ARM®-based 32-bit Cortex®-M4F MCU, 64 to 256 KB Flash, sLib, 15 timers, 1 ADC, 18 communication interfaces (CAN and OTGFS)

- **Core: ARM®32-bit Cortex®-M4F CPU with FPU**
  - 150 MHz maximum frequency, with a Memory Protection Unit (MPU), single-cycle multiplication and hardware division
  - Floating Point Unit (FPU)
  - DSP instructions
- **Memories**
  - 64 to 256 Kbytes of Flash memory
  - 20 Kbytes of boot memory used as a Bootloader or as a general instruction/data memory (one-time configured)
  - sLib: configurable part of main Flash as a library area with code executable but secured, non-readable
  - Up to 48 Kbytes of SRAM
  - External memory controller (XMC) with 16-bit data bus supporting multiplexed PSRAM and NOR memories
- **XMC as LCD parallel interface, 8080/6800 modes**
- **Power control (PWC)**
  - 2.4 V to 3.6 V power supply
  - Power-on reset (POR)/low-voltage reset (LVR), and power voltage monitor (PVM)
  - Low-power modes: Sleep, Deepsleep and Standby modes
  - 20 x 32-bit battery powered registers (ERTC_BPR)
- **Clock and reset management (CRM)**
  - 4 to 25 MHz crystal (HEXT)
  - 48 MHz internal factory-trimmed HICK (±1% at TA=25 °C, ±2.5% at TA=-40 to +105 °C), with automatic clock calibration (ACC)
  - 32 kHz crystal (LEXT)
  - Low speed internal clock (LICK)
- **Analog**
  - 1 x 12-bit 5.33 MSPS A/D converter, up to 24 input channels, 12/10/8/6-bit configurable resolution; hardware over-sampling up to equivalent 16-bit resolution
  - Temperature sensor (V_{TS}), internal reference voltage (V_{INTR})
  - 2 x 12-bit D/A converters
- **DMA: 14-channel DMA controller**
- **Up to 87 fast GPIOs**
  - All mappable on 16 external interrupts (EXINT)
  - Almost 5 V-tolerant
- **Up to 15 timers (TMR)**
  - 1 x 16-bit 7-channel advanced timer with dead-time generator and emergency break
  - Up to 8 x16-bit and 1x 32-bit general-purpose timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
  - 2 x 16-bit basic timers
  - 2 x watchdog timers (general WDT and windowed WWDT)
  - SysTick timer: a 24-bit downcounter
- **ERTC: enhanced RTC with auto wakeup, alarm, subsecond accuracy, hardware calendar and calibration feature**
- **Up to 18 communication interfaces**
  - Up to 3 x I²C interfaces (SMBus/PMBus)
  - Up to 3 x SPIs (36 Mbit/s), all with multiplexed half-duplex I²S; 2 x half-duplex I²S combined for full-duplex support
  - Up to 8 x USARTs support master synchronization SPI and modem control, ISO7816 interface, LIN, IrDA, and RS485 drive enable, TX/RX swap
  - Up to 2 x CAN (2.0B Active), each with dedicated 256KB buffer
  - USB OTG full speed controller with on-chip PHY, dedicated 1280KB buffer, supporting crystal-less in device mode
  - Infrared transmitter (IRTMR)
- **CRC calculation unit**
- **96-bit ID (UID)**
- **Debug mode**
  - SWD and JTAG interfaces
- **Temperature range: -40 to +105℃**
- **Packaging**
  - LQFP100 14 x 14 mm　LQFP64 10 x 10 mm
    LQFP64 7 x 7 mm　LQFP48 7 x 7 mm
    QFN48 6 x 6 mm　QFN36 6 x 6 mm
    QFN32 4 x 4 mm
- **List of models**

Internal Flash	Model
64 Kbytes	AT32F423K8U7-4 AT32F423T8U7 AT32F423C8U7 AT32F423C8T7 AT32F423R8T7-7 AT32F423R8T7 AT32F423V8T7
128 Kbytes	AT32F423KBU7-4 AT32F423TBU7 AT32F423CBU7 AT32F423CBT7 AT32F423RBT7-7 AT32F423RBT7 AT32F423VBT7
256 Kbytes	AT32F423KCU7-4 AT32F423TCU7 AT32F423CCU7 AT32F423CCT7 AT32F423RCT7-7 AT32F423RCT7 AT32F423VCT7

Contents

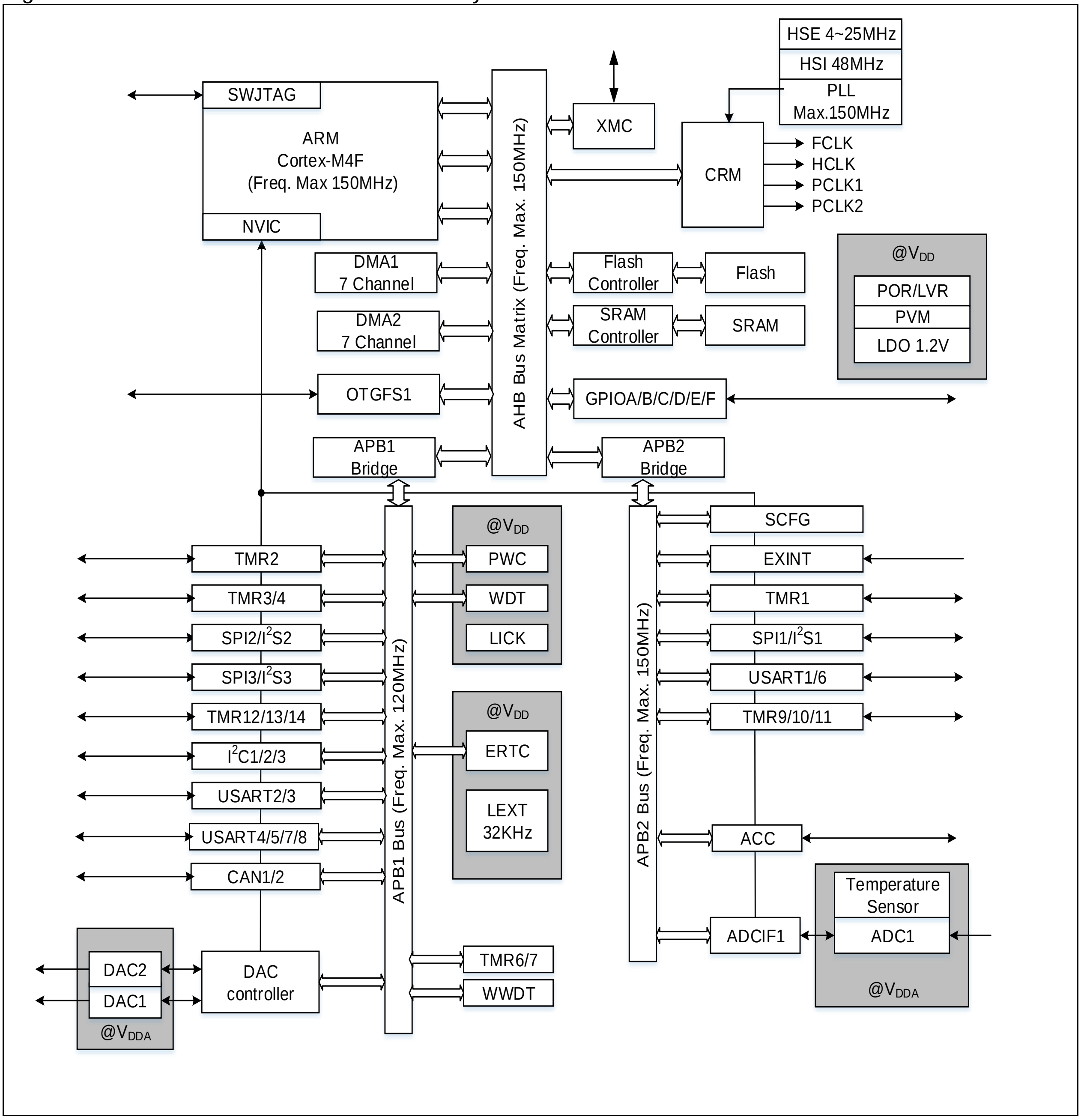
List of figures

List of tables

1 System architecture

AT32F423 series microcontrollers consist of 32-bit ARM ®Cortex®-M4F processor, multiple 16-bit and 32-bit timers, infrared transmitter (IRTMR), DMA controller, ERTC, communication interfaces such as SPI, I²C and USART, CAN bus controller, external memory controller (XMC), USB2.0 OTG full-speed interface, HICK with automatic clock calibration (ACC), 12-bit ADC, 12-bit DAC, programmable voltage monitor (PVM) and other peripherals. Cortex®-M4F processor 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, and single-precision (IEEE-754) and floating point unit (FPU), as shown in *Figure 1-1*.

Figure 1-1 AT32F423 series microcontrollers system architecture



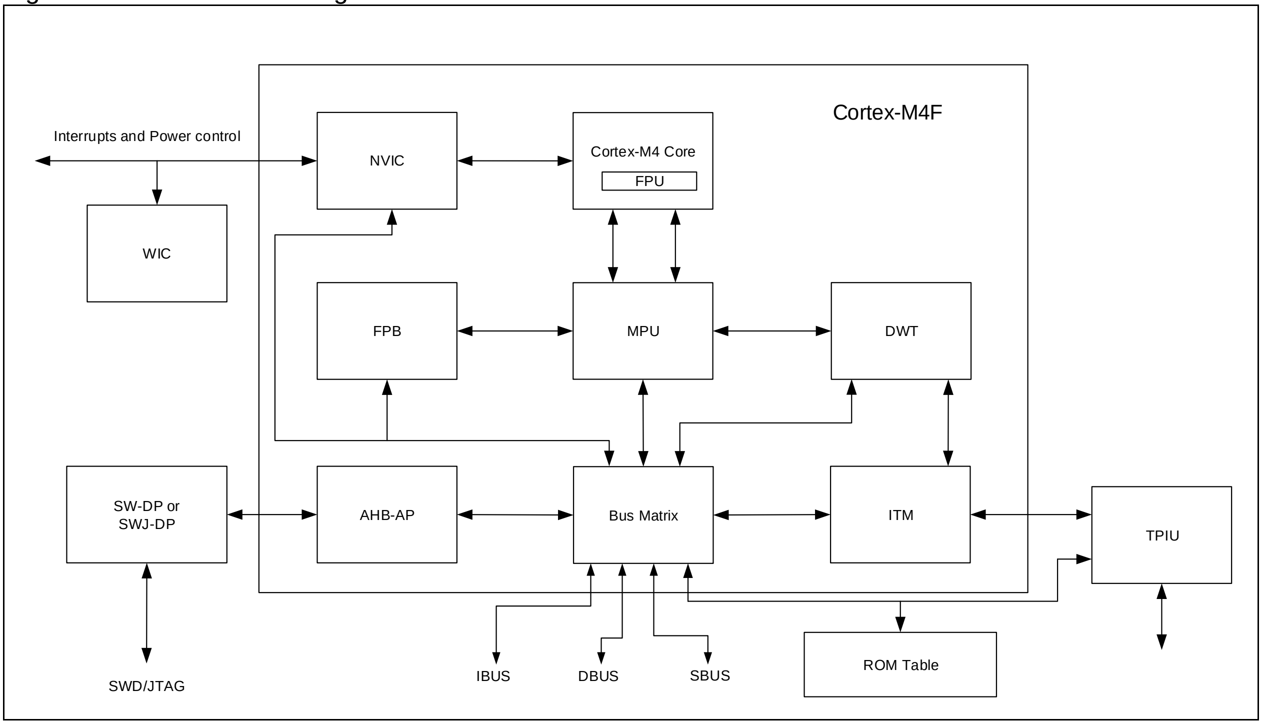
1.1 System overview

1.1.1 ARM Cortex®-M4F processor

Cortex®-M4F processor is a low-power consumption processor featuring with low gate count, low interrupt latency and low-cost debug. It supports DSP instruction set and FPU, and it is applicable to deeply embedded applications that require quicker response to interruption. Cortex®-M4F 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®-M4F processor. Please refer to *ARM®Cortex-M4 Technical Reference Manual* for more information.

Figure 1-2 Internal block diagram of Cortex®-M4F



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®-M4F memory includes two bit-band regions: the least significant 1 Mbyte of SRAM and the least significant 1 Mbyte 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 32-bit word. Thus, accessing to a bit in an alias region has the same effect as read-modify-write operation on the corresponding bit in a bit-band region.

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

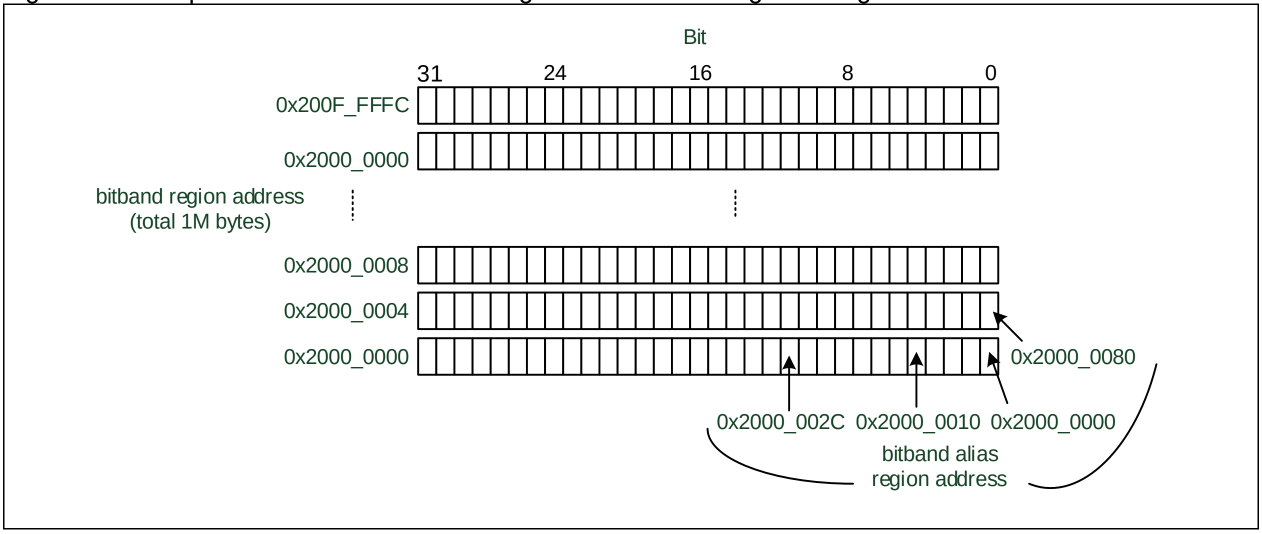
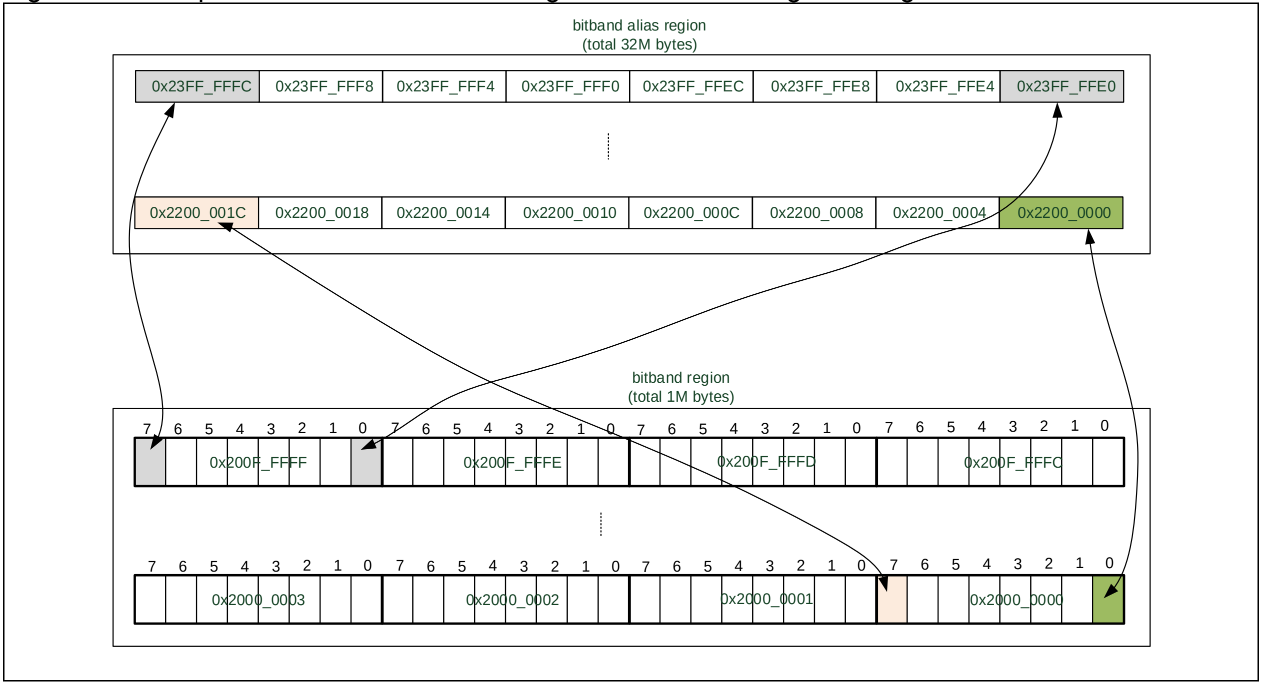


Figure 1-4 Comparison between bit-band region and its alias region: image B



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:

Least significant 1 Mbyte in SRAM: 0x2000_0000~0x200F_FFFF

Least significant 1 Mbyte in 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 terms of bit-band operation, one of the advantages is to control LED ON/OFF independently via GPIO pins. On the other hand, it brings great convenience for serial interface operations. In short, it is best suited to hardware I/O-intensive low-level applications.

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 to:

- 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 AT32F423 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	DebugLENonitor	Debug monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	Configurable	PendSV	Pendable request for system service	0x0000_0038
	6	Configurable	SysCNTRick	System tick timer	0x0000_003C
0	7	Configurable	WWDT	Window watchdog timer	0x0000_0040
1	8	Configurable	PVM	PVM from EXINT interrupt	0x0000_0044
2	9	Configurable	TAMPER	Tamper interrupt	0x0000_0048
3	10	Configurable	ERTC_WKUP	ERTC wakeup interrupt	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 line 0 interrupt	0x0000_0058
7	14	Configurable	EXINT1	EXINT line 1 interrupt	0x0000_005C
8	15	Configurable	EXINT2	EXINT line 2 interrupt	0x0000_0060
9	16	Configurable	EXINT3	EXINT line 3 interrupt	0x0000_0064
10	17	Configurable	EXINT4	EXINT line 4 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	ADC	ADC global interrupt	0x0000_0088
19	26	Configurable	CAN1_TX	CAN1 sent interrupt	0x0000_008C
20	27	Configurable	CAN1_RX0	CAN1 received 0 interrupt	0x0000_0090
21	28	Configurable	CAN1_RX1	CAN1 received 1 interrupt	0x0000_0094
22	29	Configurable	CAN_SE	CAN 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 and 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_ERR	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 line [15:10] interrupt	0x0000_00E0
41	48	Configurable	ERTCAlarm	ERTC alarm interrupt linked to EXINT	0x0000_00E4
42	49	Configurable	OTGFS1_WKUP	OTGFS1 standby wakeup interrupt linked to EXINT	0x0000_00E8
43	50	Configurable	TMR12	TMR12 global interrupt	0x0000_00EC
44	51	Configurable	TMR13	TMR13 global interrupt	0x0000_00F0
45	52	Configurable	TMR14	TMR14 global interrupt	0x0000_00F4
46	53	-	-	-	0x0000_00F8
47	54	-	-	-	0x0000_00FC
48	55	-	-	-	0x0000_0100
49	56	-	-	-	0x0000_0104
50	57	-	-	-	0x0000_0108

51	58	Configurable	SPI3	SPI3 global interrupt	0x0000_010C
52	59	Configurable	USART4	USART4 global interrupt	0x0000_0110
53	60	Configurable	USART5	USART5 global interrupt	0x0000_0114
54	61	Configurable	TMR6_DAC	TMR6 global interrupt DAC1 and DAC2 underflow error interrupt	0x0000_0118
55	62	Configurable	TMR7	TMR7 global interrupt	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	DMA2 channel 4 global interrupt	0x0000_012C
60	67	Configurable	DMA2 channel 5	DMA2 channel 5 global interrupt	0x0000_0130
61	68	-	-	-	0x0000_0134
62	69	-	-	-	0x0000_0138
63	70	Configurable	CAN2_TX	CAN2 sent interrupt	0x0000_013C
64	71	Configurable	CAN2_RX0	CAN2 received 0 interrupt	0x0000_0140
65	72	Configurable	CAN2_RX1	CAN2 received 1 interrupt	0x0000_0144
66	73	Configurable	CAN2_SE	CAN2 status error interrupt	0x0000_0148
67	74	Configurable	OTGFS1	OTGFS1 global interrupt	0x0000_014C
68	75	Configurable	DMA2 channel 6	DMA2 channel 6 global interrupt	0x0000_0150
69	76	Configurable	DMA2 channel 7	DMA2 channel 7 global interrupt	0x0000_0154
70	77	-	-	-	0x0000_0158
71	78	Configurable	USART6	USART6 global interrupt	0x0000_015C
72	79	Configurable	I2C3_EVT	I^2C2 event interrupt	0x0000_0160
73	80	Configurable	I2C3_ERR	I^2C2 error interrupt	0x0000_0164
74	81	-	-	-	0x0000_0168
75	82	-	-	-	0x0000_016C
76	83	-	-	-	0x0000_0170
77	84	-	-	-	0x0000_0174
78	85	-	-	-	0x0000_0178
79	86	-	-	-	0x0-000_017C
80	87	-	-	-	0x0000_0180
81	88	Configurable	FPU	FPU exception interrupt	0x0000_0184
82	89	Configurable	USART7	USART7 global interrupt	0x0000_0188
83	90	Configurable	USART8	USART8 global interrupt	0x0000_018C
84	91	-	-	-	0x0000_0190

85	92	-	-	-	0x0000_0194
86	93	-	-	-	0x0000_0198
87	94	-	-	-	0x0000_019C
88	95	-	-	-	0x0000_01A0
89	96	-	-	-	0x0000_01A4
90	97	-	-	-	0x0000_01A8
91	98	-	-	-	0x0000_01AC
92	99	-	-	-	0x0000_01B0
93	100	-	-	-	0x0000_01B4
94	101	Configurable	DMAMUX	DMAMUX overflow interrupt	0x0000_01B8
95	102	-	-	-	0x0000_01BC
96	103	-	-	-	0x0000_01C0
97	104	-	-	-	0x0000_01C4
98	105	-	-	-	0x0000_01C8
99	106	-	-	-	0x0000_01CC
100	107	-	-	-	0x0000_01D0
101	108	-	-	-	0x0000_01D4
102	109	-	-	-	0x0000_01D8
103	110	Configurable	ACC	ACC global interrupt	0x0000_01DC

1.1.4 System Tick (SysTick)

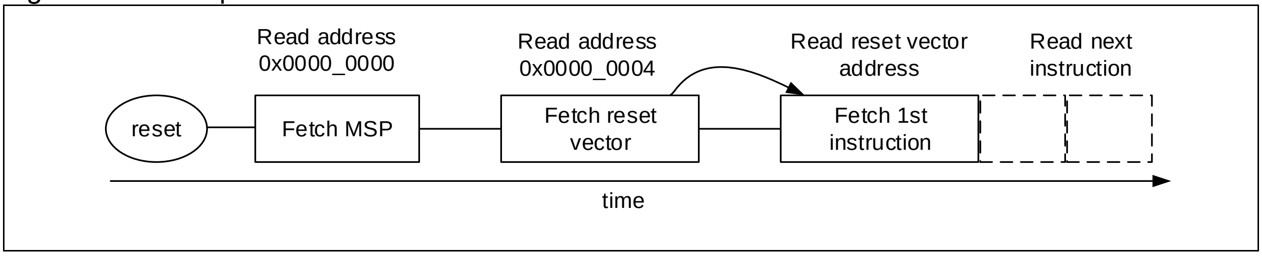
The System Tick is a 24-bit downcounter. It will be reloaded with the initial value automatically when it is decremented to zero. It can generate periodic interrupts, so it is often used as multi-task scheduling counter for embedded operating system, and also to call the periodic tasks for non-embedded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

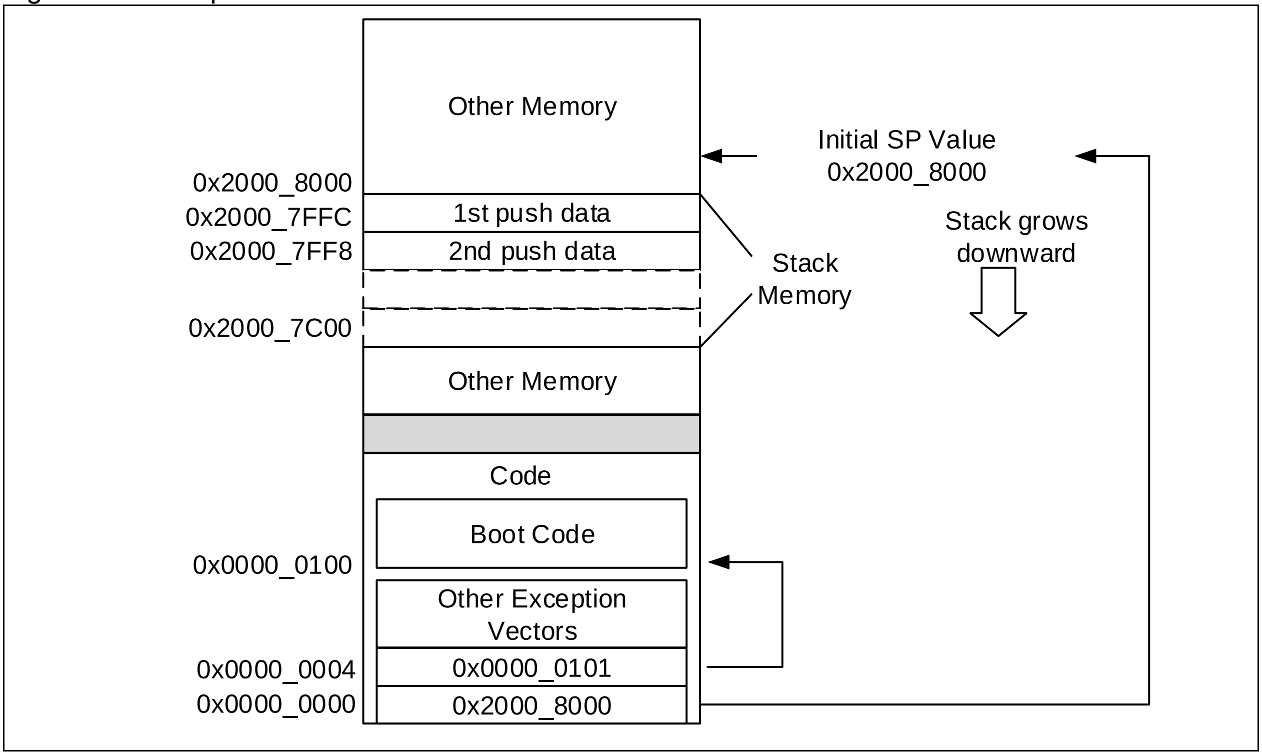
Figure 1-5 Reset process



Cortex®-M4F 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®-M4F 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 AT32F423 series, the main Flash memory, boot memory or SRAM can be remapped to the CODE area between 0x0000_0000 and 0x07FF_FFFF. nBOOT1 corresponds to the value of the bit nBOOT1 in the SSB of the User System Data (USD). nBOOT1 and BOOT0 are used to set the specific memory from which CODE starts.

