there are no data in the FIFO on a received IN token and the packet count for the endpoint is 0, the controller generates an “IN token received when FIFO is empty” interrupt, and the NAK bit for the endpoint is not set. The controller responds with a NAK handshake signal to the non-synchronous endpoints on the USB.
7. The controller rewinds the FIFO pointers internally and no timeout interrupt is generated except for the control IN endpoints.
8. When the transfer size is 0 and the packet count is also 0, the Transfer completed interrupt is generated and the endpoint enable bit is cleared.

[Application programming sequence]

1. Program the OTGFS_DIEPTSIZx register according to the transfer size and the corresponding packet count.
2. Program the OTGFS_DIEPCTLx register according to the endpoint characteristics and set the CNAK and endpoint enable bits.
3. While sending non-zero-length data packets, the application must poll the OTGFS_DTXFSTSx register (where n is the FIFO number related to that endpoint) to determine whether there is enough space in the data FIFO. The application can also use the TXFEMP bit in the OTGFS_DIEPINTx register before writing data.

20.5.4.15 Non-synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in “OTGFS Initialization”. Before communicating with a host, the application must initialize endpoints based on the process described in “Endpoint Initialization” and by referring to “Read FIFO packets”. This section describes a regular non-synchronous OUT transfers (control, bulk or interrupt transfers).

[Application requirements]

1. For OUT data transfers, the transfer size of the endpoint transfer register must be set to a multiple of the largest packet size for the endpoint, and adjusted to the WORD boundary.

```
if (mps[epnum] mod 4) == 0
transfer size[epnum] = n * (mps[epnum]) //WORD Aligned
else
transfer size[epnum] = n * (mps[epnum] + 4 - (mps[epnum] mod 4)) //Non WORD
Aligned
packet count[epnum] = n
n > 0
```

2. When an OUT endpoint interrupt occurs, the application must read the endpoint’s transfer size register to calculate the size of the data in the memory. The received payload size must be less than the programmed transfer size.

- Payload size in memory = Application-programmed initial transfer size – Controller-updated final transfer size
- Number of USB packets the payload was received = Application-programmed initial packet count – Controller-updated final packet count

[Internal data flow]

1. The application must set the transfer size and packet count bits in the endpoint control registers, clear the NAK bit, and enable the endpoint to receive the data.
2. Once the NAK bit is cleared, the controller starts receiving data and writes it to the receive FIFO as long as there is available space in the receive FIFO. For each data packet received on the USB line, the data packet and its status are written to the receive FIFO. The packet count is decremented by 1 each time a packet (largest packet size or a short packet) is written to the receive FIFO.

- OUT data packets received with Bad Data CRC are emptied from the receive FIFO
- After sending an ACK to the data packet on the USB, the controller discards non-synchronous OUT data packets that the host (which cannot detect the ACK) re-transmits. The application does

not detect multiple consecutive OUT data packets on the same endpoint with the same data PID. In this case, the packet count is not decremented.

- If there is no space in the receive FIFO, synchronous or non-synchronous data packets are ignored and not written to the receive FIFO. Besides, the non-synchronous OUT tokens receive a NAK handshake response.
- In all the above-mentioned cases, the packet count is not decremented because no data is written to the receive FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for the endpoint is set. Once the NAK bit is set, the synchronous or non-synchronous data packets are ignored and not written to the receive FIFO, and non-synchronous OUT tokens receive a NAK handshake response.
4. After the data is written to the receive FIFO, the application reads the data from the receive FIFO and writes it to the external memory, once packet at a time per endpoint.
5. At the end of data packet write to the external memory, the transfer size for the endpoint is decremented with the size of the written packet.
6. The OUT data transfer completed mode for an OUT endpoint is written to the receive FIFO one of the following conditions:

- The transfer size and packet count are both 0
- The last OUT data packet written to the receive FIFO is a short packet (0 ≤ data packet size < largest packet size)

7. When the application pops this entry (OUT data transfer complete), a transfer completed interrupt is generated and the endpoint enable bit is cleared.

[Application programming sequence]

1. Program the OTGFS_DOEPTSIZx register with the transfer size and the corresponding packet count.
2. Program the OTGFS_DOEPCTLx register with the endpoint characteristics, and set the endpoint enable and ClearNAK bits.

- OTGFS_DOEPCTLx.EPENA = 0x1
- OTG-DOEPCTLx.CNAK = 0x1

3. Wait for the RXFLVL interrupt in the OTGFS_GINTSTS register, and read out all data packets from the receive FIFO.

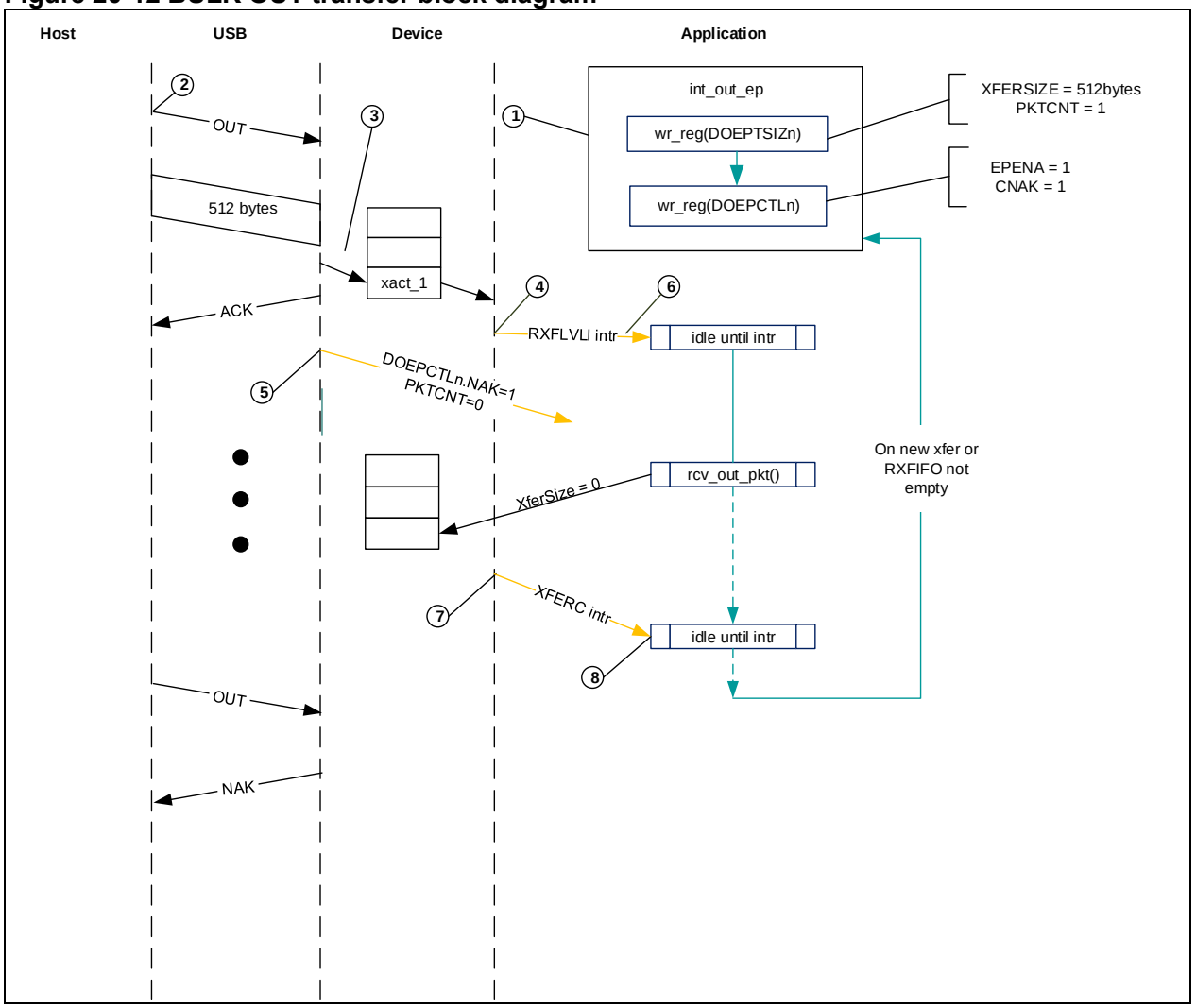
- This step can be repeated, depending on the transfer size

4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register, it indicates a successful completion of the non-synchronous OUT data transfer. Read the OTGFS_DOEPTSIZx register to determine how much data has been received.

[Bulk OUT transfer]

Figure 20-12 describes the reception of a single bulk OUT data packet from the USB to the AHB and shows the events involved in the process.

Figure 20-12 BULK OUT transfer block diagram



After a SetConfiguration/SetInterface command is received, the application initializes all OUT endpoints by setting CNAK = 0x1 and EPENA = 0x1 in the OYG_DOEPCTLx register, and setting the XFERSIZE and PKTCNT bits in the OTGFS_DOEPTSIZx register.

1. The host attempts to send data (OUT token) to the endpoint
2. When the controller receives the OUT token on the USB, it stores data in the receive FIFO because the FIFO has free space.
3. Upon writing the complete data in the receive FIFO, the controller then triggers the RXFLVL interrupt bit in the OTGFS_GINTSTS register.
4. Upon receiving the packet count of USB packets, the controller internally sets the NAK bit for the endpoint to prevent it from receiving any more packets.
5. The application processes the interrupt and reads the data from the receive FIFO.
6. When the application reads all the data (equivalent to XFERSIZE), the controller generates an XFERC interrupt in the OTGFS_DOEPINTx register.
7. The application processes the interrupt and uses the XFERC bit in the OTGFS_DOEPINTx register to judge that the expected transfer is already complete.

20.5.4.16 Synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in “OTGFS Initialization”. Before communicating with a host, the application must initialize endpoints based on the process described in “Endpoint Initialization” and by referring to “Read FIFO packets”. This section describes a regular synchronous OUT transfers.

[Application requirements]

1. All the application requirements are the same as that of non-synchronous OUT data transfers.

2. For synchronous OUT data transfers, the transfer size and packet count must be set to the number of the largest-packet-size packets that can be received in a single frame and not exceed this size. Synchronous OUT data transfer cannot span more than one frame.

- 1 ≤ packet count [epnum] ≤ 3

3. If the device supports the synchronous OUT endpoints, the application must read all synchronous OUT data packets from the receive FIFO before the end of the periodic frame (EOPF interrupt in the OTGFS_GINTSTS register)

4. To receive data in the subsequent frame, a synchronous OUT endpoint must be enabled before the generation of the EOPF and SOF interrupt in the OTGFS_GINTSTS register.

[Internal data flow]

1. The internal data flow for the synchronous OUT endpoints is the same as that for the non-synchronous OUT endpoints, just for a few differences.

2. When the synchronous OUT endpoint is enabled by setting the endpoint enable bit and by clearing the NAK bit, the even/odd frame bits are also set properly. The controller can receive data on a synchronous OUT endpoint in a particular frame only when the following condition is met:

- Even/Odd microframe (OTGFS_DOEPCTLx) =SOFFN[0] (OTGFS_DSTS)

3. When the application completely reads the synchronous OUT data packet (data and status) from the receive FIFO, the controller updates the RXDPID bit in the OTGFS_DOEPTSIZx register based on the data PID of the last synchronous OUT data packet read from the receive FIFO.

[Application programming sequence]

1. Program the transfer size and the corresponding packet count of the OTGFS_DOEPTSIZx register

2. Program the OTGFS_DOEPCTLx register with the endpoint enable, ClearNAK and Even/Odd frame bits

- Endpoint enable = 0x1
- CNAK = 0x1
- Even/Odd frame = (0x0: Even; 0x1: Odd)

3. Wait for the RXFLVL interrupt in the OTGFS_GINTSTS register, and read all the data packets from the receive FIFO. See “Read FIFO” for more information

- This step can be repeated several times, depending on the transfer size

4. When the XFERC interrupt is set in the OTGFS_DOEPINTx register, it indicates the completion of the synchronous OUT data transfers. But this interrupt does not necessarily mean that the data in memory are good.

5. This interrupt signal cannot always be detected by the synchronous OUT data transfers. However, the application can detect the INCOMPISOOUT interrupt in the OTGFS_GINTSTS register. See “Incomplete synchronous OUT data transfers” for more information.

6. Read the OTGFS_DOEPTSIZx register to determine the received transfer size and to determine whether the data received in the frame are valid or not. The application must treat the data received in memory as valid only when one of the following conditions is met:

- OTGFS_DOEPTSIZx.RxDPID = 0xD0 and the USB packet count in which the payload was received =0x1
- OTGFS_DOEPTSIZx.RxDPID = 0xD1 and the USB packet count in which the payload was received =0x2
- OTGFS_DOEPTSIZx.RxDPID = 0xD2 and the USB packet count in which the payload was received =0x3

The number of USB packets in which the payload was received= Application-programed initial packet count – Controller-updated final packet count

The application discards invalid data packets.

20.5.4.17 Enable synchronous endpoints

After sending a Set interface control command to the device, a host enables the synchronous endpoints. Then the host can send the initial synchronous IN token in any frame before transmission in the sequence of BInterval.

Instead, synchronous support in the OTGFS controller is based on a single-transfer level. The application must re-configure the controller on every frame. The OTGFS controller enables the synchronous endpoint of the frame before the frame to be transmitted.

For example, to send data on the frame n, enable the endpoint of the frame n-1. Additionally, the OTGFS controller schedules the synchronous transfers by setting Even/Odd frame bits.

[Synchronous IN transfer interrupt]

The following interrupts must be processed to ensure successful scheduling of the synchronous transfers.

- XFERC interrupt in the OTGFS_DIEPINTx register (for endpoints)
- OTG INCOMPISOIN interrupt in the OTGFS_GINTSTS register (for global interrupts)

[Handling synchronous IN transfers]

The following steps must be performed to handle a synchronous IN transfer:

1. Unmask the incompISOOUT interrupt in the OTGFS_GINTSTS register by setting the INCOMISOINMSK interrupt bit in the OTGFS_GINTMSK register
2. Unmask the XFERC interrupt in the OTGFS_DIEPINTx register by setting the XFERCMSK bit in the OTGFS_DIEPMSK register
3. Enable synchronous endpoints with the following steps:

- Program the OTGFS_DIEPTSIZx register

  OTGFS_DIEPTSIZx.XFERSIZE= n * OTGFS_DIEPCTLx.MPS + sp, where 0 <= n <= 3 and 0 <= sp <OTGFS_DIEPCTLx.MPS. When the transfer size in a frame is less than that of the MPS bit in the OTGFS_DIEPCTLx register, n=0; When the transfer size in a frame is a multiple of that of the MPS bit in the OTGFS_DIEPCTLx register, sp=0.

  OTGFS_DIEPTSIZx.PKTCNT = 0x1

  The MC bit in the OTGFS_DIEPTSIZx register is set the same value as that of the PKTCNT bit in the OTGFS_DIEPTSIZx register.

- Program the OTGFS_DIEPCTLx register

  Read the OTGFS_DSTS register to determine the current frame number

  Program the OTGFS_DIEPCTLx with the maximum packet size (MPS bit)

  Set USBACTEP = 0x1 in the OTGFS_DIEPCTLx register

  Set EPTYPE = 0x1 in the OTGFS_DIEPCTLx register, marking synchronization

  Set the FIFO number of the endpoint through the TXFNUM bit in the OTGFS_DIEPCTLx register

  Set CNAK = 0x1 in the OTGFS_DIEPCTLx register

  If.SOFFN[0] = 0x0 in OTGFS_DSTS, then SETEVENFR = 0x1 in OTGFS_DIEPCTLx (otherwise, SETEVENFR = 0x1 in OTGFS_DIEPCTLx)

  If SOFFN[0] = 0x1 in OTGFS_DSTS, then SETODDFR = 0x1 in OTGFS_DIEPCTLx (otherwise, SETODDFR = 0x0 in OTGFS_DIEPCTLx)

  Set EPENA = 0x1 in OTGFS_DIEPCTLx

4. Write endpoint data to the corresponding transmit FIFO

For example, write address ranges are as follows:

- EP1 corresponding to 0x2000 - 0x2FFC
- EP2 corresponding to 0x3000 - 0x3FFC
- EP3 corresponding to 0x3000 - 0x3FFC
- ...

5. Wait for interrupts

- When an interrupt is generated (XFERC bit in OTGFS_DIEPINTx register), clear the XFERC interrupt; for the following transaction, repeat step 3-5 until the completion of data transfers.
- When an interrupt is generated (INCOMPISOIN bit in OTGFS_GINTSTS register), clear the INCOMPISOIN interrupt; For any synchronous IN endpoint, when Odd/Even bits match the current frame number bit 0, and when the endpoint remains enabled, the controller generates an interrupt at the end of the frame. This interrupt is generated on one of the following conditions:

  (1) There is no token in a frame

  (2) Late data write to the receive FIFO. An IN token has arrived before the completion of data write

  (3) IN token error

The INCOMPISOIN interrupt in the OTGFS_GINTSTS register is a global interrupt. Therefore, when more than one synchronous endpoints are in active state, the application must determine which one of the synchronous IN endpoints has not yet completed data transfers.

To achieve this, read the DSTS and DIEPCTLx bits of all synchronous endpoints. If the current endpoint has been enabled, and the read value of the SOFFN bit in the OTGFS_DSTS register is equal to the target frame number of the endpoint, it indicates that this endpoint has not finished data transfers. The application must keep track of and update the target frame number of the synchronous endpoint.

If data transfer is not yet complete on an endpoint, then Odd/Even bits have to be toggled.

Next:

(1) When the DPID is set to 1 (an odd frame) in the OTGFS_DIEPCTLx register, write 1 to the SETD0PID bit in the OTGFS_DIEPCTLx register makes it an even frame, then data transmission starts when there is an IN token input in the next frame.

(2) When the DPID is set to 0 in the OTGFS_DIEPCTLx register, write 1 to the SETD1PID bit in the OTGFS_DIEPCTLx register makes it an odd frame, then data transmission starts when there is an IN token input in the next frame.

20.5.4.18 Incomplete synchronous OUT data transfers

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS Initializtion. Before communicating with a host, the controller must follow the steps defined in Endpoint Initializtion to initialize endpoints. This section describes the application programming sequence when the controller drops synchronous OUT data packets.

[Internal data flow]

1. For synchronous OUT endpoints, the XFERC interrupt (in the OTGFS_DOEPINTx register) may not always be generated. If the controller drops synchronous OUT data packets, the application may fail to detect the XFERC interrupt in the OTGFS_DOEPINTx register.

- When the receive FIFO cannot accommodate the complete ISO OUT data packet, the controller drops the received ISO OUT data.
- When the synchronous OUT data packet is received with CRC errors.
- When the synchronous OUT token received by the controller is corrupted.
- When the application is very slow in reading the receive FIFO

2. When the controller detects the end of periodic frames before transfer complete to all synchronous OUT endpoints, an interrupt of incomplete synchronous OUT data is generated, indicating that an XFERC interrupt in the OTGFS_DOEPINTx register is not set on at least one of the synchronous OUT endpoints. At this point, the endpoint with the incomplete data transfer remains enabled, but no valid transfers are in progress on this endpoint.

[Application programming sequence]

1. The assertion of the incomplete synchronous OUT data interrupt indicates that at least one synchronous OUT endpoint has an incomplete data transfer in the current frame.
2. If this occurs because the synchronous OUT data is not completely read out from the endpoint, the application must empty all synchronous OUT data (data and status) in the receive FIFO before proceeding.

- When all data are read from the receive FIFO, the application can detect the XFERC interrupt in the OTGFS_DOEPINTx register. In this case, the application must re-enable the endpoint to

receive the synchronous OUT data in the next frame by following the steps listed in "SETUP/Data IN/Status OUT"

3. When it receives an incomplete synchronous OUT data interrupt, the application must read the control registers of all synchronous OUT endpoints to determine which one of the endpoints has an incomplete data transfer in the current frame. An endpoint transfer is regarded as incomplete if both of the following conditions are met:

- OTGFS_DOEPCTLx. Even/Odd frame bit= OTGFS_DSTS.SOFFN[0]
- OTGFS_DOEPCTLx. Endpoint enable = 0x1

4. The pervious step must be performed before the SOF interrupt of the GINTSTS register is detected to ensure that the current frame number is not changed.
5. For synchronous OUT endpoints with incomplete transfers, the application must drop the data in memory, and disable the endpoint through the endpoint disable bit in the OTGFS_DOEPCTLx register.
6. Wait for the endpoint disable interrupt in the OTGFS_DOEPINTx register, and enable the endpoint to receive new data in the next frame by following the steps listed in "SETUP/Data IN/Status OUT". Because the controller can take some time to disable the endpoint, the application may not be able to receive the data in the next frame after receiving wrong synchronous data.

20.5.4.19 Incomplete synchronous IN data transfers

This section describes how the application behaves on incomplete synchronous IN transfers.

[Internal data flow]

1. Synchronous IN transfers are incomplete on one of the following conditions:

- The controller receives corrupted synchronous IN tokens from more than one synchronous IN endpoints. In this case, the application can detect the incomplete synchronous IN transfer interrupt in the GINTSTS register.
- The application is slow in writing complete data to the transmit FIFO, and an IN token is received before the completion of data write. In this case, the application can detect the INTKNTXFEMP interrupt in the OTGFS_DIEPINTx register. The application ignores this interrupt, which will result in the generation of the incomplete synchronous IN transfer interrupt (in OTGFS_GINTSTS register). The controller responds to the received IN token by sending a zero-length data packet to the USB.

2. Either way, the application must stop writing the transmit FIFO as soon as possible.
3. The application must set the NAK and disable bits of the endpoints.
4. The controller disables the endpoint, clears the disable bit, and triggers the endpoint disable interrupt.

[Application programming sequence]

1. When the transmit FIFO becomes empty, the application ignores the INTKNTXFEMP interrupt (in the OTGFS_DIEPINTx register) from any synchronous IN endpoint because this can trigger the incomplete synchronous IN interrupt.
2. The incomplete synchronous IN transfer interrupt (in the OTGFS_GINTSTS register) indicates that at least one synchronous IN endpoint is with incomplete synchronous IN transfers.
3. The application must read the endpoint control registers of all synchronous IN endpoints to determine which one is with incomplete synchronous IN transfers.
4. The application must write data to the periodic transmit FIFO of the endpoint.
5. Disable theses endpoints by setting the following bits in the OTGFS_DIEPCTLx register

- OTGFS_DIEPCTLx.SETNAK = 0x1
- OTGFS_DIEPCTLx. endpoint enable = 0x1

6. The endpoint disable interrupt in the DIEPINTx register indicates that the controller has disabled the endpoint.
7. At this point, the application must empty the data in the associated transmit FIFO or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next frame. The application must refresh the data through the OTGFS_GRSTCTL register.

20.5.4.20 Periodic IN (interrupt and synchronous) data transfers

This section describes a typical periodic IN data transfer.

To initialize the controller after power-on reset, the application must perform the steps list in OTGFS Initializtion. Before communicating with a host, the controller must follow the steps defined in Endpoint Initializtion to initialize endpoints.

[Application requirements]

1. Application requirements in “Non-periodic (bulk and control) IN data transfers” also apply to periodic IN data transfers, except for a slight difference of requirement 2.
   - The application can only transmit multiples of largest-packet-size data packets, and a short packet. To transmit several largest-packet-size data packets and a short packet, the following conditions must be met:

   Transfer size [epnum] = n * mps[epnum] + sp (where n and i are integers ≥ 0, and 0 ≤ sp < mps[epnum])

   If (sp > 0), packet count [epnum] = n + 1. Otherwise, packet count [epnum] = n, mc[epnum] = packet count [epnum]
   - The application cannot transmit a zero-length data packet at the end of a transfer. But it can transmit a single zero-length data packet in itself, provided packet count [epnum] = 1, mc[epnum] = packet count [epnum]
2. The application can only schedule data transfers of one frame at a time
   - (OTGFS_DIEPTSIZx.MC – 1) * OTGFS_DIEPCTLx.MPS ≤OTGFS_DIEPTSIZx.XFERSIZ ≤OTGFS_DIEPTSIZx.MC * OTGFS_DIEPCTLx.MPS
   - OTGFS_DIEPTSIZx.PKTCNT = OTGFS_DIEPTSIZx.MC
   - If OTGFS_DIEPTSIZx.XFERSIZ < OTGFS_DIEPTSIZx.MC * OTGFS_DIEPCTLx.MPS, the last data packet of the transfer is a short packet.
3. For periodic IN endpoints, one-frame data must be prefetched before the data transfer in the next frame. This can be done by enabling periodic IN endpoint 1 frame before the scheduling of the frame to be transmitted.
4. The complete data to be transmitted in a frame must be written to the transmit FIFO by the application before the periodic IN token is received. Even when one-WORD data to be transmitted per frame is missing in the transmit FIFO while the periodic IN token is received, the controller behaves as when the FIFO is empty. When the transmit FIFO is empty, a zero-length data packet would be transmitted on the USB, and An NAK handshake signal would be transmitted for INTR IN endpoints.

[Internal data flow]

1. The application must set the transfer size and packet count bits of the endpoint registers, and enable the endpoint to transmit the data.
2. The application must also write the required data to the associated transmit FIFO.
3. Each time the application writes a packet to the transmit FIFO, the transfer size for the endpoint is decremented by the packet size. Continue to write data until the transfer size for the endpoint becomes 0
4. When an IN token for a periodic endpoint is received, the application writes the data to the FIFO (If any). If the complete data for the frame is not present in the FIFO, the controller generates an INTKNTXFEMP interrupt.
   - A zero-length data packet is transmitted on the USB for synchronous IN endpoints
   - An NAK handshake signal is transmitted on the USB for interrupt IN endpoints.
5. The packet count for the endpoints is decremented by one under the following conditions:
   - For synchronous endpoints, when a zero-or non-zero-length data packet is transmitted
   - For interrupt endpoints, when an ACK handshake is transmitted
   - When the transfer size and packet count are both 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable bit is cleared.
6. In the “Periodic frame interval” (by the PERFRINT bit in the OTGFS_DCFG register), when the controller finds non-empty any one of the IN endpoint FIFOs scheduled for the current frame non-

empty, the controller generates an INCOMPISOIN interrupt in the OTGFS_GINTSTS register.

[Application programming sequence (frame transfers)]

1. Program the OTGFS_DIEPTSIZx register
2. Program the OTGFS_DIEPCTLx register based on endpoint characteristics, and set the CNAK and endpoint enable bits
3. Write the data to be transmitted into the transmit FIFO.
4. The assertion of the INTKNTXFEMP interrupt indicates that the application has not yet written all data to be transferred into the transmit FIFO.
5. If the interrupt endpoint is already enabled while this interrupt is detected, ignore the interrupt. If it is not enabled, enable the endpoint to transmit data on the next IN token. If it is enabled while the interrupt is detected, refer to “Incomplete synchronous IN data transfers”.
6. When the interrupt IN endpoint is set as a periodic endpoint, the controller internally can process the timeout on the interrupt IN endpoint, without the need of the application intervention. Therefore, the application can never detect the TIMEOUT interrupt (in the OTGFS_DIEPINTx register) on the periodic interrupt IN endpoints.
7. The assertion of the XFERC interrupt in the OTGFS_DIEPINTx register but without the INTKNTXFEMP interrupt indicates the successful completion of a synchronous IN transfer. When reading the OTGFS_DIEPTSIZx register, only transfer size =0 and packet count =0 indicate that all data are transmitted on the USB line.
8. The assertion of the XFERC interrupt in the OTGFS_DIEPINTx register, with or without the INTKNTXFEMP interrupt, indicates the successful completion of an interrupt IN transfer. When reading the OTGFS_DIEPTSIZx register, only transfer size =0 and packet count =0 indicate that all data are transmitted on the USB line.
9. The assertion of the INCOMPISOIN interrupt but without the above-mentioned interrupts indicates that the controller did not receive at least one periodic IN token in the current frame. Refer to “Incomplete synchronous IN data transfers” for more information on synchronous IN endpoints.