{nBOOT1, BOOT0}=00/10: CODE starts from the main Flash memory.

{nBOOT1, BOOT0}=11: CODE starts from boot memory.

{nBOOT1, BOOT0}=01: CODE starts from SRAM.

After a system reset or when leaving from Standby mode, the pin values of both nBOOT1 and BOOT0 will be relatched.

When CODE starts from SRAM, the status of BOOT is latched, and it is impossible to load a new boot mode through a system reset. At this point, the power-on reset must be performed to reload a new boot mode.

Boot memory contains an embedded bootloader 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 this bit. Reading it returns to 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 Base address and reset value of 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, 0x0040 = 64 Kbytes

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

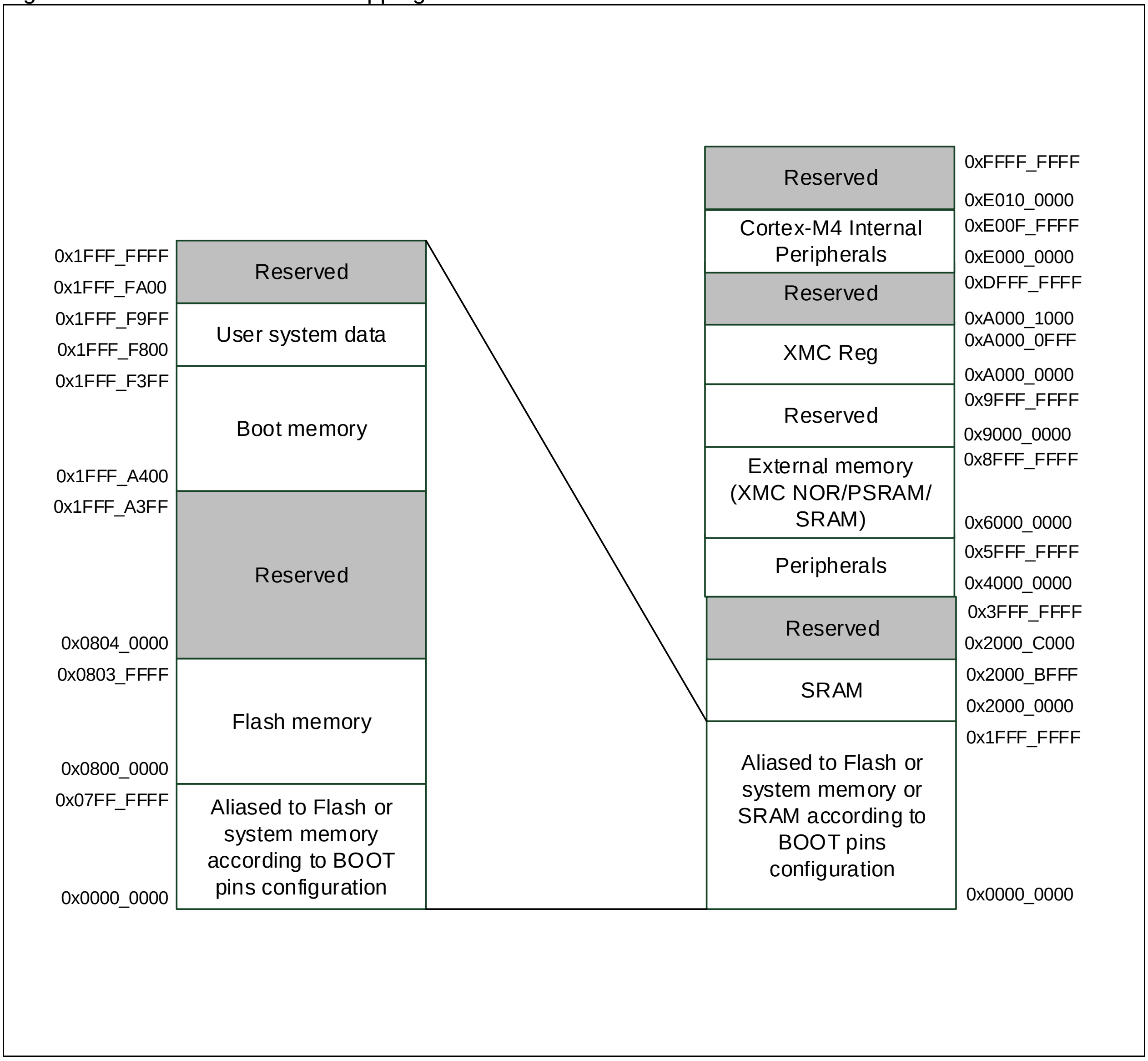
Note: UID[95:88] is Series ID, which is 0x12 for AT32F423.

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 2-1 AT32F423 address mapping



2.2 Flash memory

AT32F423 series provide up to 256 KB of on-chip Flash memory, supporting a single-cycle 32-bit read operation.

Refer to *Chapter 5* 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)

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 3	0x0800 1800 – 0x0800 1FFF
		Sector 4	0x0800 2000 – 0x0800 27FF
		…	…
		Sector 127	0x0803 F800 – 0x0803 FFFF
Information block		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512 B user system area	0x1FFF F800 – 0x1FFF F9FF

Flash memory organization (128 KB)

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

Table 2-2 Flash memory organization (128 KB)

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 3	0x0800 0C00 – 0x0800 0FFF
		Sector 4	0x0800 1000 – 0x0800 13FF
		…	…
		Sector 127	0x0801 FC00 – 0x0801 FFFF
Information block		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512 B user system data	0x1FFF F800 – 0x1FFF F9FF

Flash memory organization (64 KB)

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

Table 2-3 Flash memory organization (64 KB)

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 3	0x0800 0C00 – 0x0800 0FFF
		Sector 4	0x0800 1000 – 0x0800 13FF
		…	…
		Sector 63	0x0800 FC00 – 0x0800 FFFF
Information block		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512 B user system data	0x1FFF F800 – 0x1FFF F9FF

2.3 SRAM memory

The AT32F423 series contain a 48-KB on-chip SRAM that starts at the address of 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	0xC000 0000 - 0xFFFF FFFF	Reserved
	0xB000 0000 - 0xBFFF FFFF	Reserved
	0xA000 1000 - 0xAFFF FFFF	Reserved
	0xA000 0000 - 0xA000 0FFF	XMC_REG
	0x9000 0000 - 0x9FFF FFFF	Reserved
	0x6000 0000 - 0x8FFF FFFF	XMC
	0x5004 0000 - 0x5FFF FFFF	Reserved
	0x5000 0000 - 0x5003 FFFF	OTG_FS1
	0x4002 6800 - 0x4FFF FFFF	Reserved
	0x4002 6400 - 0x4002 67FF	DMA2
	0x4002 6000 - 0x4002 63FF	DMA1
	0x4002 4000 - 0x4002 5FFF	Reserved
	0x4002 3C00 - 0x4002 3FFF	Flash memory interface (FLASH)
	0x4002 3800 - 0x4002 3BFF	Clock and reset manage (CRM)
	0x4002 3400 - 0x4002 37FF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2000 - 0x4002 2FFF	Reserved
	0x4002 1C00 - 0x4002 1FFF	Reserved
	0x4002 1800 - 0x4002 1BFF	Reserved
	0x4002 1400 - 0x4002 17FF	GPIO port F
	0x4002 1000 - 0x4002 13FF	GPIO port E
	0x4002 0C00 - 0x4002 0FFF	GPIO port D
	0x4002 0800 - 0x4002 0BFF	GPIO port C
	0x4002 0400 - 0x4002 07FF	GPIO port B
	0x4002 0000 - 0x4002 03FF	GPIO port A
APB2	0x4001 8000 - 0x4001 FFFF	Reserved
	0x4001 7C00 - 0x4001 7FFF	Reserved
	0x4001 7800 - 0x4001 7BFF	Reserved
	0x4001 7400 - 0x4001 77FF	ACC
	0x4001 4C00 - 0x4001 73FF	Reserved
	0x4001 4800 - 0x4001 4BFF	TMR11 timer
	0x4001 4400 - 0x4001 47FF	TMR10 timer
	0x4001 4000 - 0x4001 43FF	TMR9 timer
	0x4001 3C00 - 0x4001 3FFF	EXINT

	0x4001 3800 - 0x4001 3BFF	SCFG
	0x4001 3400 - 0x4001 37FF	Reserved
	0x4001 3000 - 0x4001 33FF	SPI1/I2S1
	0x4001 2400 - 0x4001 2FFF	Reserved
	0x4001 2000 - 0x4001 23FF	ADC
	0x4001 1800 - 0x4001 1FFF	Reserved
	0x4001 1400 - 0x4001 17FF	USART6
	0x4001 1000 - 0x4001 13FF	USART1
	0x4001 0800 - 0x4001 0FFF	Reserved
	0x4001 0400 - 0x4001 07FF	Reserved
	0x4001 0000 - 0x4001 03FF	TMR1 timer
APB1	0x4000 8000 - 0x4000 FFFF	Reserved
	0x4000 7C00 - 0x4000 7FFF	USART8
	0x4000 7800 - 0x4000 7BFF	USART7
	0x4000 7400 - 0x4000 77FF	DAC
	0x4000 7000 - 0x4000 73FF	Power control (PWC)
	0x4000 6C00 - 0x4000 6FFF	Reserved
	0x4000 6800 - 0x4000 6BFF	CAN2
	0x4000 6400 - 0x4000 67FF	CAN1
	0x4000 6000 - 0x4000 63FF	Reserved
	0x4000 5C00 - 0x4000 5FFF	I2C3
	0x4000 5800 - 0x4000 5BFF	I2C2
	0x4000 5400 - 0x4000 57FF	I2C1
	0x4000 5000 - 0x4000 53FF	USART5
	0x4000 4C00 - 0x4000 4FFF	USART4
	0x4000 4800 - 0x4000 4BFF	USART3
	0x4000 4400 - 0x4000 47FF	USART2
	0x4000 4000 - 0x4000 43FF	Reserved
	0x4000 3C00 - 0x4000 3FFF	SPI3/I2S3
	0x4000 3800 - 0x4000 3BFF	SPI2/I2S2
	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	Reserved
	0x4000 2000 - 0x4000 23FF	TMR14 timer
	0x4000 1C00 - 0x4000 1FFF	TMR13 timer
	0x4000 1800 - 0x4000 1BFF	TMR12 timer
	0x4000 1400 - 0x4000 17FF	TMR7 timer
	0x4000 1000 - 0x4000 13FF	TMR6 timer
	0x4000 0C00 - 0x4000 0FFF	Reserved

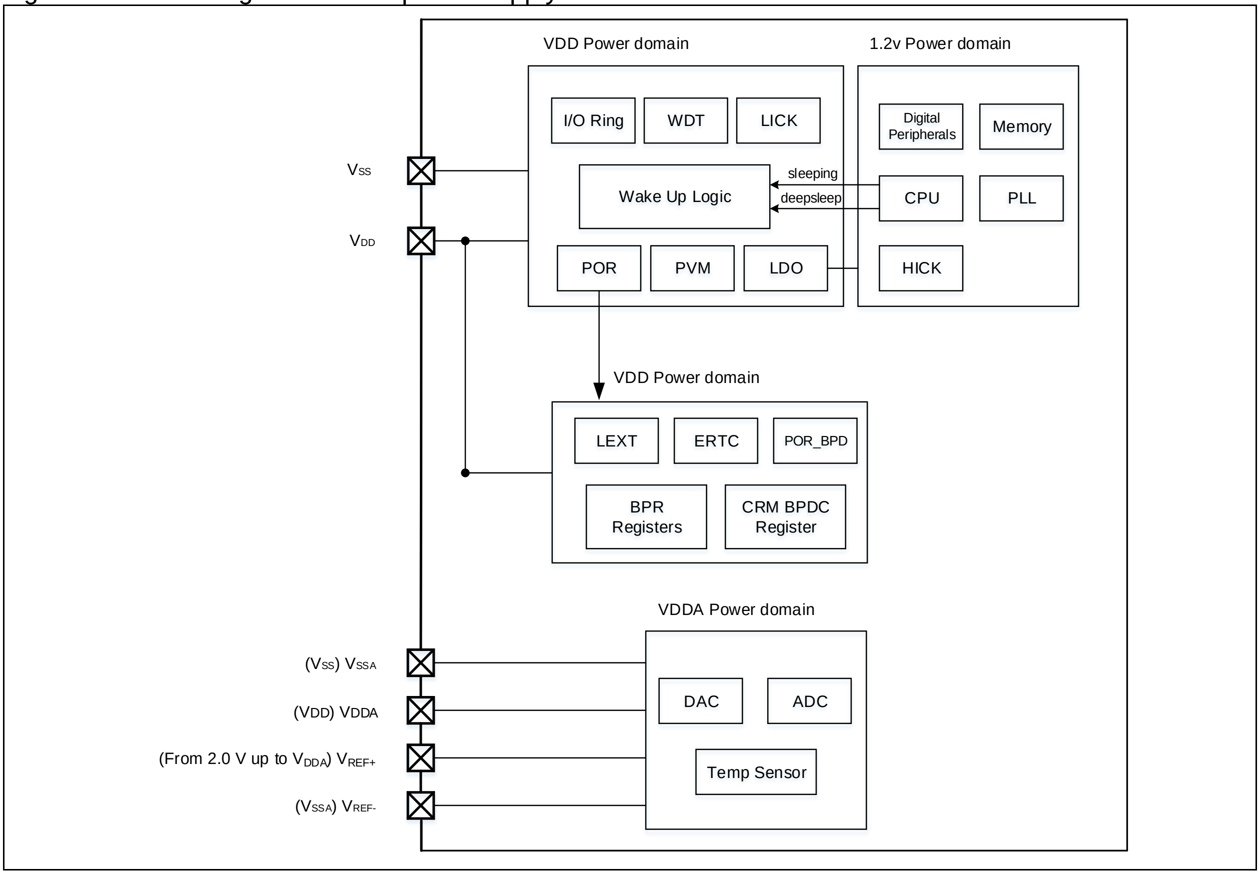
	0x4000 0800 - 0x4000 0BFF	TMR4 timer
	0x4000 0400 - 0x4000 07FF	TMR3 timer
	0x4000 0000 - 0x4000 03FF	TMR2 timer

3 Power control (PWC)

3.1 Introduction

For AT32F423 series, its operating voltage supply is 2.4 V ~ 3.6 V, with a temperature range of -40~+105ºC. 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 tradeoff among the conflicting demands of CPU operating time, speed and power consumption. The AT32F423 series have two power domains — VDD/VDDA and1.2 V domain. The VDD/VDDA domain is supplied directly by external power, and the 1.2 V domain is powered by the embedded LDO in the VDD/VDDA domain.

Figure 3-1 Block diagram of each power supply



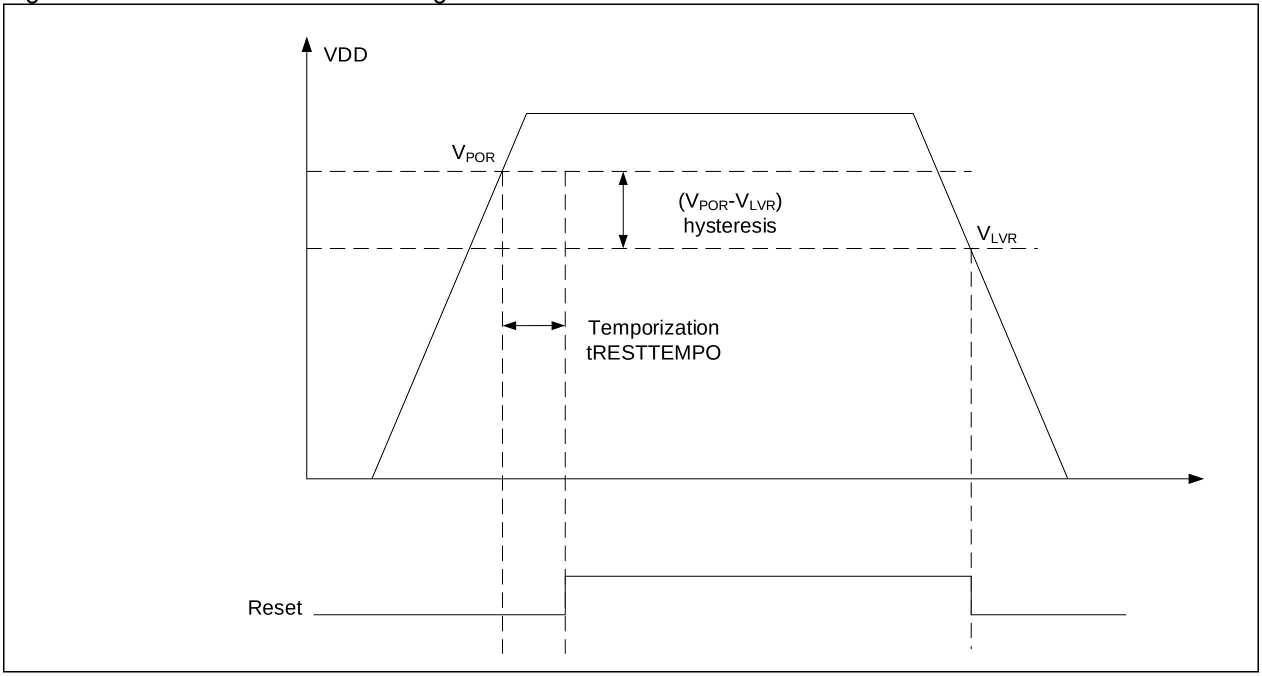
3.2 Main features

- Two power domains: VDD/VDDA domain and 1.2V 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 generate an interrupt or event when the supply voltage is lower or higher than a programmed threshold
- VDD/VDDA applies independent 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

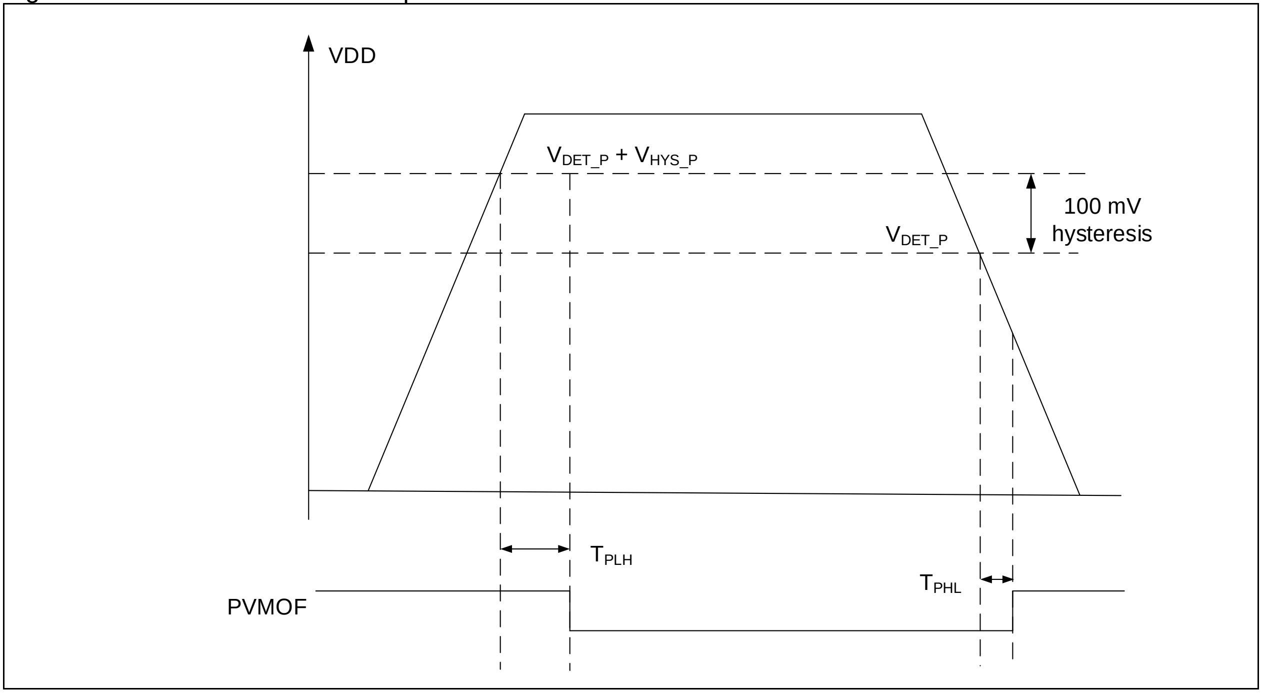


3.4 Power voltage monitor (PVM)

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 (PPL). Such power domain is supplied by LDO (voltage regulator).

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, power-on reset/low voltage reset (POR/LVR), LDO, ERTC circuit, LEXT and all PAD circuits. The VDDA domain contains DAC/ADC (DA/AD converters), temperature sensor and so on.

Typically, to ensure a better accuracy of ADC/DAC at a low voltage, the digital circuit is supplied by VDD while the analog circuit is powered by VDDA. On 64-pin packages and packages with less pins, the external reference voltage VREF+ and VREF- are connected to VDDA pin and VSSA pin, respectively.

In Run mode, the LDO supplies full power to the 1.2 V core domain. The LDO output voltage is selected through the PWC_LDOOV register. The maximum operating frequency for the system depends on the selected output voltage. Refer to *AT32F423 Series Datasheet* for details.

Note: The LDO output voltage is changeable only when the HEXT or HICK is used as system clock.

LDO output voltage regulation

1) Select HICK or HEXT as system clock
2) Change LDO voltage by setting the LDOOVSEL [1:0] bit
3) Set the FLASH_PSR register
4) Set the targeted frequency for PLL-related registers, enable PLL, and wait for PLL_STBL
5) Set pre-division factors for AHB and APB
6) Enable auto step-by-step frequency switch function when the PLL frequency is greater than 108 MHz
7) Switch the system clock to PLL

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®-M4F 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.

SLEEP-ON-EXIT mode:

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

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 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®-M4F system control register and clearing the LPSEL bit in the PWC_CTRL register before executing 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 addition, in Deepsleep mode, with VRSEL=1 and LDO being in low-power mode, the power consumption of the system can be further reduced by setting the VREXLPEN bit.

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 Deepsleep 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 Deepsleep mode is entered by executing a WFE instruction, the event 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 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.