20.6 OTGFS control and status registers

The application controls the OTGFS controller by reading from and writing to the control and status registers (CSRx) through the AHB slave interface. These registers are accessible by 32 bits, and the addresses are 32-bit aligned.

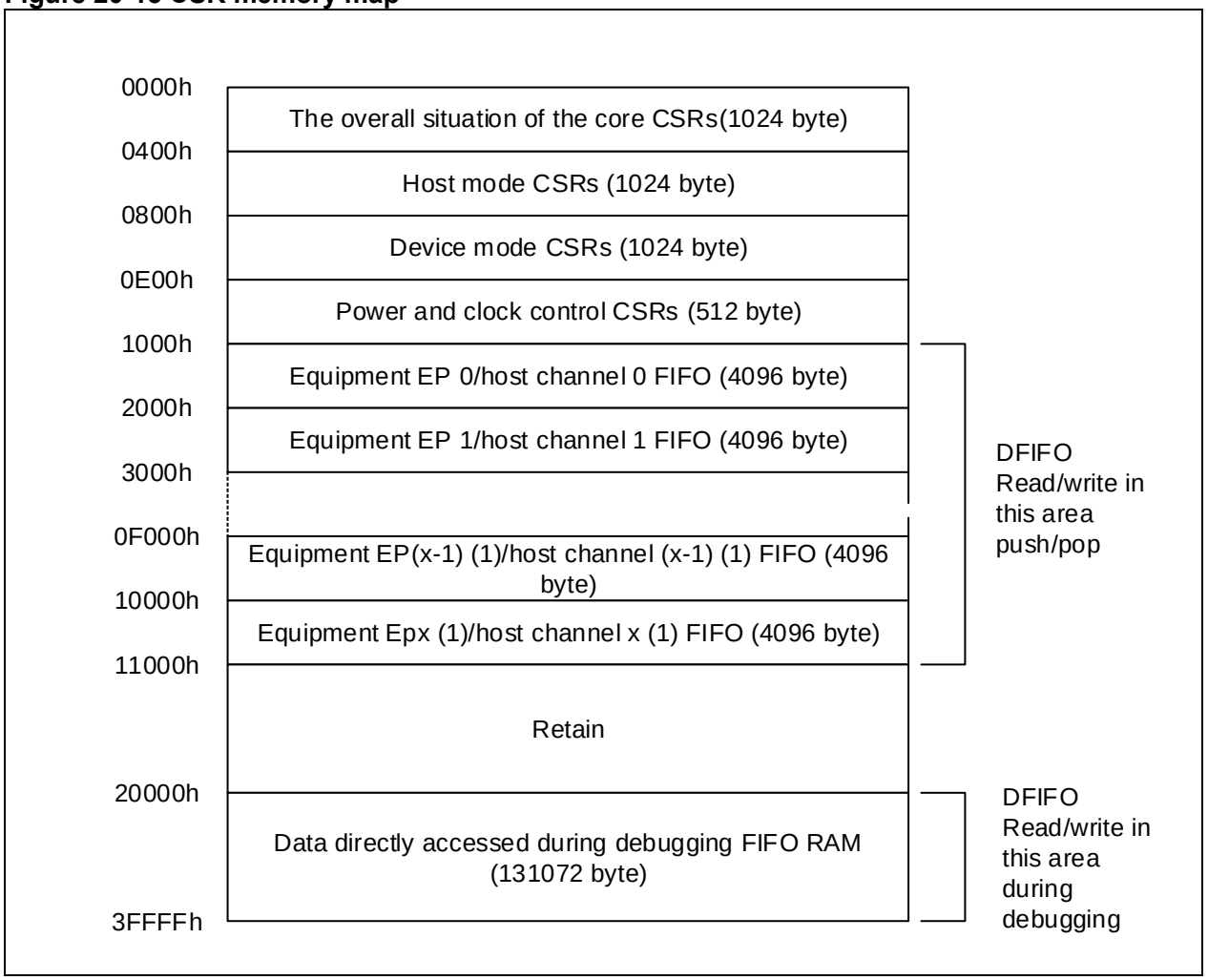
Only the controller global, power and clock control, data FIFO access and host port control and status registers are active in both host and device modes. When the OTGFS controller operates in either host or device mode, the application must not access the register group from the other mode. If an illegal access occurs, a mode mismatch interrupt is generated and the MODMIS bit (in the OTGFS_GINTSTS register) is affected.

When the controller switches from one mode to the other, the registers in the new mode must be re-initialized as they are after a power-on reset. These peripheral registers must be accessed by words (32-bit)

20.6.1 CSR register map

The host and device mode registers occupy different addresses. All registers are located in the AHB clock domain

Figure 20-13 CSR memory map



x = 4 in device mode, x =8 in host mode.

The OTGFS control and status registers contain OTGFS global register, host mode register, device mode register, data FIFO register, power and clock control register.

1. OTGFS global registers: They are active in both host and device modes. The register acronym is G.
2. Host-mode registers: They must be programmed every time the controller changes to host mode, the register acronym is H.
3. Device-mode registers: They must be programmed every time the controller changes to device mode, the register acronym is D.
4. Data FIFO access registers: These registers are valid in both in host and device modes, and are used to read or write the FIFO for a specific endpoint or channel in a given direction. If a host channel is of type IN, the FIFO can only be read. Similarly, if a host channel is of type OUT, the FIFO can only be written.
5. Power and clock control register: There is only one register for power and clock control. It is valid in both host and device modes.

20.6.2 OTGFS register address map

Table 20-4 shows the USB OTG register map and their reset values.

These peripheral registers must be accessed by words (32-bit)

Table 20-4 OTGFS register map and reset values

Register name	Offset	Reset value
OTGFS_GOTGCTL	0x000	0x0001 0000
OTGFS_GOTGINT	0x004	0x0000 0000
OTGFS_GAHBCFG	0x008	0x0000 0000
OTGFS_GUSBCFG	0x00C	0x0000 1400
OTGFS_GRSTCTL	0x010	0x2000 0000
OTGFS_GINTSTS	0x014	0x0400 0020
OTGFS_GINTMSK	0x018	0x0000 0000
OTGFS_GRXSTSR	0x01C	0x0000 0000
OTGFS_GRXSTSP	0x020	0x0000 0000
OTGFS_GRXFSIZ	0x024	0x0000 0200
OTGFS_GNPTXFSIZ/ OTGFS_DIEPTXF0	0x028	0x0200 0200
OTGFS_GNPTXSTS	0x02C	0x0008 0200
OTGFS_GCCFG	0x038	0x0000 0000
OTGFS_GUID	0x03C	0x0000 1000
OTGFS_HPTXFSIZ	0x100	0x0000 0000
OTGFS_DIEPTXF1	0x104	0x0000 0000
OTGFS_DIEPTXF2	0x108	0x0000 0000
OTGFS_DIEPTXF3	0x10C	0x0000 0000
OTGFS_HCFG	0x400	0x0000 0000
OTGFS_HFIR	0x404	0x0000 EA60
OTGFS_HFNUM	0x408	0x0000 3FFF
OTGFS_HPTXSTS	0x410	0x0008 0100
OTGFS_HAINT	0x414	0x0000 0000
OTGFS_HAINTMSK	0x418	0x0000 0000
OTGFS_HPRT	0x440	0x0000 0000
OTGFS_HCCHAR0	0x500	0x0000 0000
OTGFS_HCINT0	0x508	0x0000 0000
OTGFS_HCINTMSK0	0x50C	0x0000 0000
OTGFS_HCTSIZ0	0x510	0x0000 0000
OTGFS_HCCHAR1	0x520	0x0000 0000
OTGFS_HCINT1	0x528	0x0000 0000
OTGFS_HCINTMSK1	0x52C	0x0000 0000
OTGFS_HCTSIZ1	0x530	0x0000 0000
OTGFS_HCCHAR2	0x540	0x0000 0000
OTGFS_HCINT2	0x548	0x0000 0000

OTGFS_HCINTMSK2	0x54C	0x0000 0000
OTGFS_HCTSIZ2	0x550	0x0000 0000
OTGFS_HCCHAR3	0x560	0x0000 0000
OTGFS_HCINT3	0x568	0x0000 0000
OTGFS_HCINTMSK3	0x56C	0x0000 0000
OTGFS_HCTSIZ3	0x570	0x0000 0000
OTGFS_HCCHAR4	0x580	0x0000 0000
OTGFS_HCINT4	0x588	0x0000 0000
OTGFS_HCINTMSK4	0x58C	0x0000 0000
OTGFS_HCTSIZ4	0x590	0x0000 0000
OTGFS_HCCHAR5	0x5A0	0x0000 0000
OTGFS_HCINT5	0x5A8	0x0000 0000
OTGFS_HCINTMSK5	0x5AC	0x0000 0000
OTGFS_HCTSIZ5	0x5B0	0x0000 0000
OTGFS_HCCHAR6	0x5C0	0x0000 0000
OTGFS_HCINT6	0x5C8	0x0000 0000
OTGFS_HCINTMSK6	0x5CC	0x0000 0000
OTGFS_HCTSIZ6	0x5D0	0x0000 0000
OTGFS_HCCHAR7	0x5E0	0x0000 0000
OTGFS_HCINT7	0x5E8	0x0000 0000
OTGFS_HCINTMSK7	0x5EC	0x0000 0000
OTGFS_HCTSIZ7	0x5F0	0x0000 0000
OTGFS_HCCHAR8	0x600	0x0000 0000
OTGFS_HCINT8	0x608	0x0000 0000
OTGFS_HCINTMSK8	0x60C	0x0000 0000
OTGFS_HCTSIZ8	0x610	0x0000 0000
OTGFS_DCFG	0x800	0x0220 0000
OTGFS_DCTL	0x804	0x0000 0000
OTGFS_DSTS	0x808	0x0000 0010
OTGFS_DIEPMSK	0x810	0x0000 0000
OTGFS_DOEPMSK	0x814	0x0000 0000
OTGFS_DAINT	0x818	0x0000 0000
OTGFS_DAINTMSK	0x81C	0x0000 0000
OTGFS_DIEPEMPMSK	0x834	0x0000 0000
OTGFS_DIEPCTL0	0x900	0x0000 0000
OTGFS_DIEPINT0	0x908	0x0000 0080
OTGFS_DIEPTSIZ0	0x910	0x0000 0000
OTGFS_DTXFSTS0	0x918	0x0000 0200
OTGFS_DIEPCTL1	0x920	0x0000 0000
OTGFS_DIEPINT1	0x928	0x0000 0080
OTGFS_DIEPTSIZ1	0x930	0x0000 0000

	OTGFS_DTXFSTS1	0x938	0x0000 0200
	OTGFS_DIEPCTL2	0x940	0x0000 0000
	OTGFS_DIEPINT2	0x948	0x0000 0080
	OTGFS_DIEPTSIZ2	0x950	0x0000 0000
	OTGFS_DTXFSTS2	0x958	0x0000 0200
	OTGFS_DIEPCTL3	0x960	0x0000 0000
	OTGFS_DIEPINT3	0x968	0x0000 0080
	OTGFS_DIEPTSIZ3	0x970	0x0000 0000
	OTGFS_DTXFSTS3	0x978	0x0000 0200
	OTGFS_DOEPCTL0	0xB00	0x0000 8000
	OTGFS_DOEPINT0	0xB08	0x0000 0080
	OTGFS_DOEPTSIZ0	0xB10	0x0000 0000
	OTGFS_DOEPCTL1	0xB20	0x0000 0000
	OTGFS_DOEPINT1	0xB28	0x0000 0080
	OTGFS_DOEPTSIZ1	0xB30	0x0000 0000
	OTGFS_DOEPCTL2	0xB40	0x0000 0000
	OTGFS_DOEPINT2	0xB48	0x0000 0080
	OTGFS_DOEPTSIZ2	0xB50	0x0000 0000
	OTGFS_DOEPCTL3	0xB60	0x0000 0000
	OTGFS_DOEPINT3	0xB68	0x0000 0080
	OTGFS_DOEPTSIZ3	0xB70	0x0000 0000
	OTGFS_PCGCCTL	0xE00	0x0000 0000
	OTGFS_DEP3RMPEN	0xD0C	0x0000 0000
	OTGFS_USBDIVRST	0xE10	0x0000 0000

20.6.3 OTGFS global registers

These registers are available in both host and device modes, and do not need to be reprogrammed when switching between two modes.

20.6.3.1 OTGFS status and control register (OTGFS_GOTGCTL)

This register controls the OTG function and reflects its status.

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x0000	resd	Kept at its default value.
Bit 21	CURMOD	0x0	ro	Current Mode of Operation Accessible in both host and device modes This bit indicates the current operation mode. 0: Device mode 1: Host mode
Bit 20: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CONIDSTS	0x1	ro	Accessible in both host and device modes Connector ID Status 0: OTGFS controller is in A-device mode 1: OTGFS controller is in B-device mode
Bit 15: 0	Reserved	0x0000	resd	Kept at its default value.

20.6.3.2 OTGFS interrupt status control register (OTGFS_GOTGINT)

The application reads this register to know about which kind of OTG interrupt is generated, and writes this register to clear the OTG interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000	resd	Kept at its default value.
Bit 2	SESENDDET	0x0	rw1c	Available in both host and device modes Session end detected The controller sets this bit when a Bvalid (Vbus) signal is disconnected. This register can only be set by hardware. Writing 1 by software clears this bit.
Bit 1: 0	Reserved	0x0000	resd	Kept at its default value.

20.6.3.3 OTGFS AHB configuration register (OTGFS_GAHBCFG)

This register is used to configure the controller after power-on or mode change. This register mainly contains AHB-related parameters. Do not change this register after the initial configuration. The application must configure this register before starting transmission on either the AHB or USB.

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	PTXFEMPLVL	0x0	rw	Accessible in host mode only Periodic TxFIFO empty level It indicates when the periodic TxFIFO empty interrupt bit in the GINTSTS register is triggered. 0: PTXFEMP (GINTSTS) interrupt indicates that the periodic TxFIFO is half empty 1: PTXFEMP (GINTSTS) interrupt indicates that the periodic TxFIFO is fully empty
Bit 7	NPTXFEMPLVL	0x0	rw	Accessible in both host mode and device modes Non-Periodic TxFIFO empty level In host mode, this bit indicates when the non-periodic TxFIFO empty interrupt (NPTXFEMP in GINTSTS) is triggered. In device mode, this bit indicates when the IN endpoint TxFIFO empty interrupt (TXFEMP bit in DIEPINTn) is triggered. 0: The TxFEMP (in DIEPINTn) interrupt indicates that the IN endpoint TxFIFO is half empty 1: The TxFEMP (in DIEPINTn) interrupt indicates that the IN endpoint TxFIFO is fully empty
Bit 6: 1	Reserved	0x00	resd	Kept at its default value.
Bit 0	GLBINTMSK	0x0	rw	Accessible in both host mode and device modes Global interrupt mask The application uses this bit to mask or unmask the interrupts sent by the interrupt line to itself. 0: Mask the interrupts sent to the application 1: Unmask the interrupts sent to the application

20.6.3.4 OTGFS USB configuration register (OTGFS_GUSBCFG)

This register is used to configure the controller after power-on or a change between host mode and device mode. This register contains USB and USB-PHY related parameters. The application must program the register before handling any transaction on either the AHB or USB. Do not change this register after the initial configuration.

Bit	Register	Reset value	Type	Description
Bit 31	COTXPKT	0x0	rw	Accessible in both host mode and device modes Corrupt Tx packet This bit is for debug purpose only. Do not set this bit to 1.
Bit 30	FDEVMODE	0x0	rw	Accessible in both host mode and device modes Force device mode Writing 1 to this bit forces the controller to go into device mode, irrespective of the status of the ID input point. 0: Normal mode 1: Force device mode After setting this bit, the application must wait at least 25ms before the configuration takes effect.
Bit 29	FHSTMODE	0x0	rw	Accessible in both host mode and device modes Force host mode Writing 1 to this bit forces the controller to go into host mode, irrespective of the status of the ID input point. 0: Normal mode 1: Force host mode After setting this bit, the application must wait at least 25ms before the configuration takes effect.
Bit 28: 15	Reserved	0x0000	resd	Kept at its default value.
Bit 14	Reserved	0x0	resd	Kept at its default value.
Bit 13: 10	USBTRDTIM	0x5	rw	Accessible in device mode USB Turnaround Time This field sets the turnaround time in PHY clocks. It defines the response time when the MAC sends a request to the packet FIFO controller (PFC) to fetch data from the DFIFO (SPRAM). These bits must be configured as follows: 0101: When the MAC interface is 16-bit UTMI+ 1001: When the MAC interface is 8-bit UTMI+ Note: The aforementioned values are calculated based on a minimum of 30MHz AHB frequency. The USB turnaround time is critical for certifications with long cables and 5-Hub. If you want the AHB to run below 30 MHz, and don't care about the USB turnaround time, you can set larger values for these bits.
Bit 9: 3	Reserved	0x00	resd	Kept at its default value.
Bit 2: 0	TOUTCAL	0x0	rw	Accessible in both host mode and device modes FS Timeout calibration The number of PHY clocks that the application programs in these bits is added to the full-speed interpacket timeout duration in order to compensate for any additional latency introduced by the PHY. This action can be required, because the delay triggered by the PHY while generating the line state condition can vary from one PHY to another. In full-speed mode, the USB standard timeout value is 16~18 (inclusive) bit times. The application must program these bits based on the enumeration speed. The number of bit times added per PHY clock is 0.25 bit times.

20.6.3.5 OTGFS reset register (OTGFS_GRSTCTL)

The application resets various hardware modules in the controller through this register.

Bit	Register	Reset value	Type	Description
Bit 31	AHBIDLE	0x1	ro	Accessible in both host mode and device modes AHB master Idle This bit indicates that the AHB master state machine is in idle condition.
Bit 30: 11	Reserved	0x000	resd	Kept at its default value.
Bit 10: 6	TXFNUM	0x00	rw	Accessible in both host mode and device modes TxFIFO number This field indicates the FIFO number that must be refreshed through the TxFIFO Flush bit. Do not make changes to this field until the controller clears the TxFIFO Flush bit. 00000: - Non-periodic TxFIFO in host mode - Tx FIFO 0 in device mode 00001: - Periodic TxFIFO in host mode - TXFIFO 1 in device mode 00010: - TXFIFO 2 in device mode ... 01111: - TXFIFO 15 in device mode 10000: - Refresh all the transmit FIFOs in device or host mode
Bit 5	TXFFLSH	0x0	rw1s	Accessible in both host mode and device modes TxFIFO Flush This bit selectively refreshes a single or all transmit FIFOs, but can do so when the controller is not in the process of a transaction. The application must write this bit only after checking that the controller is neither writing to nor reading from the TxFIFO. Verify using these registers: Read: NAK effective interrupt (NAK Effective Interrupt) ensures that the controller is not reading from the FIFO Write: AHBIDLE bit in GRSTCTL ensures that the controller is not writing to the FIFO. For FIFO reprogramming, it is usually recommended to carry out flushing operation. In device endpoint disable state, it is also advised to use FIFO flushing operation. The application must wait until the controller clears this bit, before performing other operations. It takes 8 clocks to clear this bit (slowest of phy_clk or hclk)
Bit 4	RXFFLSH	0x0	rw1s	Accessible in both host mode and device modes RxFIFO flush The application can refresh the entire RxFIFO using this bit, but must first ensure that the controller is not in the process of a transaction. The application must only write to this bit after checking that the controller is neither reading from nor writing to the RxFIFO. The application must wait until the controller clears this bit, before performing other operations. It takes 8 clocks to clear this bit (slowest of PHY or AHB)
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	FRMCNTRST	0x0	rw1s	Accessible in both host mode and device modes Host frame counter reset The application uses this bit to reset the frame number counter inside the controller. After the frame counter is reset, the subsequent SOS sent out by the controller has

				a frame number of 0. If the application writes 1 to this bit, it may not be able to read the value, because this bit is cleared after a few clock cycles by the controller
Bit 1	PIUSFTRST	0x0	rw1s	Accessible in both host mode and device modes PIU FS dedicated controller soft reset This bit is used to reset PIU full-speed dedicated controller All state machines in the PIU full-speed dedicated controller are reset to the idle state. When the PHY remains in the receive state for more than one-frame time due to PHY errors (such as operation interrupted or babble), this bit can be used to reset the PIU full-speed dedicated controller. This is can be cleared automatically, the controller this clear this bit after all the necessary logic is reset in the controller.
Bit 0	CSFTRST	0x0	rw1s	Accessible in both host mode and device modes Controller soft reset Resets the hclk and phy_clock domain as follows: Clears all interrupts and CSR registers except for the following bits: - HCFG.FSLSPCS - DCFG.DECSPD - DCTL.SFTDIS Resets all state machines (except AHB slave) to the idle state, and clears all the transmit and receive FIFOs. All transactions on the AHB master are terminated as soon as possible after completing the last phase of an AHB data transfer. All transactions on the USB are terminated immediately. The application can write to this bit at any time to reset the controller. This is can be cleared automatically, the controller this clear this bit after all the necessary logic is reset in the controller. The controller could take several clocks to clear this bit, depending on the current state of the controller. Once this bit is cleared, the application must wait at least 3 PHY clocks before accessing the PHY domain (synchronization delay). Additionally, the application must ensure that the bit 31 in this register is set (AHB master is in idle state) before performing other operations. Typically, the software set is used during software development and also when the user dynamically changes the PHY selection bits in the above-listed USB configuration registers. To change the PHY, the corresponding PHY clock is selected and used in the PHY domain. After a new clock is selected, the PHY domain has to be reset for normal operation.

20.6.3.6 OTGFS interrupt register (OTGFS_GINTSTS)

This register interrupts the application due to system-level events in the current mode (device or host mode), as shown in *Figure 20-2*.

Some of the bits in this register are valid only in host mode, while others are valid in device mode only. Besides, this register indicates the current mode.

The FIFO status interrupts are read-only. The FIFO interrupt conditions are cleared automatically as soon as the software reads from or writes to the FIFO while processing these interrupts.

The application must clear the GINTSTS register at initialization before enabling an interrupt bit to avoid any interrupt generation prior to initialization.

Bit	Register	Reset value	Type	Description
Bit 31	WKUPINT	0x0	rw1c	Accessible in both host mode and device modes Resume/Remote wakeup detected interrupt) In device mode, this interrupt is generated only when a resume signal (triggered by host) is detected on the USB bus. In host mode, this interrupt is generated only when a remote wakeup signal (triggered by device) is detected on the USB bus.
Bit 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	DISCONINT	0x0	rw1c	Accessible in host mode only Disconnect detected interrupt The interrupt is generated when a device disconnect is detected.
Bit 28	CONIDSCHG	0x0	rw1c	Accessible in both host mode and device modes Connector ID status change This bit is set by the controller when there is a change in connector ID status.
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	PTXFEMP	0x1	ro	Accessible in host mode only Periodic TxFIFO Empty The interrupt is generated when the Periodic Transmit FIFO is either half or completely empty and there is space for a request to be written in the period request queue. The half or completely empty status depends on the periodic transmit FIFO empty level bit in the AHB configuration register.
Bit 25	HCHINT	0x0	ro	Host channel interrupt The controller sets this bit to indicate that an interrupt is pending on one of the channels in the controller (in host mode). The application must read the Host All Channels Interrupt register to determine the exact number of the channel on which the interrupt occurred, and then read the Host Channel-n Interrupt register to determine the interrupt event source. The application must clear the corresponding status bit in the HCINTn (Host All Channels Interrupt) register to clear this bit.
Bit 24	PRTINT	0x0	ro	Host port interrupt The controller sets this bit to indicate a change in port status one of the ports. The application must read the Host Port Control and Status register to determine the exact event source. The application must clear the Host Port Control and Status register to clear this bit.
Bit 23: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	INCOMPIP INCOMPISOOUT	0x0	rw1c	Incomplete periodic transfer Accessible in host mode only In host mode, the controller sets this interrupt bit when there are incomplete periodic transfers still pending in the current frame. Incomplete Isochronous OUT Transfer Accessible in device mode only In device mode, the controller sets this interrupt bit to

				indicate that there is at least one synchronous OUT endpoint with incomplete transfers in the current frame. This interrupt is generated along with the End of Periodic Frame Interrupt bit in this register.
Bit 20	INCOMPISOIN	0x0	rw1c	Accessible in device mode only Incomplete Isochronous IN Transfer The controller sets this interrupt to indicate that there is at least one synchronous IN endpoint with incomplete transfers in the current frame. This interrupt is generated along with the End of Periodic Frame Interrupt bit in this register.
Bit 19	OEPTINT	0x0	ro	Accessible in device mode only OUT endpoints interrupt The controller sets this bit to indicate that an interrupt is pending on one of the OUT endpoints in the controller. The application must read the Device All Endpoints Interrupt register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding Device OUT Endpoint-n Interrupt register to determine the exact source of the interrupt. The application must clear the corresponding status bit in the corresponding Device OUT Endpoint-n Interrupt register to clear this bit.
Bit 18	IEPTINT	0x0	ro	Accessible in device mode only IN Endpoints interrupt The controller sets this bit to indicate that an interrupt is pending one of the IN endpoints in the controller (in device mode). The application must read the Device All Endpoints Interrupt register to determine the exact number of the IN endpoint on which the interrupt occurred, and then read the corresponding Device IN Endpoint-n Interrupt register to determine the exact source of the interrupt. The application must clear the corresponding status bit in the corresponding Device IN Endpoint-n Interrupt register to clear this bit.
Bit 17: 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EOPF	0x0	rw1c	Accessible in device mode only End of periodic frame interrupt This bit indicates that the period programmed in the periodic frame interval bit of the Device Configuration register has been reached in the current frame.
Bit 14	ISOOUTDROP	0x0	rw1c	Accessible in device mode only Isochronous OUT packet dropped interrupt) The controller sets this bit on the following condition: the controller fails to write a synchronous OUT packet into the receive FIFO because the receive FIFO does not have enough space to accommodate a maximum size packet for the synchronous OUT endpoint.
Bit 13	ENUMDONE	0x0	rw1c	Accessible in device mode only Enumeration done The controller sets this bit to indicate that speed enumeration is done. The application must read the Device Status register to obtain the enumeration speed.
Bit 12	USBRST	0x0	rw1c	Accessible in device mode only USB Reset The controller sets this bit to indicate that a reset is detected on the USB bus.
Bit 11	USBSUSP	0x0	rw1c	Accessible in device mode only USB Suspend The controller sets this bit to indicate that a suspend is detected on the USB bus. The controller enters the Suspend state when there is no activity on the bus for a long period of time.

Bit 10	ERLYSUSP	0x0	rw1c	Accessible in device mode only Early suspend The controller sets this bit to indicate that the idle state has been detected on the USB bus for 3 ms.
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GOUTNAKEFF	0x0	ro	Accessible in device mode only Global OUT NAK effective This bit indicates that the Set Global OUT NAK bit in the Device Control register (set by the application) has taken effect. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register.
Bit 6	GINNAKEFF	0x0	ro	Accessible in device mode only Global IN Non-periodic NAK effective This bit indicates that the Set Global Non-periodic IN NA bit in the Device Control register (set by the application) has taken effect. That is, the controller has sampled the Global IN NAK bit set by the application. This bit can be cleared by writing the Clear Global Non-periodic IN NA bit in the Device Control register. This interrupt does not necessarily mean that a NAK handshake signal is sent out on the USB bus. The STALL bit has priority over the NAK bit.
Bit 5	NPTXFEMP	0x1	ro	Accessible in both host and device modes Non-periodic TxFIFO empty This interrupt is generated when the Non-periodic TxFIFO is either half or completely empty and there is enough space for at least one request to be written to the Non-periodic Transmit Request Queue. The half or completely empty depends on the Non-periodic TxFIFO Empty Level bit in the Core AHB Configuration register.
Bit 4	RXFLVL	0x0	ro	Accessible in both host and device modes RxFIFO Non-Empty Indicates that there is at least one packet to be read from the receive FIFO.
Bit 3	SOF	0x0	rw1c	Accessible in both host and device modes Start of Frame In host mode, the controller sets this bit to indicate that an SOF (full-speed) or Keep-Alive (low-speed) is transmitted on the USB bus. The application must set this bit to 1 to clear this interrupt. In device mode, the controller sets this bit to indicate that an SOF token has been received on the USB bus. The application must read the Device Status register to get the current frame number. This interrupt can be generated only when the controller is running in FS mode. This bit is set by the controller. The application must write 1 to clear this bit. Note: Reading this register immediately after power-on reset may return the value 0x1. If this register is read as 0x1 immediately after power-on reset, it does not mean that an SOF has been transmitted (in host mode) or received (in device mode). The reading of this register is valid only when an effective connection has been established between the host and the device. If this bit is set after power-on reset, the application can clear this bit.
Bit 2	OTGINT	0x0	ro	Accessible in both host and device modes OTG interrupt The controller sets this bit to indicate that an OTG protocol event is generated. The application must read the OTGFS_GOTGINT register to determine the exact source that caused this interrupt. The application must clear the corresponding status bit in the OTGFS_GOTGINT register to clear this bit.
Bit 1	MODEMIS	0x0	rw1c	Accessible in both host and device modes

				Mode mismatch interrupt The controller sets this bit when the application is attempting to access: A host-mode register, when the controller is running in device mode A device-mode register, when the controller is running in host mode An OKAY response occurs when the register access is completed on the AHB, but it is ignored by the controller internally, and does not affect the operation of the controller. This bit can be set by the controller only. The application must write 1 to clear this bit.
Bit 0	CURMOD	0x0	ro	Accessible in both host and device modes Current mode of operation This bit indicates the current mode. 0: Device mode 1: Host mode

20.6.3.7 OTGFS interrupt mask register (OTGFS_GINTMSK)

This register works with the Interrupt Register to interrupt the application. When an interrupt bit is masked, the interrupt related to this interrupt bit is not generated. However, the Interrupt Register bit corresponding to this interrupt is still set.

Interrupt mask: 0

Interrupt unmask: 1

Bit	Register	Reset value	Type	Description
Bit 31	WKUPINTMSK	0x0	rw	Accessible in both host and device modes Resume/Remote wakeup detected interrupt mask
Bit 30	Reserved	0x0	resd	Kept at its default value.
Bit 29	DISCONINTMSK	0x0	rw	Accessible in both host and device modes Disconnect detected interrupt mask
Bit 28	CONIDSCHGMSK	0x0	rw	Accessible in both host and device modes Connector ID status change mask
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	PTXFEMPMSK	0x0	rw	Accessible in host mode only Periodic TxFIFO empty mask
Bit 25	HCHINTMSK	0x0	rw	Accessible in host mode only Host channels interrupt mask
Bit 24	PRTINTMSK	0x0	ro	Accessible in host mode only Host port interrupt mask
Bit 23: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	INCOMPIPMSK INCOMPISOOUTMSK	0x0	rw	Incomplete periodic transfer mask Accessible in host mode only Incomplete isochronous OUT transfer mask Accessible in device mode only
Bit 20	INCOMISOINMSK	0x0	rw	Accessible in device mode only Incomplete isochronous IN transfer mask
Bit 19	OEPTINTMSK	0x0	rw	Accessible in device mode only OUT endpoints interrupt mask
Bit 18	IEPTINTMSK	0x0	rw	Accessible in device mode only IN endpoints interrupt mask
Bit 17	Reserved	0x0	rw	Kept at its default value.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EOPFMSK	0x0	rw	Accessible in device mode only End of periodic frame interrupt mask
Bit 14	ISOOUTDROPMSK	0x0	rw	Device only isochronous OUT packet dropped interrupt mask
Bit 13	ENUMDONEMSK	0x0	rw	Accessible in device mode only Enumeration done mask
Bit 12	USBRSTMSK	0x0	rw	Accessible in device mode only USB Reset mask
Bit 11	USBSUSPMSK	0x0	rw	Accessible in device mode only

				USB suspend interrupt mask
Bit 10	ERLYSUSPMSK	0x0	rw	Accessible in device mode only Early suspend interrupt mask
Bit 9: 8	Reserved	0x0	resd	Kept at its default value.
Bit 7	GOUTNAKEFFMSK	0x0	rw	Accessible in device mode only Global OUT NAK effective mask
Bit 6	GINNAKEFFMSK	0x0	rw	Accessible in device mode only Global Non-periodic IN NAK effective mask
Bit 5	NPTXFEMPMSK	0x0	rw	Accessible in both host and device modes Non-periodic TxFIFO empty mask
Bit 4	RXFLVLMSK	0x0	rw	Accessible in both host and device modes Receive FIFO Non-empty mask
Bit 3	SOFMSK	0x0	rw	Accessible in both host and device modes Start of Frame mask
Bit 2	OTGINTMSK	0x0	rw	Accessible in both host and device modes OTG interrupt mask
Bit 1	MODEMISMSK	0x0	rw	Accessible in both host and device modes Mode mismatch interrupt mask
Bit 0	Reserved	0x0	resd	Kept at its default value.

20.6.3.8 OTGFS receive status debug read/OTG status read and POP registers (OTGFS_GRXSTSR / OTGFS_GRXSTSP)

A read to the Receive Status Debug Read register returns the data of the top of the Receive FIFO. A read to the Receive Status Read and Pop register pops the data of the top of the Receive FIFO.

The receive status contents are interpreted differently in host and device modes. Then controller ignores the receive status pop/read when the receive FIFO is empty and returns the value of 0x0000 0000. The application can only pop the receive status FIFO when the receive FIFO non-empty bit of the Core Interrupt register is set.

Host mode:

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x000	resd	Kept at its default value.
Bit 20: 17	PKTSTS	0x0	ro	Packet status Indicates the status of the received data packet. 0010: IN data packet received 0011: IN transfer completed (triggers an interrupt) 0101: Data toggle error (triggers an interrupt) 0111: Channel halted (triggers an interrupt) Others: Reserved Reset value: 0
Bit 16: 15	DPID	0x0	ro	Data PID Indicates the data PID of the received data packet. 00: DATA0 10: DATA1 01: DATA2 11: MDATA Reset value: 0
Bit 14: 4	BCNT	0x000	ro	Byte count Indicates the byte count of the received IN data packet.
Bit 3: 0	CHNUM	0x0	ro	Channel number Indicates the channel number to which the currently received data packet belongs.

Device mode:

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 21	FN	0x0	ro	Frame number Indicates the least significant 4 bits of the frame number of the data packet received on the USB bus. This field is applicable only when the synchronous OUT endpoints are supported.
Bit 20: 17	PKTSTS	0x0	ro	Packet status Indicates the status of the received data packet. 0001: Global OUT NAK (triggers an interrupt) 0010: OUT data packet received 0011: OUT transfer completed (triggers an interrupt) 0100: SETUP transaction completed (triggers an interrupt) 0110: SETUP data packet received Others: Reserved
Bit 16: 15	DPID	0x0	ro	Data PID Indicates the data PID of the received OUT data packet. 00: DATA0 10: DATA1 01: DATA2 11: MDATA
Bit 14: 4	BCNT	0x000	ro	Byte count Indicates the byte count of the received data packet.
Bit 3: 0	EPTNUM	0x0	ro	Endpoint number Indicates the endpoint number to which the currently received data packet belongs.

20.6.3.9 OTGFS receive FIFO size register (OTGFS_GRXFSIZ)

The application can program the SRAM size that must be allocated to the receive FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	RXFDEP	0x0200	ro/rw	RxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 512 The power-on reset value of this register is defined as the largest receive data FIFO depth during the configuration.

20.6.3.10 OTGFS non-periodic Tx FIFO size (OTGFS_GNPTXFSIZ)/Endpoint 0 Tx FIFO size registers (OTGFS_DIEPTXF0)

The application can program the SRAM size and start address of the non-periodic transmit FIFO. The fields of this register varies with host mode or device mode.

Host:

Bit	Register	Reset value	Type	Description
Bit 31: 16	NPTXFDEP	0x0000	ro/rw	Non-periodic TxFIFO depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 256
Bit 15: 0	NPTXFSTADDR	0x0200	ro/rw	Non-periodic transmit SRAM start address This field contains the memory start address of the Non-periodic Transmit FIFO SRAM.

Device:

Bit	Register	Reset value	Type	Description
Bit 31: 16	INEPT0TXDEP	0x0000	ro/rw	N Endpoint TxFIFO 0 depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 256
Bit 15: 0	INEPT0TXSTADDR	0x0200	ro/rw	IN Endpoint FIFO0 transmit SRAM start address This field contains the memory start address of the IN Endpoint FIFO0 transmit SRAM.

20.6.3.11 OTGFS non-periodic Tx FIFO size/request queue status register (OTGFS_GNPTXSTS)

This register is valid in host mode only. It is a read-only register that contains the available space information for the Non-periodic TxFIFO and the Non-periodic Transmit Request Queue.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 24	NPTXQTOP	0x00	ro	Top of the Non-periodic transmit request queue Indicates that the MAC is processing the request from the non-periodic transmit request queue. Bit [30: 27]: Channel/Endpoint number Bit [26: 25]: 00: IN/OUT token 01: Zero-length transmit packet (device IN/host OUT) 10: PING/CSPLIT token 11: Channel halted command Bit [24]: Terminate (last request for the selected channel/endpoint)
Bit 23: 16	NPTXQSPCAVAIL	0x08	ro	Non-periodic transmit request queue space available Indicates the amount of space available in the non-periodic transmit request queue. This queue supports both IN and OUT requests in host mode. 00: Non-periodic transmit request queue is full 01: 1 location available 02: 2 locations available N: n locations available (0 ≤ n ≤ 8) Others: Reserved Reset value: Configurable
Bit 15: 0	NPTXFSPCAVAIL	0x0200	ro	Non-periodic TxFIFO space available Indicates the amount of space available in the non-periodic TxFIFO. Values are in terms of 32-bit words. 00: Non-periodic transmit FIFO is full 01: 1 location available 02: 2 locations available N: n locations available (0 ≤ n ≤ 256) Others: Reserved Reset value: Configurable

20.6.3.12 OTGFS general controller configuration register (OTGFS_GCCFG)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21	VBUSIG	0x0	rw	VBUS ignored When this bit is set, the OTGFS controller does not monitor the Vbus pin voltage, and assumes that the Vbus is always active in both host and device modes, and leaves the Vbus pin for other purposes. 0: Vbus is not ignored 1: Vbus is ignored, and is deemed as always active
Bit 20	SOFOUTEN	0x0	rw	SOF output enable 0: No SOF pulse output 1: SOF pulse output on PIN
Bit 19	BVALIDSESEN	0x0	rw	Bvalid sense enable 0: Disabled 1: Enabled
Bit 18	AVALIDSESEN	0x0	rw	Avalid sense enable 0: Disabled 1: Enabled
Bit 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	PWRDOWN	0x0	rw	Power down This bit is used to activate the transceiver in transmission/reception. It must be pre-configured to allow USB communication. 0: Power down enable 1: Power down disable (Transceiver active)
Bit 15: 0	Reserved	0x0000	resd	Kept at its default value.

20.6.3.13 OTGFS controller ID register (OTGFS_GUID)

This is a read-only register containing the production ID.

Bit	Register	Reset value	Type	Description
31: 0	USERID	0x0000 1000	rw	Product ID field The application can program the ID field.

20.6.3.14 OTGFS host periodic Tx FIFO size register (OTGFS_HPTXFSIZ)

This register contains the size and memory start address of the periodic transmit FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	PTXFSIZE	0x02000	ro/rw	Host periodic TxFIFO depth Values are in terms of 32-bit words. Minimum value is 16 Maximum value is 512
Bit 15: 0	PTXFSTADDR	0x0600	ro/rw	Host Periodic TxFIFO start address The power-on reset value of this register is the sum of the largest receive FIFO depth and the largest non-periodic transmit FIFO depth.

20.6.3.15 OTGFS device IN endpoint Tx FIFO size register (OTGFS_DIEPTXFn) (x=1…3, where n is the FIFO number)

This register holds the depth and memory start address of the IN endpoint transmit FIFO in device mode. Each of the FIFOs contains an IN endpoint data. This register can be used repeatedly for instantiated IN endpoint FIFO1~15 . The GNPTXFSIZ register is used to program the depth and memory start address of the IN endpoint FIFO 0.

Bit	Register	Reset value	Type	Description
Bit 31: 16	INEPTXFDEP	0x0200	ro/rw	IN Endpoint TxFIFO depth Values are in terms of 32-bit words. Minimum value is 16 Maximum value is 512 The reset value is the maximum possible IN endpoint transmit FIFO depth
Bit 15: 0	INEPTXFSTADDR	0x0400	ro/rw	IN Endpoint FIFOn transmit SRAM start address This field contains the SRAM start address of the IN endpoint n transmit FIFO

20.6.4 Host-mode registers

Host-mode registers affect the operation of the controller in host mode. Host-mode register are not accessible in device mode (as the results are undefined in device mode). Host-mode registers contain as follows:

20.6.4.1 OTGFS host mode configuration register (OTGFS_HCFG)

This register is used to configure the controller after power-on. Do not change this register after initialization.

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 2	FSLSSUPP	0x0	ro	FS- and LS-only support The application uses this bit to control the controller's enumeration speed. With this bit, the application can make the controller enumerate as a full-speed host mode, even if the connected device supports high-speed communication. Do not change this bit after initial programming. 0: FS/LS, depending on the largest speed supported by the connected device. 1: FS/LS-only, even if the connected device supports high-speed.
Bit 1: 0	FSLSPCLKSEL	0x0	rw	FS/LS PHY clock select When the controller is in FS host mode: 01: PHY clock is running at 48MHz Others: Reserved When the controller is in LS host mode: 00: Reserved 01: PHY clock is running at 48 MHz 10: PHY clock is running at 6 MHz. If 6 MHz clock is selected, reset must be done by software. 11: Reserved

20.6.4.2 OTGFS host frame interval register (OTGFS_HFIR)

This register is used to program the frame interval at current enumeration speed.

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	HFIRRLDCTRL	0x0	rw	Reload control This bit is used to disable/enable dynamic reload for the host frame register at runtime. 1: Reload control disable 0: Reload control enable This bit must be configured at initialization. Do not change its value at runtime.
Bit 15: 0	FRINT	0xEA60	rw	Frame interval The application uses this filed to program the interval between two consecutive SOFs (full speed) The number of PHY locks in this field indicates the frame interval. The application can write a value to the host frame interval register only after the port enable bit in the host port control and status register has been set. If no value is programmed, the controller calculates the value based on the PHY clock frequency defined in the FS/LS PHY clock select bit of the host configuration register. Do not change the value of this field after initial configuration. 1 ms * (FS/LS PHY clock frequency)

20.6.4.3 OTGFS host frame number/frame time remaining register (OTGFS_HFNUM)

This register indicates the current frame number, and also the time remaining in the current frame (in terms of the number of PHY clocks).

Bit	Register	Reset value	Type	Description
Bit 31: 16	FTREM	0x0000	ro	Frame time remaining Indicates the time remaining in the current frame (FS/HS), in terms of the number of PHY clocks. This field decrements with the number of PHY clocks. When it reaches zero, this filed is reloaded with the value of the frame interval register, and a new SOF is transmitted on the USB bus.
Bit 15: 0	FRNUM	0x3FFF	ro	Frame number This field increments every time a new SOP is transmitted on the USB bus, and is cleared to 0 when the value reaches 16'h3FFF.

20.6.4.4 OTGFS host periodic Tx FIFO/request queue register (OTGFS_HPTXSTS)

This is a ready-only register containing the free space information of the period Tx FIFO and the periodic transmit request queue.

Bit	Register	Reset value	Type	Description
Bit 31: 24	PTXQTOP	0x00	ro	Top of the periodic transmit request queue) Indicates that the MAC is processing the request from the periodic transmit request queue. This register is used for debugging. Bit [31]: Odd/Even frame 0: Transmit in even frame 1: Transmit in odd frame Bit [30: 27]: Channel/Endpoint number Bit [26: 25]: Type 00: IN/OUT 01: Zero-length packet 10: Reserved 11: Channel command disable Bit [24]: Terminate (last request for the selected channel

				or endpoint)
Bit 23: 16	PTXQSPCAVAIL	0x08	ro	Periodic transmit request queue space available Indicates the number of free space available to be written in the periodic transmit request queue. This queue contains both IN and OUT requests. 00: Periodic transmit request queue is full 01: 1 space available 10: 2 space available N: n space available (0 ≤ n ≤ 8) Others: Reserved
Bit 15: 0	PTXFSPCAVAIL	0x0100	rw	Periodic transmit data FIFO space available Indicates the number of free space available to be written in the periodic transmit FIFO, in terms of 32-bit words. 0000: Periodic transmit FIFO is full 0001: 1 space available 0010: 2 space available N: n space available (0 ≤ n ≤ 512) Others: Reserved

20.6.4.5 OTGFS host all channels interrupt register (OTGFS_HAINT)

When a flag event occurs on a channel, the host all channels interrupt register interrupts the application through the host channels interrupt bit of the controller interrupt register, as shown in *Figure 20-2*. There is one interrupt bit for each channel, up to 16 bits. The application sets or clears this register by setting or clearing the appropriate bit in the corresponding host channel-n interrupt register.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	HAINT	0x0000	ro	Channel interrupts One bit per channel: bit 0 for channel 0, bit 15 for channel 15.

20.6.4.6 OTGFS host all channels interrupt mask register (OTGFS_HAINTMSK)

The host all channels interrupt mask register ((OTGFS_HAINTMSK)) works with the host all channels interrupt register (OTGFS_HAINT) to interrupt the application when an event occurs on a channel. There is one interrupt mask bit per one channel, 16 bits in total.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	HAINTMSK	0x0000	rw	Channel interrupt mask One bit per channel: bit 0 for channel 0, bit 15 for channel 15.

20.6.4.7 OTGFS host port control and status register (OTGFS_HPRT)

This register is valid only in host mode. Currently, the OTG host supports only one port.

This register contains USB port-related information such as USB reset, enable, suspend, resume, connect status and test mode, as show in *Figure 21-2*.

The register of type rw1c can interrupt the application through the host port interrupt bit in the controller interrupt register. Upon a port interrupt, the application must read this register and clear the bit that caused the interrupt. For the register of type rw1c, the application must write 1 to clear the interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at its default value.
Bit 18: 17	PRTSPD	0x0	ro	Port speed Indicates the speed of the device connected to this port. 00: Reserved 01: Full speed 10: Low speed 11: Reserved
Bit 16: 13	PRTTSTCTL	0x0	rw	Port test control The application writes a non-zero value to this field to put the port into test mode, and the port gives a corresponding signal.

				0000: Test mode disabled 0001: Test_J mode 0010: Test_K mode 0011: Test_SE0_NAK mode 0100: Test_Packet mode 0101: Test_Force_Enable Others: Reserved
Bit 12	PRTPWR	0x0	rw	Port power The application uses this bit to control power supply to this port (by writing 1 or 0) 0: Power off 1: Power on Note: This bit is not associated with interfaces. The application must follow the programming manual to set this bit for various interfaces.
Bit 11: 10	PRTLNSTS	0x0	ro	Port line status Indicates the current logic status of the USB data lines. Bit [10]: Logic level of D+ Bit [11]: Logic level of D–
Bit 9	Reserved	0x0	resd	Kept at its default value.
Bit 8	PRTRST	0x0	rw	Port reset When this bit is set by the application, a reset sequence is started on this port. The application must calculate the time required for the reset sequence, and clear this bit after the reset sequence is complete. 0: Port not in reset 1: Port in reset The application must keep this bit set for a minimum duration defined in Section 7.1.7.5 of USB 2.0 specification to start a reset on the port. In addition to this, the application can make this bit set for another 10 ms to the minimum duration, before clearing this bit. There is no maximum limit set by the USB standard.
Bit 7	PRTSUSP	0x0	rw1s	Port suspend The application sets this bit to put this port in suspend mode. In this case, the controller only stops sending SOF. The application must set the port clock stop bit in order to disable the PHY clock. The read value of this bit reflects the current suspend status of the port. This bit is cleared by the controller when a remote wakeup signal is detected or when the application sets the port reset bit or port resume bit in this register, or sets the resume/remote wakeup detected interrupt bit or disconnect detected interrupt bit in the controller interrupt register. The controller can still clear this bit, even if the device is disconnected with the host. 0: Port not in suspend mode 1: Port in suspend mode
Bit 6	PRTRES	0x0	rw	Port resume The application sets this bit to drive resume signaling on the port. The controller continues to trigger the resume signal until the application clears this bit. If the controller detects a USB remote wakeup sequence (as indicated by the port resume/remote wakeup detected interrupt bit of the controller interrupt register), the controller starts driving resume signaling without the intervention of the application. The read value of this bit indicates whether the controller is currently driving resume signaling. 0: No resume triggered 1: Resume triggered
Bit 5	PRTOVRCCHNG	0x0	rw1c	Port overcurrent change The controller sets this bit when the status of the port

				overcurrent active bit (bit 4) in this register changes. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 4	PRTOVRCACT	0x0	ro	Port overcurrent active Indicates the overcurrent status of the port. 0: No overcurrent 1: Overcurrent condition
Bit 3	PRTENCHNG	0x0	rw1c	Port enable/disable change The controller sets this bit when the status of the port enable bit 2 in this register changes. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 2	PRTENA	0x0	rw1c	Port enable A port is enabled only by the controller after a reset sequence. This port is enabled by an overcurrent condition, a disconnected condition ro by the application. The application cannot set this bit by a register write operation. It can only clear this bit to disable the port. This bit does not trigger any interrupt. 0: Port disabled 1: Port enabled
Bit 1	PRTCONDET	0x0	rw1c	Port connect detected On a device connection detected, the controller sets this bit using the host port interrupt bit in the controller register. This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 0	PRTCONSTS	0x0	ro	Port connect status 0: No device is connected to the port 1: A device is connected to the port

20.6.4.8 OTGFS host channelx characteristics register (OTGFS_HCCHARx) (x = 0...8, where x= channel number)

Bit	Register	Reset value	Type	Description
Bit 31	CHENA	0x0	rw1s	Channel enable This bit is set by the application and cleared by the OTG host. 0: Channel disabled 1: Channel enabled
Bit 30	CHDIS	0x0	rw1s	Channel disable The application sets this bit to stop transmitting or receiving data on a channel, even before the transfer on that channel is complete. The application must wait for the generation of the channel disabled interrupt before treating the channel as disabled.
Bit 29	ODDFRM	0x0	rw	Odd frame This bit is set / cleared by the application to indicate that the OTG host must perform a transfer in an odd frame. This bit is applicable for periodic transfers (synchronous and interrupt) only. 0: Even frame 1: Odd frame
Bit 28: 22	DEVADDR	0x00	rw	Device address This field is used to select the device that can serve as the data source or receiver.
Bit 21: 20	MC	0x0	rw	Multi count (MC) This field indicates to the host the number of transfers that must be performed per frame for the periodic endpoint. 00: Reserved. This field generates undefined results. 01: 1 transaction 10: 2 transactions per frame 11: 3 transactions per frame This field must be set to at least 0x01.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type

				Indicates the transfer type selected. 00: Control transfer 01: Synchronous transfer 10: Bulk transfer 11: Interrupt transfer
Bit 17	LSPDDEV	0x0	rw	Low-speed device The application sets this bit to indicate that this channel is communicating to a low-speed device.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	EPTDIR	0x0	rw	Endpoint direction Indicates whether the transfer is in IN or OUT. 0: OUT 1: IN
Bit 14: 11	EPTNUM	0x0	rw	Endpoint number Indicates the endpoint number on the device (serving as data source or receiver)
Bit 10: 0	MPS	0x000	rw	Maximum packet size Indicates the maximum packet size of the corresponding port.

20.6.4.9 OTGFS host channelx interrupt register (OTGFS_HCINTx) (x = 0...8, where x= channel number)

This register contains the status of a channel related to USB and AHB events, as shown in ***Figure 20-2***.

The application must read this register when the host channels interrupt bit is set in the controller interrupt register. Before reading this register, the application must read the host all channels interrupt register to get the exact channel number of the host channel-n interrupt register. The application must clear the corresponding bit in this register to clear the corresponding bits in the OTGFS_HAIN and OTGFS_GINTSTS registers.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Kept at its default value.
Bit 10	DTGLERR	0x0	rw1c	Data toggle error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 9	FRMOVRUN	0x0	rw1c	Frame overrun This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 8	BBLERR	0x0	rw1c	Babble error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 7	XACTERR	0x0	rw1c	Transaction error Indicates one of the following errors occurred on the USB bus: CRC check failure Timeout Bit stuffing error EOP error This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 6	Reserved	0x0	resd	Kept at its default value.
Bit 5	ACK	0x0	rw1c	ACK response received/Transmitted interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 4	NAK	0x0	rw1c	NAK response received interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 3	STALL	0x0	rw1c	STALL response received interrupt This bit can only be set by the controller. The application must write 1 to clear this bit.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	CHHLTD	0x0	rw1c	Channel hated Indicates that the transfer completed abnormally either because of any transfer error or in response to a disable

				request by the application.
Bit 0	XFERC	0x0	rw1c	Transfer completed Transfer completed normally, without any error. This bit can only be set by the controller. The application must write 1 to clear this bit.