Low-power Deepsleep LDO voltage regulation process (note that Sleep and Standby modes have no such limits)

1) Select HICK as system clock
2) Change LDO voltage to 1 V by setting the LDOOVSEL[1:0] bit and set the VREXLPEN bit
3) Set the LDO in low-power mode by setting the VRSEL bit
4) System enters Deepsleep state
5) System exits Deepsleep state (if wakeup conditions are met)
6) Change LDO voltage by setting the LDOOVSEL[1:0] bit
7) Set the FLASH_PSR register
8) If the HEXT is used as PLL clock, enable HEXT and wait for HEXTSTBL
9) Set the targeted frequency for PLL-related registers
10) Enable PLL and wait for PLL_STBL
11) Set pre-division factors for AHB and APB
12) Enable auto step-by-step frequency switch function when PLL frequency is greater than 108 MHz
13) Switch the system clock to the PLL

Note: If the clock, after low-power mode is waken up, needs to keep the same state as in low-power mode, the above-mentioned steps 7/9/11 can be ignored.

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. SRAM and register contents are lost. Only VDD/VDDA domain remains supplied.

The Standby mode is entered by the following procedures:

- Set the SLEEPDEEP bit in the Cortex®-M4F 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 or 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 enabled.

The MCU exits the Standby mode when a rising edge on the WKUP pin, a rising edge of an ERTC alarm event, an ERTC tamper event, ERTC timestamp, ERTC periodic automatic wakeup, an external reset (NRST pin) or a WDT reset 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®-M4F 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 half-words (16 bits) or words (32 bits).

Table 3-1 PWC register map and reset values

Register abbr.	Offset	Reset values
PWC_CTRL	0x00	0x0000 0000
PWC_CTRLSTS	0x04	0x0000 0000
PWC_LDOOV	0x10	0x0000 0012

3.7.1 Power control register (PWC_CTRL)

Bit	Name	Reset value	Type	Description
Bit 31:9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	BPWEN	0x0	rw	Battery powered domain write enable 0: Disabled 1: Enabled Note: After reset, the battery powered domain write access is disabled. To write, this bit must be set.
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	0x0	rw	Power voltage monitoring enable 0: Disabled 1: Enabled
Bit 3	CLSEF	0x0	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	0x0	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	0x0	rw	Low power mode select when Cortex®-M4F sleepdeep 0: Enter Deepsleep mode

				1: Enter Standby mode
Bit 0	VRSEL	0x0	rw	Voltage regulator state select when Deepsleep mode 0: Enabled 1: Low-power consumption mode

3.7.2 Power control/status register (PWC_CTRLSTS)

Bit	Name	Reset value	Type	Description
Bit 31:15	Reserved	0x0000 0	resd	Kept at its default value.
Bit 14	SWPEN7	0x0	rw	Standby wake-up pin7 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.
Bit 13	SWPEN6	0x0	rw	Standby wake-up pin6 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.
Bit 12:10	Reserved	0x0	resd	Kept at its default value.
Bit 9	SWPEN2	0x0	rw	Standby wake-up pin2 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.
Bit 8	SWPEN1	0x0	rw	Standby wake-up pin1 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.
Bit 7:3	Reserved	0x00	resd	Kept at its default value.
Bit 2	PVMOF	0x0	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	0x0	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	0x0	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.

3.7.3 LDO output voltage select register (PWC_LDOOV)

Bit	Name	Reset value	Type	Description
Bit 31:5	Reserved	0x0000000	resd	Kept at its default value.
Bit 4	VREXLPEN	0x1	rw	Voltage regulator extra low power mode enable This bit works together with the LPSEL and VRSEL bits in the PWC_CTRL register, and is valid when VRSEL = 1 and the chip enters Deepsleep mode. 0: Disabled 1: Enabled Note: To enable the extra low power mode, set VREXLPEN before setting the LPSEL and VRSEL bits.
Bit 3:2	Reserved	0x0	resd	Kept at its default value.
Bit 1:0	LDOOVSEL	0x2	rw	LDO output voltage select 00: 1.0 V 01: Reserved 10: 1.2 V 11: 1.3 V

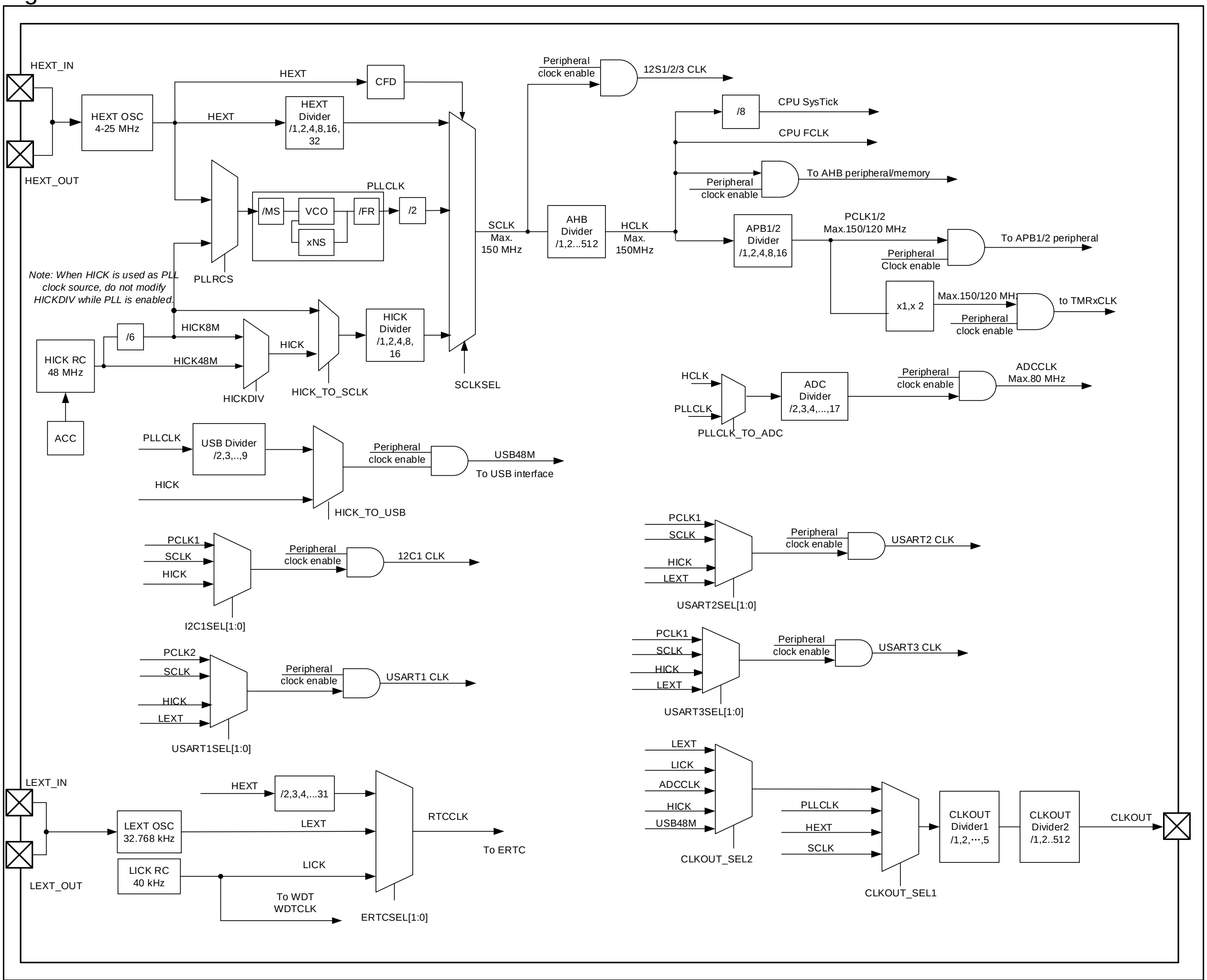
4 Clock and reset manage (CRM)

4.1 Clock

AT32F423 series provide different clock sources:

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

Figure 4-1 AT32F423 clock tree



AHB, APB2 and APB1 all support multiple frequency division. The AHB domain has a maximum frequency of 150 MHz; APB1 is up to 120 MHz and APB2 is up to 150 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 released for GPIO control.

- 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, its startup time is shorter than the HEXT crystal oscillator. The HICK clock frequency of each device is calibrated by ARTERY to ±1% accuracy (25°C) in factory. The factory-trimmed 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 by setting the HICKTRIM[5: 0] bit in the clock control register.

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

When HICK is used as system clock, set the HICK_TO_SCLK bit in the CRM_MISC2 register by the following procedures:

1. Set the HICK_TO_SCLK_DIV bit in the CRM_MISC2 register to 0x4
2. Wait for 50 system clock cycles
3. Change the value of HICK_TO_SCLK in the CRM_MISC2 register
4. Set the HICK_TO_SCLK_DIV bit to the desired value

When HICK is used as system clock, set the HICKDIV bit in the CRM_MISC1 register by the following procedures:

1. Set the HICK_TO_SCLK_DIV bit in the CRM_MISC2 register to 0x4
2. Wait for 50 system clock cycles
3. Set the HICK_TO_SCLK bit in the CRM_MISC2 register to 0x1
4. Change the value of HICKDIV in the CRM_MISC1 register
5. Restore the value of HICK_TO_SCLK bit to the original value
6. Set the HICK_TO_SCLK_DIV bit to the desired value

- PLL clock

The HICK or HEXT clock can be used as an input clock source of the PLL. The PLL input clock, after being divided by a pre-divider internally, is sent to the VCO for frequency multiplication, and the VCO output frequency is output after being divided by a post-divider. At the same time, the clock after pre-divider must remain between 2 MHz and 16 MHz, and the VCO operating frequency must be kept between 500 MHz and 1000 MHz. The PLL must be configured before being enabled. The reason is that the configuration parameters cannot be changed once PLL is enabled. The PLL clock signal is not released before it becomes stable.

PLL formula:

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

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

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

For example, when the PLL input clock is 25 MHz, the PLL output frequency = 25 x 192 / (5 x 4) = 240 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 low-power and accurate 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 pin must remain in 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 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 if 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 HCLK or PLL divided by 2, 3, 4, 5…17.

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.

The USB clock source can be switched between HICK and PLL frequency divider. If the HICK is selected as a clock source, the USB clock should be set as 48 MHz; If the PLL frequency divider is selected as a clock source, the USB frequency divider provides 48 MHz USBCLK, and thus the PLL must be set as (48*N) MHz (N=2,3,4,5…).

ERTC clock source: divided HEXT oscillator, LEXT oscillator and LICK oscillator. Once the clock source is selected, it cannot be altered unless a reset is performed. When the VDD is powered off, the ERTC state is not guaranteed because HEXT, LEXT and LICK are all powered off.

I^2C1 clock source: system clock SCLK, PCLK1 clock and HICK oscillator.

USART1 clock source: system clock SCLK, PCLK2 clock, HICK oscillator and LEXT oscillator.

USART2/3 clock source: system clock SCLK, PCLK1 clock, HICK oscillator and LEXT oscillator.

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 TMR1/9/10/11/12/13/14 and a CFD 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 or when the AHB prescaler factor is changed. Once it is enabled, the AHB bus is halted by hardware until 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 output signal to be output to external CLKOUT pin. That is, ADCCLK, USB48M, SCLK, LICK, LEXT, HICK, HEXT and PLLCLK can be used as CLKOUT clock. When being used as the CLKOUT clock output pin, the corresponding GPIO port registers must be configured accordingly.

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

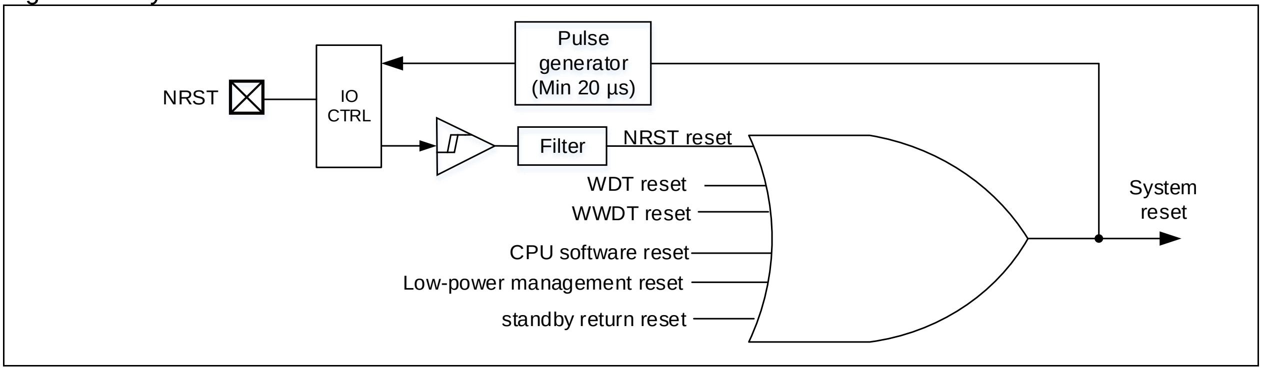
4.2.1 System reset

AT32F423 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®-M4F 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 bit in the user system data area).
- When exiting Standby mode

NRST reset, WDT reset, WWDT reset, software reset and low-power management reset set all registers to their reset values except the clock control/status register (CRM_CTRLSTS) and the register in the battery powered domain; the 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 power on (if it has 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_PLLCFG	0x004	0x0003 3002
CRM_CFG	0x008	0x4000 0000
CRM_CLKINT	0x00C	0x0000 0000
CRM_AHBRST1	0x010	0x0000 0000
CRM_AHBRST2	0x014	0x0000 0000
CRM_AHBRST3	0x018	0x0000 0000
CRM_APB1RST	0x020	0x0000 0000
CRM_APB2RST	0x024	0x0000 0000
CRM_AHBEN1	0x030	0x0000 0000
CRM_AHBEN2	0x034	0x0000 0000
CRM_AHBEN3	0x038	0x0000 0000
CRM_APB1EN	0x040	0x0000 0000
CRM_APB2EN	0x044	0x0000 0000
CRM_AHBLPEN1	0x050	0x0141 903F
CRM_AHBLPEN2	0x054	0x0000 0080
CRM_AHBLPEN3	0x058	0x0000 0001
CRM_APB1LPEN	0x060	0xF6FE C9F7
CRM_APB2LPEN	0x064	0x2007 5131
CRM_PICLKS	0x068	0x0000 0000
CRM_BPDC	0x070	0x0000 0000
CRM_CTRLSTS	0x074	0x0C00 0000
CRM_MISC1	0x0A0	0x000F 0000
CRM_MISC2	0x0A4	0x0000 000D

4.3.1 Clock control register (CRM_CTRL)

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

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: Disabled 1: Enabled
Bit 23:20	Reserved	0x0	resd	Kept at its default value.

Bit 19	CFDEN	0x0	rw	Clock Failure Detection enable 0: Disabled 1: Enabled
Bit 18	HEXTBYPS	0x0	rw	High speed external crystal bypass This bit can be set by software only when the HEXT is disabled. 0: Disabled 1: Enabled
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: Disabled 1: Enabled
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, it needs adjust 40 kHz (design value) 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% accuracy.
Bit 1	HICKSTBL	0x1	ro	High speed internal clock stable This bit is 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 PLL clock configuration register (CRM_PLLCFG)

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

Bit	Name	Reset value	Type	Description
Bit 31:23	Reserved	0x000	resd	Kept at its default value.
Bit 22	PLLRCS	0x0	rw	PLL reference clock select The PLL reference clock source is selected by setting this bit to "1" or clearing this bit. It can be written only when the PLL is disabled. 0: HICK is used as PLL reference clock 1: HEXT is used as PLL reference clock
Bit 21:19	Reserved	0x0	resd	Kept at its default value.

Bit 18:16	PLL_FR	0x3	rw	PLL post-division PLL_FR range (0~5) 000: PLL post-division=1 001: PLL post-division=2 010: PLL post-division=4 011: PLL post-division=8 100: PLL post-division=16 101: PLL post-division=32 Others: Reserved Attention should be paid to the correlation between the PLL_FR value and post-division factor.
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14:6	PLL_NS	0x0C0	rw	PLL multiplication factor PLL_NS range (31~500) 000000000 ~ 000011110: Forbidden 000011111: 31 000100000: 32 000100001: 33 …… 111110011: 499 111110100: 500 111110101~111111111: Forbidden
Bit 5:4	Reserved	0x0	resd	Kept at its default value.
Bit 3:0	PLL_MS	0x2	rw	PLL pre-division PLL_MS range (1~15) 0000: Forbidden 0001: 1 0010: 2 0011: 3 …… 1110: 14 1111: 15

Note: PLL clock formulas:

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

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

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

4.3.3 Clock configuration register (CRM_CFG)

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

Bit	Name	Reset value	Type	Description
Bit 31:30	CLKOUT_SEL1	0x1	rw	Clock output selection 1 This field is set and cleared by software. 00: System clock (SCLK) selected 01: Secondary clock output selected by the CLKOUT_SEL2 bit in the CRM_MISC1 register 10: External oscillator clock (HEXT) selected 11: PLL clock output Note: This clock out may be cut off during the startup and switch of CLKOUT clock source. While being used as an output to the CLKOUT pin, the system clock output must be no more than 50 MHz (the maximum frequency of an IO port).

Bit 29:27	CLKOUTDIV1	0x0	rw	Clock output division1 0xx: CLKOUT 100: CLKOUT/2 101: CLKOUT/3 110: CLKOUT/4 111: CLKOUT/5
Bit 26:21	Reserved	0x00	resd	Kept at its default value.
Bit 20:16	ERTCDIV	0x00	rw	HEXT division for ERTC clock This field is set and cleared by software to divide the HEXT for ERTC clock. These bits must be configured before selecting the ERTC clock source. 00000: Forbidden 00001: Forbidden 00010: HEXT/2 00011: HEXT/3 00100: HEXT/4 … 11110: HEXT/30 11111: HEXT/31
Bit 15:13	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
Bit 12:10	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
Bit 9:8	Reserved	0x0	resd	Kept at its default value.
Bit 7:4	AHBDIV	0x0	rw	AHB division 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 Note: Enable auto step-by-step system clock switch by setting the AUTO_STEP_EN bit before modifying the AHBDIV bit.
Bit 3:2	SCLKSTS	0x0	ro	System clock select status 00: HICK 01: HEXT 10: PLL/2 11: Reserved. Kept at its default value.
Bit 1:0	SCLKSEL	0x0	rw	System clock select 00: HICK 01: HEXT 10: PLL/2 11: Reserved. Kept at its default value. Note: Enable auto step-by-step system clock switch by setting the AUTO_STEP_EN bit before modifying the SCLKSEL bit.

4.3.4 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 interrupt 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 failu occurs. 0: No clock failure 1: Clock failure
Bit 6:5	Reserved	0x0	resd	Kept 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 flag Set by hardware. 0: LICK is not ready 1: LICK is ready

4.3.5 AHB peripheral reset register 1 (CRM_AHBRST1)

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

Bit	Name	Reset value	Type	Description
Bit 31:25	Reserved	0x00	resd	Kept at its default value.
Bit 24	DMA2RST	0x0	rw	DMA2 reset 0: Does not reset DMA2 1: Reset DMA2
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1RST	0x0	rw	DMA1 reset 0: Does not reset DMA1 1: Reset DMA1
Bit 21:13	Reserved	0x000	resd	Kept at its default value.
Bit 12	CRCRST	0x0	rw	CRC reset 0: Does not reset CRC 1: Reset CRC
Bit 11:6	Reserved	0x00	resd	Kept at its default value.
Bit 5	GPIOFRST	0x0	rw	IO port F reset 0: Does not reset IO port F 1: Reset IO port F
Bit 4	GPIOERST	0x0	rw	IO port E reset 0: Does not reset IO port E 1: Reset IO port E
Bit 3	GPIODRST	0x0	rw	IO port D reset 0: Does not reset IO port D 1: Reset IO port D
Bit 2	GPIOCRST	0x0	rw	IO port C reset 0: Does not reset IO port C 1: Reset IO port C
Bit 1	GPIOBRST	0x0	rw	IO port B reset 0: Does not reset IO port B 1: Reset IO port B
Bit 0	GPIOARST	0x0	rw	IO port A reset 0: Does not reset IO port A 1: Reset IO port A

4.3.6 AHB peripheral reset register 2 (CRM_AHBRST2)

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

Bit	Name	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	OTGFS1RST	0x0	rw	OTGFS1 reset 0: Does not reset OTGFS1 1: Reset OTGFS1
Bit 6:0	Reserved	0x00	resd	Kept at its default value.

4.3.7 AHB peripheral reset register 3 (CRM_AHBRST3)

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

Bit	Name	Reset value	Type	Description
Bit 31:1	Reserved	0x00000000	resd	Kept at its default value.
Bit 0	XMCRST	0x0	rw	XMC reset 0: Does not reset XMC 1: Reset XMC

4.3.8 APB1 peripheral reset register (CRM_APB1RST)

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

Bit	Name	Reset value	Type	Description
Bit 31	USART8RST	0x0	rw	USART8 reset 0: Does not reset USART8 1: Reset USART8
Bit 30	USART7RST	0x0	rw	USART7 reset 0: Does not reset USART7 1: Reset USART7
Bit 29	DACRST	0x0	rw	DAC interface reset 0: Does not reset DAC interface 1: Reset DAC interface
Bit 28	PWCRST	0x0	rw	Power interface reset 0: Does not reset power interface 1: Reset power interface
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAN2RST	0x0	rw	CAN2 reset 0: Does not reset CAN2 1: Reset CAN2
Bit 25	CAN1RST	0x0	rw	CAN1 reset 0: Does not reset CAN1 1: Reset CAN1
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3RST	0x0	rw	I^2C3 reset 0: Does not reset I^2C3 1: Reset I^2C3
Bit 22	I2C2RST	0x0	rw	I^2C2 reset 0: Does not reset I^2C2 1: Reset I^2C2
Bit 21	I2C1RST	0x0	rw	I^2C1 reset 0: Does not reset I^2C1 1: Reset I^2C1
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 Set and cleared by software. 0: No effect 1: Reset USART3
Bit 17	USART2RST	0x0	rw	USART2 reset 0: Does not reset USART2 1: Reset USART2
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3RST	0x0	rw	SPI3 reset 0: Does not reset SPI3 1: Reset SPI3
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	Window watchdog reset 0: Does not reset window watchdog 1: Reset window watchdog
Bit 10:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14RST	0x0	rw	Timer14 reset 0: Does not reset Timer14 1: Reset Timer14
Bit 7	TMR13RST	0x0	rw	Timer13 reset 0: Does not reset Timer13 1: Reset Timer13
Bit 6	TMR12RST	0x0	rw	Timer12 reset 0: Does not reset Timer12 1: Reset Timer12
Bit 5	TMR7RST	0x0	rw	Timer7 reset 0: Does not reset Timer7 1: Reset Timer7
Bit 4	TMR6RST	0x0	rw	Timer6 reset 0: Does not reset Timer6 1: Reset Timer6
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	TMR4RST	0x0	rw	Timer4 reset 0: Does not reset Timer4 1: Reset Timer4
Bit 1	TMR3RST	0x0	rw	Timer3 reset 0: Does not reset Timer3 1: Reset Timer3
Bit 0	TMR2RST	0x0	rw	Timer2 reset 0: Does not reset Timer2 1: Reset Timer2

4.3.9 APB2 peripheral reset register (CRM_APB2RST)

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

Bit	Name	Reset value	Type	Description
Bit 31:30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCRST	0x0	rw	ACC reset 0: Does not reset ACC 1: Reset ACC
Bit 28:19	Reserved	0x000	resd	Kept at its default value.
Bit 18	TMR11RST	0x0	rw	Timer11 reset 0: Does not reset Timer11 1: Reset Timer11
Bit 17	TMR10RST	0x0	rw	Timer10 reset 0: Does not reset Timer10 1: Reset Timer10
Bit 16	TMR9RST	0x0	rw	Timer9 reset 0: Does not reset Timer9 1: Reset Timer9
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGRST	0x0	rw	SCFG reset 0: Does not reset SCFG 1: Reset SCFG
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 11:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ADCRST	0x0	rw	ADC interface reset 0: Does not reset ADC interface 1: Reset ADC interface
Bit 7:6	Reserved	0x0	resd	Kept at its default value.
Bit 5	USART6RST	0x0	rw	USART6 reset 0: Does not reset USART6

				1: Reset USART6
Bit 4	USART1RST	0x0	rw	USART1 reset) 0: Does not reset USART1 1: Reset USART1
Bit 3:1	Reserved	0x0	resd	Kept at its default value.
Bit 0	TMR1RST	0x0	rw	TMR1 timer reset) 0: Does not reset TMR1 1: Reset TMR1

4.3.10 AHB peripheral clock enable register 1 (CRM_AHBEN1)

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

Bit	Name	Reset value	Type	Description
Bit 31:25	Reserved	0x00	resd	Kept at its default value.
Bit 24	DMA2EN	0x0	rw	DMA2 clock enable 0: Disabled 1: Enabled
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1EN	0x0	rw	DMA1 clock enable 0: Disabled 1: Enabled
Bit 21:13	Reserved	0x000	resd	Kept at its default value.
Bit 12	CRCEN	0x0	rw	CRC clock enable 0: Disabled 1: Enabled
Bit 11:6	Reserved	0x00	resd	Kept at its default value.
Bit 5	GPIOFEN	0x0	rw	IO port F clock enable 0: Disabled 1: Enabled
Bit 4	GPIOEEN	0x0	rw	IO port E clock enable 0: Disabled 1: Enabled
Bit 3	GPIODEN	0x0	rw	IO port D clock enable 0: Disabled 1: Enabled
Bit 2	GPIOCEN	0x0	rw	IO port C clock enable 0: Disabled 1: Enabled
Bit 1	GPIOBEN	0x0	rw	IO port B clock enable 0: Disabled 1: Enabled
Bit 0	GPIOAEN	0x0	rw	IO port A clock enable 0: Disabled 1: Enabled

4.3.11 AHB peripheral clock enable register 2 (CRM_AHBEN2)

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

Bit	Name	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	OTGFS1EN	0x0	rw	OTGFS1 clock enable 0: Disabled 1: Enabled
Bit 6:0	Reserved	0x00	resd	Kept at its default value.