20.6.4.10 OTGFS host channelx interrupt mask register (OTGFS_HCINTMSKx) (x = 0...8, where x= channel number)

This register is used to mask the channels described in the previous section.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x000000	resd	Kept at its default value.
Bit 10	DTGLERRMSK	0x0	rw	Data toggle error mask
Bit 9	FRMOVRUNMSK	0x0	rw	Frame overrun mask
Bit 8	BBLERRMSK	0x0	rw	Babble error mask
Bit 7	XACTERRMSK	0x0	rw	Transaction error mask
Bit 6	NYETMSK	0x0	rw	NYET response received interrupt mask
Bit 5	ACKMSK	0x0	rw	ACK response received/transmitted interrupt mask
Bit 4	NAKMSK	0x0	rw	NAK response received interrupt mask
Bit 3	STALLMSK	0x0	rw	STALL response received interrupt mask
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	CHHLTDMSK	0x0	rw	Channel halted mask
Bit 0	XFERCMSK	0x0	rw	Transfer completed mask

20.6.4.11 OTGFS host channelx transfer size register (OTGFS_HCTSIZx) (x = 0...8, where x= channel number)

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	PID	0x0	rw	PID (Pid) The application programs this field with the type of PID used for the initial transfer. The host controls this filed for the rest of transfers. 00: DATA0 01: DATA2 10: DATA1 11: MDATA(non-control)/SETUP(control)
Bit 28: 19	PKTCNT	0x000	rw	Packet count The application programs this field with the expected number of packets to be transmitted or received. The host decrements the packet count on every successful transmission or reception of an OUT/IN packet. When this count reaches zero, the application is interrupted to indicate normal completion of the transfer.
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer size For an OUT transfer, this field indicates the number of data bytes the host sends during a transfer. For an IN transfer, this field indicates the buffer size that the application has reserved for the transfer. For an IN transfer (periodic and non-periodic), the application must program this field as an integer multiple of the maximum packet size.

20.6.5 Device-mode registers

These registers are applicable in device mode only. They are not supported in host mode due to unknown access results. Some of the registers affect all the endpoints, while some affect only one endpoint.

20.6.5.1 OTGFS device configure register (OTGFS_DCFG)

This register configures the controller in device mode after power-on or after certain control commands or enumeration. Do not change this register after initial programming.

Bit	Register	Reset value	Type	Description
Bit 31: 13	Reserved	0x0110	resd	Kept at its default value.
Bit 12: 11	PERFRINT	0x0	rw	Periodic frame interval

				This field indicates the time within a frame at which the periodic frame end interrupt is generated. The application can use this interrupt to determine if the synchronous transfer has been completed in a frame. 00: 80% of the frame interval 01: 85% of the frame interval 10: 90% of the frame interval 11: 95% of the frame interval
Bit 10: 4	DEVADDR	0x00	rw	Device address The application must program this field every time a SetAddress command is received.
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	NZSTSOUTHSHK	0x0	rw	Non-zero-length status OUT handshake The application can use this field to select the handshake the controller sends on receiving a non-zero-length data packet during a control transfer' status stage. 1: Send a STALL handshake on a non-zero-length status OUT transfer and do not send the received OUT packet to the application 0: Send the received OUT packet to the application (zero-length or non-zero-length), and send a handshake based on the NAK and STALL bits in the device endpoint control register.
Bit 1: 0	DEVSPD	0x0	rw	Device speed This field indicates the speed at which the application needs the controller to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the entire sequence is complete, and is based on the speed of the USB host to which the controller is connected. 00: Reserved 01: Reserved 10: Reserved 11: Full speed (USB1.1 transceiver, clock is 48MHz)

20.6.5.2 OTGFS device control register (OTGFS_DCTL)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	PWROPRGDNE	0x0	wo	Power-on programming done The application uses this bit to indicate that the register configuration is complete after a wakeup from power-down mode.
Bit 10	CGOUTNAK	0x0	wo	Clear global OUT NAK Writing 1 to this bit clears the global OUT NAK.
Bit 9	SGOUTNAK	0x0	wo	Set global OUT NAK Writing to this bit sets the global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set this bit only after checking that the global OUT NAK effective bit in the controller interrupt register is cleared.
Bit 8	CGNPINNAK	0x0	wo	Clear Global Non-periodic IN NAK Writing to this bit clears the global Non-periodic OUT NAK.
Bit 7	SGNPINNAK	0x0	wo	Set global Non-periodic IN NAK Writing to this bit sets the global Non-periodic OUT NAK. The application uses this bit to send a NAK handshake on all non-periodic IN endpoints. The application must set this bit only after checking that the global IN NAK effective bit in the controller interrupt register is cleared.
Bit 6: 4	TSTCTL	0x0	rw	Test control 000: Test mode disabled 001: Test_J mode 010: Test_K mode 011: Test_SE0_NAK mode

				100: Test_Packet mode 101: Test_Force_Enable Others: Reserved
Bit 3	GOUTNAKSTS	0x0	ro	Global OUT NAK status 0: A handshake is sent based on the FIFO status, NAK and STALL bit settings. 1: No data is written to the receive FIFO, irrespective of space availability. Sends a NAK handshake on all packets (except on SETUP transfers). Drops all synchronous OUT packets.
Bit 2	GNPINNAKSTS	0x0	ro	Global Non-periodic IN NAK status 0: A handshake is sent based on the data status in the transmit FIFO 1: A NAK handshake is sent on all non-periodic IN endpoints, irrespective of the data status in the transmit FIFO.
Bit 1	SFTDISCON	0x0	rw	Software disconnect The application uses this bit to indicate the OTGFS controller to perform software disconnected. Once this bit is set, the host finds the device disconnected, and the device does not receive signals on the USB bus. The controller stays in the disconnected state until the application clears this bit. 0: Normal operation. When this bit is cleared after a software disconnect, the controller issues a device connect event to the host. Then the USB host restarts device enumeration.
Bit 0	RWKUPSIG	0x0	rw	Remote wakeup signaling When this bit is set by the application, the controller initiates a remote signal to wakeup the USB host. The application must set this bit to indicate the controller to exit the suspend mode. Per USB2.0 standards, the application must clear this bit 1-15 ms after setting it.

Table 20-5 lists the minimum duration at which the software disconnect bit must be set in various states for the USB host to detect a device disconnect. To accommodate clock jitter, it is advised that the application adds some extra delay to the specified minimum duration.

Table 20-5 Minimum duration for software disconnect

Operating speed	Device state	Minimum duration
Full speed	Suspend	1ms + 2.5us
Full speed	Idle	2.5us
Full speed	No idle or suspend (performing transfers)	2.5us

20.6.5.3 OTGFS device status register (OTGFS_DSTS)

This register indicates the status of the controller related to OTGFS events. It must be read on interrupt events from the device all interrupts register (OTGFS_DAINT).

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at its default value.
Bit 21: 8	SOFFN	0x0000	ro	Frame number of the received SOF Note: The read value of this field immediately after power-on reset reflects a non-zero value. If a non-zero value is returned after reading this field immediately after power-on reset, it does not mean that the host has received a SOP. The read value of this field is valid only when the host is connected to the device.
Bit 7: 4	Reserved	0x1	resd	Kept at its default value.
Bit 3	ETICERR	0x0	ro	Erratic error This error causes the controller to enter suspend mode, and interrupt is generated with the early suspend bit of the

				controller interrupt register. If the early suspend is asserted due to an erratic error, the application can only perform a software disconnect recover.
Bit 2: 1	ENUMSPD	0x0	ro	Enumerated speed Indicates the speed at which the controller has determined after speed detection through a sequence. 01: Reserved 10: Reserved 11: Full speed (PHY clock is running at 48MHz) Others: Reserved
Bit 0	SUSPSTS	0x0	ro	Suspend status In device mode, this bit is set as long as a suspend condition is detected on the USB bus. The controller enters the suspend state when there is no activity on the USB bus. The controller exits the suspend state on the following conditions: When there is an activity on the USB bus When the application writes to the remote wakeup signal bit in the device control register.

20.6.5.4 OTGFS device OTGFSIN endpoint common interrupt mask register (OTGFS_DIEPMSK)

This register works with each of the device IN endpoint interrupt register for all endpoints to generate an IN endpoint interrupt. The IN endpoint interrupt for a specific status in the OTGFS_DIEPINTx register can be masked by writing to the corresponding bit in the OTGFS_DIEPMSK register. Status bits are masked by default.

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	BNAINMSK	0x0	rw	BNA interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 8	TXFIFOUDRMSK	0x0	rw	FIFO underrun mask 0: Interrupt masked 1: Interrupt unmasked
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	INEPTNAKMSK	0x0	rw	IN endpoint NAK effective mask 0: Interrupt masked 1: Interrupt unmasked
Bit 5	INTKNEPTMISMSK	0x0	rw	IN token received with EP mismatch mask 0: Interrupt masked 1: Interrupt unmasked
Bit 4	INTKNTXFEMPMSK	0x0	rw	IN token received when TxFIFO empty mask 0: Interrupt masked 1: Interrupt unmasked
Bit 3	TIMEOUTMSK	0x0	rw	Timeout condition mask (Non-isochronous endpoints)) 0: Interrupt masked 1: Interrupt unmasked
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISMSK	0x0	rw	Endpoint disabled interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 0	XFERCMSK	0x0	rw	Transfer completed interrupt mask 0: Interrupt masked 1: Interrupt unmasked

20.6.5.5 OTGFS device OUT endpoint common interrupt mask register (OTGFS_DOEPMSK)

This register works with each of the OTGFS_DOEPINTx registers for all endpoints to generate an OUT endpoint interrupt. Each of the bits in the OTGFS_DOEPINTx registers can be masked by writing to the register. All interrupts are masked by default.

Bit	Register	Reset value	Type	Description
Bit 31: 10	Reserved	0x000000	resd	Kept at its default value.
Bit 9	BNAOUTMSK	0x0	rw	BNA interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 8	OUTPERRMSK	0x0	rw	OUT packet error mask 0: Interrupt masked 1: Interrupt unmasked
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6	B2BSETUPMSK	0x0	rw	Back-to-back SETUP packets received mask 0: Interrupt masked 1: Interrupt unmasked
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	OUTTEPDMSK	0x0	rw	OUT token received when endpoint disabled mask 0: Interrupt masked 1: Interrupt unmasked
Bit 3	SETUPMSK	0x0	rw	SETUP phase done mask Applies to control endpoints only. 0: Interrupt masked 1: Interrupt unmasked
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISMSK	0x0	rw	Endpoint disabled interrupt mask 0: Interrupt masked 1: Interrupt unmasked
Bit 0	XFERCMSK	0x0	rw	Transfer completed interrupt mask 0: Interrupt masked 1: Interrupt unmasked

20.6.5.6 OTGFS device all endpoints interrupt mask register (OTGFS_DAINT)

When an event occurs on an endpoint, The IN/OUT endpoint interrupt bits in the OTGS_DAINT register can be used to interrupt the application. There is one interrupt pit per endpoint, up to 8 interrupt bits for OUT endpoints and 8 bits for IN endpoints. For a bidirectional endpoint, the corresponding IN and OUT interrupt bits are used at the same time. The corresponding bits in this register are set and cleared when the application sets and clears the bits in the corresponding device endpoint-x interrupt register.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at its default value.
Bit 23: 16	OUTEPTINT	0x0000	ro	OUT endpoint interrupt bits One OUT endpoint per bit. Bit 16 for OUT endpoint 0, bit 18 for OUT endpoint 2.
Bit 15: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTINT	0x0000	ro	IN endpoint interrupt bits One IN endpoint per bit. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7.

20.6.5.7 OTGFS all endpoints interrupt mask register (OTGFS_DAINTMSK)

When an event occurs on a device endpoint, the device endpoint interrupt mask register works with the device endpoint interrupt register to interrupt the application. However, the device all endpoints interrupt register corresponding to this interrupt is still set.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at its default value.
Bit 23: 16	OUTEPTMSK	0x0000	rw	OUT EP interrupt mask bits One OUT endpoint per bit. Bit 16 for OUT endpoint 0, bit 18 for OUT endpoint 2. 0: Interrupt masked 1: Interrupt unmasked
Bit 15: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTMSK	0x0000	rw	IN EP interrupt mask bits One IN endpoint per bit. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7. 0: Interrupt masked 1: Interrupt unmasked

20.6.5.8 OTGFS device IN endpoint FIFO empty interrupt mask register (OTGFS_DIEPEMPMSK)

This register works with the TXFE_OTGFS_DIEPINTx register to generate an interrupt.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x0000	resd	Kept at its default value.
Bit 7: 0	INEPTXFEMSK	0x0000	rw	IN endpoint Tx FIFO empty interrupt mask bits These bits serve as mask bits for the device IN endpoint interrupt register. A transmit FIFO empty interrupt bit per IN endpoint. Bit 0 for IN endpoint 0, bit 7 for IN endpoint 7. 0: Interrupt masked 1: Interrupt unmasked

20.6.5.9 OTGFS device control IN endpoint 0 control register (OTGFS_DIEPCTL0)

This section describes the control IN endpoint 0 control register. Nonzero control endpoint uses registers for endpoints 1-7.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start data transmission on the endpoint 0. The controller clears this bit before generating the following interrupts: Endpoint disabled Transfer completed.
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application sets this bit to stop data transmission on an endpoint. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint is enabled.
Bit 29: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit of the endpoint. The application can use this bit to control the transmission of NAK handshakes on the endpoint. The controller also sets this bit when a SETUP data packet is received on the endpoint.
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for the endpoint.

Bit 25: 22	TXFNUM	0x0	rw	TxFIFO number The endpoint 0 can only use FIFO0.
Bit 21	STALL	0x0	rw1s	STALL handshake The application sets this bit, and the controller clears this bit when a SETUP token is received. If a NAK bit, a global non-periodic IN NAK or global OUT NAK bit is set along with this bit, the STALL bit has priority.
Bit 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 18	EPTYPE	0x0	ro	Endpoint type Set to 0 by hardware for control endpoints.
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following: 0: The controller is transmitting non-NAK handshakes based on the FIFO status 1: The controller is transmitting NAK handshakes on this endpoint When this bit is set, either by the application or controller, the controller stops transmitting data, even if there are space available in the receive FIFO. The controller always responds to SETUP data packets with an ACK handshake, irrespective of this bit's setting.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	USBACEPT	0x0	ro	USB active endpoint This bit is always set to 1, indicating that the control endpoint 0 is always active in all configurations and interfaces.
Bit 14: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1: 0	MPS	0x0	rw	Applies to IN and OUT endpoints The application uses this bit to program the maximum packet size for the current logical endpoint. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

20.6.5.10 OTGFS device IN endpoint-x control register (OTGFS_DIEPCTLx) (x=x=1…3, where x is endpoint number)

The application uses this register to control the behavior of the endpoints other than endpoint 0.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start transmitting data on an endpoint. The controller clears this bit before the generation one of the following interrupts on this endpoint: SETUP stage done Endpoint disabled Transfer completed
Bit 30	EPTDIS	0x0	rw1s	Endpoint disable The application sets this bit to stop transmitting data on an endpoint, even if the transfer on that endpoint is incomplete. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint enabled set.
Bit 29	SETD1PID/ SETODDFR	0x0	wo	Set DATA1 PID Applies to interrupt/bulk IN endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA1. Set odd frame Applies to synchronous IN endpoints only. Writing to this bit sets the Even/Odd frame to odd frame. 0: Disabled Set DATA1 PID disabled or Do not force odd frame

				1: Set DATA1 PID enabled or forced odd frame
Bit 28	SETD0PID/ SETEVENFR	0x0	rw	Set DATA0 PID Applies to interrupt/bulk IN endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA0. Set Even frame Applies to synchronous IN endpoints only. Writing to this bit sets the Even/Odd frame to even frame. 0:Disabled Set DATA0 PID disabled or Do not force even frame 1: Set DATA0PID or set the EOFRNUM to even frame
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for the endpoint. The application uses this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a Transfer completed interrupt or after receiving a SETUP packet. Values: 0: Do not set NAK 1: Set NAK
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for this endpoint. 0: Not clear NAK 1: Clear NAK
Bit 25: 22	TXFNUM	0x0	rw	TxFIFO number Allocate FIFO number to the corresponding endpoint. A separate FIFO number is allocated to each valid IN endpoint. This bit applies to IN endpoints only.
Bit 21	STALL	0x0	rw	STALL handshake Applies to non-control, non-synchronous IN and OUT endpoints. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit , global non-periodic IN NAK bit or global OUT NAK bit is set along with this bit, the STALL bit has priority. Only the application can clear this bit, but the controller never. 0: Stall all invalid tokens 1: Stall all valid tokens
Bit 20	Reserved	0x0	resd	Kept at its default value.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type This is the transfer type supported by this logical endpoint. 00: Control 01: Synchronous 10: Bulk 11: Interrupt
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following status: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO to accommodate the incoming data packets. For non-synchronous IN endpoints: the controller stops transmitting data on the endpoint, even if there is data pending in the transmit FIFO. For synchronous IN endpoints: the controller sends a zero-length data packet, even if there is space in the transmit FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.
Bit 16	DPID/ EOFRNUM	0x0	ro	Endpoint data PID Applies to interrupt/bulk IN endpoints only.

				This bit contains the PID of the packet to be transmitted on this endpoint. The application must program the PID of the initial data packet to be received or transmitted on this endpoint, after the endpoint is enabled. The application programs DATA0 or DATA1 PID through the SetD1PID and SetD0PID of this register. 0: DATA0 1: DATA1 Even/Odd frame Applies to synchronous IN endpoints only. Indicates the frame number in which the controller transmits synchronous data on this endpoint. The application must program the even/odd frame number in which it tends to transmit or receive synchronous data through the SETEVNFR and SETODDFR bits in this register. 0: Even frame 1: Odd frame
Bit 15	USBACEPT	0x0	rw	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The controller clears this bit for all endpoints except for endpoint 0 after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers and set this bit. 0: Inactive 1: Active
Bit 14: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10: 0	MPS	0x000	rw	Maximum packet size The application uses this field to set the maximum packet size for the current logical endpoint. The values are in bytes.

20.6.5.11 OTGFS device control OUT endpoint 0 control register (OTGFS_DOEPCTL0)

This section describes the control OUT endpoint 0 control register. Non-zero control endpoints use registers for endpoints 1-7.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable The application sets this bit to start transmitting data on endpoint 0.The controller clears this bit before setting any one of the following interrupts on this endpoint: SETUP stage done Endpoint disabled Transfer completed
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application cannot disable control OUT endpoint 0.
Bit 29: 28	Reserved	0x0	resd	Kept at its default value.
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for this endpoint. The application can use this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a transfer completed interrupt or when a SETUP data packet is received.
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK for the endpoint.
Bit 25: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	STALL	0x0	rw1s	STALL handshake The application sets this bit and the controller clears this bit when a SETUP token is received for this endpoint. If a NAK bit , global non-periodic OIT NAK bit is set along with this bit, the STALL bit has priority. The controller always responds to SETUP data packets, regardless of whether

				this bit is set or not.
Bit 20	SNP	0x0	rw	Snoop mode This bit configures the endpoint to Snoop mode. In this mode, the controller does not check the correctness of OUT packets before transmitting OUT packets to the application memory.
Bit 19: 18	EPTYPE	0x0	ro	Endpoint type Hardware sets this bit to 0 to control endpoint type.
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	USBACEPT	0x1	ro	USB active endpoint This bit is always set to 1, indicating that a control endpoint 0 is always active in all configurations and interfaces.
Bit 14: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1: 0	MPS	0x0	ro	Maximum packet size The maximum packet size of the control OUT endpoint 0 is the same as that of the control IN endpoint 0. 00: 64 bytes 01: 32 bytes 10: 16 bytes 11: 8 bytes

20.6.5.12 OTGFS device control OUT endpoint-x control register (OTGFS_DOEPCTLx) (x= x=1...3, where x if endpoint number)

This application uses this register to control the behavior of all endpoints other than endpoint 0.

Bit	Register	Reset value	Type	Description
Bit 31	EPTENA	0x0	rw1s	Endpoint enable Indicates that the descriptor structure and data buffer for data reception has been configured. The controller clears this bit before setting any one of the following interrupts on this endpoint: – SETUP stage done – Endpoint disabled – Transfer completed
Bit 30	EPTDIS	0x0	ro	Endpoint disable The application sets this bit to stop transmitting data on an endpoint, even if the transfer on that endpoint is incomplete. The application must wait for the endpoint disabled interrupt before treating the endpoint as disabled. The controller clears this bit before setting the endpoint disabled interrupt. The application must set this bit only when the endpoint enabled set. 0: No effect 1: Endpoint disabled
Bit 29	SETD1PID/ SETODDFR	0x0	rw	Set DATA1 PID Applies to interrupt/bulk OUT endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA1. Set odd frame Applies to synchronous OUT endpoints only. Writing to

				this bit sets the Even/Odd frame to odd frame. 0: Disabled Set DATA1 PID disabled or Do not force odd frame 1: Set DATA1 PID enabled or forced odd frame
Bit 28	SETD0PID/ SETEVENFR	0x0	rw	Set DATA0 PID Applies to interrupt/bulk OUT endpoints only. Writing to this bit sets the endpoint data PID bit in this register to DATA0. Set Even frame Applies to synchronous OUT endpoints only. Writing to this bit sets the Even/Odd frame to even frame. 0:Disabled Set DATA0 PID disabled or Do not force even frame 1: Set DATA0PID or set the EOFRNUM to even frame
Bit 27	SNAK	0x0	wo	Set NAK A write to this bit sets the NAK bit for the endpoint. The application uses this bit to control the transmission of NAK handshakes on an endpoint. The controller sets this bit on a Transfer completed interrupt or after receiving a SETUP packet. Values: 0: Do not set NAK 1: Set NAK
Bit 26	CNAK	0x0	wo	Clear NAK A write to this bit clears the NAK bit for the endpoint. 0: Not clear NAK 1: Clear NAK
Bit 25: 22	Reserved	0x0	resd	Kept at its default value.
Bit 21	STALL	0x0	rw	Applies to non-control, non-synchronous IN and OUT endpoints. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, global non-periodic IN NAK bit or global OUT NAK bit is set along with this bit, the STALL bit has priority. Only the application can clear this bit, but the controller never.
Bit 20	SNP	0x0	rw	Snoop mode This bit configures the endpoint to Snoop mode. In this mode, the controller does not check the correctness of OUT packets before transmitting OUT packets to the application memory.
Bit 19: 18	EPTYPE	0x0	rw	Endpoint type This is the transfer type supported by this logical endpoint. 00: Control 01: Synchronous 10: Bulk 11: Interrupt
Bit 17	NAKSTS	0x0	ro	NAK status Indicates the following: 0: The controller is sending non-NAK handshakes based on the FIFO status 1: The controller is sending NAK handshakes When this bit is set (either by the application or the controller), the controller stops receiving any data on an OUT endpoint, even if there is space in the receive FIFO to accommodate the incoming data packets. For non-synchronous IN endpoints: the controller stops transmitting data on the endpoint, even if there is data pending in the transmit FIFO. For synchronous IN endpoints: the controller sends a zero-length data packet, even if there is space in the transmit FIFO. The controller always responds to SETUP data packets with an ACK handshake, regardless of whether this bit is set or not.

Bit 16	DPID/ EOFRNUM	0x0	ro	Endpoint data PID Applies to interrupt/bulk OUT endpoints only. This bit contains the PID of the packet to be transmitted on this endpoint. The application must program the PID of the initial data packet to be received or transmitted on this endpoint, after the endpoint is enabled. The application programs DATA0 or DATA1 PID through the SetD1PID and SetD0PID of this register. 0: DATA0 1: DATA1 Even/Odd frame Applies to synchronous OUT endpoints only. Indicates the frame number in which the controller transmits synchronous data on this endpoint. The application must program the even/odd frame number in which it tends to transmit or receive synchronous data through the SETEVNFR and SETODDFR bits in this register. 0: Even frame 1: Odd frame
Bit 15	USBACEPT	0x0	rw	USB active endpoint Indicates whether this endpoint is active in the current configuration and interface. The controller clears this bit for all endpoints except for endpoint 0 after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program the endpoint registers and set this bit. 0: Inactive 1: Active
Bit 14: 11	Reserved	0x0	resd	Kept at its default value.
Bit 10: 0	MPS	0x000	rw	Maximum packet size The application uses this field to set the maximum packet size for the current logical endpoint. The values are in bytes.

20.6.5.13 OTGFS device IN endpoint-x interrupt register (OTGFS_DIEPINTx) (x=0…3, where x if endpoint number)

This register indicates the status of an endpoint when USB and AHB-related events occurs, as shown in *Figure 20-2*. When the IEPINT bit of the OTGFS_GINTSTS register is set, the application must first read the OTGFS_DAINT register to get the exact endpoint number in which the event occurs, before reading the endpoint interrupt registers. The application must clear the appropriate bit in this register to clear the corresponding bits in the OTGFS_DAINT and OTGFS_GINTST registers.

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	TXFEMP	0x0	ro	Transmit FIFO empty This interrupt is generated when the transmit FIFO for this endpoint is half or completely empty. The half or completely empty status depends on the transmit FIFO empty level bit in the controller AHB configuration register.
Bit 6	INEPTNAK	0x0	rw1c	IN endpoint NAK effective This bit can be cleared by writing 1 to the CNAK bit in the DIEPCTLx register. This interrupt indicates that the IN endpoint NAB bit set by the application has taken effect. This interrupt does not guarantee that a NAK handshake is sent on the USB line. A STALL bit has priority over a NAK bit. This bit applies to the scenario only when the endpoint is enabled.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	INTKNTXFEMP	0x0	rw1c	N token received when TxFIFO is empty Indicates that an IN token was received when the

				associated transmit FIFO (periodic or non-periodic) was empty. An interrupt is generated on the endpoint for which an IN token was received.
Bit 3	TIMEOUT	0x0	rw1c	Timeout condition Applies to control IN endpoints only. This bit indicates that the controller has detected a timeout condition for the last IN token on this endpoint.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISD	0x0	rw1c	Endpoint disabled interrupt This bit indicates that the endpoint is disabled according to the application's request.
Bit 0	XFERC	0x0	rw1c	Transfer completed interrupt Indicates that the programmed transfers are complete on the AHB and on the USB for this endpoint.

20.6.5.14 OTGFS device OUT endpoint-x interrupt register (OTGFS_DOEPINTx) (x=0…3, where x if endpoint number)

This register indicates the status of an endpoint with respect to USB and AHB-related events, as shown in *Figure 20-2*. When the OEPINT bit of the OTGFS_GINTSTS register is set, the application must first read the OTGFS_DAINT register to get the exact endpoint number in which the event occurs, before reading the endpoint interrupt registers. The application must clear the appropriate bit in this register to clear the corresponding bits in the OTGFS_DAINT and OTGFS_GINTST registers.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x0000001	resd	Kept at its default value.
Bit 6	B2BSTUP	0x0	rw1c	Back-to-back SETUP packets received Indicates that more than three back-to-back SETUP packets are received.
Bit 5	Reserved	0x0	resd	Kept at its default value.
Bit 4	OUTTEPD	0x0	rw1c	OUT token received when endpoint disabled Applies to control OUT endpoints only. Indicates that an OUT token was received when the endpoint has not yet been enabled. An interrupt is generated on the endpoint for which an OUT token was received.
Bit 3	SETUP	0x0	rw1c	SETUP phase done Applies to control OUT endpoints only. Indicates that the SETUP stage for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. Upon this interrupt, the application can decode the received SETUP data packets.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	EPTDISD	0x0	rw1c	Endpoint disabled interrupt Indicates that the endpoint is disabled according to the application's request.
Bit 0	XFERC	0x0	rw1c	Transfer completed interrupt Indicates that the programmed transfers are complete on the AHB and on the USB for this endpoint.