4.3.12 AHB peripheral clock enable register 3 (CRM_AHBEN3)

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

Bit	Name	Reset value	Type	Description
Bit 31:1	Reserved	0x00000000	resd	Kept at its default value.
Bit 0	XMCEN	0x0	rw	XMC clock enable 0: Disabled 1: Enabled

4.3.13 APB1 peripheral clock enable register (CRM_APB1EN)

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

Bit	Name	Reset value	Type	Description
Bit 31	USART8EN	0x0	rw	USART8 clock enable 0: Disabled 1: Enabled
Bit 30	USART7EN	0x0	rw	USART7 clock enable 0: Disabled 1: Enabled
Bit 29	DACEN	0x0	rw	DAC interface clock enable 0: Disabled 1: Enabled
Bit 28	PWCEN	0x0	rw	Power interface clock enable 0: Disabled 1: Enabled
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAN2EN	0x0	rw	CAN2 clock enable 0: Disabled 1: Enabled
Bit 25	CAN1EN	0x0	rw	CAN1 clock enable 0: Disabled 1: Enabled
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3EN	0x0	rw	I^2C3 clock enable 0: Disabled 1: Enabled
Bit 22	I2C2EN	0x0	rw	I^2C2 clock enable 0: Disabled 1: Enabled
Bit 21	I2C1EN	0x0	rw	I^2C1 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	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3EN	0x0	rw	SPI3 clock enable 0: Disabled 1: Enabled
Bit 14	SPI2EN	0x0	rw	SPI 2 clock enable 0: Disabled 1: Enabled
Bit 13:12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTEN	0x0	rw	Window watchdog clock enable

				0: Disabled 1: Enabled
Bit 10:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14EN	0x0	rw	Timer14 clock enable 0: Disabled 1: Enabled
Bit 7	TMR13EN	0x0	rw	Timer13 clock enable 0: Disabled 1: Enabled
Bit 6	TMR12EN	0x0	rw	Timer12 clock enable 0: Disabled 1: Enabled
Bit 5	TMR7EN	0x0	rw	Timer 7 clock enable 0: Disabled 1: Enabled
Bit 4	TMR6EN	0x0	rw	Timer 6 clock enable 0: Disabled 1: Enabled
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	TMR4EN	0x0	rw	Timer 4 clock enable 0: Disabled 1: Enabled
Bit 1	TMR3EN	0x0	rw	Timer 3 clock enable 0: Disabled 1: Enabled
Bit 0	TMR2EN	0x0	rw	Timer 2 clock enable 0: Disabled 1: Enabled

4.3.14 APB2 peripheral clock enable register (CRM_APB2EN)

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

Bit	Name	Reset value	Type	Description
Bit 31:30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCEN	0x0	rw	ACC clock enable 0: Disabled 1: Enabled
Bit 28:19	Reserved	0x00	resd	Kept at its default value.
Bit 18	TMR11EN	0x0	rw	Timer11 clock enable 0: Disabled 1: Enabled
Bit 17	TMR10EN	0x0	rw	Timer10 clock enable 0: Disabled 1: Enabled
Bit 16	TMR9EN	0x0	rw	Timer9 clock enable 0: Disabled 1: Enabled
Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGEN	0x0	rw	SCFG 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 11:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ADCEN	0x0	rw	ADC interface clock enable 0: Disabled 1: Enabled
Bit 7:6	Reserved	0x0	resd	Kept at its default value.
Bit 5	USART6EN	0x0	rw	USART6 clock enable 0: Disabled 1: Enabled

Bit 4	USART1EN	0x0	rw	USART1 clock enable 0: Disabled 1: Enabled
Bit 3:1	Reserved	0x0	resd	Kept at its default value.
Bit 0	TMR1EN	0x0	rw	TMR1 timer clock enable 0: Disabled 1: Enabled

4.3.15 AHB peripheral clock enable in low power mode register (CRM_AHBLPEN1)

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

Bit	Name	Reset value	Type	Description
Bit 31:25	Reserved	0x00	resd	Kept at its default value.
Bit 24	DMA2LPEN	0x1	rw	DMA2 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 23	Reserved	0x0	resd	Kept at its default value.
Bit 22	DMA1LPEN	0x1	rw	DMA1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 21:17	Reserved	0x00	resd	Kept at its default value.
Bit 16	SRAMLPEN	0x1	rw	SRAM clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 15	FLASHLPEN	0x1	rw	FLASH clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 14:13	Reserved	0x0	resd	Kept at its default value.
Bit 12	CRCLPEN	0x1	rw	CRC clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 11:6	Reserved	0x0	resd	Kept at its default value.
Bit 5	GPIOFLPEN	0x1	rw	IO port F clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 4	GPIOELPEN	0x1	rw	IO port E clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 3	GPIODLPEN	0x1	rw	IO port D clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 2	GPIOCLPEN	0x1	rw	IO port C clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 1	GPIOBLPEN	0x1	rw	IO port B clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 0	GPIOALPEN	0x1	rw	IO port A clock enable in Sleep mode 0: Disabled 1: Enabled

4.3.16 AHB peripheral clock enable in low power mode register 2 (CRM_AHBLPEN2)

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

Bit	Name	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	resd	Kept at its default value.
Bit 7	OTGFS1LPEN	0x1	rw	OTGFS1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 6:0	Reserved	0x00	resd	Kept at its default value.

4.3.17 AHB peripheral clock enable in low power mode register 3 (CRM_AHBLPEN3)

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

Bit	Name	Reset value	Type	Description
Bit 31:1	Reserved	0x00000000	resd	Kept at its default value.
Bit 0	XMCLPEN	0x1	rw	XMC clock enable in Sleep mode 0: Disabled 1: Enabled

4.3.18 APB1 peripheral clock enable in low power mode register (CRM_APB1LPEN)

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

Bit	Name	Reset value	Type	Description
Bit 31	USART8LPEN	0x1	rw	USART8 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 30	USART7LPEN	0x1	rw	USART7 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 29	DACLPEN	0x1	rw	DAC interface clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 28	PWCLPEN	0x1	rw	Power interface clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 27	Reserved	0x0	resd	Kept at its default value.
Bit 26	CAN2LPEN	0x1	rw	CAN2 clock enable in Sleep mode) 0: Disabled 1: Enabled
Bit 25	CAN1LPEN	0x1	rw	CAN1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 24	Reserved	0x0	resd	Kept at its default value.
Bit 23	I2C3LPEN	0x1	rw	I^2C3 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 22	I2C2LPEN	0x1	rw	I^2C2 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 21	I2C1LPEN	0x1	rw	I^2C1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 20	USART5LPEN	0x1	rw	USART5 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 19	USART4LPEN	0x1	rw	USART4 clock enable in Sleep mode 0: Disabled 1: Enabled

Bit 18	USART3LPEN	0x1	rw	USART3 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 17	USART2LPEN	0x1	rw	USART2 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 16	Reserved	0x0	resd	Kept at its default value.
Bit 15	SPI3LPEN	0x1	rw	SPI3 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 14	SPI2LPEN	0x1	rw	SPI 2 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 13:12	Reserved	0x0	resd	Kept at its default value.
Bit 11	WWDTLPEN	0x1	rw	Window watchdog clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 10:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	TMR14LPEN	0x1	rw	Timer14 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 7	TMR13LPEN	0x1	rw	Timer13 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 6	TMR12LPEN	0x1	rw	Timer12 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 5	TMR7LPEN	0x1	rw	Timer 7 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 4	TMR6LPEN	0x1	rw	Timer 6 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 3	Reserved	0x0	resd	Kept at its default value.
Bit 2	TMR4LPEN	0x1	rw	Timer 4 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 1	TMR3LPEN	0x1	rw	Timer 3 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 0	TMR2LPEN	0x1	rw	Timer 2 clock enable in Sleep mode 0: Disabled 1: Enabled

4.3.19 APB2 peripheral clock enable in low power mode register (CRM_APB2LPEN)

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

Bit	Name	Reset value	Type	Description
Bit 31:30	Reserved	0x0	resd	Kept at its default value.
Bit 29	ACCLPEN	0x1	rw	ACC clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 28:19	Reserved	0x000	resd	Kept at its default value.
Bit 18	TMR11LPEN	0x1	rw	Timer11 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 17	TMR10LPEN	0x1	rw	Timer10 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 16	TMR9LPEN	0x1	rw	Timer9 clock enable in Sleep mode 0: Disabled 1: Enabled

Bit 15	Reserved	0x0	resd	Kept at its default value.
Bit 14	SCFGLPEN	0x1	rw	SCFG clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at its default value.
Bit 12	SPI1LPEN	0x1	rw	SPI1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 11:9	Reserved	0x0	resd	Kept at its default value.
Bit 8	ADCLPEN	0x1	rw	ADC interface clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 7:6	Reserved	0x0	resd	Kept at its default value.
Bit 5	USART6LPEN	0x1	rw	USART6 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 4	USART1LPEN	0x1	rw	USART1 clock enable in Sleep mode 0: Disabled 1: Enabled
Bit 3:1	Reserved	0x0	resd	Kept at its default value.
Bit 0	TMR1LPEN	0x1	rw	TMR1 timer clock enable in Sleep mode 0: Disabled 1: Enabled

4.3.20 Peripheral independent clock select register (CRM_PICLKS)

Bit	Name	Reset value	Type	Description
Bit 31:14	Reserved	0x0000	resd	Kept at its default value.
Bit 13:12	I2C1SEL	0x0	rw	I^2C1 clock source select 00: PCLK1 01: SCLK 10: HICK 11: Reserved
Bit 11:6	Reserved	0x00	resd	Kept at its default value.
Bit 5:4	USART3SEL	0x0	rw	USART3 clock source select 00: PCLK1 01: SCLK 10: HICK 11: LEXT
Bit 3:2	USART2SEL	0x0	rw	USART2 clock source select 00: PCLK1 01: SCLK 10: HICK 11: LEXT
Bit 1:0	USART1SEL	0x0	rw	USART1 clock source select 00: PCLK2 01: SCLK 10: HICK 11: LEXT

4.3.21 Battery powered domain control register (CRM_BPDC)

It is only reset by setting the battery powered domain 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.

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 (PWC_CTRL). These bits could be reset only by battery powered domain software 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: Does 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 source selection Once the ERTC clock source is selected, it cannot be changed until the BPDRST bit is set. 00: No clock 01: LEXT 10: LICK 11: Divided HEXT (with the ERTC_DIV bit in the CRM_CFG)
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	Low-speed external oscillator enable 0: Disabled 1: Enabled

4.3.22 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 and 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 reset flag Set by hardware. Cleared by writing to the RSTFC bit. 0: No NRST reset occurs 1: NRST 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

4.3.23 Additional register 1 (CRM_MISC1)

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

Bit	Name	Reset value	Type	Description
Bit 31:28	CLKOUTDIV2	0x0	rw	Clock output division2 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 27:20	Reserved	0x00	resd	Kept at its default value.
Bit 19:16	CLKOUT_SEL2	0xF	rw	Clock output sel2 0000: USB clock output 0001: ADC clock output 0010: Internal RC oscillator (HICK) output frequency divider 0011: LICK clock output 0100: LEXT clock output 0101~0111: Reserved 1000~1111: Reserved
Bit 15	PLLCLK_TO_ADC	0x0	rw	ADC clock source select 0: HCLK is selected as ADC clock source 1: PLLCLK is selected as ADC clock source Note: When the PLL is selected as ADC clock source, the SCLKSEL must select PLL/2.
Bit 14	HICK_TO_SCLK	0x0	rw	HICK as system clock frequency select When the HICK is selected as the clock source of 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 13	HICK_TO_USB	0x0	rw	USB 48 MHz clock source select 0: PLL or divided PLL 1: HICK or HICK/6 Note: Since USB must work at 48 MHz, HICKDIV=1 must be guaranteed to ensure that the HICK 48 MHz is selected as the clock source of USB 48 MHz.
Bit 12	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: HICKDIV must not be modified during PLL enable when HICK is used as PLL clock source.
Bit 11:8	Reserved	0x0	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 as 0x5A.

4.3.24 Additional register 2 (CRM_MISC2)

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 21:19	HEXT_TO_SCLK_DIV	0x0	rw	HEXT as system clock frequency division 000: HEXT 001: HEXT/2 010: HEXT/4 011: HEXT/8 100: HEXT/16 101: HEXT/32 Others: Reserved
Bit 18:16	HICK_TO_SCLK_DIV	0x0	rw	HICK as system clock frequency division 000: HICK 001: HICK/2 010: HICK/4 011: HICK/8 100: HICK/16 Others: Reserved
Bit 15:12	USBDIV	0x0	rw	USB division The PLL clock, after frequency division, is used as USB clock. 0000: PLL clock divided by 3 0001: PLL clock divided by 2 0010: PLL clock divided by 5 0011: PLL clock divided by 4 0100: PLL clock divided by 7 0101: PLL clock divided by 6 0110: PLL clock divided by 9 0111: PLL clock divided by 8 Others: Reserved
Bit 11:6	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 or when the AHB prescaler is changed, it is recommended to enable auto step-by-step system clock switch. Once it is enabled, the AHB bus is halted by hardware until 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 AHB resumes. 00: Disabled 01: Reserved 10: Reserved 11: Enabled. When AHBDIV or SCLKSEL is modified, the auto step-by-step system clock switch is activated automatically.
Bit 3:0	Reserved	0xD	resd	It is fixed to 0xD. Do not change.

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 20 KB bootloader and the user system data area. The bootloader uses USART1, USART2 or USB serial interface for ISP programming.

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

Table 5-1 Flash memory architecture (256 K)

Bank		Name	Address range
Main memory	Bank 1 256 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		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512B user system data	0x1FFF F800 – 0x1FFF F9FF

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

Table 5-2 Flash memory architecture (128 K)

Bank		Name	Address range
Main memory	Bank 1 128 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		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512B user system data	0x1FFF F800 – 0x1FFF F9FF

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

Table 5-3 Flash memory architecture (64 K)

Bank		Name	Address range
Main memory	Bank 1 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		20 KB bootloader	0x1FFF A400 – 0x1FFF F3FF
		512B user system data	0x1FFF F800 – 0x1FFF F9FF

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 byte corresponds to the contents in the system data area and the high byte represents 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 bytes 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 [1] and [26]bits in the FLASH_USD register) 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 bit (stored in the FLASH_USD[9:2])	
		Bit 7	Reserved
		Bit 6 (nSTDBY_WDT)	0: WDT stops counting while entering Standby mode 1: WDT does not stop counting while entering Standby mode
		Bit 5 (nDEPSLP_WDT)	0: WDT stops counting while entering Deepsleep mode 1: WDT does not stop counting while entering Deepsleep mode
		Bit 4 (nBOOT1)	nBOOT1: It defines boot mode together with BOOT0 pin. When BOOT0 = 1, 0: Boot from SRAM 1: Boot from boot memory
		Bit 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 (stored in the FLASH_USD[17:10])	
	[15:8]	**nData0[7:0]**: Inverse code of Data0[7:0]	
	[23:16]	**Data1[7:0]**: User data 1 (stored in the FLASH_USD[25:18])	
	[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 the main Flash memory (256 KB) and sector0~sector31 of the main Flash memory (128 KB/64 KB). Each bit takes care of 4 KB sectors. 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 (in the FLASH_EPPS[15:8]) This field is used to protect sector16~sector31 of the main Flash memory (256 KB) and sector32~sector63 of the main Flash memory (128 KB/64 KB). Each bit takes care of 4 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 (in the FLASH_EPPS[23:16]) This field is used to protect sector32~sector47 of the main Flash memory (256 KB) and sector64~sector95 of the main Flash memory (128 KB). Each bit takes care of 4 KB sectors. This bit is reserved and unused for the 64 KB main Flash memory. 0: Erase/write protection is enabled 1: Erase/write protection is disabled	

	[15:8]	**nEPP2[7:0]**: Inverse code of EPP2[7:0]
	[23:16]	**EPP3[7:0]**: Flash erase/write protection byte 3 (in the FLASH_EPPS[31:24]) Bit [6:0] is used to protect sector48~sector61 of the main Flash memory (256 KB) and sector96~sector123 of the main Flash memory (128 KB). Each bit takes care of 4 KB sectors. Bit [6:0] is reserved for the 64 KB main Flash memory. Bit [7] is used to protect sector62~sector127 of the main Flash memory (256 KB) and sector124~sector127 of the main Flash memory (128 KB). Bit [7] is also used in the main Flash memory (256 KB/128 KB/64 KB) 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 data 2
	[15:8]	**nData2[7:0]**: Inverse code of Data2[7:0]
	[23:16]	**Data3[7:0]**: User data 3
	[31:24]	**nData3[7:0]**: Inverse code of Data3[7:0]
0x1FFF_F814	[7:0]	**Data4[7:0]**: User data 4
	[15:8]	**nData4[7:0]**: Inverse code of Data4[7:0]
	[23:16]	**Data5[7:0]**: User data 5
	[31:24]	**nData5[7:0]**: Inverse code of Data5[7:0]
. .	. .	. .
0x1FFF_F9FC	[7:0]	**Data248[7:0]**: User data 248
	[15:8]	**nData248[7:0]**: Inverse code of Data248[7:0]
	[23:16]	**Data249[7:0]**: User data 249
	[31:24]	**nData249[7:0]**: Inverse code of Data249[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 sector erase and mass erase.

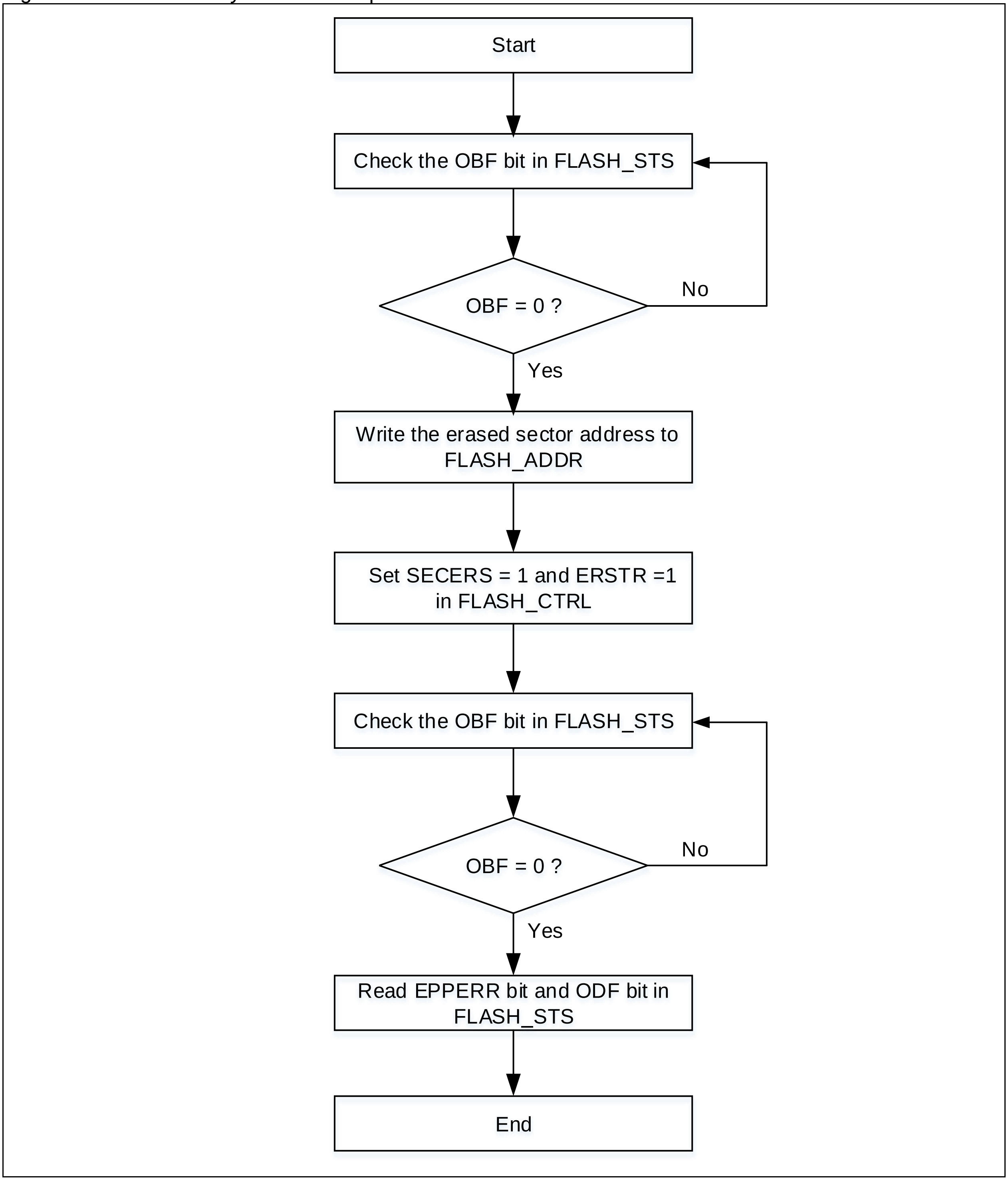
Sector erase

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

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress.
- Write the sector to be erased in the FLASH_ADDR register.
- Set the SECERS and ERSTR bits (set to 1) in the FLASH_CTRL register to enable sector erase.
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the EPPERR and ODF bits in the FLASH_STS register to verify the erased sectors.

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

Figure 5-1 Flash memory sector erase process



Mass erase

Mass erase function can erase all the 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 bits in the FLASH_CTRL register to enable mass erase.
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the EPPERR and ODF bits in the FLASH_STS register to verify the erase result.

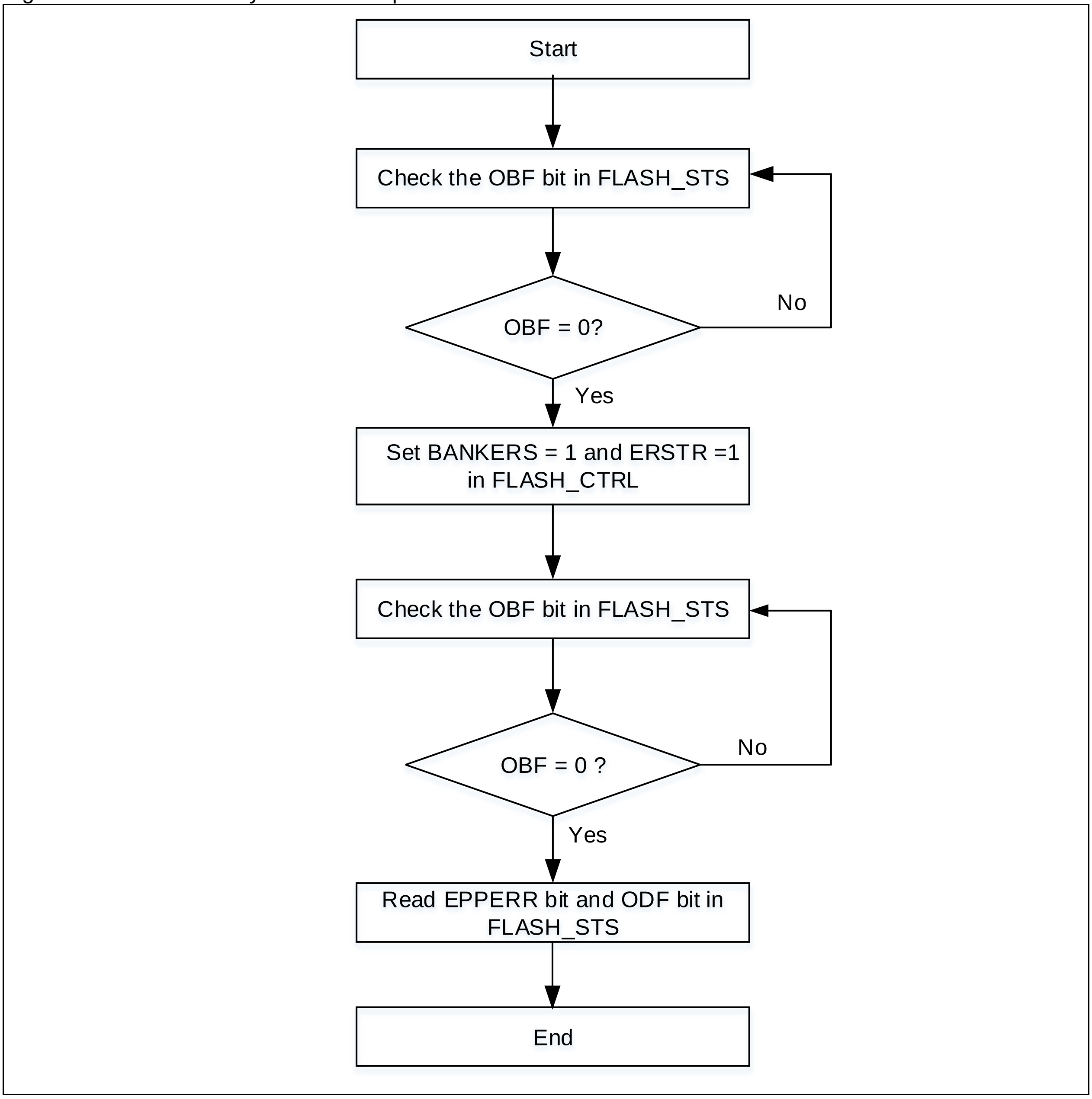
Note:

1) When the boot memory 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 becomes “0” in the FLASH_STS register. Read the EPPERR, PRGMERR and ODF bits in the FLASH_STS register to verify the programming result.

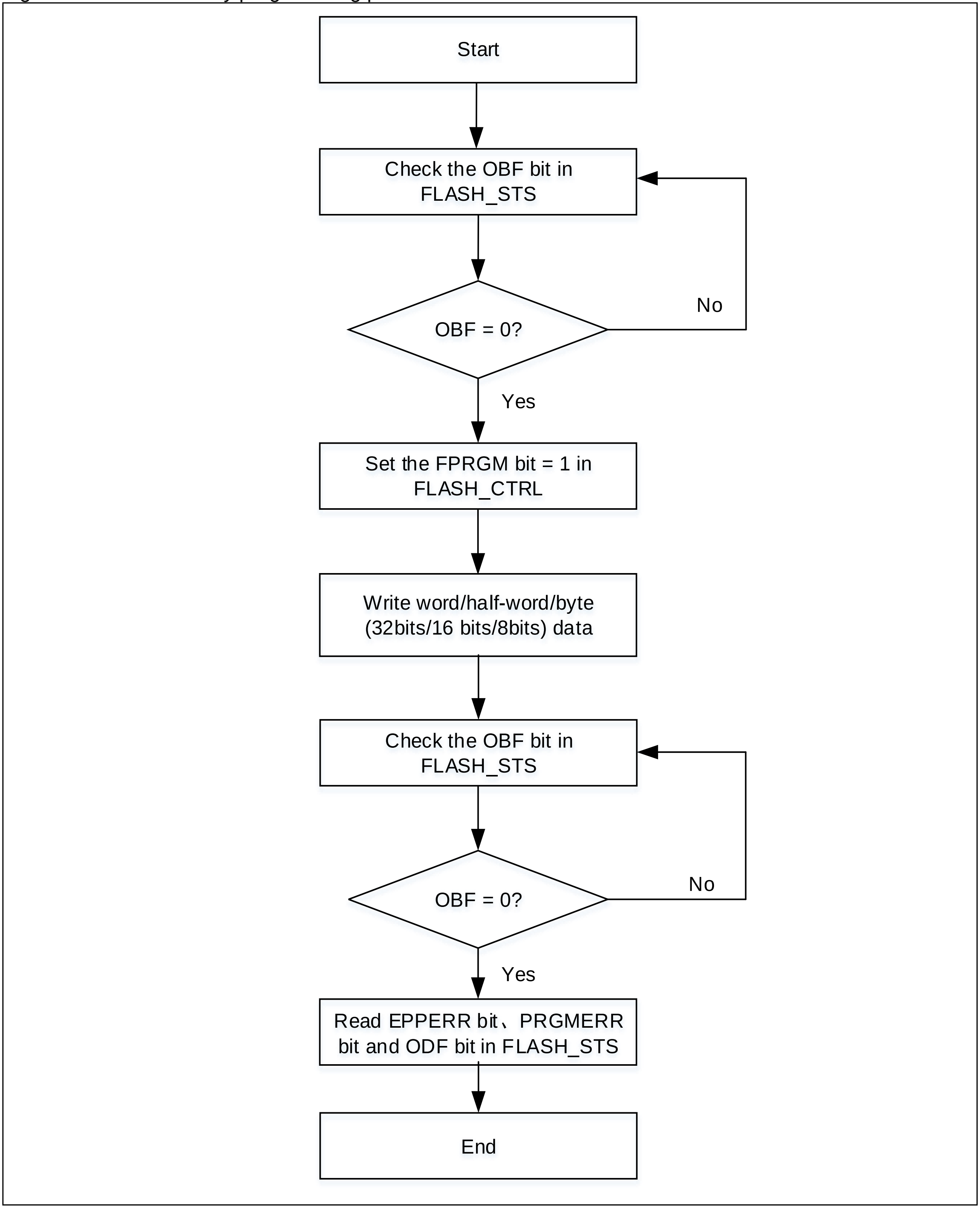
Note:

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 the 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

Boot memory 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 operation

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

When KEY1 (0x45670123) and KEY2 (0xCDEF89AB) are 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.

The following process is recommended:

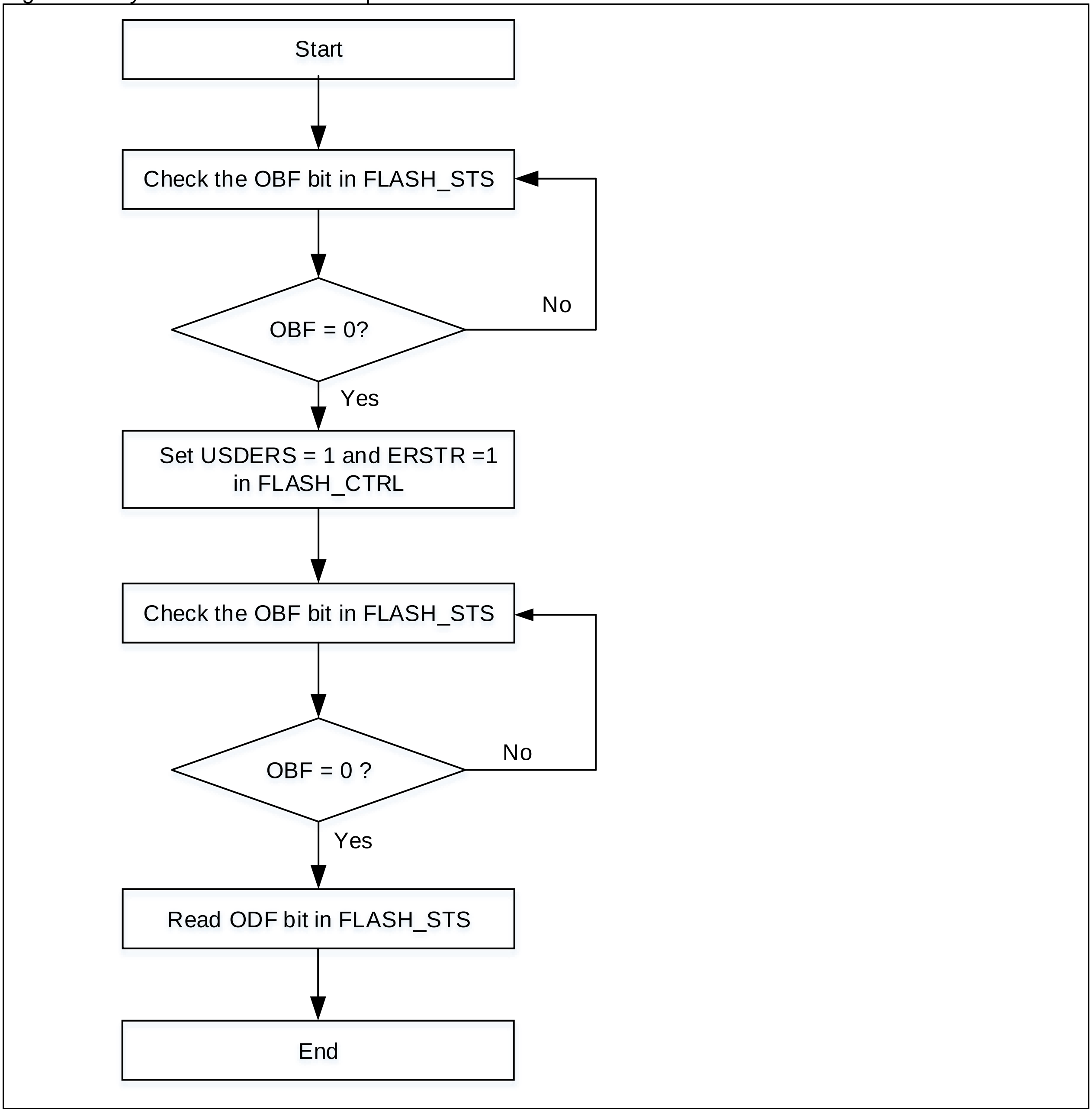
- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress.
- Set the USDERS and ERSTR bits (set to 1) in the FLASH_CTRL register to enable erase operation.
- Wait until the OBF bit becomes “0” in the FLASH_STS register. Read the ODF bit in the FLASH_STS register to verify the erase result.

Note:

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

2) 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:

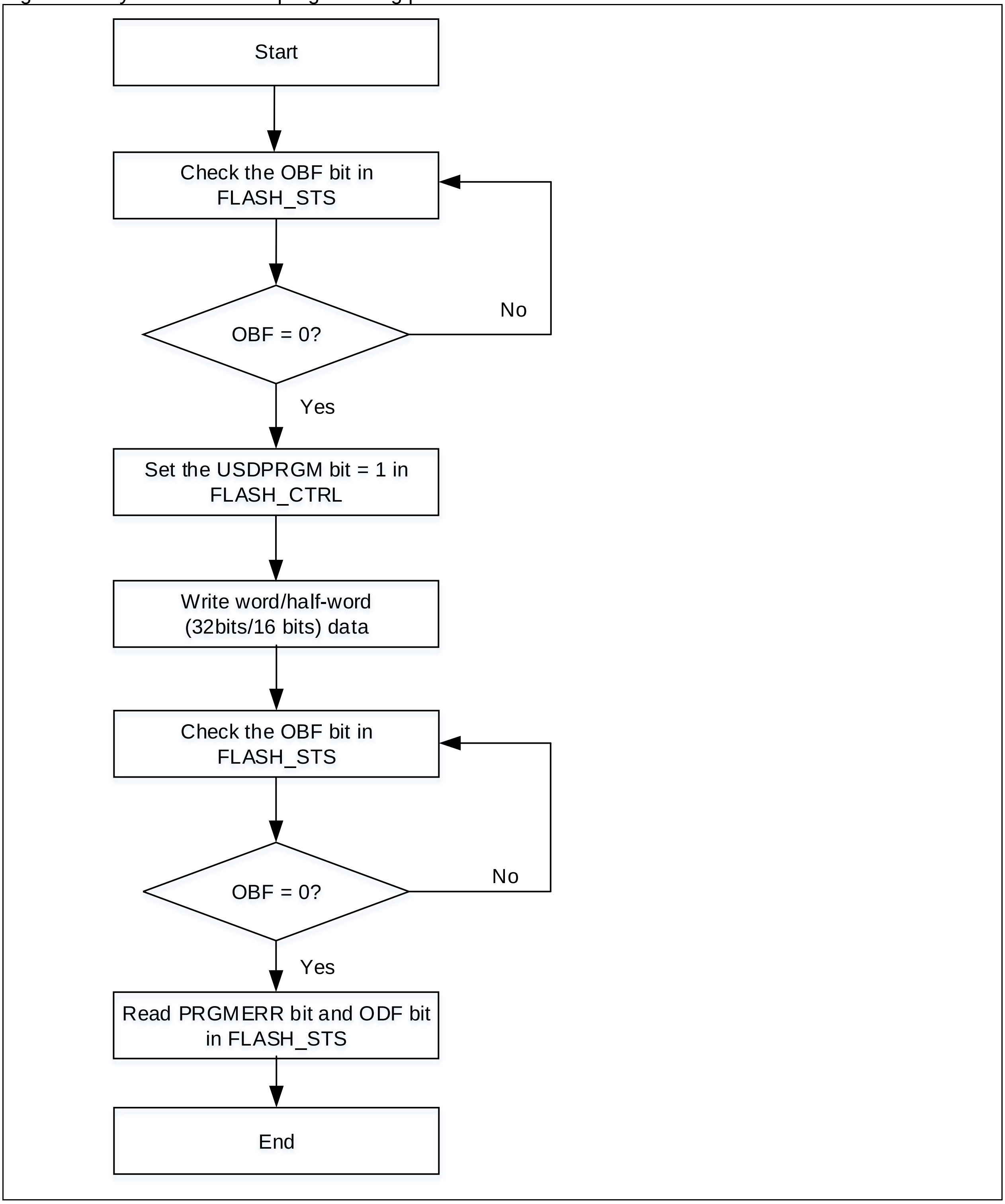
- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress.
- Set the USDPRGM bit in the FLASH_CTRL register, so that the programming instructions for the user system data area can be received.
- Write the data (word/half-word) to be programmed to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

Note:

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

2) 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 protection 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 low-level protection to unprotected state will trigger mass erase on the Flash memory and its extension area 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 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 Flash 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 5-5 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 or boot memory			Boot from main Flash memory		
		Read	Write	Erase	Read	Write	Erase
Main Flash memory	Low-level protection	Not allowed		Not allowed (1)(2)	Accessible		
	High-level protection	None (3)			Accessible		
User system data area	Low-level protection	Not allowed	Accessible		Accessible		
	High-level protection	None (3)			Accessible	Not allowed	

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

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

(3) When the high-level access protection is enabled, the system automatically boots from the main Flash.

5.5.2 Erase/program protection

Erase/program protection is performed on the basis of 4 KB. 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 when

- Erasing/programming the pages (in Flash memory and its extension area) where erase/program protection is enabled;
- Performing mass erase on the sectors and main Flash memory extension area where erase/program protection enabled;

- When the Flash access protection is enabled, the sector0~sector1 in the 256 KB main Flash memory and sector0~sector3 in the 128 KB/64 KB Flash memory will be protected against erase/program automatically;
- When the Flash access protection is enabled, the main Flash memory is protected against sector erase and programming operation when it is the main Flash memory and its extension area are in debug mode or when it boots from non-main Flash memory.

5.6 Read access

To increase system clock frequency, program the number of wait states to access the Flash memory through the WTCYC bit in the FLASH_PSR register.

The Flash read times can be decreased through the PFT_EN, PFT_EN2 and PFT_LAT_DIS bits in the FLASH_PSR register.

5.7 Special functions

5.7.1 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 written or deleted unless a correct code is keyed in. Security library includes instruction security library (cannot be read) and data security library (can be read).

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 the 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 code must be programmed by sector, with its start address aligned with the main memory address;

Only CPU instruction is allowed to read instruction security library;

In an attempt of writing or erasing the security library code, a warning message will be issued by EPPERR =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 to the security library setting register.

The steps to enable security library are as follows:

- Check the OBF bit in the FLASH_STS register to confirm that there is no other programming operation in progress.
- Write 0xA35F6D24 to the SLIB_UNLOCK register to unlock the security library.
- Check the SLIB_ULKF bit in the SLIB_MISC_STS register to verify that it is unlocked successfully;
- If the security library is set in the main Flash memory, set the sectors to be protected in the SLIB_SET_RANGE register (including the addresses of instruction/data security library). If the security library is set in the main Flash memory extension area, set the EM_SLIB_SET register.
- Wait until the OBF bit becomes “0”.
- Set the 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 word.

- Read the SLIB_STS0/STS1 register to verify the security library settings.

Note:

The main Flash memory and its extension area cannot be set as security library at the same time.

Security library should be enabled when the Flash access protection is not activated.

The steps to unlock security library are as follows:

- 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 its extension, as well as the security library setting block.

5.7.2 Boot memory used as Flash memory extension

There is only one chance for users to program the boot memory as the main Flash 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 boot memory;
- Write 0xA35F6D24 to the SLIB_UNLOCK register to unlock the current mode of boot memory;
- 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 the setting result.

Note: The above-mentioned process must be performed when the Flash memory access protection is disabled. Once enabled, the boot memory is forced to start from the main Flash memory.

5.7.3 CRC verify

The optional CRC check for security library code or user code is performed on a sector level.

CRC verify procedure is 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 CRC check in the FLASH_CRC_ADDR register;
- Program the code count (in terms of sectors) to be CRC check through the bit [15:0] in the FLASH_CRC_CTRL register;
- Enable CRC verify by setting the bit 16 in the FLASH_CRC_CTRL register;
- Wait until the OBF bit becomes "0";
- Read the FLASH_CRC_CHKR register to verify the result.

Note:

The values of the FLASH_CRC_ADDR register must be aligned with the start address of the sector.

CRC verify must not cross the main Flash memory and its extension area.

5.8 FLASH memory registers

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

Table 5-6 Flash memory register map and reset value

Register	Offset	Reset value
FLASH_PSR	0x00	0x0000 01F0
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	0x00FF 0000
SLIB_STS1	0x78	0xFFFF FFFF
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

5.8.1 Flash performance select register (FLASH_PSR)

Bit	Abbr.	Reset value	Type	Description
Bit 31:9	Reserved	0x000000	resd	Kept at its default value.
Bit 8	PFT_LAT_DIS	1	rw	Prefetch latency disable 0: Prefetch buffer latency enabled, one-wait state for buffer access 1: Prefetch buffer latency disabled, 0-wait state for buffer access It is recommended to set this bit to 1 and do not change.
Bit 7	PFT_ENF2	1	ro	Prefetch enabled flag2 This bit is set to enable prefetch buffer 2.
Bit 6	PFT_EN2	1	rw	Prefetch enable2 0: Prefetch buffer 2 is disabled 1: Prefetch buffer 2 is enabled It is recommended to set this bit to 1 and do not change.
Bit 5	PFT_ENF	1	ro	Prefetch enabled flag This bit is set to enable prefetch buffer.
Bit 4	PFT_EN	1	rw	Prefetch enable 0: Prefetch buffer is disabled 1: Prefetch buffer is enabled
Bit 3	Reserved	0	rw	Kept at 0.
Bit 2:0	WTCYC	0x0	rw	Wait cycle The wait states depend on the size of the system clock, and they are in terms of system clocks. 000: Zero wait state, 0 MHz<system clock≤32 MHz 001: One wait state, 32 MHz<system clock≤64 MHz 010: Two wait states, 64 MHz<system clock≤96 MHz 011: Three wait states, 96 MHz<system clock≤128 MHz 100: Four wait states, 128 MHz<system clock≤150 MHz

5.8.2 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 the Flash memory and its extension area.

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

5.8.3 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.

5.8.4 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	rwc1	Operation done flag This bit is set by hardware when Flash memory operations (program/erase) are completed. It is cleared by writing "1".
Bit 4	EPPERR	0	rwc1	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	rwc1	Program error When the programming address is not in erase state, 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 operations are in progress. It is cleared when operations are completed.

5.8.5 Flash control register (FLASH_CTRL)

Bit	Abbr.	Reset value	Type	Description
Bit 31:13	Reserved	0x0000	resd	Kept at its default value.
Bit 12	ODFIE	0	rw	Operation done flag interrupt enable 0: Interrupt is disabled 1: Interrupt is enabled
Bit 11,8,3	Reserved	0	resd	Kept at its default value.
Bit 10	ERRIE	0	rw	Error interrupt enable This bit enables EPPERR or PRGMERR 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	Erasing start An erase operation is triggered when this bit is set by software. 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 operation.

Bit 4	USDPRGM	0	rw	User system data program It indicates the user system data program.
Bit 2	BANKERS	0	rw	Bank erase It indicates bank erase operation.
Bit 1	SECERS	0	rw	Sector erase It indicates sector erase operation.
Bit 0	FPRGM	0	rw	Flash program It indicates Flash programming operation.

5.8.6 Flash address register (FLASH_ADDR)

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

5.8.7 User system data register (FLASH_USD)

Bit	Abbr.	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-0xCC and non-0xA5 10: Reserved 11: High-level Flash access protection enabled, 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 It includes the system setting bytes in the loaded user system data area. Bit 9: Unused Bit 8: nSTDBY_WDT Bit 7: nDEPSLP_WDT Bit 6: nBOOT1 Bit 5: Unused Bit 4: nSTDBY_RST Bit 3: nDEPSLP_RST Bit 2: nWDT_ATO_EN
Bit 1	FAP	0	ro	Flash access protection
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.

5.8.8 Erase/program protection status register (FLASH_EPPS)

Bit	Abbr.	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.

5.8.9 Flash security library status register 0 (SLIB_STS0)

For Flash security library only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:24	Reserved	0x00	resd	Kept at its default value.
Bit 23:16	EM_SLIB_INST_SS	0xFF	ro	Extension memory sLib instruction start sector 00000000: Sector 0 00000001: Sector 1 00000010: Sector 2 … 00001001: Sector 9 (the last sector of 256 KB main Flash memory) … 00010011: Sector 19 (the last sector of 64 KB/128 KB main Flash memory) 11111111: None Others: Invalid
Bit 15:4	Reserved	0x000	resd	Kept at its default value.
Bit 3	SLIB_ENF	0	ro	sLib enabled 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 enabled flag When this bit is set, it indicates that the boot memory 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 boot memory can be used as main Flash extension area to store user application code; otherwise, it is only used for system boot code.

5.8.10 Flash security library status register 1 (SLIB_STS1)

For Flash security library only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:22	SLIB_ES	0x3FF	ro	sLib end sector 0000000000: Sector 0 0000000001: Sector 1 0000000010: Sector 2 … 0000111111: Sector 63 (the last sector of 64 KB main Flash memory) … 0001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory)
Bit 21:11	SLIB_INST_SS	0x7FF	ro	sLib instruction start sector 00000000000: Sector 0 00000000001: Sector 1 00000000010: Sector 2 … 00000111111: Sector 63 (the last sector of 64 KB main Flash memory) … 00001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory) 11111111111: None
Bit 10:0	SLIB_SS	0x7FF	ro	sLib start sector 00000000000: Sector 0 00000000001: Sector 1 00000000010: Sector 2 … 00000111111: Sector 63 (the last sector of 64 KB main Flash memory)

				… 00001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory)

5.8.11 Security library password clear register (SLIB_PWD_CLR)

For Flash security library only.

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

5.8.12 Security library additional status register (SLIB_MISC_STS)

For Flash security library only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:3	Reserved	0x0000000	resd	Kept at its default value.
Bit 2	SLIB_ULKF	0	ro	sLib unlock flag When this bit is set, it indicates that sLib-related setting registers can be configured.
Bit 1	SLIB_PWD_OK	0	ro	sLib password ok This bit is set by hardware when the password is correct.
Bit 0	SLIB_PWD_ERR	0	ro	sLib 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.

5.8.13 Flash CRC address register (FLASH_CRC_ADDR)

For the main Flash memory and its extension area.

Bit	Abbr.	Reset value	Type	Description
Bit 31:0	CRC_ADDR	0x0000 0000	wo	CRC address The register is used to select the start address of a sector to be CRC checked. Note: The CRC address must align with the sector start address.

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

5.8.14 Flash CRC control register (FLASH_CRC_CTRL)

For the main Flash memory and its extension area.

Bit	Abbr.	Reset value	Type	Description
Bit 31:17	Reserved	0x0000	resd	Kept at its default value.
Bit 16	CRC_STRT	0	wo	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 sector number This bit defines the sectors to be CRC checked.

5.8.15 Flash CRC check result register (FLASH_CRC_CHKR)

For the main Flash memory and its extension area.

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

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

5.8.16 Security library password setting register (SLIB_SET_PWD)

For Flash security library password setting only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:0	SLIB_PSET_VAL	0x0000 0000	wo	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.

5.8.17 Security library address setting register (SLIB_SET_RANGE)

For Flash security library address setting only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:22	SLIB_ES_SET	0x000	wo	sLib end sector setting These bits are used to set the security library end sector. 0000000000: Sector 0 0000000001: Sector 1 0000000010: Sector 2 … 0000111111: Sector 63 (the last sector of 64 KB main Flash memory) … 0001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory)
Bit 21:11	SLIB_ISS_SET	0x000	wo	sLib instruction start sector setting These bits are used to set the security library instruction start sector. 00000000000: Sector 0 00000000001: Sector 1 00000000010: Sector 2 … 00000111111: Sector 63 (the last sector of 64 KB main Flash memory) … 00001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory) 11111111111: No security library instruction area
Bit 10:0	SLIB_SS_SET	0x000	wo	sLib start sector setting These bits are used to set the security library start sector. 00000000000: Sector 0 00000000001: Sector 1 00000000010: Sector 2 … 00000111111: Sector 63 (the last sector of 64 KB main Flash memory) … 00001111111: Sector 127 (the last sector of 256KB/128KB main Flash memory)

Note:

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

This register can be written only when security library is unlocked.