20.6.5.15 OTGFS device IN endpoint 0 transfer size register (OTGFS_DIEPTSIZ0)

The application must set this register before enabling endpoint 0. Once the endpoint 0 is enabled using the endpoint enable pin in the device endpoint 0 control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31: 21	Reserved	0x000	resd	Kept at its default value.
Bit 20: 19	PKTCNT	0x0	rw	Packet count Indicates the total number of USB packets that constitute the transfer size of data for the endpoint 0. This field is decremented every time a packet is read from the transmit FIFO (maximum packet size or short packet)
Bit 18: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 0	XFERSIZE	0x00	rw	Transfer size Indicates the transfer size (in bytes) for the endpoint 0. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint at the end of each packet. The controller decrements this field every time a packet from the external memory is written to the transmit FIFO.

20.6.5.16 OTGFS device OUT endpoint 0 transfer size register (OTGFS_DOEPTSIZ0)

The application must set this register before enabling endpoint 0. Once the endpoint 0 is enabled using the endpoint enable pin in the device endpoint 0 control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	SUPCNT	0x0	rw	SETUP packet count Indicates the number of back-to-back SETUP data packets the endpoint can receive. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 20	Reserved	0x000	resd	Kept at its default value.
Bit 19	PKTCNT	0	rw	Packet count This bit is decremented to 0 after a packet is written to the receive FIFO.
Bit 18: 7	Reserved	0x000	resd	Kept at its default value.
Bit 6: 0	XFERSIZE	0x00	rw	Transfer size Indicates the transfer size (in bytes) for the endpoint 0. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet from the external memory is written to the transmit FIFO. The controller decrements this field every time a packet from the receive FIFO is written to the external memory.

20.6.5.17 OTGFS device IN endpoint-x transfer size register (OTGFS_DIEPTSIZx) (x=1…3, where x is endpoint number)

The application must set this register before enabling endpoint x. Once the endpoint x is enabled using the endpoint enable pin in the device endpoint x control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	MC	0x0	rw	Multi count For periodic IN endpoints, this field indicates the number of packets to be transmitted on the USB for each frame. The controller uses this field to calculate the data PID transmitted on synchronous IN endpoints. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 19	PKTCNT	0x000	rw	Packet count Indicates the total number of USB packets (data transfer size on the endpoint) this field is decremented every time a packet is read from the transmit FIFO (maximum packet size and short packet).
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer Size Indicates the transfer size (in bytes) for the current endpoint. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet from the external memory is written to the transmit FIFO.

20.6.5.18 OTGFS device IN endpoint transmit FIFO status register (OTGFS_DTXFSTSx) (x=0…3, where x is endpoint number)

This is a ready-only register containing the free space information for the device IN endpoint transmit FIFO.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at its default value.
Bit 15: 0	INEPTXFSAV	0x0200	ro	IN endpoint TxFIFO space available Indicates the amount of free space in the endpoint transmit FIFO. Values are in terms of 32-bit words. 0x0: Endpoint transmit FIFO is full 0x1: 1 word available 0x02: 2 words available 0xn: n words available (0 < n < 512) 0x200: Remaining 512 words Others: Reserved

20.6.5.19 OTGFS device OUT endpoint-x transfer size register (OTGFS_DOEPTSIZx) (x=1...3, where x is endpoint number)

The application must set this register before enabling endpoint x. Once the endpoint x is enabled using the endpoint enable pin in the device endpoint x control register, the controller modifies this register. The application can only read this register as long as the controller clears the endpoint enable bit.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30: 29	RXDPID	0x0	ro	Received data PID Applies to synchronous OUT endpoints only. This is the data PID received in the last packet. 00: DATA0 01: DATA2 10: DATA1 11: MDATA SETUP packet count Applies to synchronous OUT endpoints only. Indicates the number of back-to-back SETUP data packets the endpoint can receive. 01: 1 packet 10: 2 packets 11: 3 packets
Bit 28: 19	PKTCNT	0x000	rw	Packet count Indicates the number of USB packets transmitted on the endpoint. This field is decremented every time a packet is written to the receive FIFO (maximum packet size and short packet)
Bit 18: 0	XFERSIZE	0x00000	rw	Transfer size Indicates the transfer size (in bytes) for the current endpoint. The controller interrupts the application when the transfer size becomes 0. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The controller decrements this field every time a packet is read from the receive FIFO and written to the external memory.

20.6.6 Power and clock control registers

20.6.6.1 OTGFS power and clock gating control register (OTGFS_PCGCCTL)

This register is available in host and device modes.

Bit	Register	Reset value	Type	Description
Bit 31: 5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	SUSPENDM	0x0	ro	PHY suspend Indicates that the PHY has been suspended.
Bit 3: 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	STOPPCLK	0x0	rw	Stop PHY clock The application uses this bit to stop PHY clock when the USB is suspended, session is invalid or device is disconnected. The application clears this bit when the USB is resumed or a new session starts.

21 SDIO interface

21.1 SDIO introduction

The SD/SDIO MMC card host interface (SDIO) provides an interface between the AHB peripheral bus and MultiMediaCards (MMC), SD memory cards and SDIO cards.

SD memory card and SDI/O card system specifications are available through the SD card association website www.sdcard.org.

The MultiMediaCard system specifications published by the MMCA technical committee are available through the MultiMediaCard association website www.mmca.org.

21.2 SDIO main features

- Full compatibility with SD memory card specifications version 2.0
- Full compatibility with SDI/O card specification version 2.0 and support 1-bit and 4-bit databus modes.
- Full compatibility with MultiMedia card specification version 4.2 and support 1-bit, 4-bit and 8-bit databus modes.
- Full compatibility with previous versions of MultiMedia card specifications
- DMA transfer
- Data transfer up to 50 MHz in 8-bit bus mode
- Interrupt requests

Note: The SDIO is not compatible with SPI communication mode. It supports only one SD/SDIO/MMC 4.2 card at any one time.

Communication on the bus is based on command and data transfers.

— Command: A command is a token that starts an operation. Commands are sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). Commands are transferred serially on the CMD line.

— Response: A response is a token that is sent from a card to the host as an answer to a previously received command. Responses are transferred serially on the CMD line.

— Data: Data can be transferred from the card to the host or vice versa. Data is transferred via the SDIO_D data line.

The basic operation on the MMC card/SD/SDI I/O bus is the command/response structure. These types of bus operation transfer their information through the command or bus mechanism. In addition, some operations have a data token.

Data transfers to/from SD/SDIO memory cards are done in data blocks. Data blocks are always followed by CRC bit, defining single and multiple block operations. Data transfers to/from MMC are done in data blocks or streams, as shown in Figure below.

Figure 21-1 SDIO “no response” and “no data” operations

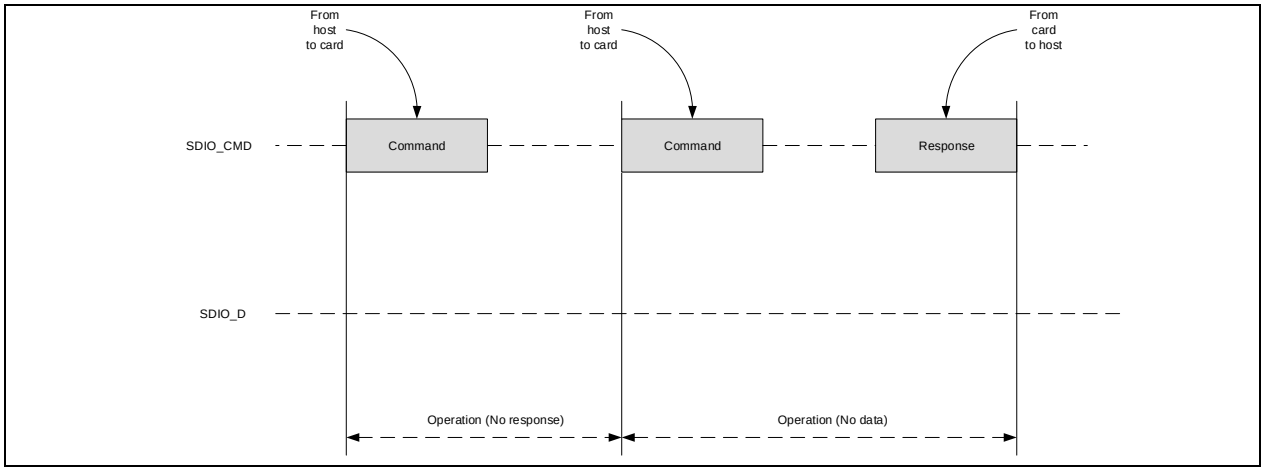


Figure 21-2 SDIO multiple block read operation

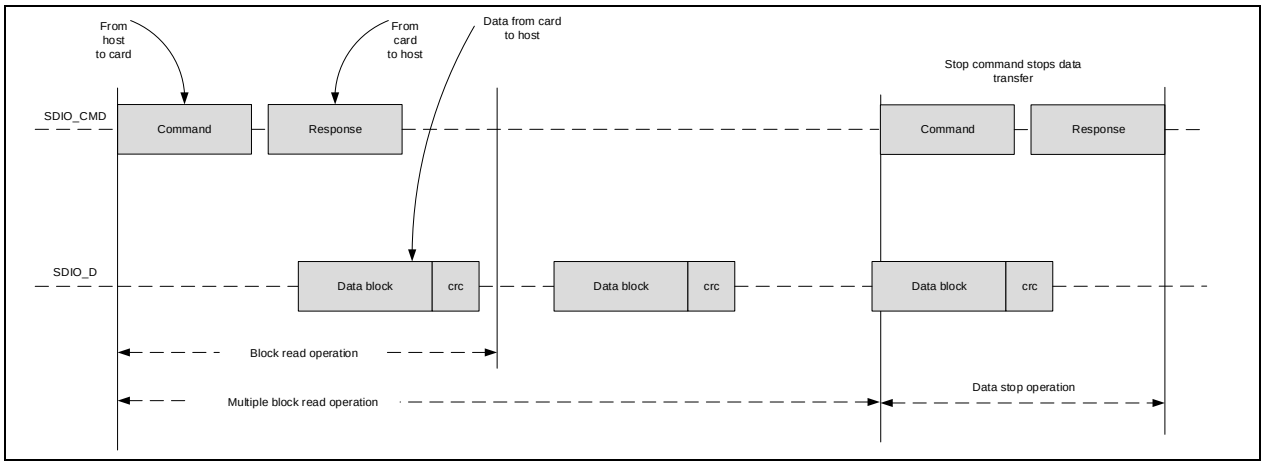
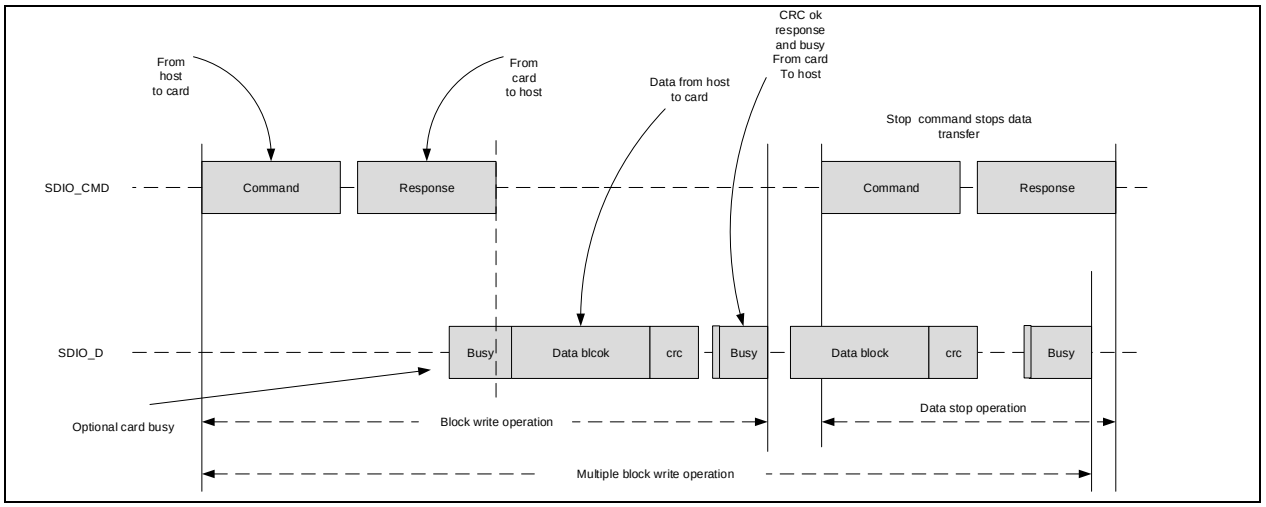


Figure 21-3 SDIO multiple block write operation



Note: The SDIO will send no data as long as the Busy signal is set (SDIO_D0 pulled low).

Figure 21-4 SDIO sequential read operation

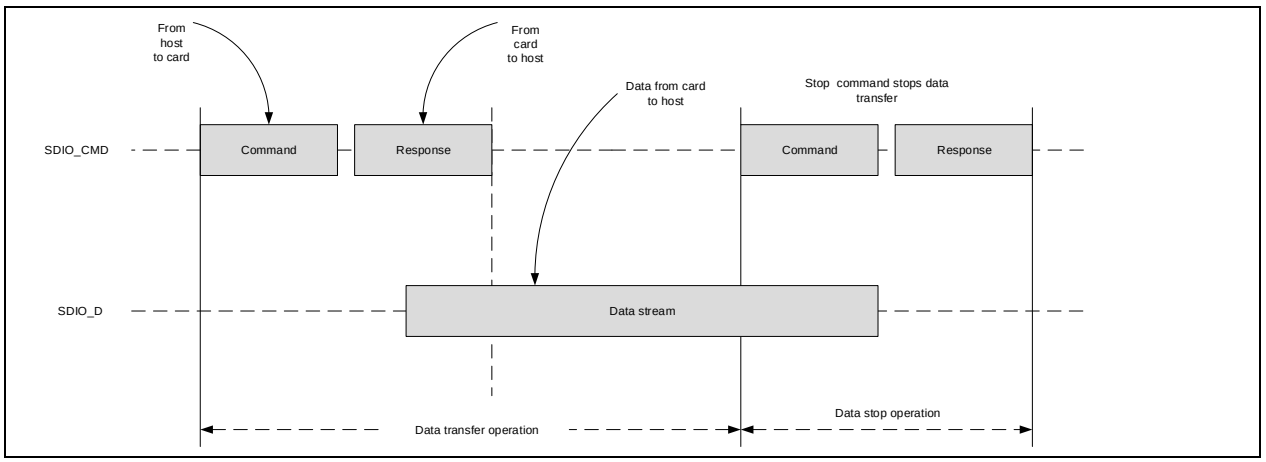
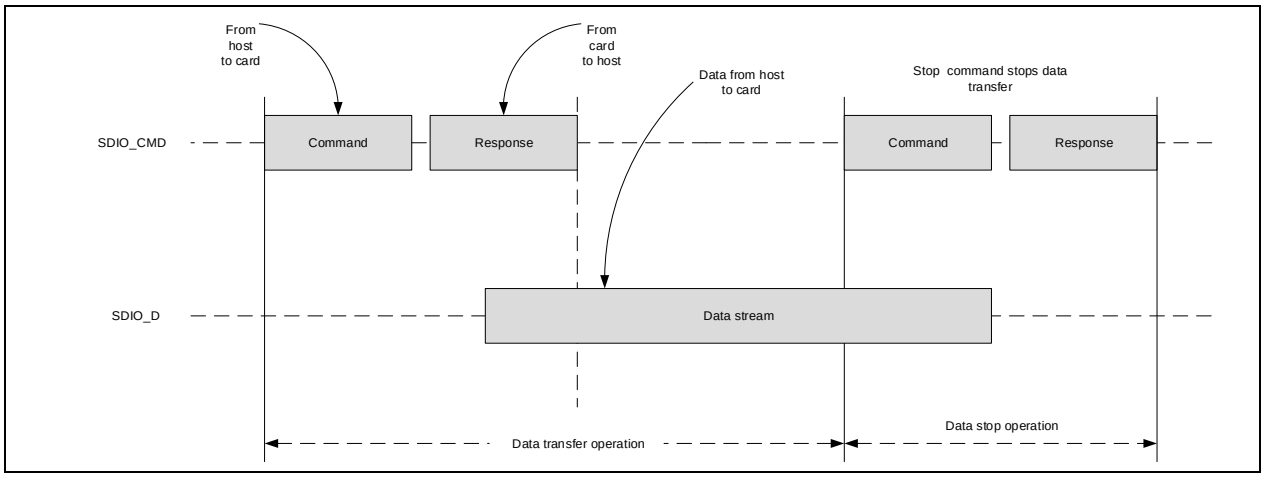


Figure 21-5 SDIO sequential write operation



21.3 SDIO main features

21.3.1 Card functional description

All the communications between the host and the cards are controlled by the card. The host sends two different types of commands: broadcast command and addressed (point-to-point) command.

- Broadcast command: applicable to all cards, some need responses
- Addressed command: sent to the addressed card, and responses received from the card

Memory card defines two types of operational modes:

- Card identification mode
- Data transfer mode

21.3.1.1 Card identification mode

In card identification mode, the host resets all cards, validates the operation voltage range, identifies cards and sets a relative card address (RCA) for each card on the CMD. All communications in the card identification mode use the command line (CMD).

Card identification process

The card identification process varies from card to card, and the host sends different commands. There are SD, SDI/O and MMC cards. It is possible to send a CMD5 command to identify the type of a card. If the host receives a response, it is a SDI/O card. If no response is received, the host will continue to send ACMD41 command, if a response is received, it is a SD card, otherwise a MMC card.

The card identification process is described as follows:

1. The bus is activated to confirm whether the card is connected or not. The clock frequency is at 0-400kHz during the card identification process.
2. The SDIO host sends a SD card, SDI/O card or MMC card.
3. Card Initialization
   - SD card: The SDIO host sends CMD2 (ALL_SEND_CID) to obtain its unique CID number. After receiving a response (CID number) from the card, the host will send CMD3 (SEND_RELATIVE_ADDR), instructing the card to issue the relative card address (RCA), which is shorter than the CID and is used to address the card in the data transfer mode.
   - SDI/O card: The SDIO host sends CMD3 (SEND_RELATIVE_ADDR) to instruct the card to release the relative card address (RCA), which is shorter than the CID and is used to address the card in the data transfer mode.
   - MMC card: The SDIO host sends CMD1 (SEND_OP_COND), followed by CMD2 and CMD3.
4. If the host wants to assign another RCA number, it can instruct the card to issue a new number by sending another CMD3 command. The last RCA is the actual RCA number of the card. The host repeats the card identification process (CMD2 and CMD3 cycle of each card)

21.3.1.2 Data transfer mode

The host will enter data transfer mode after identifying all cards on the bus. In data transfer mode, the host can operate cards within the range of 0 - 50MHz. It can send CMD9 (SEND_CSD) to get data specific to a card (CSD register) such as block length and car memory size. All communications between the host and the selected card are point-to-point transferred, and the CMD bus will confirm all addressed commands as a response. Data transfer read/write can be done in data block mode or stream mode, configured by the TFRMODE bit in the SDIO_DTCTRL register. In the data stream mode, data is transferred in bytes and without CRC appended to the end of each data block.

Wide bus selection/deselection

Wide bus (4-bit bus width) operation mode is selected or deselected by using ACMD6 (SET_BUS_WIDTH). The default bus width after power-up or CMD0 (GO_IDLE_STATE) is 1 bit. The ACMD6 is only valid in a transfer state, indicating that the bus width can be changed only after a card is selected by CMD7.

Stream read/write (MultiMedia card only)

Read**:**

1. The host sends CMD11 (READ_DAT_UNTIL_STOP) for stream read.
2. Until the host sends CMD12 (STOP_TRANSMISSION). The stop command has an execution delay due to the serial command transmission and the data transfers stops after the end bit of the stop command.

Write:

1. The host sends CMD20 (WRITE_DAT_UNTIL_STOP) for stream write.
2. Until the host sends CMD12 (STOP_TRANSMISSION). As the amount of data to be transferred is not determined in advance, the CRC cannot be used. When the memory range is reached, the command will be discarded by the card and remain in a transfer state, and a response is issued by setting the ADDRESS_OUT_OF_RANGE bit.

Data block read

In block read mode, the basic unit of data transfer is a block whose maximum size (fixed length 512 bytes) is defined in the CSD (READ_BL_LEN). If the READ_BL_PARTIAL is set, smaller blocks whose start and end addresses are entirely contained within 512 bytes may also be transmitted. A CRC is appended to the end of each block to ensuring data transfer integrity. Several commands related to data block read are as follows:

- CMD17 (READ_SINGLE_BLOCK): initiates a data block read and returns to the transfer state after the completion of the transfer.
- CMD18 (READ_MULTIPLE_BLOCK): starts a transfer of several consecutive data blocks.

  Data blocks will keep transferring until the host sends CMD12(STOP_TRANSMISSION). The stop command has an execution delay due to the serial command transmission and the data transfers stops after the end bit of the stop command.

Data block write

During block write (CMD24-27), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block. If the CRC failed, the card indicates a failure on the SDIO_D signal line and the transferred data are discarded and not written, and all transmitted data blocks are ignored.

If the host uses partial blocks with accumulated length is not block aligned, and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card will detect the block misalignment error before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR bit in the SDIO_STS register and waits the stop command in a receive state while ignoring all further data transfer. If the host attempts to perform write operation on a write-protected area, the write operation will be aborted. In this case, however, the card should set the WP_VIOLATION bit.

Programming of the CID and CSD registers does not require a block length setting in advance. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in the ROM, then the unchangeable part must match the corresponding part of the receive buffer. If this match failed, then

the card will report an error and does not change any register contents. Some cards may require long and unpredictable times to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the SDIO_D signal line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host can check the status of the card with SEND_STATUS command (CMD13) at any time, and the card will respond with its status.

The READY_FOR_DATA status bit indicates whether the car can accept new data or whether the write process in still in progress. The host can deselect the card by issuing CMD7 (select another card), which will place the card in the disconnect state and release the SDIO_D line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling SDIO_D line to low if the programming is still in progress and the write buffer is unavailable.

21.3.1.3 Erase

The erasable unit of the MultiMedia card and SD card is the erase group. The erase group is calculated in write blocks, which are the basic writable units of the card. The size of the erase group is a parameter specific to the card and defined in the CSD.

The host can erase a contiguous range of erase groups. There are three steps to start the erase process, but the commands sent by the MultiMedia card and SD card are different.

1. The host defines the start address of the range using the following command:
   - SD card: issue CMD32 (ERASE_WR_BLK_START)
   - MMC car: issue CMD35 (ERASE_GROUP_START)
2. The host defines the end address of the rang using the following command:
   - SD card: issue CMD33 (ERASE_WR_BLK_END)
   - MMD car: issue CMD36 (ERASE_GROUP_END)
3. The host starts the erase process by sending CMD38 (ERASE)

21.3.1.4 Protection management

Three write protection methods are supported in the SDIO card host module to ensure that the protected data is not erased or changed.

Mechanical write protect switch

There is a mechanical sliding switch on the side of the card to allow the user to set/clear the write protection on the card. When the sliding switch is positioned with the window open, the card is write-protected, and when the window is closed, the card is not write-protected.

Internal card write protection

Card data can be protected against write and erase. The entire card can be permanently write-protected by the manufacturer or the content provider by setting the permanent or temporary write-protect bits in the CSD. For cards that support write protection of groups of sectors by setting the WP_GRP_ENABLE bit in the CSD, part of the data can be protected, and the write protection can be changed by the application. The SET_WRITE_PROT commands set the write protection of the addressed group. The CLR_WRITE_PROT commands clear the write protection of the addressed group. The SEND_WRITE_PROT command is similar to a single block read command. The card sends a data block containing 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits. The address filed in the write protect commands is a group address in byte units.

Password protect

The password protection function enables the SDIO card host to lock and unlock a card with a password. The password is stored in the 128-bit PWD register and its size is set in the 8-bit PWD_LEN register. These registers are nonvolatile so that the content is not erased after power-off.

Locked cards can support certain commands, indicating that the host is allowed to reset, initialize and query for status, but not allowed to access data on the card. When the password is set (PWD_LEN is nonzero value), the card is locked automatically after power-up.

Like the CSD and CID register write commands, the lock/unlock commands are valid only in a transfer state. The command does not include an address parameter and thus the card must be selected before using it.

The card lock/unlock commands have the structure and bus transaction types of a regular single-block write command. The transferred data block include all the information required for the command (the password setting mode, PWD content and card lock/unlock indication). The command data block size is defined by the SDIO card host module before it sends the card lock/unlock command. The lock/unlock command structure is shown below:

Table 21-1 Lock/unlock command structure

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	Reserved(set to 0)				ERASE	LOCK_UNLOCK	CLR_PWD	SET_PWD
1	PWDS_LEN							
2	password data							
…								
PWDS_LEN+1								

- ERASE: Setting it will force an erase operation. All other bits must be zero, and only the command byte is sent.
- LOCK_UNLOCK: Setting it will lock the card. Clearing it will unlock the card. LOCK_UNLOCK can be set simultaneously with SET_PWD, but not with the CLR_PWD.
- CLR_PWD: Setting it will clear the password data.
- SET_PWD: Setting this bit will save the password data to memory.
- PWD_LEN: This bit defines the length of the password in bytes. When the password is changed, the length is the combination of the old and new passwords.
- PWD: password (new or currently used, depending on the command)

  The data block size should be defined by the host before it sends the card lock/unlock command. The block length should be equal to or greater than the data structure required for the lock/unlock command.

  The following sections list the command sequence to set/clear a password, lock/unlock a card, and force an erase operation.

Setting the password

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password. When a password is replaced, the block size must take into account the length of both the old and the new passwords sent with the command.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD). When a password replacement is done, the length value includes the length of the both passwords, the old and the new one, and the PWD field includes the old password (currently used) followed by the new password.
4. When the old password is matched, the new password and its size are saved into the PWD and PWD_LEN fields, respectively. When the old password sent is not correct (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the old password is not changed. When the old password sent is correct (in size and content), the given new password and its size will be saved in the PWD and the PWD_LEN registers, respectively.

The password length field (PWD_LEN) indicates whether a password is currently set or not. When the field is a zero value, it means that a password is not set currently. When the field is nonzero, it means that a password is set and the card locks itself after power-up. It is possible to lock the card immediately in the current power session by setting the LOCK_UNLOCK bit or sending an additional card lock command.

Clearing the password

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously.
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including

the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD). When a password is matched, the PWD field is cleared and PWD_LEN is set to 0. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the password is not changed.

Locking a card

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously.
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD).
4. When a password is matched, the card is locked and CARD_IS_LOCKED is set in the SDIO_STS register. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the lock fails.

If the password is previously set (PWD_LEN is not 0), the card is locked automatically after power-on reset. An attempt to lock a locked card or to lock a card that does not have a password fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

Unlocking a card

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (SET_PWD=1), the length (PWD_LEN) and the password (PWD).
4. When a password is matched, the card is unlocked and CARD_IS_LOCKED is cleared in the SDIO_STS register. If the password sent does not correspond to the expected password (in size and/or content), the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the lock remains locked.

The unlocking function is valid only for the current power session. The card is locked automatically on the next power-up as long as the PWD field is not cleared.

An attempt to unlock an unlocked card fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

Forcing erase

If the user forgot the password (PWD content), it is possible to access the card after clearing all the data on the card. This forced erase operation will erase all card data and all password data.