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

5.8.18 Flash extension memory security library setting register (EM_SLIB_SET)

For Flash extension area only.

Bit	Abbr.	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 sector setting It is used to set the security library instruction area start sector. 00000000: Sector 0 00000001: Sector 1 00000010: Sector 2 … 00001001: Sector 9 (the last sector of 256 KB main Flash memory) … 00010011: Sector 19 (the last sector of 64KB/128KB main Flash memory) 11111111: No sLib instruction area Others: Invalid Note: When it is set to 0xFF, it indicates that the extension area from sector 0 to sector 3 is the security library, read-only.
Bit 15:0	EM_SLIB_SET	0x000	wo	Extension memory sLib setting Writing 0x5AA5 can configure the Flash extension area to store the sLib code.

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

5.8.19 Boot memory mode setting register (BTM_MODE_SET)

For boot memory only.

Bit	Abbr.	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: Boot memory serves as a system area that stores system boot code Others: Boot memory serves a Flash extension area that stores application code Note: This register is set when the Flash access protection is disabled.

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

5.8.20 Security library unlock register (SLIB_UNLOCK)

For security library register unlock only.

Bit	Abbr.	Reset value	Type	Description
Bit 31:0	SLIB_UKVAL	0x0000 0000	wo	sLib 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 GPIOs and IOMUX

6.1 Introduction

AT32F423 series supports up to 87 bidirectional I/O pins, namely PA0-PA15, PB0-PB15, PC0-PC15, PD0-PD15, PE0-PE15, PF0-PF2 PF6 PF8-PF10. Each of these pins features 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 with individual weak pull-up/pull-down capability;
- Each pin's output drive capability is configurable by software;
- Each pin can be configured as external interrupt input;
- Each pin can be locked.

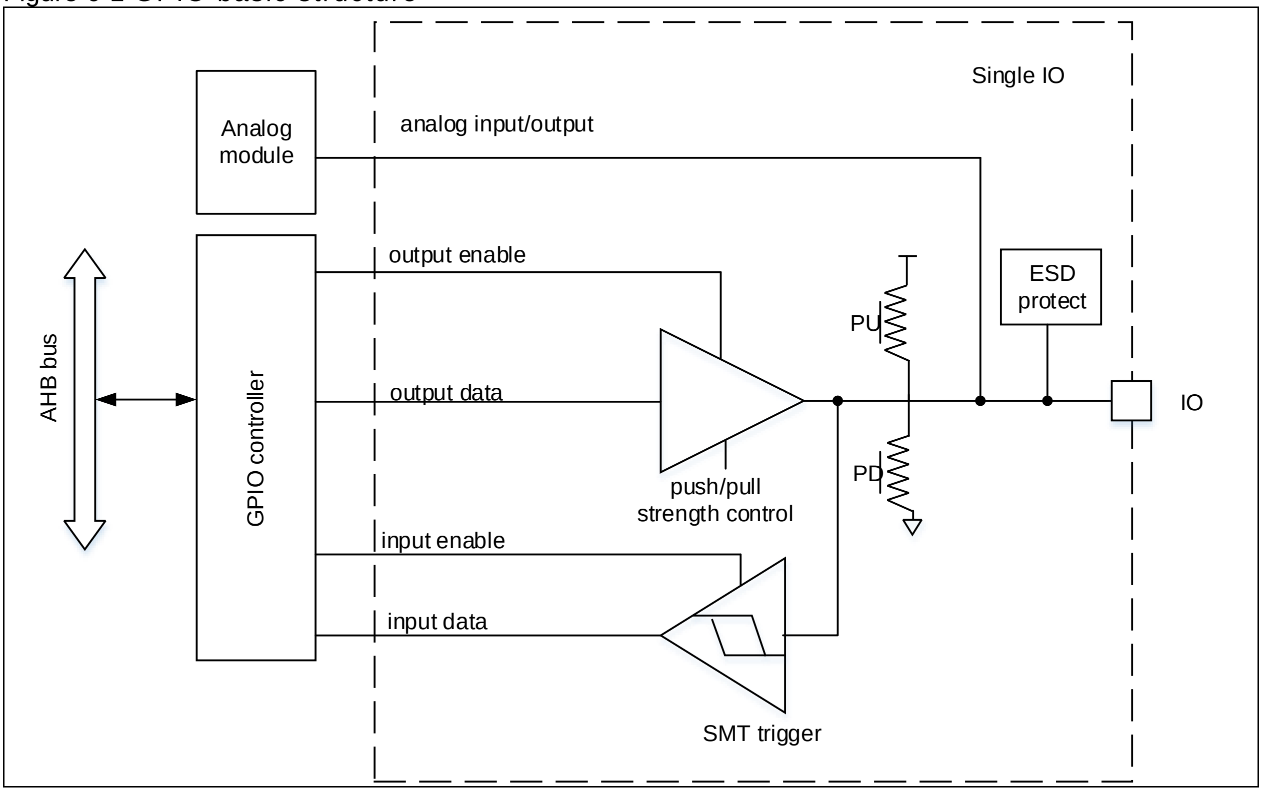
6.2 Function overview

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 half-words or bytes.

Figure 6-1 GPIO basic structure



Note: The corresponding GPIO function of PC13 and related RTC functions cannot be directly used during the initial power-on. For details, refer to GPIO section in ES0010_AT32F423_Errata_Sheet.

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 configurations are as follows:

- PA15/JTDI, PA13/JTMS and PB4/JNTRST in multiplexed pull-up mode;
- PA14/JTCK in multiplexed pull-down mode;
- PB3/TDO in multiplexed mode without pull-up/pull-down capability.

6.2.3 General-purpose input configuration

Mode	IOMC	PUPD
Floating input	00	00
Pull-down input		10
Pull-up input		01

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 are 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 intereferences from unused pins in a complex environment.

6.2.4 Analog input/output configuration

Mode	IOMC	PUPD
Analog input/output	11	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	IOMC	OM	HDRV	ODRV[1:0]	PUPD
Push-pull without pull-up/pull-down	01	0	000: Output mode, normal sourcing/sinking strength 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		00 or 11
Push-pull with pull-up	01	0			01
Push-pull with pull-down	01	0			10
Open-drain without pull-up/pull-down	01	1	000: Output mode, normal sourcing/sinking strength 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		00 or 11
Open-drain with pull-up	01	1			01
Open-drain with pull-down	01	1			10

When I/O port is configured as output:

- Schmitt-trigger input is enabled;
- Output through output register;
- In open-drain mode, forced output 0, and use pull-up resistor to output 1;
- In push-pull mode, output register is used to output 0/1;
- GPIO set/clear register is used to set/clear the corresponding GPIO output data registers.

Note: If both IOCB and IOSB bits are set in the GPIO set/clear register, the IOSB takes priority.

6.2.6 I/O 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.

6.2.7 IOMUX structure

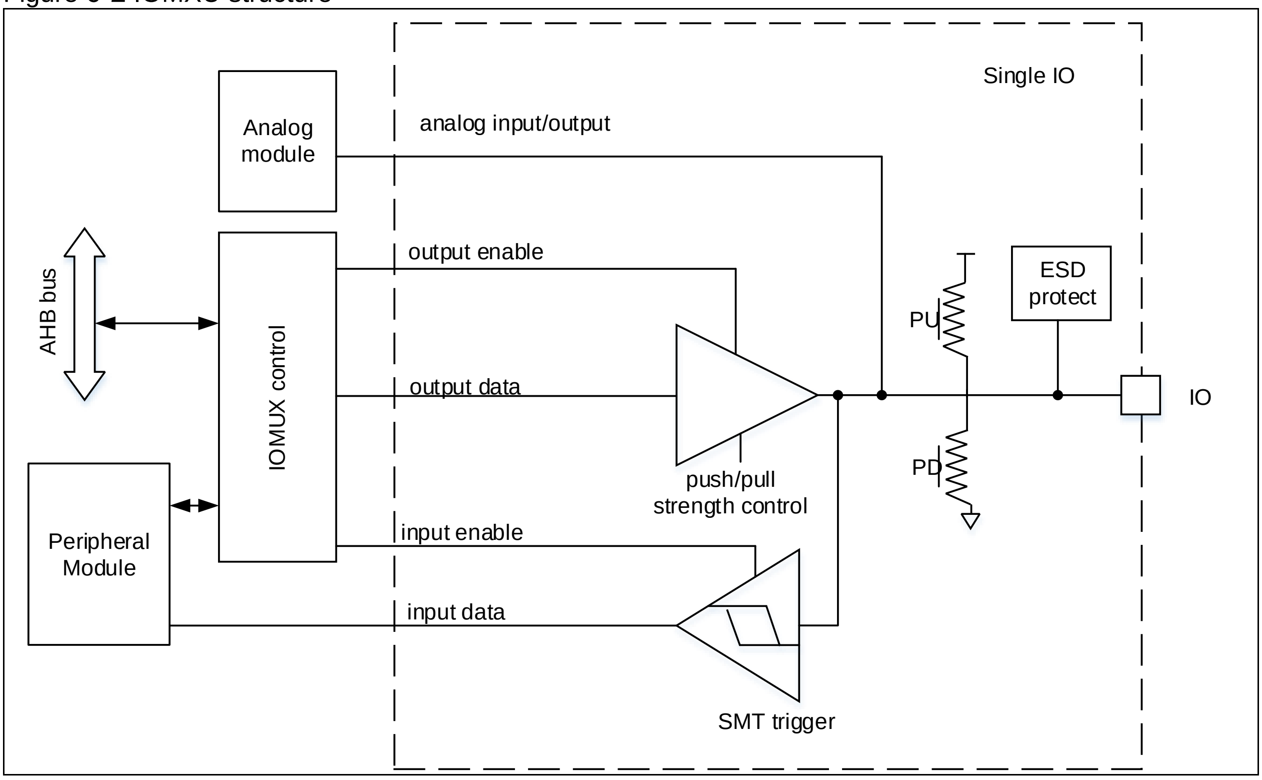
Several peripheral functions can be mapped on each IO pin. Peripheral input/output corresponding to an I/O pin is selected through IOMUX input/output table. Each I/O pin has up to 16 IOMUX mapping options for flexible selection, configured through the GPIOx_MUXL (for pin 0 to 7) and GPIOx_MUXH (for pin 8 to 15) registers.

Each I/O pin is connected to only one peripheral's pin by setting the GPIOx_MUXL or GPIOx_MUXH register so that there can be no conflict between peripherals sharing the same pin.

To enable multiplexed function output, the port is configured as multiplexed function mode push-pull or open-drain mode by setting GPIOx_CFGR or GPIOx_OMODE register. In this case, the pins are disconnected from GPIO controller, and controlled by IOMUX controller, instead.

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

Figure 6-2 IOMXU structure



6.2.8 Multiplexed function pull-up/down configuration

Mode	IOMC	PUPD
Multiplexed function floating	10	00
Multiplexed function pull-down		10
Multiplexed function pull-up		01

When an I/O port is configured as input:

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

6.2.9 IOMUX input/output

The multiplexed function of each IO port line is configured through the GPIOx_MUXL (from pin 0 to pin 7) or GPIOx_MUXH (from pin 8 to pin 15) register.

Table 6-1 Port A multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PA0		TMR2_CH1 TMR2_EXT		TMR9_CH2C	I2C2_SCL		USART2_RX	USART2_CTS
PA1		TMR2_CH2		TMR9_CH1C	I2C2_SDA	I2C1_SMBA	SPI3_CS/I2S3_WS	USART2_RTS_DE
PA2		TMR2_CH3		TMR9_CH1				USART2_TX
PA3		TMR2_CH4		TMR9_CH2		I2S2_MCK		USART2_RX
PA4					I2C1_SCL	SPI1_CS/I2S1_WS	SPI3_CS/I2S3_WS	USART2_CK
PA5		TMR2_CH1 TMR2_EXT				SPI1_SCK/I2S1_CK	USART3_CK	USART3_RX
PA6		TMR1_BRK	TMR3_CH1			SPI1_MISO/I2S1_MCK	I2S2_MCK	USART3_CTS
PA7		TMR1_CH1C	TMR3_CH2		I2C3_SCL	SPI1_MOSI/I2S1_SD		USART3_TX
PA8	CLKOUT	TMR1_CH1		TMR9_BRK	I2C3_SCL			USART1_CK
PA9	CLKOUT	TMR1_CH2			I2C3_SMBA	SPI2_SCK/I2S2_CK		USART1_TX
PA10	ERTC_REFIN	TMR1_CH3				SPI2_MOSI/I2S2_SD		USART1_RX
PA11		TMR1_CH4			I2C2_SCL	SPI2_CS/I2S2_WS	I2C1_SMBA	USART1_CTS
PA12		TMR1_EXT			I2C2_SDA	SPI2_MISO/I2S2_MCK		USART1_RTS_DE
PA13	JTMS SWDIO	IR_OUT			I2C1_SDA	I2S_SDEXT	SPI3_MISO/I2S3_MCK	
PA14	JTCK SWCLK				I2C1_SMBA		SPI3_MOSI/I2S3_SD	
PA15	JTDI	TMR2_CH1 TMR2_EXT				SPI1_CS/I2S1_WS	SPI3_CS/I2S3_WS	USART1_TX

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PA0	USART4_TX							EVENTOUT
PA1	USART4_RX							EVENTOUT
PA2		CAN2_RX			XMC_D4			EVENTOUT
PA3		CAN2_TX			XMC_D5			EVENTOUT
PA4	USART6_TX	TMR14_CH1	OTGFS_OE		XMC_D6			EVENTOUT
PA5	USART6_RX	TMR13_CH1C			XMC_D7			EVENTOUT
PA6	USART3_RX	TMR13_CH1						EVENTOUT
PA7		TMR14_CH1						EVENTOUT
PA8	USART2_TX	USART7_RX	OTGFS_SOF					EVENTOUT
PA9	I2C1_SCL	TMR14_BRK	OTGFS_VBUS					EVENTOUT
PA10	I2C1_SDA		OTGFS_ID					EVENTOUT
PA11	USART6_TX	CAN1_RX						EVENTOUT
PA12	USART6_RX	CAN1_TX						EVENTOUT
PA13			OTGFS_OE					EVENTOUT
PA14	USART2_TX							EVENTOUT
PA15	USART2_RX	USART7_TX	USART4_RTS_DE		XMC_NE2			EVENTOUT

Table 6-2 Port B multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PB0		TMR1_CH2C	TMR3_CH3			SPI1_MISO / I2S1_MCK	SPI3_MOSI/I2S3_SD	USART2_RX
PB1		TMR1_CH3C	TMR3_CH4			SPI1_MOSI / I2S1_SD	SPI2_SCK/I2S2_CK	USART2_CK
PB2		TMR2_CH4	TMR3_EXT		I2C3_SMBA		SPI3_MOSI/I2S3_SD	
PB3	JTDO SWO	TMR2_CH2			I2C2_SDA	SPI1_SCK/I2S1_CK	SPI3_SCK/I2S3_CK	USART1_RX
PB4	JNTRST		TMR3_CH1	TMR11_BRK	I2C3_SDA	SPI1_MISO/I2S1_MCK	SPI3_MISO/I2S3_MCK	USART1_CTS
PB5			TMR3_CH2	TMR10_BRK	I2C3_SMBA	SPI1_MOSI/I2S1_SD	SPI3_MOSI/I2S3_SD	USART1_CK
PB6			TMR4_CH1	TMR10_CH1C	I2C1_SCL	I2S1_MCK	SPI3_CS / I2S3_WS	USART1_TX
PB7			TMR4_CH2	TMR11_CH1C	I2C1_SDA		SPI3_SCK / I2S3_CK	USART1_RX
PB8		TMR2_CH1 TMR2_EXT	TMR4_CH3	TMR10_CH1	I2C1_SCL		SPI3_MISO / I2S3_MCK	USART1_TX
PB9	IR_OUT	TMR2_CH2	TMR4_CH4	TMR11_CH1	I2C1_SDA	SPI2_CS/I2S2_WS	SPI3_MOSI / I2S3_SD	I2C2_SDA
PB10		TMR2_CH3			I2C2_SCL	SPI2_SCK/I2S2_CK	I2S3_MCK	USART3_TX
PB11		TMR2_CH4			I2C2_SDA			USART3_RX
PB12		TMR1_BRK		TMR12_BRK	I2C2_SMBA	SPI2_CS/I2S2_WS	SPI3_SCK/I2S3_CK	
PB13	CLKOUT	TMR1_CH1C		TMR12_CH1C	I2C3_SMBA	SPI2_SCK/I2S2_CK		I2C3_SCL
PB14		TMR1_CH2C			I2C3_SDA	SPI2_MISO/I2S2_MCK	I2S_SDEXT	USART3_RTS_DE
PB15	ERTC_REFIN	TMR1_CH3C		TMR12_CH1C	I2C3_SCL	SPI2_MOSI/I2S2_SD		

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PB0	USART3_CK							EVENTOUT
PB1	USART3_RTS_DE	TMR14_CH1						EVENTOUT
PB2		TMR14_CH1C						EVENTOUT
PB3	USART1_RTS_DE	USART7_RX	USART5_TX					EVENTOUT
PB4	I2S_SDEXT	USART7_TX	USART5_RX					EVENTOUT
PB5	USART5_RX	CAN2_RX	USART5_RTS_DE					EVENTOUT
PB6	USART5_TX	CAN2_TX	USART4_CK					EVENTOUT
PB7	USART4_CTS				XMC_NADV			EVENTOUT
PB8	USART5_RX	CAN1_RX						EVENTOUT
PB9	USART5_TX	CAN1_TX	I2S1_MCK					EVENTOUT
PB10					XMC_NOE			EVENTOUT
PB11		TMR13_BRK						EVENTOUT
PB12	USART3_CK	CAN2_RX			XMC_D13			EVENTOUT
PB13	USART3_CTS	CAN2_TX						EVENTOUT
PB14		TMR12_CH1			XMC_D0			EVENTOUT
PB15		TMR12_CH2						EVENTOUT

Table 6-3 Port C multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PC0					I2C3_SCL			I2C1_SCL
PC1					I2C3_SDA	SPI3_MOSI/I2S3_SD	SPI2_MOSI/I2S2_SD	I2C1_SDA
PC2						SPI2_MISO/I2S2_MCK	I2S_SDEXT	
PC3						SPI2_MOSI/I2S2_SD		
PC4				TMR9_CH1		I2S1_MCK		USART3_TX
PC5				TMR9_CH2	I2C1_SMBA			USART3_RX
PC6		TMR1_CH1	TMR3_CH1		I2C1_SCL	I2S2_MCK		
PC7		TMR1_CH2	TMR3_CH2		I2C1_SDA	SPI2_SCK/I2S2_CK	I2S3_MCK	
PC8		TMR1_CH3	TMR3_CH3					USART8_TX
PC9	CLKOUT	TMR1_CH4	TMR3_CH4		I2C3_SDA			USART8_RX
PC10							SPI3_SCK/I2S3_CK	USART3_TX
PC11						I2S_SDEXT	SPI3_MISO/I2S3_MCK	USART3_RX
PC12				TMR11_CH1	I2C2_SDA		SPI3_MOSI/I2S3_SD	USART3_CK
PC13								
PC14								
PC15								

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PC0	USART6_TX	USART7_TX						EVENTOUT
PC1	USART6_RX	USART7_RX						EVENTOUT
PC2	USART8_TX				XMC_NWE			EVENTOUT
PC3	USART8_RX				XMC_A0			EVENTOUT
PC4		TMR13_CH1			XMC_NE4			EVENTOUT
PC5		TMR13_CH1C			XMC_NOE			EVENTOUT
PC6	USART6_TX	USART7_TX			XMC_D1			EVENTOUT
PC7	USART6_RX	USART7_RX			XMC_NADV			EVENTOUT
PC8	USART6_CK							EVENTOUT
PC9	I2C1_SDA		OTGFS_OE					EVENTOUT
PC10	USART4_TX							EVENTOUT
PC11	USART4_RX				XMC_D2			EVENTOUT
PC12	USART4_CK		USART5_TX		XMC_D3			EVENTOUT
PC13								EVENTOUT
PC14								EVENTOUT
PC15								EVENTOUT

Table 6-4 Port D multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PD0							SPI3_MOSI/I2S3_SD	SPI2_CS/I2S2_WS
PD1							SPI2_SCK/I2S2_CK	SPI2_CS/I2S2_WS
PD2			TMR3_EXT					USART3_RTS_DE
PD3						SPI2_SCK/I2S2_CK	SPI2_MISO/I2S2_MCK	USART2_CTS
PD4							SPI2_MOSI/I2S2_SD	USART2_RTS_DE
PD5								USART2_TX
PD6						SPI3_MOSI/I2S3_SD		USART2_RX
PD7								USART2_CK
PD8								USART3_TX
PD9								USART3_RX
PD10								USART3_CK
PD11					I2C2_SMBA			USART3_CTS
PD12			TMR4_CH1		I2C2_SCL			USART3_RTS_DE
PD13			TMR4_CH2		I2C2_SDA			
PD14			TMR4_CH3		I2C3_SCL			
PD15			TMR4_CH4		I2C3_SDA			

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PD0	USART4_RX	CAN1_RX			XMC_D2			EVENTOUT
PD1	USART4_TX	CAN1_TX			XMC_D3			EVENTOUT
PD2	USART5_RX				XMC_NWE			EVENTOUT
PD3					XMC_CLK			EVENTOUT
PD4					XMC_NOE			EVENTOUT
PD5					XMC_NWE			EVENTOUT
PD6					XMC_NWAIT			EVENTOUT
PD7					XMC_NE1			EVENTOUT
PD8		TMR12_CH2C			XMC_D13			EVENTOUT
PD9					XMC_D14			EVENTOUT
PD10	USART4_TX				XMC_D15			EVENTOUT
PD11					XMC_A16			EVENTOUT
PD12	USART8_CK_RTS_DE				XMC_A17			EVENTOUT
PD13	USART8_TX				XMC_A18			EVENTOUT
PD14	USART8_RX				XMC_D0			EVENTOUT
PD15		USART7_CK_RTS_DE			XMC_D1			EVENTOUT

Table 6-5 Port E multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PE0			TMR4_EXT					
PE1		TMR1_CH2C						
PE2			TMR3_EXT	TMR9_BRK				
PE3			TMR3_CH1	TMR9_CH2C				
PE4			TMR3_CH2	TMR9_CH1C				
PE5			TMR3_CH3	TMR9_CH1				
PE6			TMR3_CH4	TMR9_CH2				
PE7		TMR1_EXT						
PE8		TMR1_CH1C						
PE9		TMR1_CH1						
PE10		TMR1_CH2C						
PE11		TMR1_CH2						
PE12		TMR1_CH3C				SPI1_CS/I2S1_WS		
PE13		TMR1_CH3				SPI1_SCK/I2S1_CK		
PE14		TMR1_CH4				SPI1_MISO/I2S1_MCK		
PE15		TMR1_BRK				SPI1_MOSI/I2S1_SD		

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PE0	USART8_RX	TMR13_CH1			XMC_LB			EVENTOUT
PE1	USART8_TX	TMR14_CH1			XMC_UB			EVENTOUT
PE2		TMR14_CH1C			XMC_A23			EVENTOUT
PE3		TMR14_BRK			XMC_A19			EVENTOUT
PE4					XMC_A20			EVENTOUT
PE5					XMC_A21			EVENTOUT
PE6					XMC_A22			EVENTOUT
PE7	USART5_CK	USART7_RX			XMC_D4			EVENTOUT
PE8	USART4_TX	USART7_TX			XMC_D5			EVENTOUT
PE9	USART4_RX				XMC_D6			EVENTOUT
PE10	USART5_TX				XMC_D7			EVENTOUT
PE11	USART5_RX				XMC_D8			EVENTOUT
PE12					XMC_D9			EVENTOUT
PE13					XMC_D10			EVENTOUT
PE14					XMC_D11			EVENTOUT
PE15					XMC_D12			EVENTOUT

Table 6-6 Port F multiplexed function configuration with GPIOA_MUX* register

Pin	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PF0		TMR1_CH1			I2C1_SDA			
PF1		TMR1_CH2C			I2C1_SCL	SPI2_CS / I2S2_WS		
PF2						SPI2_SCK/I2S2_CK		
PF6		TMR2_CH1			I2C2_SCL			
PF8		TMR2_CH2			I2C2_SDA			
PF9			TMR4_CH1					
PF10			TMR4_CH2					

Pin	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PF0								EVENTOUT
PF1								EVENTOUT
PF2		USART7_CK_RTS_DE						EVENTOUT
PF6		USART7_RX						EVENTOUT
PF8		USART7_TX						EVENTOUT
PF9	USART6_TX	TMR12_CH1						EVENTOUT
PF10	USART6_RX	TMR12_CH2						EVENTOUT

Note: EVENTOUT is the TXEV signal of Cortex-M.

6.2.10 Peripheral MUX function configuration

IOMUX function configuration as follows:

- To use a peripheral pin in MUX output, it is configured as multiplexed push-pull/open-drain output.
- To use a peripheral pin in MUX input, it is configured as floating input/pull-up/pull-down input.
- For ADC peripherals, the pins of analog channels should be configured as analog input/output mode.
- For I^2C peripherals that intend to use pins as bidirectional functions, open-drain mode is required.
- For USB OTGFS_ID pin, configure the corresponding IOMUX and enable corresponding clocks in CRM; there is no need of GPIO status configuration.

6.2.11 IOMUX mapping priority

The unique peripheral multiplexed function can be configured through the GPIOx_MUXL/GPIOx_MUXH register, except individual pins that may be directly owned by hardware.

Some pins have been directly owned by specific hardware feature, whatever GPIO configuration.

Table 6-7 Pins owned by hardware

Pin	Enable bit	Description
PA0	PWC_CTRLSTS[8] =1	Once enabled, PA0 pin acts as WKUP1 of PWC.
PC13	PWC_CTRLSTS[9] = 1	Once enabled, PA0 pin acts as WKUP2 of PWC.
PB5	PWC_CTRLSTS[13] =1	Once enabled, PB5 pin acts as WKUP6 of PWC.

PB15	PWC_CTRLSTS[14] =1	Once enabled, PB15 pin acts as WKUP7 of PWC.
PC13	(ERTC_CTRL[23]=1)| (ERTC_CTRL[22:21]!=00)| (ERTC_CTRL[11]=1& ERTC_TAMP[17]=0)| (ERTC_TAMP[0]=1& ERTC_TAMP[16]=0)	Once enabled, PC13 pin is used as RTC channel.
PA0	(ERTC_CTRL[11]=1& ERTC_TAMP[17]=1)| (ERTC_TAMP[0]=1& ERTC_TAMP[16]=1)| (ERTC_TAMP[3]=1)	Once enabled, PA0 pin is used as TAMPER2_BPR.
PC14	CRM_BPDC[0]=1	Once enabled, PC14 is used as LEXT channel.
PC15	CRM_BPDC[0]=1 & CRM_BPDC[2]=0	Once enabled, PC15 is used as LEXT channel.
PA4	DAC_CTRL[2] =1	Once enabled, PA4 is used as DAC1 analog channel.
PA5	DAC_CTRL[18] =1	Once enabled, PA5 is used as DAC2 analog channel.
PF0	CRM_CTRL[16]=1	Once enabled, PF0 is used as HEXT channel.
PF1	CRM_CTRL[16]=1& CRM_CTRL[18]=0	Once enabled, PF1 is used as HEXT channel.
PA11	CRM_AHBEN2[7] & OTGFS_GCCFG[16]	Once enabled, PA11 is used as OTGFS_D- channel.
PA12	CRM_AHBEN2[7] & OTGFS_GCCFG[16]	Once enabled, PA12 is used as OTGFS_D+ channel.

Note: PA0 and PC13 cannot enable TAMPER_BPR function and WKUP of PWC at the same time.

6.2.12 External interrupt/wake-up lines

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

6.3 GPIO registers

The table below lists GPIO register map and their reset values. These peripheral registers can be accessed by bytes (8 bits), half-words (16 bits) or words (32 bits).

Table 6-8 GPIO register map and reset values

Register	Offset	Reset value
GPIOA_CFGR	0x00	0xA800 0000
GPIOx_CFGR(x =B,C,F)	0x00	0x0000 0280(B) 0x0000 0000
GPIOx_OMODER	0x04	0x0000 0000
GPIOx_ODRVR	0x08	0x0000 00C0(B) 0x0000 0000
GPIOA_PULL	0x0C	0x6400 0000(A)
GPIOx_PULL(x = B,C,F)	0x0C	0x0000 0100(B) 0x0000 0000
GPIOx_IDT	0x10	0x0000 XXXX
GPIOx_ODT	0x14	0x0000 0000
GPIOx_SCR	0x18	0x0000 0000
GPIOx_WPR	0x1C	0x0000 0000
GPIOx_MUXL	0x20	0x0000 0000
GPIOx_MUXH	0x24	0x0000 0000
GPIOx_CLR	0x28	0x0000 0000
GPIOx_TGR	0x2C	0x0000 0000
GPIOx_HDRV	0x3C	0x0000 0000

6.3.1 GPIO configuration register (GPIOx_CFGR) (x=A..F)

Reset value: 0xa8000000 for port A,
0x0000 0280 for port B,
0x00000000 for other ports.

Bit	Register	Reset value	Type	Description
Bit 2y+1:2y	IOMCy	0xA800 0000	rw	GPIOx mode configuration (y=0~15) This field is used to configure the GPIOx mode: 00: Input mode (reset state) 01: General-purpose output mode 10: Multiplexed function mode 11: Analog mode

6.3.2 GPIO output mode register (GPIOx_OMODE) (x=A..F)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Always 0.
Bit 15:0	OM	0x0000	rw	GPIOx output mode configuration (y=0..15) This field is used to configure the output mode of GPIOx: 0: Push-pull (reset state) 1: Open-drain

6.3.3 GPIO drive capability register (GPIOx_ODRVR) (x=A..F)

Reset value: 0x0000 00C0 for port B,
0x00000000 for other ports.

Bit	Register	Reset value	Type	Description
Bit 2y+1:2y	ODRVy	0x0000 0000	rw	GPIOx drive capability (y=0..15) This field is used to configure the IO port drive capability. x0: Normal sourcing/sinking strength 01: Large sourcing/sinking strength 11: Normal sourcing/sinking strength

6.3.4 GPIO pull-up/pull-down register (GPIOx_PULL) (x=A..F)

Reset value: 0x6400 0000 for port A,
0x0000 0100 for port B,
0x00000000 for other ports.

Bit	Register	Reset value	Type	Description
Bit 2y+1:2y	PULLy	0x6400 0000	rw	GPIOx pull-up/pull-down configuration (y=0…15) This field is used to configure the pull-up/pull-down of the IO port. 00: No pull-up/pull-down 01: Pull-up 10: Pull-down

6.3.5 GPIO input data 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 It indicates the input status of I/O port. Each bit corresponds to an I/O.

6.3.6 GPIO output data 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. It indicates the output status of I/O port. 0: Low 1: High

6.3.7 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. If both IOCB and IOSB bits are set to 1, the IOSB takes the priority. 0: No action to the corresponding ODT bits 1: Set the corresponding ODT bits

6.3.8 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 for four times in the order below: write "1" ->write "0" ->write "0"-> 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

6.3.9 GPIO multiplexed function low register (GPIOx_MUXL) (x=A..F)

Bit	Register	Reset value	Type	Description
Bit 4y+3:4y	MUXLy	0x0	rw	Multiplexed function select for GPIOx pin y (y=0…7) This field is used to configure multiplexed function IOs. 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13

				1110: MUX14 1111: MUX15

6.3.10 GPIO multiplexed function high register (GPIOx_MUXH) (x=A..F)

Bit	Register	Reset value	Type	Description
Bit 4y+3:4y	MUXHy	0x0	rw	Multiplexed function select for GPIOx pin y (y=8…15) This field is used to configure multiplexed function IOs. 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13 1110: MUX14 1111: MUX15

6.3.11 GPIO port 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

6.3.12 GPIO port bit toggle register (GPIOx_TOGR) (x=A..F)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:0	IOTB	0x0000	wo	GPIOx toggle bit 0: No effect on the corresponding bit (equivalent to ODT register operation) 1: Toggle the corresponding bit

6.3.13 GPIO huge current control register (GPIOx_HDRV) (x=A..F)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:0	HDRV	0x0000	rw	Huge sourcing/sinking strength control 0: Not active 1: GPIO is configured as maximum sourcing/sinking strength

7 System configuration controller (SCFG)

7.1 Introduction

This device contains a set of system configuration register. The system configuration controller is mainly set to:

- Manage the external interrupts connected to the GPIOs
- Control the memory mapping mode
- Manage IRTMR GPIO configurations

7.2 SCFG registers

The table below shows SCFG register map and their reset values.

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

Table 7-1 SCFG register map and reset value

Register	Offset	Reset value
SCFG_CFG1	0x00	0x0000 000X
SCFG_CFG2	0x04	0x0000 0000
SCFG_EXINTC1	0x08	0x0000 0000
SCFG_EXINTC2	0x0C	0x0000 0000
SCFG_EXINTC3	0x10	0x0000 0000
SCFG_EXINTC4	0x14	0x0000 0000
SCFG_UHDRV	0x2C	0x0000 0000

7.2.1 SCFG configuration register 1 (SCFG_CFG1)

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x00000 00	resd	Kept at its default value.
Bit 7:6	IR_SRC_SEL	0x0	rw	Infrared modulation envelope signal source selection This field is used to select the infrared modulation envelope signal source. 00: TMR10 01: USART1 10: USART2 11: Reserved
Bit 5	IR_POL	0x0	rw	Infrared output polarity selection 0: Infrared output (IR_OUT) is not inversed 1: Infrared output (IR_OUT) is inversed
Bit 4:2	Reserved	0x0	resd	Kept at its default value.
Bit 1:0	MEM_MAP_SEL	0xX	ro	Boot mode status bit This bit is read-only, indicating the boot mode after reset. X0: Boot from main Flash memory 01: Boot from system memory 11: Boot from internal SRAM

7.2.2 SCFG configuration register 2 (SCFG_CFG2)

Bit	Register	Reset value	Type	Description
Bit 31:30	I2S_FD	0x0	rw	I^2S full duplex configuration bit It is used to configure I^2S full-duplex mode. This bit must remain 00 if there is no need for I^2S full duplex configuration. Refer to Section 13.3.2 for details. 00: SPI/I^2S1~3 operates separately 01: I^2S1 and I^2S3 are configured as full-duplex mode 10: I^2S2 and I^2S3 are configured as full-duplex mode 11: I^2S1 and I^2S2 are configured as full-duplex mode

Bit 29:3	Reserved	0x0000 000	resd	Kept at its default value.
Bit 2	PVM_LK	0x0	rw	PVM lock enable 0: Disconnect the PVM interrupt with TIM1/TIM9/ TIM10/11/12/13/14 break input. The PVMSEL and PVMEN bits can be modified by software. 1: Connect the break input. Both PVMSEL and PVMEN bits are read-only, and cannot be modified by software.
Bit 1	Reserved	0x0	resd	Kept at its default value.
Bit 0	CPU_LK	0x0	rw	CPU lock enable 0: Disconnect CPU lock from the break input of TIM1/TIM9/TIM10/11/12/13/14 1: Connect CPU lock with the break input of TIM1/TIM9/TIM10/11/12/13/14

7.2.3 SCFG external interrupt configuration register 1 (SCFG_EXINTC1)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:12	EXINT3	0x0	rw	EXINT3 input source configuration These bits are used to select the input source for the EXINT3 external interrupt. 0000: GPIOA pin 3 0001: GPIOB pin 3 0010: GPIOC pin 3 0011: GPIOD pin 3 0100: GPIOE pin 3 Others: Reserved
Bit 11:8	EXINT2	0x0	rw	EXINT2 input source configuration These bits are used to select the input source for the EXINT2 external interrupt. 0000: GPIOA pin 2 0001: GPIOB pin 2 0010: GPIOC pin 2 0011: GPIOD pin 2 0100: GPIOE pin 2 0101: GPIOF pin 2 Others: Reserved
Bit 7:4	EXINT1	0x0	rw	EXINT1 input source configuration These bits are used to select the input source for the EXINT1 external interrupt. 0000: GPIOA pin 1 0001: GPIOB pin 1 0010: GPIOC pin 1 0011: GPIOD pin 1 0100: GPIOE pin 1 0101: GPIOF pin 1 Others: Reserved
Bit 3:0	EXINT0	0x0	rw	EXINT0 input source configuration These bits are used to select the input source for the EXINT0 external interrupt. 0000: GPIOA pin 0 0001: GPIOB pin 0 0010: GPIOC pin 0 0011: GPIOD pin 0 0100: GPIOE pin 0 0101: GPIOF pin 0 Others: Reserved

7.2.4 SCFG external interrupt configuration register 2 (SCFG_EXINTC2)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:12	EXINT7	0x0	rw	EXINT7 input source configuration These bits are used to select the input source for the EXINT7 external interrupt. 0000: GPIOA pin 7 0001: GPIOB pin 7 0010: GPIOC pin 7 0011: GPIOD pin 7 0100: GPIOE pin 7 Others: Reserved
Bit 11:8	EXINT6	0x0	rw	EXINT6 input source configuration These bits are used to select the input source for the EXINT6 external interrupt. 0000: GPIOA pin 6 0001: GPIOB pin 6 0010: GPIOC pin 6 0011: GPIOD pin 6 0100: GPIOE pin 6 0101: GPIOF pin 6 Others: Reserved
Bit 7:4	EXINT5	0x0	rw	EXINT5 input source configuration These bits are used to select the input source for the EXINT5 external interrupt. 0000: GPIOA pin 5 0001: GPIOB pin 5 0010: GPIOC pin 5 0011: GPIOD pin 5 0100: GPIOE pin 5 Others: Reserved
Bit 3:0	EXINT4	0x0	rw	EXINT4 input source configuration These bits are used to select the input source for the EXINT4 external interrupt. 0000: GPIOA pin 4 0001: GPIOB pin 4 0010: GPIOC pin 4 0011: GPIOD pin 4 0100: GPIOE pin 4 Others: Reserved

7.2.5 SCFG external interrupt configuration register 3 (SCFG_EXINTC3)

Bit	**Register**	**Reset value**	**Type**	**Description**
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:12	EXINT11	0x0	rw	EXINT11 input source configuration These bits are used to select the input source for the EXINT11 external interrupt. 0000: GPIOA pin 11 0001: GPIOB pin 11 0010: GPIOC pin 11 0011: GPIOD pin 11 0100: GPIOE pin 11 Others: Reserved
Bit 11:8	EXINT10	0x0	rw	EXINT10 input source configuration These bits are used to select the input source for the EXINT10 external interrupt. 0000: GPIOA pin 10 0001: GPIOB pin 10 0010: GPIOC pin 10 0011: GPIOD pin 10 0100: GPIOE pin 10 0101: GPIOF pin 10

				Others: Reserved
Bit 7:4	EXINT9	0x0	rw	EXINT9 input source configuration These bits are used to select the input source for the EXINT9 external interrupt. 0000: GPIOA pin 9 0001: GPIOB pin 9 0010: GPIOC pin 9 0011: GPIOD pin 9 0100: GPIOE pin 9 0101: GPIOF pin 9 Others: Reserved
Bit 3:0	EXINT8	0x0	rw	EXINT8 input source configuration These bits are used to select the input source for the EXINT8 external interrupt. 0000: GPIOA pin 8 0001: GPIOB pin 8 0010: GPIOC pin 8 0011: GPIOD pin 8 0100: GPIOE pin 8 0101: GPIOF pin 8 Others: Reserved

7.2.6 SCFG external interrupt configuration register 4 (SCFG_EXINTC4)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Kept at its default value.
Bit 15:12	EXINT15	0x0	rw	EXINT15 input source configuration These bits are used to select the input source for the EXINT15 external interrupt. 0000: GPIOA pin 15 0001: GPIOB pin 15 0010: GPIOC pin 15 0011: GPIOD pin 15 0100: GPIOE pin 15 Others: Reserved
Bit 11:8	EXINT14	0x0	rw	EXINT14 input source configuration These bits are used to select the input source for the EXINT14 external interrupt. 0000: GPIOA pin 14 0001: GPIOB pin 14 0010: GPIOC pin 14 0011: GPIOD pin 14 0100: GPIOE pin 14 Others: Reserved
Bit 7:4	EXINT13	0x0	rw	EXINT13 input source configuration These bits are used to select the input source for the EXINT13 external interrupt. 0000: GPIOA pin 13 0001: GPIOB pin 13 0010: GPIOC pin 13 0011: GPIOD pin 13 0100: GPIOE pin 13 Others: Reserved
Bit 3:0	EXINT12	0x0	rw	EXINT12 input source configuration These bits are used to select the input source for the EXINT12 external interrupt. 0000: GPIOA pin 12 0001: GPIOB pin 12 0010: GPIOC pin 12 0011: GPIOD pin 12 0100: GPIOE pin 12 Others: Reserved

7.2.7 SCFG ultra high souring/sinking strength (SCFG_UHDRV)

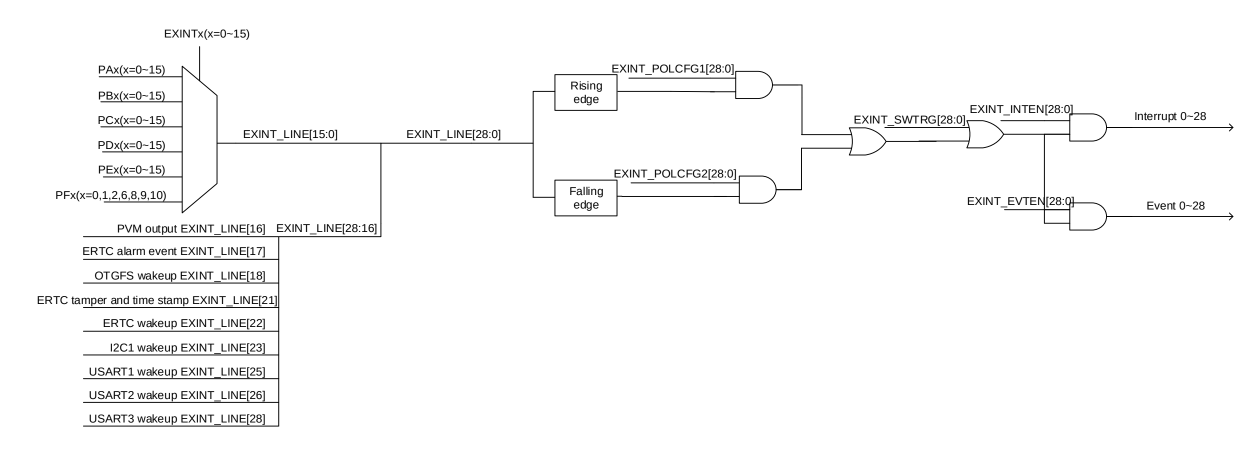
Bit	Register	Reset value	Type	Description
Bit 31:7	Reserved	0x0000 000	resd	Kept at its default value.
Bit 6	PD13_UH	0x0	rw	PD13 Ultra high sourcing/sinking strength This bit is written by software to control PD13 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra high sourcing/sinking strength. When this bit is set, the control bits of GPIOx_ODRVR&GPIOx_HDRV becomes invalid.
Bit 5	PD12_UH	0x0	rw	PD12 Ultra high sourcing/sinking strength This bit is written by software to control PD12 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra high sourcing/sinking strength. When this bit is set, the control bits of GPIOx_ODRVR&GPIOx_HDRV becomes invalid.
Bit 4	Reserved	0x0	resd	Kept at its default value.
Bit 3	PB8_UH	0x0	rw	PB8 Ultra high sourcing/sinking strength This bit is written by software to control PD8 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra high sourcing/sinking strength. When this bit is set, the control bits of GPIOx_ODRVR&GPIOx_HDRV becomes invalid.
Bit 2	Reserved	0x0	resd	Kept at its default value.
Bit 1	PB9_UH	0x0	rw	PB9 Ultra high sourcing/sinking strength This bit is written by software to control PD9 PAD sourcing/sinking strength. 0: Not active 1: Corresponding GPIO is switched to ultra high sourcing/sinking strength. When this bit is set, the control bits of GPIOx_ODRVR&GPIOx_HDRV becomes invalid.
Bit 0	Reserved	0x0	resd	Kept at its default value.

8 External interrupt/event controller (EXINT)

8.1 EXINT introduction

EXINT consists of 25 interrupt lines EXINT_LINE[28:0] (in which 19, 20, 24 and 27 bits are reserved), 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 trigger source in order to generate an interrupt or event.

Figure 8-1 External interrupt/event controller block diagram



Main features:

- EXINT interrupt lines 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 25 software triggers 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 25 interrupt lines EXINT_LINE[28:0] (in which 19, 20, 24 and 27 bits are reserved), EXINT can detect not only GPIO external interrupt sources but also nine internal sources such as PVM output, ERTC alarm events, ERTC tamper and time stamp events, ERTC wakeup events, OTGFS wakeup events, USART1/USART2/USART3 wakeup events and I^2C1 wakeup events through edge detection mechanism, where, GPIO interrupt sources can be selected with the SCFG_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 register, indicating that the corresponding interrupt or event must be enabled prior to either edge detection or software trigger.

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.

Interrupt initialization procedure

- Select an interrupt by setting the SCFG_EXINTCx register (this is required if GPIO is used as an interrupt source);
- Select a trigger mode by setting the EXINT_POLCFG1 and EXINT_POLCFG2 registers;
- Enable interrupt or event by setting the EXINT_INTEN and EXINT_EVTEN registers;
- Generate software trigger by setting the EXINT_SWTRG register (this is applied to only software trigger interrupt).

Note: To modify the interrupt source configuration, disable the EXINT_INTEN and EXINT_EVTEN registers and then restart interrupt initialization.

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

The table below shows EXINT register map and their reset value.

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

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)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	INTENx	0x00000	rw	Interrupt enable or disable on line x 0: Interrupt request is disabled 1: Interrupt request is enabled Note: The 19, 20, 24 and 27 bits are reserved and unused.

8.3.2 Event enable register (EXINT_EVTEN)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	EVTENx	0x00000	rw	Event enable or disable on line x 0: Event request is disabled 1: Event request is enabled Note: The 19, 20, 24 and 27 bits are reserved and unused.

8.3.3 Polarity configuration register 1 (EXINT_POLCFG1)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	RPx	0x00000	rw	Rising polarity configuration bit of line x These bits are used to select a rising edge to trigger an interrupt and event on line x. 0: Rising edge trigger on line x is disabled 1: Rising edge trigger on line x is enabled Note: The 19, 20, 24 and 27 bits are reserved and unused.

8.3.4 Polarity configuration register 2 (EXINT_POLCFG2)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	FRx	0x00000	rw	Falling polarity event configuration bit of line x These bits are used to select a falling edge to trigger an interrupt and event on line x. 0: Falling edge trigger on line x is disabled 1: Falling edge trigger on line x is enabled Note: The 19, 20, 24 and 27 bits are reserved and unused.

8.3.5 Software trigger register (EXINT_SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	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 register automatically to generate an interrupt. If the corresponding bit in 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. Note: The 19, 20, 24 and 27 bits are reserved and unused.

8.3.6 Interrupt status register (EXINT_INTSTS)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x0	resd	Forced to 0 by hardware.
Bit 28:0	LINEx	0x00000	rw1c	Line x state bit 0: No interrupt occurred 1: Interrupt occurred Note: This bit is cleared by writing “1”. Note: The 19, 20, 24 and 27 bits are reserved and unused.

9 DMA controller (DMA)

9.1 Introduction

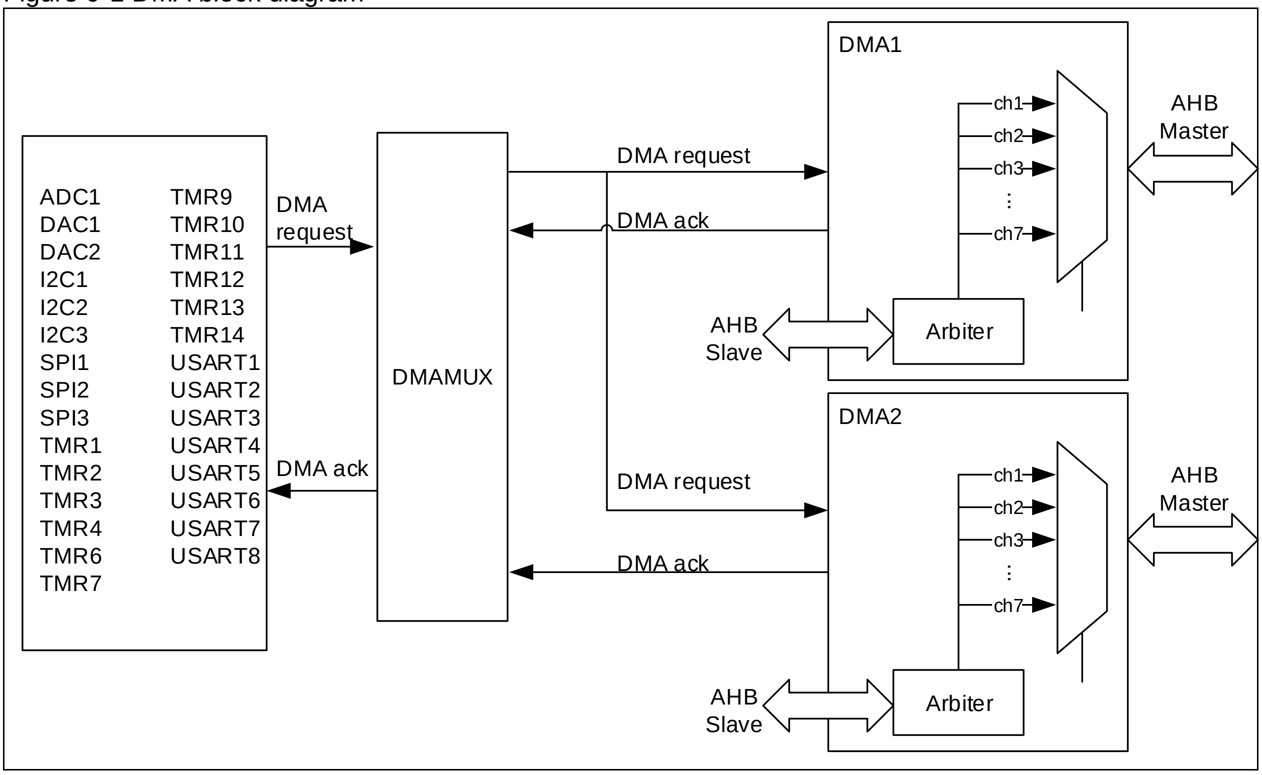
Direct memory access (DMA) controller is designed for high speed data transmission between peripherals and memory or between memories. The data can be transmitted through DMA at a high speed without CPU interference, which saves CPU capacity.