1. Select a card using CMD7 (SELECT/DESELECT_CARD), if none is selected previously
2. Define the block length with CMD16(SET_BLOCKLEN) to send in the 8-bit card lock/unlock mode, 8-bit PWD_LEN, and the number of bytes of the new password.
3. Send CMD42(LOCK/UNLOCK) with the appropriate data block size on the data line including the 16-bit CRC. The data block indicates the operation mode (ERASE =1). All other bits must be zero.
4. When the ERASE bit is the only bit in the data field, all card content will be erased, including the PWD and the PWD_LEN field, and the card is no longer locked. When any other bits are not zero, the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register, and the card data are retained, and the card remains locked.

An attempt to force erase an unlocked card fails and the LOCK_UNLOCK_FAILED error bit is set in the SDIO_STS register.

21.3.2 Commands and responses

21.3.2.1 Commands

Command types

Four commands are available to control the SD memory card:

1. Broadcast command: sent to all cards, no responses returned
2. Broadcast command with response: sent to all cards, responses received from all cards simultaneously
3. Addressed command: sent to the selected card, and no data transfer on the SDIO_D line
4. Addressed data transfer command: sent to the selected card, and data transfer is present on the SDIO_D line

Command description

The SDIO host module system is designed to provide a standard interface for a variety of application types. In the meantime, specific customer/application features must also be taken into account. Because of this, two types of general commands are defined in the standard: general commands (GEN_CMD) and application-specific commands (ACMD).

To use the application-specific commands, the SDIO host must send CMD55(APP_CMD) first, and waits the response from the card, which indicates that the APP_CMD bit is set and an ACMD is expected, before sending ACMD.

Table 21-2 Commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD0	bc	[31: 0]=stuff bits	-	GO_IDLE_ STATE	Reset all cards to idle state
CMD1	bc	[31: 0]=OCR	R3	SEND_OP_ COND	In idle state, request a card to send OCR register content through the CMD bus
CMD2	bcr	[31: 0]=stuff bits	R2	ALL_Send_CID	Request all cards to send CID data through the CMD bus
CMD3	bcr	[31: 0]= stuff bits	R6	SEND_RELATIV E_AD DR	Request a card to issue a new relative card address (RCA)
CMD4	bc	[31: 16]=DSR [15: 0]= stuff bits	-	SET_DSR	Set the DSR register of all cards
CMD5	bcr	[31: 24]Reserved [23: 0] I/O OCR	R4	IO_SEND_OP_C OND	For SDI/O card only, to query the voltage range of the required I/O card
CMD6	ac	[31: 26] set to 0 [25: 24] access [23: 16] index [15: 8] value [7: 3] set to 0 [2: 0] command set	R1b	SWITCH	For the MMC card only, to switch the operation modes or modify the EXT_CSD register
CMD7	ac	[31: 16]=RCA [15: 0]= stuff bits	R1b	SELECT/DESEL ECT_ CARD	This command is used to switch a card between the standby state and the send state, or between the programmed and the disconnected state. The relative card address is used to select a card. Address 0 is used to deselect the card.
CMD8 (SD)	bcr	[31: 12] Reserved [11:8]operating voltage (VHS) [7: 0]check mode	R7	SEND_IF_ COND	Send the host power supply voltage to the SD card and check whether the card supports the voltage or not.
CMD8 (MMC)	adtc	[31: 0]= stuff bits	R1	SEND_EXT_ CSD	For MMC card only, to send its own EXT_CSD register as a data block

CMD9	ac	[31: 16]=RCA [15: 0]= stuff bits	R2	SEND_CID	The selected card sends CID (card flag) through the CMD bus
CMD10	ac	[31: 16]=RCA [15: 0]= stuff bits	R2	SEND_CID	The selected card sends CID (card flag) through the CMD bus
CMD12	ac	[31: 0]= stuff bits	R1b	STOP_TRANSMISSION	Force the card to stop transmission
CMD13	ac	[31: 16]=RCA [15: 0]= stuff bits	R1	SEND_STATUS	Selected card send status register
CMD15	ac	[31: 16]=RCA [15: 0]= stuff bits	-	GO_INACTIVE_STATE	Selected card switch to inactive state

Table 21-3 Data block read commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD16	ac	[31: 0]=data block length	R1	SET_BLOCKLEN	This command is used to set the length of data blocks (in bytes) for all block commands. The default value is 512 bytes.
CMD17	adtc	[31: 0]=data address	R1	READ_SINGLE_BLOCK	Read a data block of the size set by CMD16
CMD18	adtc	[31: 0]=data address	R1	READ_MULTIPLE_BLOCK	Continuously read data from the card to the host until the STOP_TRANSMISSION is received

Table 21-4 Data stream read/write commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD11	adtc	[31: 0]= data address	R1	READ_DAT_UNTIL_STOP	Read data stream form the card starting from a given address until the STOP_TRANSMISSION is received.
CMD20	adtc	[31: 0]= data address	R1	WRITE_DAT_UNTIL_STOP	Read data stream form the host starting from a given address until the STOP_TRANSMISSION is received.

Table 21-5 Data block write commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD16	ac	[31: 0]=data block length	R1	SET_BLOCKLEN	This command is used to set the length of data blocks (in bytes) for all block commands. The default value is 512 bytes
CMD23	ac	[31: 16]=set to 0 [15: 0]=data block size	R1	SET_BLOCK_COUNT	Define the number of blocks to be transferred in the data block read/write that follows
CMD24	adtc	[31: 0]=data address	R1	WRITE_BLOCK	Write a data block of the size set by the CMD16

CMD25	adtc	[31: 0]=data address	R1	WRITE_MULTIPLE BLOCK	Continuously write data blocks until the STOP_TRANSMISSION is received
CMD26	adtc	[31: 0]= stuff bits	R1	PROGRAM_CID	Program the card identification register
CMD27	adtc	[31: 0]= stuff bits	R1	PROGRAM_CSD	Program the programmable bits of the CSD

Table 21-6 Block-based write protect command

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD28	ac	[31: 0]= data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the specified group. The properties of write protection are placed in the card-specific area (WP_GRP_SIZE).
CMD29	ac	[31: 0]= data address	R1b	CLR_WRITE_PROT	If the card has write protection features, this command clears the write protection bit of the specified group.
CMD30	adtc	[31: 0]=write protect data address	R1	SEND_WRITE_PROT	If the card has write protection features, this command asks the card to send the status of the write protection bits.

Table 21-7 Erase commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD32 … CMD34	Reserved. These command indexes cannot be used in order to maintain backward compatibility with older versions of the MultiMedia card.				
CMD35	ac	[31: 0]=data address	R1	ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase
CMD36	ac	[31: 0]=data address	R1	ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase
CMD37	Reserved. These command indexes cannot be used in order to maintain backward compatibility with older versions of the MultiMedia card.				
CMD38	ac	[31: 0]=stuff bits	R1b	ERASE	Erase all previously selected data blocks

Table 21-8 I/O mode commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD39	ac	[31: 16]=RCA [15]=register write flag [14: 8]=register address [7: 0]=register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command specifies a card and a register and provides the data for writing if the write flag is set. The R4 response contains data read from the specified register. This command accesses application-specific registers that are not defined in the MultiMedia card standard.
CMD40	bcr	[31: 0]=stuff bits	R5	GO_IRQ_STATE	Place the system in the interrupt mode

Table 21-9 Card lock commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD42	adtc	[31: 0]=stuff bits	R1	LOCK_ UNLOCK	Sets/clears the password or locks/unlocks the card. The size of the data block is set by CMD16.

Table 21-10 Application-specific commands

CMD index	Type	Parameter	Response format	Abbreviation	Description
CMD55	ac	[31: 16]=RCA [15: 0]=stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application-specific command rather than a standard command.
CMD56	adtc	[31: 1]=stuff bits [0]=RD/WR	R1	GEN_CMD	Used either to transfer a data block to the card or to read a data block from the card for general-purpose/application-specific commands. The size of the data block is defined by the SET_BLOCK_LEN command.
CMD57 … CMD59	Reserved.				
CMD60 … CMD63	Reserved for manufacturer				

21.3.2.2 Response formats

All responses are sent via the CMD bus. The response transmission always starts with the left bit of the bit string corresponding to the response code word. The length depends on the response type.

A response always starts with a start bit (always 0), followed by the transmission bit indicating the direction of transmission (card =0). A value denoted by – in the tables below stands for a variable entry. All responses, except the R3 response type, are protected by a CRC. Every command code word is terminated with the end bit (always 1).

21.3.2.2.1 R1 (normal response command)

Code length = 48 bits. The 45:40 bits indicate the index of the command to be responded to. This value is interpreted as a binary-coded number (between 0 and 63). The status of the card is coded in 32 bits. Note that if it involves a data transfer to a card, a busy signal may appear on the data line after each data block is transmitted. The host should check the busy signal after data block transfer.

Table 21-11 R1 response

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Field width	1	1	6	32	7	1
Value	0	0	-	-	-	1
Description	Start bit	Transmission bit	Command index	Card status	CRC7	End bit

21.3.2.2.2 R1b

It is the same as R1 with an optional busy signal transmitted on the data line. The card may become busy after receiving these commands based on its state prior to the command reception. The host should check the busy signal.

21.3.2.2.3 R2 (CID & CSD registers)

Code length = 136 bits. The contents of the CID register are sent as a response to the CMD2 and CMD10 commands. The contents of the CSD register are sent as a response to the CMD9. Only the bits [127 ... 1] of the CID and CSD are transmitted, and the reserved bit [0] of these registers is replaced with the end bit of the response.

Table 21-12 R2 response

Bit	135	134	[133 : 128]	[127 : 1]	0
Field width	1	1	6	127	1
Value	1	0	111111	-	1
Description	Start bit	Transmission bit	Reserved	CID or CSD register	End bit

21.3.2.2.4 R3 (OCR register)

Code length = 48 bits. The contents of the OCR register are sent as a response to ACMD41.

Table 21-13 R3 response

Bit	47	46	[45 : 40]	[39: 8]	[7: 1]	0
Field width	1	1	6	32	7	1
Value	1	0	111111	-	111111	1
Description	Start bit	Transmission bit	Reserved	OCR register	Reserved	End bit

21.3.2.2.5 R4 (Fast I/O)

Code length = 48 bits. The parameter field contains the RCA of the specified card, the register address to be read out or written to, and its contents.

Table 21-14 R4 response

Bit	47	46	[45: 40]	[39: 8]			[7: 1]	0
Field width	1	1	6	16	8	8	7	1
Value	1	0	100111	-	-	-	-	1
Description	Start bit	Transfer bit	CMD39	RCA	Register address	Read register contents	CRC7	End bit

21.3.2.2.6 R4b

For SD I/O only, an SDIO card will respond with a unique SDIO response R4 after receiving the CMD5.

Table 21-15 R4b response

Bit	47	46	[45: 40]			[39: 8]			[7: 1]	0
Field width	1	1	6	1	3	1	3	24	7	1
Value	1	0	-	-	-	-	-	-	-	1
Description	Start bit	Tx bit	Res.	Card ready	Number of I/O functions	Current memory	Stuff bit	I/O OCR	Res.	End bit

21.3.2.2.7 R5 (interrupt request)

For MultiMedia card only. Code length = 48 bits. If the response is generated by the host, the RCA field in the parameter will be 0x0.

Table 21-16 R5 response

Bit	47	46	[45: 40]	[39: 8]		[7: 1]	0
Field width	1	1	6	16	16	7	1
Value	1	0	101000	-	-	-	1
Description	Start bit	Start bit	Tx bit	RCA[31:16] of a successful card or of the host	Not defined. Maybe used for interrupt data	CRC7	End bit

21.3.2.2.8 R6 (interrupt request)

For SD I/O card only. This is a normal response to CMD3 by a memory device.

Table 21-17 R6 response

Bit	47	46	[45: 40]	[39: 8]		[7: 1]	0
Field width	1	1	6	16	16	7	1
Value	1	0	000011	-	-	-	1
Description	Start bit	Tx bit	CMD3	RCA[31:16] of a successful card or of the host	Card status	CRC7	End bit

The card status bit [23: 8] will be changed when the CMD3 is sent to an I/O-only card. In this case, the 16 bits of response are the SD I/O-only values.

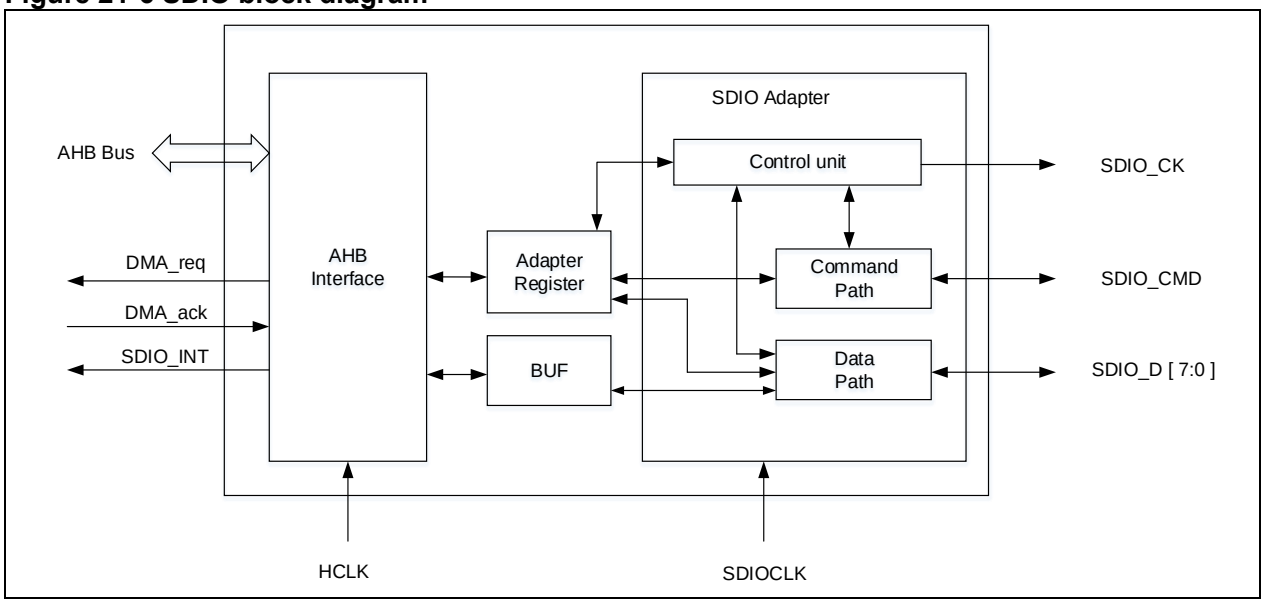
- Bit 15=COM_CRC_ERROR
- Bit 14=ILLEGAL_COMMAND
- Bit 13=ERROR
- Bit [12: 0]=Reserved

21.3.3 SDIO functional description

SDIO consists of four parts:

- SDIO adapter block: contains a control unit, command path and data path that provides all functions specific to the MMC/SD/SD I/O card such as the clock generation, command and data transfer
  - Control unit: manages and generates clock signals
  - Command path: manages command transfer
  - Data path: manages data transfer
- AHB interface: generates interrupt and DMA request signals
- Adapter register: system register
- BUF: used for data transfer

Figure 21-6 SDIO block diagram



21.3.3.1 SDIO adapter

SDIO_CK is a clock to the MultiMedia/SD/SDIO car provided by the host. One bit of command or data is transferred on both command and data lines with each clock cycle. The clock frequency can very between different cards and different protocols.

- MultiMedia card
  - V3.31 protocol 0 – 20MHz
  - V4.0/4.2 protocol 0 – 50MHz
- SD card
  - 0 – 50MHz
- SD I/O card
  - 0 – 50MHz

SDIO_CMD is a bidirectional command channel and used for the initialization of a card and command transfer. When the host sends a command to a card, the card will issue a response to the host. The SDIO_CMD has two operational modes:

- Open-drain mode for initialization (only for MMCV3.31 or previous)
- Push-pull mode for command transfer (SD/SD I/O card and MMC V4.2 also use push-pull drivers for initialization)

SDIO_D [7:0] is a bidirectional data channel. After initialization, the host can change the width of the data bus. After reset, the SDIO_D0 is used for data transfer by default. MMCV3.31 or previous supports only one bit of data line, so only SDIO_DO can be used.

The table below is used for the MultiMedia card/SD/SD I/O card bus:

Table 21-18 SDIO pin definitions

Pin	Direction	Description
SDIO_CK	Output	MultiMedia card/SD/SDIO card clock. This pin is the clock from the Host to a card.
SDIO_CMD	Bidirectional	MultiMedia card/SD/SDIO card command. This pin is the bidirectional command/response signal.
SDIO_D[7: 0]	Bidirectional	MultiMedia card/SD/SDIO card data. This pin is the bidirectional databus.

Control unit

The control unit consists of a power management sub-unit and a clock management sub-unit. The power management subunit is controlled by the SDIO_PWRCTRL register. The PS bit is used to define power-up/power-off state. During the power-off and power-up phases, the power management subunit will disable the card bus output signals. The clock management subunit is controlled by the SDIO_CLKCTRL

register where the CLKDIV bit is used to define the divider factor between the SDIOCLK and the SDIO output clock. If BYPSEN = 0, the SDIO_CK output signal is driven by the SDIOCLK divided according to the CLKDIV bit; if BYPSEN = 1, the SDIO_CK output signal is directly driven by the SDIOCLK. The HFCEN is set to enable hardware flow control feature in order to avoid the occurrence of an error at transmission underflow or reception overflow. The PWRSVEN bit can be set by software to enable power save mode, and the SDIO_CK can be output only when the bus is active.

Command path

The command path unit sends commands to and receives responses from the cards. When the CCSMEN bit is set in the SDIO_CMDCTRL register, a command transfer starts. First sends a command to a card by the SDIO_CMD, the command length is 48 bits (refer to *Table 21-19*). The data on the SDIO_CMD is synchronized with the rising edge of the SDIO_CK. A block of data is transferred with each SDIO_CK, including start bit, transfer bit, command index defined by the SDIO_CMDCTRL_CMDIDX bit, parameters defined by the SDIO_ARG, 7-bit CRC and end bit. Then receives responses from the card. There are two response types: 48-bit short response and 136-bit long response. Both use CRC error check. The received responses are saved in the area from SDIO_RSP1 to SDIO_RSP4. The command path can generate command flag, which can be defined by the SDIO_STS register.

Table 21-19 Command formats

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Width	1	1	6	32	7	1
Value	0	1	-	-	-	1
Description	Start bit	Tx bit	Command index	Parameter	CRC7	End bit

— Response: A response is sent from a specified card to the host (or synchronously from all cards for MMCV3.31 or previous), as an answer to a previously received command. Responses are transferred serially on the CMD line.

Table 21-20 Short response format

Bit	47	46	[45: 40]	[39: 8]	[7: 1]	0
Width	1	1	6	32	7	1
Value	0	0	-	-	-	1
Description	Start bit	Tx bit	Command index	Parameter	CRC7 (or 1111111)	End bit

Table 21-21 Long response format

Bit	135	134	[133: 128]	[127: 1]	0
Width	1	1	6	127	1
Value	0	0	111111	-	1
Description	Start bit	Tx	Reserved	CID or CSD (including internal CRC7)	End bit

Table 21-22 Command path status flags

Flag	Description
CMDRSPCMPL	A response is already received (CRC OK)
CMDFAIL	A command response is already received (CRC fails)
CMDCMPL	A command is sent (does not require a response)
CMDTIMEOUT	Command response timeout (64 SDIO_CK cycles)
DOCMD	Command transfer is in progress

Command channel status machine (CCSM)

When the CCSMEN bit is set in the SDIO_CMDCTR register, command transfer starts. When the command has been sent, the command channel state machine (CCSM) will set the status flags and enters the idle state if a response is not required (*Figure 21-7*). When a response is received, the received CRC code and the internally generated CRC code are compared, and the appropriate status flags are set.

- The CCSM remains in the idle state for at least 8 SDIO_CK cycles to meet the Ncc (the minimum delay between two host commands) and NRC (the minimum delay between the host command and the card response).
- When the wait state is entered, the command timer is enabled. If the NCR timeout (response time to a command), that is, 64 SDIO_CK periods, is reached before the CCSM moves to the receive state, the timeout flag is set (CMDTIMEOUT) and the idle state is entered.

If the interrupt bit is set in the command register, the timer is disabled and the CCSM waits for an interrupt request from one card. If the pending bit is set in the command register, the CCSM will enter pend state and wait for a CmdPend signal from the data path subunit. When detecting the CmdPend, the CCSM goes to the send state, which triggers the data counter to send the stop command.

Figure 21-7 Command channel state machine (CCSM)

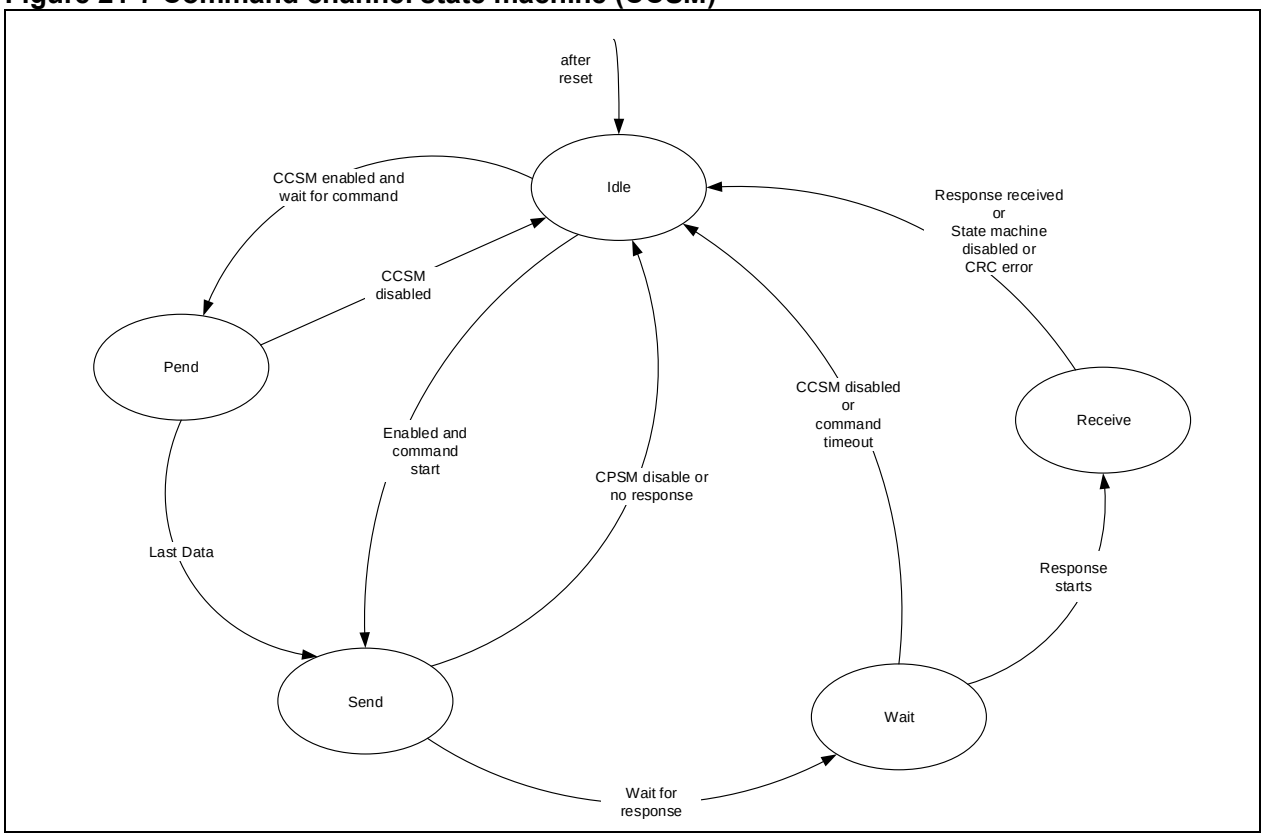
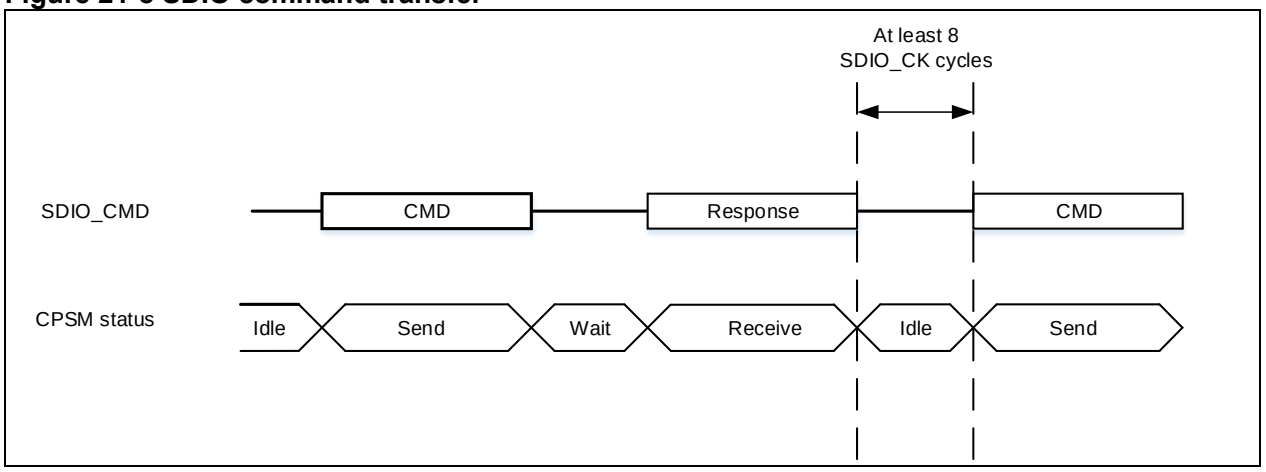


Figure 21-8 SDIO command transfer



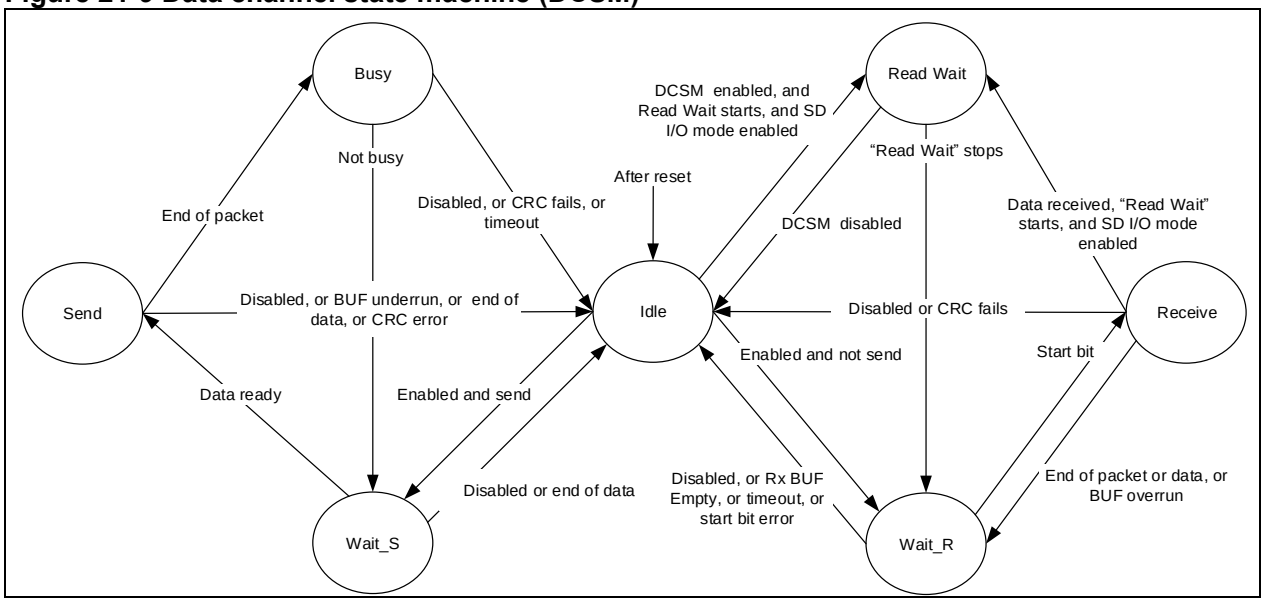
Data path

The data path subunit transfers data between the host and the cards. The databus width can be configured using the BUSWS bit in the SDIO_CLKCTRL register. By default, only the SDIO_DO signal line is used for transfer. Only one bit of data is transferred with each clock cycle. If the 4-bit wide bus mode is selected, four bits are transferred per clock cycle over the SDIO_D [3:0] signal line. If the 8-bit wide bus mode is selected, eight bits are transferred per clock cycle over the SDIO_D [7: 0] signal line. The TFRDIR bit is set in the SDIO_DTCTR register to define the transfer direction. If TFRDIR=0, it indicates that the data is transferred from the controller to the card; if TFRDIR=1, it indicates that the data is transferred from the card to the controller. The TFRMODE bit can be used to select block data transfer or stream transfer for the MultiMedia card. If the TFREN bit is set, data transfer starts. Depending on the TFRDIR bit, the DCSM enters Wait_S or Wait_R state.

Data channel state machine (DCSM)

The DCSM has seven states, in send and receive mode, as shown in the Figure below:

Figure 21-9 Data channel state machine (DCSM)



Send mode

- Idle: The data channel is inactive, either in the Wait_S or the Wait_R state.
- Wait_S: Waits until the BUF flag becomes empty or the data transmission is completed. The DCSM must remain in the Wait_S state for at least two clock periods to meet the N_{WR} timing requirements where the N_{WR} is the interval between the reception of the card response and the start of the data transfer from the host.
- Send: The DCSM sends data to a card, and the data transfer mode can be either block or stream, depending on the SDIO_DTCTRL_TFRMODE bit. If an overflow error occurred, the DCSM then moves to the idle state

- Busy: The DCSM waits for the CRC flag. If the DCSM receives a correct CRC status and is not busy, it will enter the Wait_S state. If it does not receive a correct CRC status or a timeout occurs while the DCSM is in the busy state, a CRC fail flag or timeout flag is generated.
- Wait_R: The start bit of the Wait_R. If a timeout occurs before it detects a start bit, the DCSM moves to the idle state and generates a timeout flag.
- Receive: Data is received from a card and written to the BUF. The data transfer mode can be either block or stream, depending on the SDIO_DTCTRL _TFRMODE bit. If an overflow error occurs, it then returns to Wait_R state.

Table 21-23 Data token formats

Description	Start bit	Data	CRC16	End bit
Block data	0	-	Y	1
Stream data	0	-	N	1

21.3.3.2 Data BUF

The data BUF contains a transmit and receive unit. It is a 32-bit wide and 32-word deep data buffer. Because the data BUF operates in the AHB clock domain (HCLK), all signals connected to the SDIO clock domain (SDIOCLK) are resynchronized.

- Transmit BUF: Data can be written to the transmit BUF via the AHB interface when the SDIO transmission feature is enabled.
  The transmit BUF has 32 sequential addresses. It contains a data output register that holds the data word pointed by the read pointer. When the data path has loaded its shift register, it moves its read pointer to the next data and outputs data.
  If the transmit BUF is disabled, all status flags are inactive. The data path sets the DOTX when it transmits data.
- Receive BUF: When the data path receives a data word, it will write the data to the BUF. The write pointer is incremented automatically after the end of the write operation. On the other side, a read pointer always points to the current data in the BUF. If the receive BUF is disabled, all status flags are cleared, and the read and write pointers are reset as well. The data path sets the DORX when it receives data.

21.3.3.3 SDIO AHB interface

The AHB interface generates the interrupt and DMA requests, and access the SDIO interface registers and the data BUF.

SDIO interrupts

The interrupt logic generates an interrupt request when one of the selected status flags is high. The SDIO_INTEN register is used to select the conditions that will generate an interrupt.

SDIO/DMA interface: data transfer process between the SDIO and memory

In the following examples, data is transferred from the host to the card. The SDIO BUF is filled with data stored in a memory through the DMA controller.

1. Card identification process
2. Increase the SDIO_CK frequency
3. Select a card by sending CMD7
4. Enable the DMA2 controller and clear all interrupt flag bits, configure the DMA2 channel4 source address register as the memory buffer's base address, and the DMA2 channel4 destination address register as the SDIO_BUF register address. Then configure the DMA2 channel4 control register (memory increment, non-peripheral increment, and peripheral and source data width is word width). Finally enable DMA2 channel4.
5. Send CMD24 (WRITE_BLOCK) as follows:

   Program the SDIO data length register (SDIO_DTLEN), the BLKSIZE bit in the SDIO data control register (SDIO_DTCTRL), and the SDIO parameter register (SDIO_ARG) with the address of the card where data is to be transferred, and program the SDIO command register (SDIO_CMD), enable the CCSMEN bit, wait for SDIO_STS [6]=CMDRSPCMPL interrupt,

and then program the SDIO data control register (SDIO_DTCTRL): TFREN=1 (enable the SDIO card host to send data), TFRDIR=0 (from the controller to the card), TFRMODE=0 (block data transfer), DMAEN=1 (enable DMA), BLKSIZE=9 (512 bytes), and wait from SDIO_STS [10]=DTBLKCMPL.

6. Check that no channels are still enabled by confirming the DMA channel enable SDIO status register (SDIO_STS)

21.3.3.4 Hardware flow control

The HFCEN bit is set in the SDIO_CLKCTRL register to enable hardware flow control, which is sued to avoid BUF underflow and overflow errors. Read/write access to the BUF is still active even if flow control is enabled.

21.3.4 SDIO I/O card-specific operations

The SDIO can support the following operations (except read suspend, for it does not require specific hardware operation) when the SDIO_DTCTRL [11] is set.

SDIO read wait operation by SDIO_D2 signal lines

The optional read wait operation is used only for SD card 1-bit or 4-bit mode. The read wait operation can instruct the host to stop data transfer temporarily while the host is reading from multiple registers (IO_RW_EXTENDED, CMD53), and also allows the host to send commands to other functions in the SD I/O device in order to start a read wait process after the reception of the first data block. The detailed process as follows:

- Enable data path (SDIO_DTCTRL [0] = 1)
- Enable SDIO-specific operation (SDIO_DTCTRL [11] = 1)
- Start read wait (SDIO_DTCTRL [10]=0 and SDIO_DTCTRL [8]=1)
- Data direction is from a card to the SDIO host (SDIO_DTCTRL [1]=1)
- The data unit in the SDIO adapter will enter read wait state, and drive the SDIO_D2 to 0 after 2 SDIO_CK cycles
- The data unit starts waiting to receive data from a card. The DCSM will not enter read wait even if read wait start is set. The read wait process will start after the CRC is received. The RDWTSTOP has to be cleared to start a new read wait operation.

  During the read wait period, the SDIO host can detect the SDIO interrupts over the SDIO_D1.

SDIO read wait operation by stopping clock

If the SDIO card does not support the mentioned above read wait operation, the SDIO can enter a read wait by stopping SDIO_CK, described as follows:

- Enable data path (SDIO_DTCTRL [0] = 1)
- Enable SDIO-specific operation (SDIO_DTCTRL [11] = 1)
- Start read wait (SDIO_DTCTRL [10]=0 and SDIO_DTCTRL [8]=1)

The DCSM stops the clock two SDIO_CK cycles after the end bit of the current received data block and starts the clock again after the read wait end bit is set.

Note that as the SDIO_CK is stopped, the SDIO host cannot send any command to the card. The SDIO host can detect the SDIO interrupt over the SDIO_D1.

SDIO suspend/resume operation (write and read operation suspend)

To free the bus to provide higher-priority transfers for other functions or memories, the host can suspend data transfer to certain functions or memories. As soon as the higher-priority transfer is completed, the previous transfer operation will restart at the suspended location.

While sending data to a card, the SDIO can suspend the write operation. The SDIO_CMD [11] bit is set and indicates to the CCSM that the current command is a suspend command. The CCSM analyzes the response and when an ACK signal is received from the card (suspend accepted), it acknowledges the DCSM that enters the idle state after receiving the CRC of the current data block.

SDIO interrupts

There is a pin with interrupt feature on the SD interface in order to enable the SD I/O card to interrupt the MultiMedia card/SD module. In 4-bit SD mode, this pin is SDIO_D1. The SD I/O interrupts are

detected when the level is active. In other words, the interrupt signal line must be active (low) before it is recognized and responded by the MultiMedia card/SD module, and will remain inactive (high) at the end of the interrupt routine.

When the SDIO_DTCTRL [11] bit is set, the SDIO interrupts are detected on the SDIO_D1 signal line.

21.4 SDIO registers

The device communicates with the system through 32-bit control registers accessible via AHB.

The peripheral registers must be accessed by words (32-bit).

Table 21-24 SDIO register map and reset values

Register	Offset	Reset value
SDIO_PWRCTRL	0x00	0x0000 0000
SDIO_CLKCTRL	0x04	0x0000 0000
SDIO_ARG	0x08	0x0000 0000
SDIO_CMD	0x0C	0x0000 0000
SDIO_RSPCMD	0x10	0x0000 0000
SDIO_RSP1	0x14	0x0000 0000
SDIO_RSP2	0x18	0x0000 0000
SDIO_RSP3	0x1C	0x0000 0000
SDIO_RSP4	0x20	0x0000 0000
SDIO_DTTMR	0x24	0x0000 0000
SDIO_DTLEN	0x28	0x0000 0000
SDIO_DTCTRL	0x2C	0x0000 0000
SDIO_DTCNTR	0x30	0x0000 0000
SDIO_STS	0x34	0x0000 0000
SDIO_INTCLR	0x38	0x0000 0000
SDIO_INTEN	0x3C	0x0000 0000
SDIO_BUFCNTR	0x48	0x0000 0000
SDIO_BUF	0x80	0x0000 0000

21.4.1 SDIO power control register (SDIO_ PWRCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 1: 0	PS	0x0	rw	Power switch These bits are set or cleared by software. They are used to define the current status of the card clock. 00: Power-off, the card clock is stopped. 01: Reserved 10: Reserved 11: Power-on, the card clock is started.

Note: Write access to this register is not allowed within seven HCLK clock periods after data is written.

21.4.2 SDIO clock control register (SDIO_ CLKCTRL)

The SDIO_CLKCTRL register controls the SDIO_CK output clock.

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at its default value.
Bit 16: 15	CLKDIV	0x0	rw	Clock division This field is set or cleared by software. It defines the clock division relations between the SDIOCLK and the SDIO_CK: SDIO_CK frequency=SDIOCLK / [CLKDIV[9: 0] + 2].
Bit 14	HFCEN	0x0	rw	Hardware flow control enable This bit is set or cleared by software. 0: Hardware flow control disabled 1: Hardware flow control enabled Note: When hardware flow control is enabled, refer to the SDIO_STS register for the meaning of the TXBUF_E and RXBUF_F interrupt signals.
Bit 13	CLKEGS	0x0	rw	SDIO_CK edge selection This bit is set or cleared by software. 0: SDIO_CK generated on the rising edge of the master clock SDIOCLK 1: SDIO_CK generated on the falling edge of the master clock SDIOCLK
Bit 12: 11	BUSWS	0x0	rw	Bus width selection This bit is set or cleared by software. 00: Default bus mode, SDIO_D0 used 01: 4-bit bus mode,SDIO_D[3: 0] used 10: 8-bit bus mode, SDIO_D[7: 0] used
Bit 10	BYPSEN	0x0	rw	Clock divider bypass enable bit This bit is set or cleared by software. When disabled, the SDIO_CK output signal is driven by the SDIOCLK that is divided according to the CLKDIV value. When enabled, the SDIO_CK output signal is directly driven by the SDIOCLK. 0: Clock divider bypass disabled 1: Clock divider bypass enabled
Bit 9	PWRSVEN	0x0	rw	Power saving mode enable This bit is set or cleared by software. When disabled, the SDIO_CK is always output; when enabled, the SDIO_CK is only output when the bus is active. 0: Power saving mode disabled 1: Power saving mode enabled
Bit 8	CLKOEN	0x0	rw	Clock output enable This bit is set or cleared by software. 0: Clock output disabled 1: Clock output enabled
Bit 7: 0	CLKDIV	0x00	rw	Clock division This field is set or cleared by software. It defines the clock division relations between the SDIOCLK and the SDIO_CK: SDIO_CK frequency=SDIOCLK / [CLKDIV[9: 0] + 2].

Note: 1. While the SD/SDIO card or MultiMedia car is in identification mode, the SDIO_CK frequency must be less than 400kHz.

2. When all card are assigned with relative card addresses, the clock frequency can be changed to the maximum card frequency.

3. This register cannot be written within seven HCLK clock periods after data is written. The

SDIO_CK can be stopped during the read wait period for SD I/O cards. In this case, the SDIO_CLKCTRL register does not control the SDIO_CK.

21.4.3 SDIO argument register (SDIO_ARG)

The SDIO_ARG register contains 32-bit command argument, which is sent to a card as part of a command.

Bit	Register	Reset value	Type	Description
Bit 31: 0	ARGU	0x0000 0000	rw	Command argument Command argument is sent to a card as part of a command. If a command contains an argument, it must be loaded into this register before writing a command to the command register.

21.4.4 SDIO command register (SDIO_CMD)

The SDIO_CMD register contains the command index and command type bits. The command index is sent to a card as part of a command. The command type bits control the command channel state machine (CCSM).

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	IOSUSP	0x0	rw	SD I/O suspend command This bit is set or cleared by software. If this bit is set, the command to be sent is a suspend command (used for SDIO only). 0: SD I/O suspend command disabled 1: SD I/O suspend command enabled
Bit 10	CCSMEN	0x0	rw	Command channel state machine (CCSM) enable bit This bit is set or cleared by software. 0: Command channel state machine disabled 1: Command channel state machine enabled
Bit 9	PNDWT	0x0	rw	CCSM Waits for ends of data transfer (CmdPend internal signal) This bit is set or cleared by software. If this bit is set, the CCSM waits for the end of data transfer before it starts sending a command. 0: Disabled 1: Enabled
Bit 8	INTWT	0x0	rw	CCSM waits for interrupt request This bit is set or cleared by software. If this bit is set, the CCSM disables command timeout and waits for an interrupt request. 0: Disabled 1: Enabled
Bit 7: 6	RSPWT	0x0	rw	Wait for response bits This bit is set or cleared by software. This bit indicates whether the CCSM is to wait for a response, and if yes, it will indicates the response type. 00: No response 01: Short response 10: No response 11: Long response
Bit 5: 0	CMDIDX	0x00	rw	Command index The command index is sent to a card as part of a command.

Note: 1.This register cannot be written within seven HCLK clock periods after data is written.
2.MultiMedia card can sent two types of responses: 48-bit short response or 136-bit short response. The SD card and SD I/O card can send only short responses, and the argument

can vary according to the type of response. The software will distinguish the type of response according to the command sent.

21.4.5 SDIO command response register (SDIO_RSPCMD)

The SDIO_RSPCMD register contains the command index of the last command response received. If the command response transmission does not contain the command index (long or OCR response), the SDIO_RSPCMD field is unknown, although it should have contained 111111b (the value of the reserved field from a response)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at its default value.
Bit 5: 0	RSPCMD	0x00	ro	Response command index This field contains the command index of the command response received.

21.4.6 SDIO response 1..4 register (SDIO_RSPx)

The SDIO_RSPx (x=1..4) register contains the status of a card, which is part of the response received.

Bit	Register	Reset value	Type	Description
Bit 31: 0	CARDSTSx	0x0000 0000	ro	See Table 23-25

The card status size is 32 or 127 bits, depending on the response type.

Table 21-25 Response type and SDIO_RSPx register

Register	Short response	Long response
SDIO_RSP1	Card status [31: 0]	Card status [127: 96]
SDIO_RSP2	Unused	Card status [95: 64]
SDIO_RSP3	Unused	Card status [63: 32]
SDIO_RSP4	Unused	Card status [31: 1]

The most significant bit of the card status is always received first. The least significant bit of the SDIO_RSP4 register is always 0.

21.4.7 SDIO data timer register (SDIO_DTTMR)

The SDIO_DTTMR register contains the data timeout period in the unit of card bus clock periods. A counter loads the value from the SDIO_DTTMR register and starts decrementing when the DCSM enters the Wait_R or busy state. If the counter reaches 0 while the DCSM is in either of these states, a timeout status flag will be set.

Bit	Register	Reset value	Type	Description
Bit 31: 0	TIMEOUT	0x0000 0000	rw	Data timeout period Data timeout period in card bus clock cycles.

Note: A data transfer must be written to the SDIO_DTCNTR and the SDIO_DTLEN register before being written to the SDIO data control register (SDIO_DTCTRL).

21.4.8 SDIO data length register (SDIO_DTLEN)

The SDIO_DTLEN register contains the number of data bytes to be transferred. the value is loaded into the data counter when data transfer starts.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 0	DTLEN	0x0000000	rw	Data length value Number of data bytes to be transferred.

Note: For a block data transfer, the value in the SDIO_DTLEN must be a multiple of the block data size (See Section 21.4.9).

A data transfer must be written to the SDIO_DTCNTR and the SDIO_DTLEN register before being written to the SDIO data control register (SDIO_DTCTRL).

21.4.9 SDIO data control register (SDIO_DTCTRL)

The SDIO_DTCTRL register controls the data channel statue machine (DCSM).

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at its default value.
Bit 11	IOEN	0x0	rw	SD I/O enable functions This bit is set or cleared by software. If the bit is set, the DCSM performs an SD IO card-specific operation. 0: Disabled 1: Enabled
Bit 10	RDWTMODE	0x0	rw	Read wait mode This bit is set or cleared by software. If disabled, the SDIO_D2 controls the read wait; if enabled, the SDIO_CK controls the read wait. 0: Disabled 1: Enabled
Bit 9	RDWTSTOP	0x0	rw	Read wait stop This bit is set or cleared by software. While the RDWTSTART is set, If this bit is set, it indicates that read wait is stopped; if this bit cleared, it indicates that the read wait is in progress. 0: Read wait is in progress if the RDWTSTART is set. 1: Read wait is stopped if the RDWTSTART is set.
Bit 8	RDWTSTART	0x0	rw	Read wait start This bit is set or cleared by software. When this bit is set, read wait starts; when this bit is cleared, no actions occurs. 0: Read wait disabled 1: Read wait enabled
Bit 7: 4	BLKSIZE	0x0	rw	Data block size This bit is set or cleared by software. This field defines the length of data block when the block data transfer is selected. 0000: block length = 2^0 = 1 byte 0001: block length = 2^1 = 2 bytes 0010: block length = 2^2= 4 bytes 0011: block length = 2^3= 8 bytes 0100: block length = 2^4= 16 bytes 0101: block length = 2^5= 32 bytes 0110: block length = 2^6= 64 bytes 0111: block length = 2^7= 128 bytes 1000: block length = 2^8= 256 bytes 1001: block length = 2^9= 512 bytes 1010: block length = 2^{10}= 1024 bytes 1011: block length = 2^{11}= 2048 bytes 1100: block length = 2^{12}= 4096 bytes 1101: block length = 2^{13}= 8192 bytes 1110: block length = 2^{14} = 16384 bytes 1111: Reserved
Bit 3	DMAEN	0x0	rw	DMA enable bit This bit is set or cleared by software. 0: Disabled 1: Enabled
Bit 2	TFRMODE	0x0	rw	Data transfer mode selection This bit is set or cleared by software. If this bit is set, it indicates stream data transfer; if this bit cleared, it

				indicates block data transfer. 0: Disabled 1: Enabled
Bit 1	TFRDIR	0x0	rw	Data transfer direction selection This bit is set or cleared by software. If this bit is set, data transfer is from a card to a controller; if this bit is cleared, data transfer is from a controller to a card. 0: Disabled 1: Enabled
Bit 0	TFREN	0x0	rw	Data transfer enabled bit This bit is set or cleared by software. If this bit is set, data transfer starts. The DCSM enters the Wait_S or Wait_R state, depending on the direction bit TFRDIR. The DCSM goes to the read wait state if the RDWTSTART bit is set from the beginning of the transfer. It is not necessary to clear the enable bit after the end of data transfer but the SDIO_DTCTRL must be updated to enable a new data transfer. 0: Disabled 1: Enabled

Note: This register cannot be written within seven HCLK clock periods after data is written.

21.4.10 SDIO data counter register (SDIO_DTCNTR)

The SDIO_DTCNTR register loads the value from the SDIO_DTLEN register (See *Section 21.4.8*) when the DCSM moves from the idle state to the Wait_R or Wait_S state. During the data transfer, the counter value decrements to 0, and then the DCSM enters the idle state and sets the data status end flag bit DTCMPL.

Bit	Register	Reset value	Type	Description
Bit 31: 25	Reserved	0x00	resd	Kept at its default value.
Bit 24: 0	CNT	0x0000000	ro	Data count value When this register is read, the number of data bytes to be transferred is returned. Write access has no effect.

Note: This register can be read only when the data transfer is complete.

21.4.11 SDIO status register (SDIO_STS)

The SDIO_STS is a read-only register, containing two types of flags:

- Static flags (bits [23: 22, 10: 0]): These bits can be cleared by writing to the SDIO_INTCLR register (See *Section 21.4.12*).
- Dynamic flags (bit [21: 11]): These bit status changes with the state of the corresponding logic (for example, BUT full or empty flag is set or cleared as data written to the BUF)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIF	0x0	ro	SD I/O interrupt received
Bit 21	RXBUF	0x0	ro	Data available in receive BUF
Bit 20	TXBUF	0x0	ro	Data available in transmit BUF
Bit 19	RXBUFE	0x0	ro	Receive BUF empty
Bit 18	TXBUFE	0x0	ro	Transmit BUF empty If hardware flow control is enabled, the TXBUF_E signal becomes valid when the BUF contains two words.
Bit 17	RXBUFF	0x0	ro	Receive BUF full If hardware flow control is enabled, the RXBUF_F becomes valid two words before the BUF is full.
Bit 16	TXBUFF	0x0	ro	Transmit BUF full
Bit 15	RXBUFH	0x0	ro	Receive BUF half full There are at least 8 words in the BUF. This flag bit can be

				used as DMA request.
Bit 14	TXBUFH	0x0	ro	Transmit BUF half empty: At least 8 words can be written to the BUF. This flag bit can be used as DMA request.
Bit 13	DORX	0x0	ro	Data receive in progress
Bit 12	DOTX	0x0	ro	Data transmit in progress
Bit 11	DOCMD	0x0	ro	Command transfer in progress
Bit 10	DTBLKCMPL	0x0	ro	Data block sent/received CRC check passed)
Bit 9	SBITERR	0x0	ro	Start bit not detected on all data signals in wide bus mode
Bit 8	DTCMPL	0x0	ro	Data end (data counter, SDIO CNT, is zero)
Bit 7	CMDCMPL	0x0	ro	Command sent (no response required)
Bit 6	CMDRSPCMPL	0x0	ro	Command response (CRC check passed)
Bit 5	RXERRO	0x0	ro	Received BUF overrun error
Bit 4	TXERRU	0x0	ro	Transmit BUF underrun error
Bit 3	DTTIMEOUT	0x0	ro	Data timeout
Bit 2	CMDTIMEOUT	0x0	ro	Command response timeout The command timeout is a fixed value of 64 SDIO_CK clock periods.
Bit 1	DTFAIL	0x0	ro	Data block sent/received (CRC check failed)
Bit 0	CMDFAIL	0x0	ro	Command response received (CRC check failed)

21.4.12 SDIO clear interrupt register (SDIO_INTCLR)

The SDIO_INTCLR is a read-only register. Writing 1 to the corresponding register bit will clear the correspond bit in the SDIO_STS register.

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIF	0x0	rw	SD I/O interface flag clear bit This bit is set by software to clear the IOIF flag.
Bit 21: 11	Reserved	0x000	resd	Kept at its default value.
Bit 10	DTBLKCMPL	0x0	rw	DTBLKCMPL flag clear bit This bit is set by software to clear the DTBLKCMPL flag.
Bit 9	SBITERR	0x0	rw	SBITERR flag clear bit This bit is set to clear the SBITERR flag.
Bit 8	DTCMPL	0x0	rw	DTCMPL flag clear bit This bit is set by software to clear the DTCMPL flag.
Bit 7	CMDCMPL	0x0	rw	CMDCMPL flag clear bit This bit is set by software to clear the CMDCMPL flag.
Bit 6	CMDRSPCMPL	0x0	rw	MDRSPCMPL flag clear bit This bit is set by software to clear the CMDRSPCMPL flag.
Bit 5	RXERRO	0x0	rw	RXERRO flag clear bit This bit is set by software to clear the RXERRO flag.
Bit 4	TXERRU	0x0	rw	TXERRU flag clear bit This bit is set by software to clear the TXERRU flag.
Bit 3	DTTIMEOUT	0x0	rw	DTTIMEOUT flag clear bit This bit is set by software to clear the DTTIMEOUT flag.
Bit 2	CMDTIMEOUT	0x0	rw	CMDTIMEOUT flag clear bit This bit is set by software to clear the CMDTIMEOUT flag.
Bit 1	DTFAIL	0x0	rw	DTFAIL flag clear bit This bit is set by software to clear the DTFAIL flag.
Bit 0	CMDFAIL	0x0	rw	CMDFAIL flag clear bit This bit is set by software to clear the CMDFAIL flag.

21.4.13 SDIO interrupt mask register (SDIO_INTEN)

The SDIO_INTEN register determines which status bit generates an interrupt by setting the corresponding bit.

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at its default value.
Bit 22	IOIFIEN	0x0	rw	SD I/O mode received interrupt enable This bit is set or cleared by software to enable/disable the SD I/O mode received interrupt function. 0: Disabled 1: Enabled
Bit 21	RXBUFIEN	0x0	rw	Data available in RxBUF interrupt enable This bit is set or cleared by software to enable/disable the Data Available in RxBUF Interrupt. 0: Disabled 1: Enabled
Bit 20	TXBUFIEN	0x0	rw	Data available in TxBUF interrupt enable This bit is set or cleared by software to enable/disable the Data Available in TxBUF Interrupt. 0: Disabled 1: Enabled
Bit 19	RXBUFEIEN	0x0	rw	RxBUF empty interrupt enable This bit is set or cleared by software to enable/disable the RxBUF empty interrupt. 0: Disabled 1: Enabled
Bit 18	TXBUFEIEN	0x0	rw	TxBUF empty interrupt enable This bit is set or cleared by software to enable/disable the TxBUF empty interrupt. 0: Disabled 1: Enabled
Bit 17	RXBUFFIEN	0x0	rw	RxBUF full interrupt enable This bit is set or cleared by software to enable/disable the RxBUF full interrupt. 0: Disabled 1: Enabled
Bit 16	TXBUFFIEN	0x0	rw	TxBUF full interrupt enable This bit is set or cleared by software to enable/disable the TxBUF full interrupt. 0: Disabled 1: Enabled
Bit 15	RXBUFHIEN	0x0	rw	RxBUF half full interrupt enable This bit is set or cleared by software to enable/disable the RxBUF half full interrupt. 0: Disabled 1: Enabled
Bit 14	TXBUFHIEN	0x0	rw	TxBUF half empty interrupt enable This bit is set or cleared by software to enable/disable the TxBUF half empty interrupt. 0: Disabled 1: Enabled
Bit 13	DORXIEN	0x0	rw	Data receive acting interrupt enable This bit is set or cleared by software to enable/disable the Data receive acting interrupt. 0: Disabled

				1: Enabled
Bit 12	DOTXIEN	0x0	rw	Data transmit acting interrupt enable This bit is set or cleared by software to enable/disable the Data transmit acting interrupt. 0: Disabled 1: Enabled
Bit 11	DOCMDIEN	0x0	rw	Command acting interrupt enable This bit is set or cleared by software to enable/disable the Command acting interrupt. 0: Disabled 1: Enabled
Bit 10	DTBLKCMPLIEN	0x0	rw	Data block end interrupt enable This bit is set or cleared by software to enable/disable the Data block end interrupt. 0: Disabled 1: Enabled
Bit 9	SBITERRIEN	0x0	rw	Start bit error interrupt enable This bit is set or cleared by software to enable/disable the Start bit error interrupt. 0: Disabled 1: Enabled
Bit 8	DTCMPLIEN	0x0	rw	Data end interrupt enable This bit is set or cleared by software to enable/disable the Data end interrupt. 0: Disabled 1: Enabled
Bit 7	CMDCMPLIEN	0x0	rw	Command sent interrupt enable This bit is set or cleared by software to enable/disable the Command sent interrupt. 0: Disabled 1: Enabled
Bit 6	CMDRSPCMPLIEN	0x0	rw	Command response received interrupt enable This bit is set or cleared by software to enable/disable the Command response received interrupt. 0: Disabled 1: Enabled
Bit 5	RXERROIEN	0x0	rw	RxBUF overrun error interrupt enable This bit is set or cleared by software to enable/disable the RxBUF overrun error interrupt. 0: Disabled 1: Enabled
Bit 4	TXERRUIEN	0x0	rw	TxBUF underrun error interrupt enable This bit is set or cleared by software to enable/disable the TxBUF underrun error interrupt. 0: Disabled 1: Enabled
Bit 3	DTTIMEOUTIEN	0x0	rw	Data timeout interrupt enable This bit is set or cleared by software to enable/disable the Data timeout interrupt. 0: Disabled 1: Enabled
Bit 2	CMDTIMEOUTIEN	0x0	rw	Command timeout interrupt enable This bit is set or cleared by software to enable/disable the Command timeout interrupt. 0: Disabled

				1: Enabled
Bit 1	DTFAILIEN	0x0	rw	Data CRC fail interrupt enable This bit is set or cleared by software to enable/disable the Data CRC fail interrupt. 0: Disabled 1: Enabled
Bit 0	CMDFAILIEN	0x0	rw	Command CRC fail interrupt enable This bit is set or cleared by software to enable/disable the Command CRC fail interrupt. 0: Disabled 1: Enabled

21.4.14 SDIOBUF counter register (SDIO_BUFCNTR)

The SDIO_BUFCNTR register contains the number of words to be written to or read from the BUF. The BUF counter loads the value from the SDIO_DTLEN register (*Section 21.4.8*) when the data transfer bit TFREN is set in the SDIO_DTCTRL register. If the data length is not word-aligned, the remaining 1 to 3 bytes are regarded as a word.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x00	resd	Kept at its default value.
Bit 23: 0	CNT	0x000000	ro	Number of words to be written to or read from the BUF.

21.4.15 SDIO data BUF register (SDIO_BUF)

The receive and data BUF is group of a 32-bit wide registers that can be written or read. The BUF contains 32 registers on 32 sequential addresses. The CPU can use BUF for read/write multiple operations.

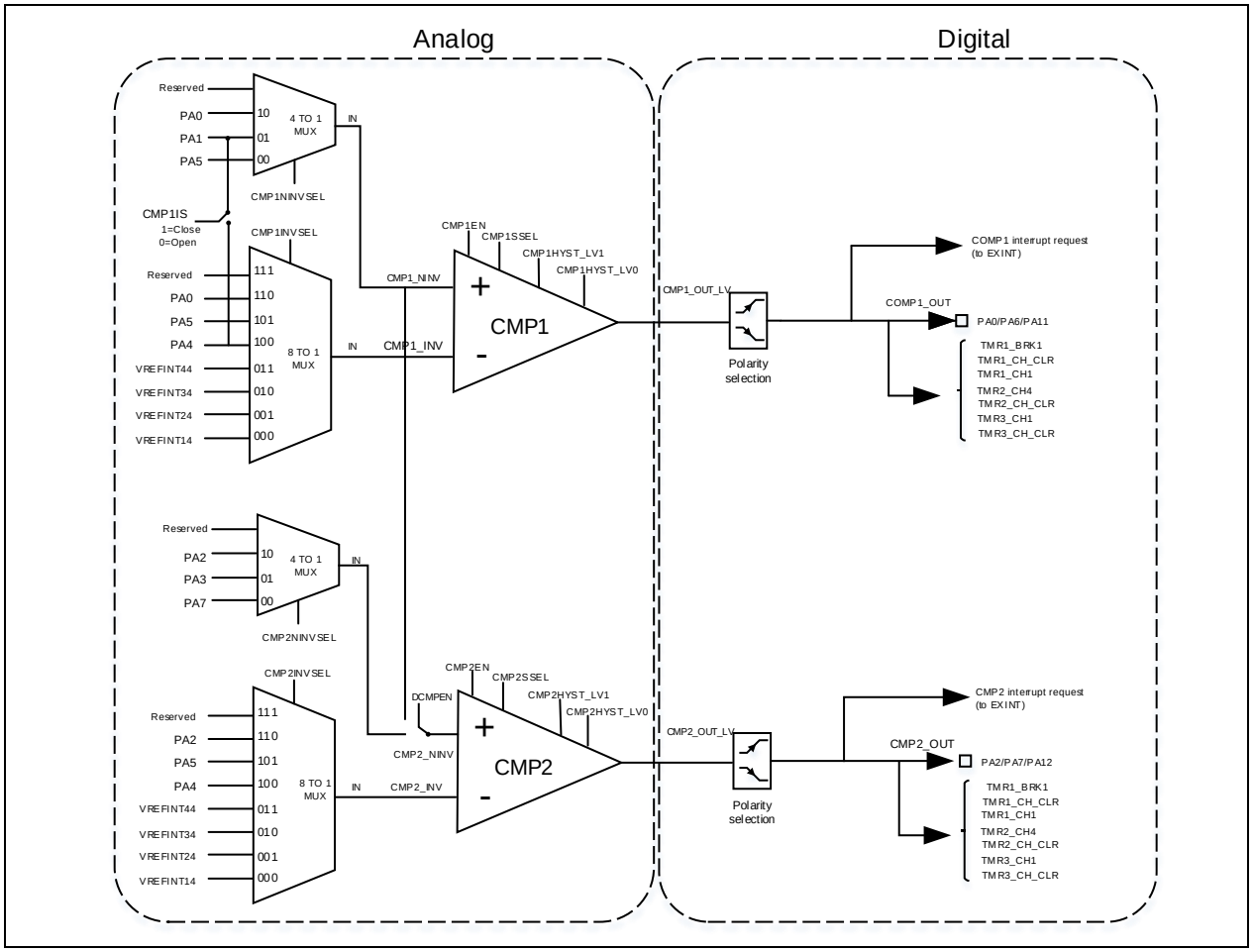
Bit	Register	Reset value	Type	Description
Bit 31: 0	DT	0x0000 0000	rw	Receive and transmit BUF data The BUF data occupies 32x 32-bit words, the address: SDIO base + 0x80 to SDIO base + 0xFC

22 Comparator (COMP)

22.1 COMP introduction

AT32F415 has two embedded ultra-low-power comparators, COMP1 and COMP2. They can be used for various purposes, such as, external analog signal monitor/control and wakeup from low-power mode, and working with other timers for pulse width measurement and PWM signal control.

Figure 22-1 Block Diagram of Comparator 1 and Comparator 2



22.2 Main features

- Programmable hysteresis level
- Programmable output polarity
- Programmable output speed
- Selectable positive/negative input sources
  - I/O pins
  - Internal reference voltage and three divider values (1/4, 1/2, 3/4)
- Output redirectioning
  - General-purpose I/O
  - Timer break input TMRx_BRK
  - Timer input capture TMR_CH
  - Timer output compare reference value clear TMR_CH_CLR
- COMP 1 and COMP2 are combined as a window comparator
- Wakeup device from low-power mode through EXINT controller

22.3 Interrupt management

Comparator 1 generates an external interrupt or event via EXTI line 19 to wakeup device from low-power mode;

Comparator 2 generates an external interrupt or event via EXTI line 20 to wakeup device from low-power mode.

For more detailed information, please refer to the section of interrupts and events.

22.4 Design tips

The following information can be used for design reference:

- Input/Output configuration

  As a comparator input, the I/Os must be configured as an analog mode. The comparator output can be remapped onto external I/Os through the CMP_MUX[1: 0] bit in the IOMUX_REMAP2 register.

Comparator output configuration:

Multiplexed	CMP_MUX [1: 0]=00	CMP_MUX [1: 0]=01	CMP_MUX [1: 0]=10
CMP1_OUT	PA0	PA6	PA11
CMP2_OUT	PA2	PA7	PA12

- Lock

  The CMP_CTRLSTS1 register can be write-protected. By setting CMPxWP=1, the corresponding bits in the CMP_CTRLSTS1 and CMP_CTRLSTS2 registers can be read-only after the completion of programming, and even the CMPxWP can be unlocked only after a system reset. This feature can be used for the applications with specific security requirements.

- Low-power mode

  The comparator is clocked by the PCLK, and uses system reset as its reset signal. The comparator still works in Deepsleep mode, which can be used as an EXINT interrupt source to wakeup device from low-power mode.

22.5 Functional overview

22.5.1 Analog comparator

Positive/Negative input selection

Select an I/O as a positive input source through the CMPxNINVSEL[1: 0] bit in the CMP_CTRLSTS1 register; Select an internal reference voltage, three voltage divider values or an I/O as a negative input source through the CMPxINVSEL[2: 0] bit.

Hysteresis

The hysteresis feature can be selected through the CMPxHYST[1: 0] bit in the CMP_CTRLSTS1 register.

This is used to avoid fake signal caused by noise. Hysteresis can be disabled (exit low-power mode) if not needed.

Operating mode

The controller can operate in fast speed/maximum power consumption, low speed/lowest power consumption in order to achieve the best trade-off between performance and power consumption, which is selected through the CMPxSSEL bit in the CMP_CTRLSTS1 register.

22.6 CMP registers

These registers must be accessed by words (32 bits).

Table 22-1 CMP register map and reset values

Register name	Offset	Reset value
CMP_CTRLSTS1	0x00	0x0000 0080
CMP_CTRLSTS2	0x04	0x0001 0001

22.6.1 Comparator control and status register 1 (COMP_CTRLSTS1)

Bit	Register	Reset value	Type	Description
Bit 31	CMP2WP	0x0	rw0c	Comparator 2 write protected 0: Disabled 1: Enabled Note: The COMP_CTRLSTS1[31:16] and COMP_CTRLSTS2[31:16] can be write-protected through this bit. This bit can be cleared through system reset.
Bit 30	CMP2VALUE	0x0	ro	Comparator 2 output value This bit is read-only, indicating the status of the current comparator 2 output (affected by the COMP2P bit).
Bit 29:28	CMP2HYST	0x0	rw	Comparator2 hysteresis 00: No hysteresis 01: Low hysteresis 10: Intermediate hysteresis 11: High hysteresis Please refer to hysteresis electrical characteristics
Bit 27	CMP2P	0x00	rw	Comparator2 polarity 0: Comparator 2 output value is not inverted 1: Comparator 2 output value is inverted
Bit 26:24	CMP2TAG	0x0	rw	Comparator output target This field controls the COMP2 output target. 000: No selection 001: Timer 1 brake input 010: Timer 1input capture 1 011: Timer 1 output compare clear 100: Timer 2 input capture 4 101: Timer 2 output compare clear 110: Timer 3 input capture 1 111: Timer 3 output compare clear
Bit 23	DCMPEN	0x0	rw	Double comparator mode enable 0: Double comparator mode disabled 1: Double comparator mode enabled Note: This bit is used to enable dual comparator mode, connecting the positive input of COMP2 with that of COMP1.
Bit 22:20	CMP2INVSEL	0x0	rw	Comparator2 inverting selection 000: 1/4 VREFIN 001: 1/2 VREFINT 010: 3/4 VREFINT 011: VREFINT 100: INM4(PA4) 101: INM5(PA5) 110: INM6(PA2) 111: Reserved
Bit 19	Reserved	0x0	resd	Kept at its default value.
Bit 18	CMP2SSEL	0x0	rw	Comparator 2 speed selection This bit is used to control the operating mode of comparators in order to adjust speed and power consumption. 0: High-speed/maximum power consumption 1: Low-speed/minimum power consumption
Bit 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	CMP2EN	0x0	rw	Comparator 2 enable

				This bit is used to enable/disable a comparator. 0: Comparator 2 disabled 1: Comparator 2 enabled
Bit 15	CMP1WP	0x0	rw0c	Comparator 1 write protect 0: Disabled 1: Enabled Note: The COMP_CTRLSTS1[15:0] and COMP_CTRLSTS2[15:0] bits are write-protected through this bit. This bit is cleared only by system reset.
Bit 14	CMP1VALUE	0x0	ro	Comparator 1 output value This bit is read-only, indicating the status of the current comparator 1 output (affected by the COMP1P bit).
Bit 13:12	CMP1HYST	0x0	rw	Comparator1 hysteresis 00: No hysteresis 01: Low hysteresis 10: Intermediate hysteresis 11: High hysteresis Please refer to hysteresis electrical characteristics
Bit 11	CMP1P	0x0	rw	Comparator1 polarity 0: Comparator 1 output value is not inverted 1: Comparator 1 output value is inverted
Bit 10:8	CMP1TAG	0x0	rw	Comparator output target This field controls the COMP2 output target. 000: No selection 001: Timer 1 brake input 010: Timer 1 input capture 1 011: Timer 1 output compare clear 100: Timer 2 input capture 4 101: Timer 2 output compare clear 110: Timer 3 input capture 1 111: Timer 3 output compare clear
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6:4	CMP1INVSEL	0x0	rw	Comparator1 inverting selection 000: 1/4 VREFIN 001: 1/2 VREFINT 010: 3/4 VREFINT 011: VREFINT 100: INM4(PA4) 101: INM5(PA5) 110: INM6(PA0) 111: Reserved
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	CMP1SSEL	0x0	rw	Comparator1 speed selection This bit is used to control the operating mode of comparators in order to adjust speed and power consumption. 0: High-speed/maximum power consumption 1: Low-speed/minimum power consumption
Bit 1	CMP1IS	0x0	rw	Comparator1 input shift 0: The switch is off. 1: The switch is on. Note: This bit is used to switch the connection between PA1 and PA4 of the comparator inverting input. It is only used for re-direction of input to high-impedance input, such as the non-inverting input of Comparator 1 (high-impedance switch).
Bit 0	CMP1EN	0x0	rw	Comparator1 enable This bit enables or disables a comparator. 0: Comparator 1 disabled 1: Comparator 1 enabled

22.6.2 Comparator Control/Status Register 2 (COMP_CTRLSTS2)

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at its default value.
Bit 17: 16	COMP2NINVSEL	0x1	rw	Comparator2 non-inverting input selection 00: INP0(PA7) 01: INP1(PA3) (by default) 10: INP2(PA2) 11: Reserved Note: This field is read-only when CMP2WP=1.
Bit 15: 2	Reserved	0x0000	resd	Kept at its default value.
Bit 1: 0	COMP1NINVSEL	0x1	rw	omparator1 non-inverting input selection 00: INP0(PA5) 01: INP1(PA1) (by default) 10: INP2(PA0) 11: Reserved Note: This field is read-only when CMP1WP=1.

23 Debug (DEBUG)

23.1 Debug introduction

Cortex®-M4 core provides powerful debugging features including halt and single step support, as well as trace function that is used for checking the details of the program execution. The debug features are implemented with a serial wire debug interface. The track information can be collected via a serial wire viewing interface, or TRACE interface (for a large track bandwidth). The track and debug interfaces can be combined into a single interface.

ARM Cortex®-M4 reference documentation:

- Cortex®-M4 Technical Reference Manual (TRM)
- ARM Debug Interface V5
- ARM CoreSight Design Kit revision r1p0 Technical Reference Manual

23.2 Debug and Trace

It is possible to support debugging for different peripherals, and configure the status of peripherals during debugging. For timers and watchdogs, the user can select whether or not to stop or continue counting during debugging; For CAN, the user can select whether or not to stop or continue updating receive registers during debugging; For I2C, the user can select whether or not to stop or continue SMBUS timeout counting.

In addition, code debugging is supported in Low-power mode. In Sleep mode, the clock programmed by code remains active for HCLK and FCLK to continue to work. In Deepsleep mode, HICK oscillator is enabled to feed FCLK and HCLK.

There are several ID codes inside the MCU, which is accessible by the debugger using the DEBUG_IDCODE at address 0xE0042000. It is part of the DEBUG and is mapped on the external PPB bus. These codes are accessible using the JTAG debug port or the SWD debug port or by the user software. They are even accessible while the MCU is under system reset.

Two trace interface modes supported: single-pin mode for serial wire view and multi-pin trace interface.

23.3 I/O pin control

SWJ-DP is supported in different packages of AT32F415. It uses 5 general-purpose I/O ports. After reset, the SWJ-DP can be immediately used by the debugger as a default function.

When the user wants to switch to a different debug port or disable debug feature, either IOMUX_MAPR or IOMUX_MAPR7 register can be configured to release these dedicated I/O pins. Once a corresponding debug I/O is released by the user, the GPIO controller takes control, and then these I/Os can be used as general-purpose I/Os.

For trace feature, it is possible to set the TRACE_IOEN and TRACE_MODE bits in the DEBUG_CTRL register to enable trace function and select trace modes.

Table 23-1 Trace function enable

TRACE_IOEN	Description
0	No Trace (default state)
1	Trace enabled

Table 23-2 Trace function mode

TRACE_MODE[1: 0]		PB3/JTDO/TRACESWO	PE2/TRACECK	PE3/TRACED[0]	PE4/TRACED[1]	PE5/TRACED[2]	PE6/TRACED[3]
00	Asynchronous trace	TRACESWO	Released (can be used as general-purpose I/Os)				
01	Synchronous trace	Released (can be used as general-purpose I/Os)	TRACECK	TRACED[0]	Released (can be used as general-purpose I/Os)		
10	Synchronous trace		TRACECK	TRACED[0]	TRACED[1]	Released (can be used as general-purpose I/Os)	
11	Synchronous trace		TRACECK	TRACED[0]	TRACED[1]	TRACED[2]	TRACED[3]

23.4 DEGUB registers

Table 23-3 shows DEBUG register map and reset values.

These peripheral registers must be accessed by word (32 bits)

Table 23-3 DEBUG register address and reset value

Register name	Offset	Reset value
DEBUG_IDCODE	0xE004 2000	0xXXXX XXXX
DEBUG_CTRL	0xE004 2004	0x0000 0000

23.4.1 DEBUG device ID (DEBUG_IDCODE)

MCU integrates an ID code that is used to identify MCU's revision code. The DEBUG_IDCODE register is mapped on the external PPB bus at address 0xE0042000. This code is accessible by the SW debug port or by the user code.

Bit	Register	Reset value	Type	Description
Bit 31: 0	PID	0xXXXX XXXX	ro	PID information

PID [31: 0]	AT32 part number	FLASH size	Packages
0x7003_0240	AT32F415RCT7	256KB	LQFP64_10x10
0x7003_0241	AT32F415CCT7	256KB	LQFP48_7x7
0x7003_0242	AT32F415KCU7-4	256KB	QFN32_4x4
0x7003_0243	AT32F415RCT7-7	256KB	LQFP64_7x7
0x7003_01C4	AT32F415RBT7	128KB	LQFP64_10x10
0x7003_01C5	AT32F415CBT7	128KB	LQFP48_7x7
0x7003_01C6	AT32F415KBU7-4	128KB	QFN32_4x4
0x7003_01C7	AT32F415RBT7-7	128KB	LQFP64_7x7
0x7003_0108	AT32F415R8T7	64KB	LQFP64_10x10
0x7003_0109	AT32F415C8T7	64KB	LQFP48_7x7
0x7003_010A	AT32F415K8U7-4	64KB	QFN32_4x4
0x7003_024C	AT32F415CCU7	256KB	QFN48_6x6
0x7003_01CD	AT32F415CBU7	128KB	QFN48_6x6

23.4.2 DEBUG control register (DEBUG_CTRL)

The DEBUG_CTRL register is mapped onto external PPB bus, base address 0xE0042000. This register is asynchronously reset by POR Reset (not reset by system reset). It can be written by the debugger under reset.

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at its default value.
Bit 30	TMR11_PAUSE	0x0	rw	TMR11 pause control bit 0: Work normally 1: Timer is disabled
Bit 29	TMR10_PAUSE	0x0	rw	TMR10 pause control bit 0: Work normally 1: Timer is disabled
Bit 28	TMR9_PAUSE	0x0	rw	TMR9 pause control bit 0: Work normally 1: Timer is disabled
Bit 27: 19	Reserved	0x000	resd	Kept at its default value.
Bit 18	TMR5_PAUSE	0x0	rw	TMR5 pause control bit 0: Work normally 1: Timer is disabled
Bit 17	Reserved	0x0	resd	Kept at its default value.
Bit 16	I2C2_SMBUS_TIMEOUT	0x0	rw	I^2C2 pause control bit 0: Work normally 1: I^2C2 SMBUS timeout control is disabled
Bit 15	I2C1_SMBUS_TIMEOUT	0x0	rw	I^2C1 pause control bit 0: Work normally 1: I^2C1 SMBUS timeout control is disabled
Bit 14	CAN1_PAUSE	0x0	rw	CAN1 pause control bit 0: CAN1 works normally 1: CAN1 receive registers do not continue to receive data
Bit 13	TMR4_PAUSE	0x0	rw	TMR4 pause control bit 0: Work normally 1: Timer is disabled
Bit 12	TMR3_PAUSE	0x0	rw	TMR3 pause control bit 0: Work normally 1: Timer is disabled
Bit 11	TMR2_PAUSE	0x0	rw	TMR2 pause control bit 0: Work normally 1: Timer is disabled
Bit 10	TMR1_PAUSE	0x0	rw	TMR1 pause control bit 0: Work normally 1: Timer is disabled
Bit 9	WWDT_PAUSE	0x0	rw	Window watchdog pause control bit 0: Window watchdog works normally 1: Window watchdog is stopped
Bit 8	WDT_PAUSE	0x0	rw	watchdog pause control bit 0: Watchdog works normally 1: Watchdog is stopped
Bit 7: 6	TRACE_MODE	0x0	rw	Trace pin assignment control 00: Asynchronous mode 01: Synchronous mode with a data length of 1 10: Synchronous mode with a data length of 2 11: Synchronous mode with a data length of 4

Bit 5	TRACE_IOEN	0x0	rw	Trace pin assignment enable 0: No trace (default state) 1: Trace is enabled
Bit 4: 3	Reserved	0x0	resd	Always 0
Bit 2	STANDBY_DEBUG	0x0	rw	Debug Standby mode control bit 0: The whole 1.2V digital circuit is unpowered in Standby mode 1: The whole 1.2V digital circuit is not unpowered in Standby mode, and the system clock is provided by the internal RC oscillator (HICK)
Bit 1	DEEPSLEEP_DEBUG	0x0	rw	Debug Deepsleep mode control bit 0: In Deepsleep mode, all clocks in the 1.2V domain are disabled. When exiting from Deepsleep mode, the internal RC oscillator (HICK) is enabled, and HICK is used as the system clock source, and the software must reprogram the system clock according to application requirements. 1: In Deepsleep mode, system clock is provided by the internal RC oscillator (HICK). When exiting from Deepsleep mode, HICK is used as the system clock source, and the software must reprogram the system clock. According to application requirements.
Bit 0	SLEEP_DEBUG	0x0	rw	Debug Sleep mode control bit 0: When entering Sleep mode, CPU HCLK clock is disabled, but other clocks remain active. When exiting from Sleep mode, it is not necessary to reprogram the clock system. 1: When entering Sleep mode, all clocks keep running.

24 Revision history

Document Revision History

Date	Version	Revision Note
2021.12.01	2.00	Initial release.
2022.06.27	2.01	1.Updated the descriptions in *11.5.1 Control register1 (I2C_CTRL1)* 2.Updated the descriptions in *18.5.3 ADC control register2 (ADC_CTRL2)* 3.Updated *Figure 19-7* 4.Updated the descriptions in *19.6.7 Error management*
2022.11.11	2.02	1. Updated descriptions of *Chapter 7* 2. Updated descriptions of *Chapter 10* 3. Updated descriptions of *Section 12.6.1* 4. Updated descriptions of *Section 12.6.2* 5. Updated descriptions of *Chapter 14*
2023.08.02	2.03	1. Updated descriptions of *Section 4.1.1 Clock sources* 2. Updated descriptions of *Section 7.4.11 IOMUX remap register6 (IOMUX_REMAP6)* 3. Updated descriptions of *Section 14 Timer* 4. Updated descriptions of *Section 12.8.3 Start bit and noise detection*

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services, and ARTERY assumes no liability whatsoever relating to the choice, selection or use of the ARTERY products and services described herein

No license, express or implied, to any intellectual property rights is granted under this document. If any part of this document deals with any third party products or services, it shall not be deemed a license granted by ARTERY for the use of such third party products or services, or any intellectual property contained therein, or considered as a warranty regarding the use in any manner of such third party products or services or any intellectual property contained therein.

Unless otherwise specified in ARTERY's terms and conditions of sale, ARTERY provides no warranties, express or implied, regarding the use and/or sale of ARTERY products, including but not limited to any implied warranties of merchantability, fitness for a particular purpose (and their equivalents under the laws of any jurisdiction), or infringement on any patent, copyright or other intellectual property right.

Purchasers hereby agree that ARTERY's products are not designed or authorized for use in: (A) any application with special requirements of safety such as life support and active implantable device, or system with functional safety requirements; (B) any aircraft application; (C) any aerospace application or environment; (D) any weapon application, and/or (E) or other uses where the failure of the device or product could result in personal injury, death, property damage. Purchasers' unauthorized use of them in the aforementioned applications, even if with a written notice, is solely at purchasers' risk, and Purchasers are solely responsible for meeting all legal and regulatory requirements in such use.

Resale of ARTERY products with provisions different from the statements and/or technical characteristics stated in this document shall immediately void any warranty grant by ARTERY for ARTERY's products or services described herein and shall not create or expand any liability of ARTERY in any manner whatsoever.