There are two DMA controllers in the microcontroller. Each controller contains 7 DMA channels that manage 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 multiplexing

Figure 9-1 DMA block diagram



Note: The number of DMA peripherals may be different depending on different models.

9.3 Function overview

9.3.1 DMA configuration

1. Set the peripheral address in the DMA_CPBAx register

The initial peripheral address for data transfer remains unchanged during transmission.

2. Set the memory address in the DMA_CMBAx register

The initial memory address for data transfer remains unchanged during transmission.

3. Configure the amount of data to be transferred in the DMA_DTCNTx 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_CHCTRLx 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) and 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_CHCTRLx register

9.3.2 Handshake mechanism

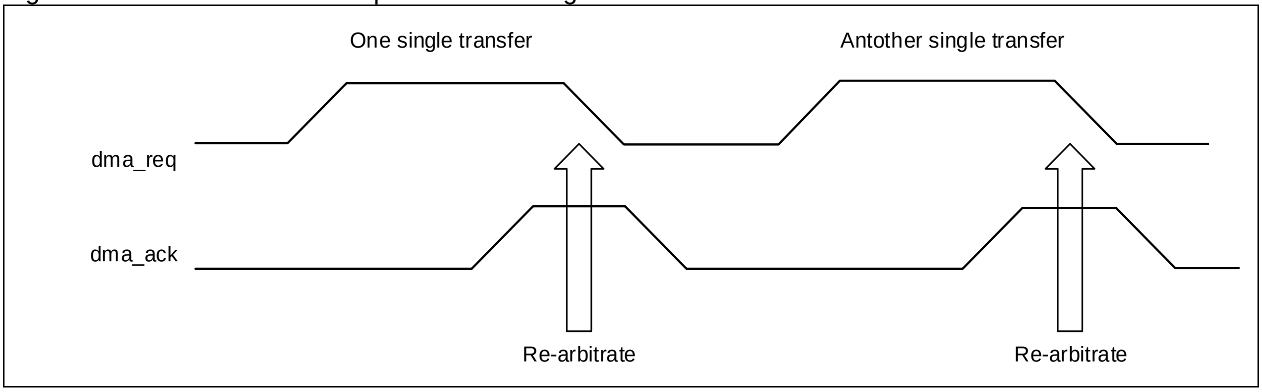
In P2M and M2P mode, the peripherals need to send a request signal to the DMA controller. The DMA channel will send the peripheral transfer request (single) until the signal is acknowledged. After the completion of peripheral transmission, the DMA controller sends the acknowledge signal to the peripheral. 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_CHCTRLx 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

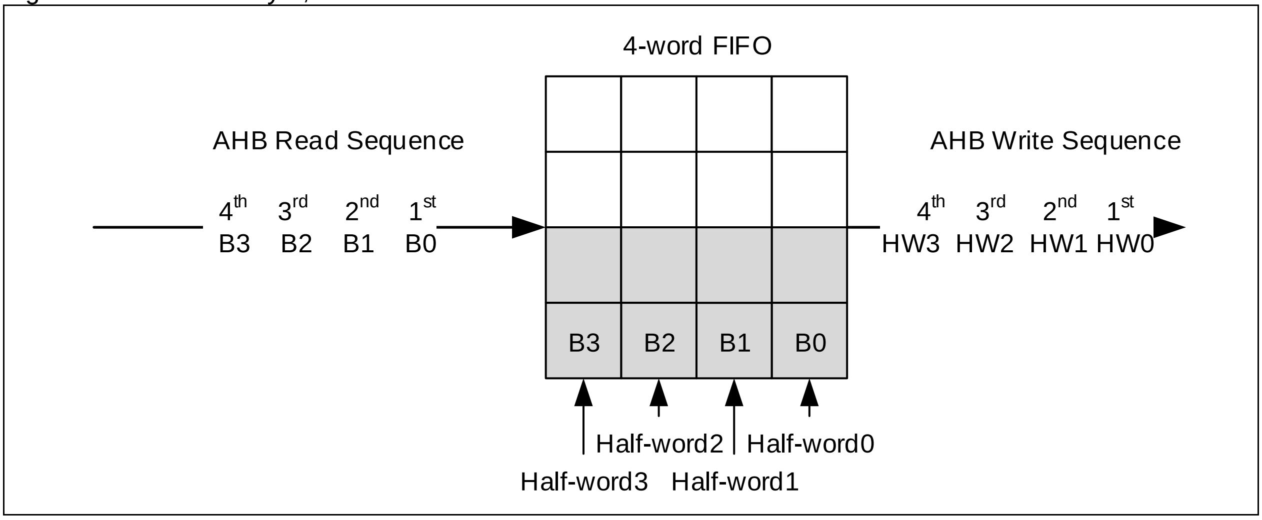


Figure 9-4 PWIDTH: half-word, MWIDTH: word

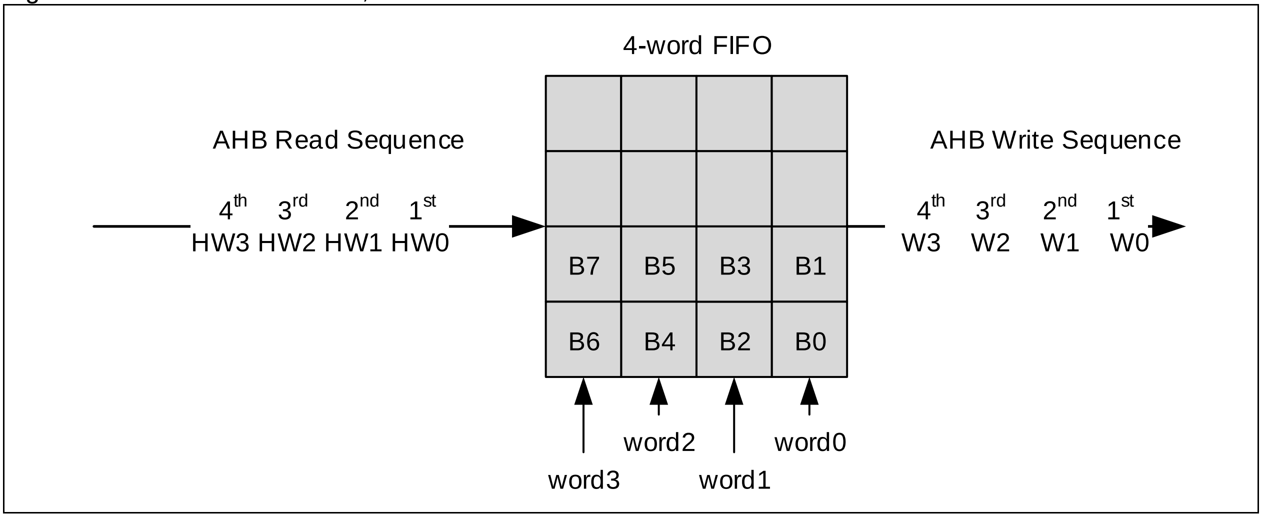
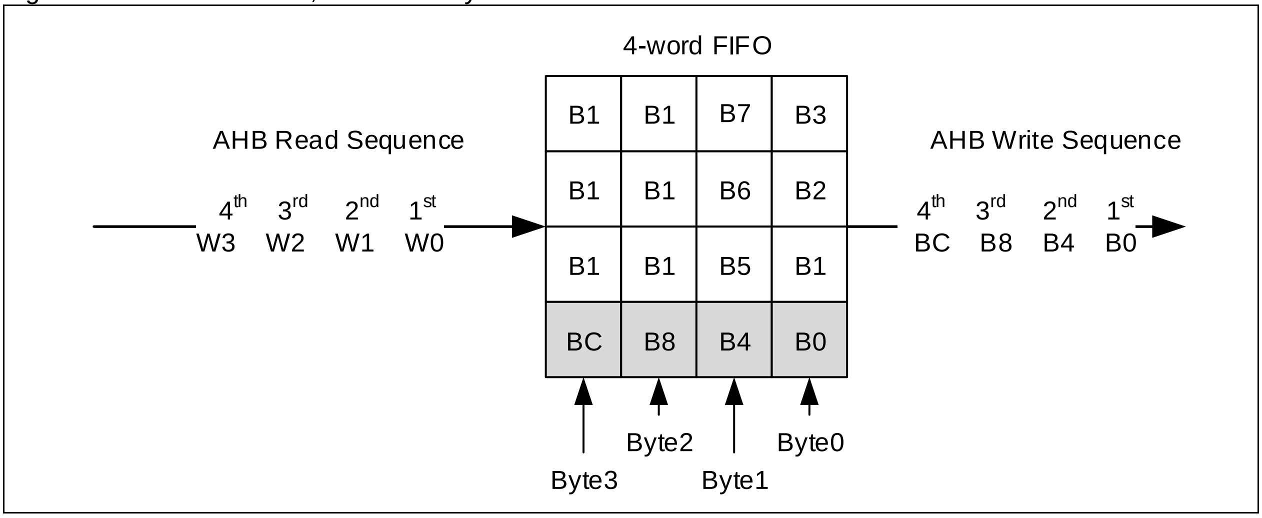


Figure 9-5 PWIDTH: word, MWIDTH: byte



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 interrupts

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

9.4 DMA multiplexer (DMAMUX)

DMAMUX manages DMA requests/acknowledge between peripherals and DMA controller.

The DMA controller selects the DMA mapping table with the TBL_SEL bit in the DMA_MUXSEL register. Each DMA controller stream selects only one DMA request from the flexible mapping table. In flexible mapping mode, each channel can bypass or synchronize 127 possible channel requests from peripherals or generators through the REQSEL [6: 0] bit in the DMA_MUXCxCTRL register.

9.4.1 DMAMUX function overview

The DMAMUX consists of a request generator and a request multiplexer.

Each of the DMAMUX generator channel x has a GEN enable bit in the DMA_MUXGxCTRL register. The SIGSEL bit is used to select the trigger input of the DMAMUX generator. Typically, the number of DMA requests equals GREQCNT + 1. The GPOL bit is used in the DMA_MUXGxCTRL register to select a trigger event that can be on a rising edge, falling edge or either of them.

Each of the DMAMUX stream x comes from all_req [127:1].

In flexible mapping mode, the SYNCEN bit in the DMA_MUXSxCTRL register is used to synchronize the selected DMA request input. In synchronous mode, the SYNCSEL bit in the DMA_MUXSxCTRL register is used to select synchronized input. The selected DMA request input will be transferred to chx_mux_req [7: 0] as soon as a valid edge of the synchronized input is detected by the SYNCPOL [1: 0] in the DMA_MUXSxCTRL register. In addition, when the EGE bit is set in the DMA_MUXCxCTRL register, the programmable request counter (REQCNT) is used to generate a request output and event output.

Figure 9-6 DMAMUX block diagram

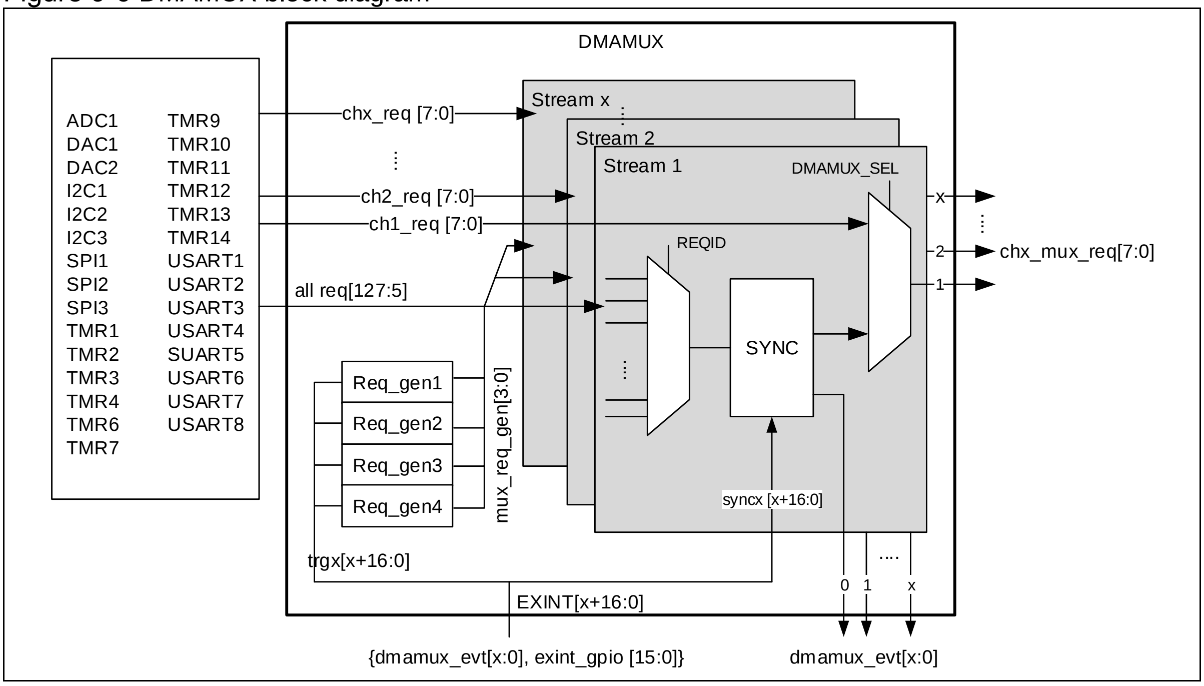


Table 9-3 Flexible DMA1 / DMA2 request mapping

CHx_ SRC	Request source	CHx_ SRC	Request source	CHx_ SRC	Request source	CHx_ SRC	Request source
1	DMA_MUXREQG1	33	USART5_TX	65	TMR3_OVERFLOW	97	TMR12_TRIG
2	DMA_MUXREQG2	34	reserved	66	TMR3_TRIG	98	TMR12_HALL
3	DMA_MUXREQG3	35	reserved	67	TMR4_CH1	99	reserved
4	DMA_MUXREQG4	36	reserved	68	TMR4_CH2	100	reserved
5	ADC1	37	reserved	69	TMR4_CH3	101	reserved
6	DAC1	38	reserved	70	TMR4_CH4	102	reserved
7	reserved	39	reserved	71	TMR4_OVERFLOW	103	reserved
8	TMR6_OVERFLOW	40	reserved	72	reserved	104	reserved
9	TMR7_OVERFLOW	41	DAC2	73	reserved	105	reserved
10	SPI1_RX	42	TMR1_CH1	74	reserved	106	reserved
11	SPI1_TX	43	TMR1_CH2	75	reserved	107	reserved
12	SPI2_RX	44	TMR1_CH3	76	reserved	108	reserved
13	SPI2_TX	45	TMR1_CH4	77	reserved	109	reserved
14	SPI3_RX	46	TMR1_OVERFLOW	78	TMR9_CH1	110	reserved
15	SPI3_TX	47	TMR1_TRIG	79	TMR9_OVERFLOW	111	reserved
16	I2C1_RX	48	TMR1_HALL	80	TMR9_TRIG	112	reserved
17	I2C1_TX	49	reserved	81	TMR9_HALL	113	reserved
18	I2C2_RX	50	reserved	82	TMR10_CH1	114	USART6_RX
19	I2C2_TX	51	reserved	83	TMR10_OVERFLOW	115	USART6_TX
20	I2C3_RX	52	reserved	84	TMR11_CH1	116	USART7_RX
21	I2C3_TX	53	reserved	85	TMR11_OVERFLOW	117	USART7_TX
22	reserved	54	reserved	86	reserved	118	USART8_RX
23	reserved	55	reserved	87	reserved	119	USART8_TX
24	USART1_RX	56	TMR2_CH1	88	reserved	120	TMR13_CH1
25	USART1_TX	57	TMR2_CH2	89	reserved	121	TMR13_OVERFLOW
26	USART2_RX	58	TMR2_CH3	90	reserved	122	TMR14_CH1
27	USART2_TX	59	TMR2_CH4	91	reserved	123	TMR14_OVERFLOW
28	USART3_RX	60	TMR2_OVERFLOW	92	reserved	124	TMR9_CH2
29	USART3_TX	61	TMR3_CH1	93	reserved	125	TMR12_CH2
30	USART4_RX	62	TMR3_CH2	94	reserved	126	TMR2_TRIG
31	USART4_TX	63	TMR3_CH3	95	TMR12_CH1	127	TMR4_TRIG
32	USART5_RX	64	TMR3_CH4	96	TMR12_OVERFLOW		

Table 9-4 DMAMUX EXINT LINE for trigger input and synchronized input

EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source	EXINT LINE	Source
0	exint_gpio[0]	8	exint_gpio[8]	16	DMA_MUXevt1	24	reserved
1	exint_gpio[1]	9	exint_gpio[9]	17	DMA_MUXevt2	25	reserved
2	exint_gpio[2]	10	exint_gpio[10]	18	DMA_MUXevt3	26	reserved
3	exint_gpio[3]	11	exint_gpio[11]	19	DMA_MUXevt4	27	reserved
4	exint_gpio[4]	12	exint_gpio[12]	20	DMA_MUXevt5	28	reserved
5	exint_gpio[5]	13	exint_gpio[13]	21	DMA_MUXevt6	39	reserved
6	exint_gpio[6]	14	exint_gpio[14]	22	DMA_MUXevt7	30	reserved
7	exint_gpio[7]	15	exint_gpio[15]	23	reserved	31	reserved

9.4.2 DMAMUX overflow interrupts

During DMAMUX request generation, when a new trigger input occurs before the GREQCNT underflows, the TRGOVFx bit will be set in the DMA_MUXGSTS register. It is cleared by setting TRGOVFCx=1 in the DMA_MUXGCLR register. An interrupt will be generated if the interrupt enable bit TRGOVIEN is set in the DMA_MUXGxCTRL register.

In DMAMUX synchronous mode, when a new synchronized input occurs before the REQCNT underflows, the SYNCOVFx bit will be set in the DMA_MUXSYNCSTS register. It is cleared by setting SYNCOVFCx=1 in the DMA_MUXSYNCCLR register. An interrupt will be generated if the interrupt enable bit SYNCOVIEN is set in the DMA_MUXSxCTRL register.

Figure 9-7 DMAMUX request synchronized mode

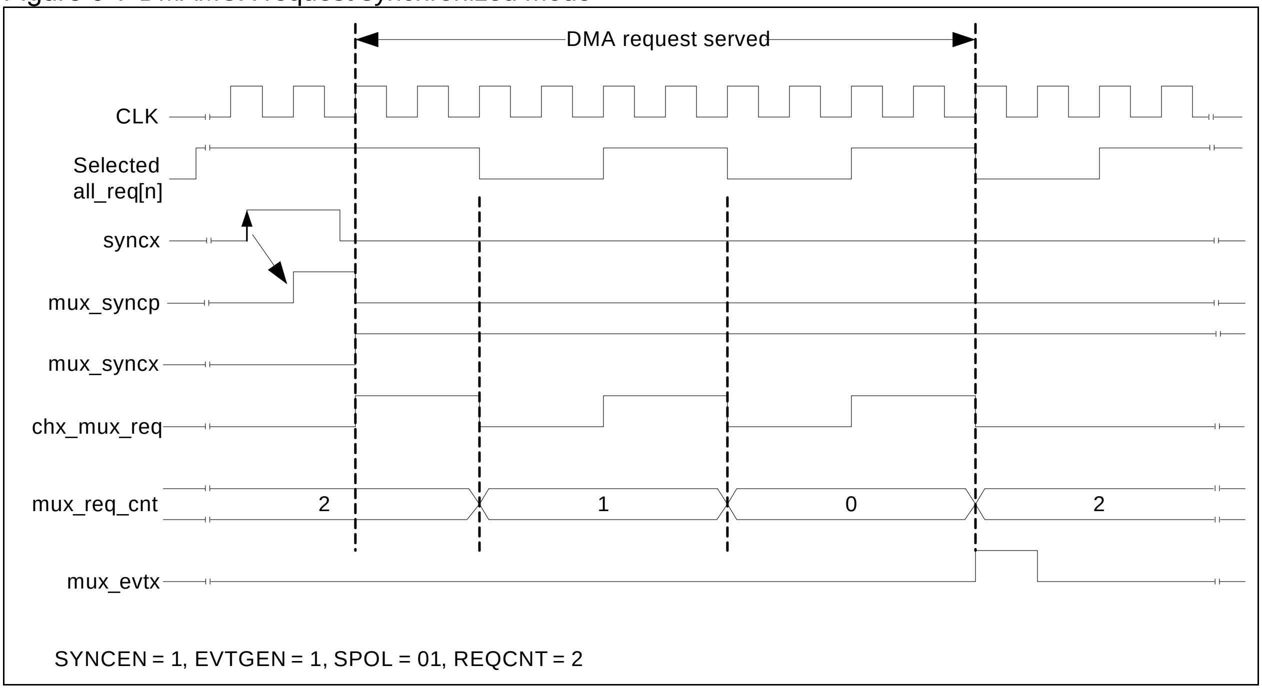
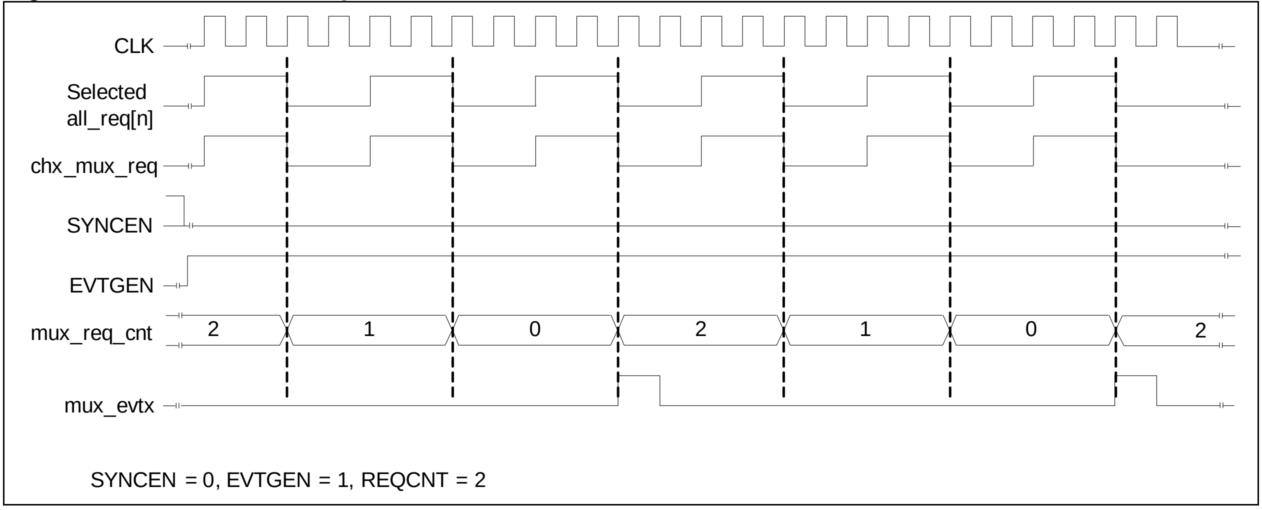


Figure 9-8 DMAMUX event generation



9.5 DMA registers

The table below lists DMA register map and their reset values.

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

Table 9-5 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_MUXSEL	0x100	0x0000 0000
DMA_MUXC1CTRL	0x104	0x0000 0000
DMA_MUXC2CTRL	0x108	0x0000 0000
DMA_MUXC3CTRL	0x10c	0x0000 0000
DMA_MUXC4CTRL	0x110	0x0000 0000
DMA_MUXC5CTRL	0x114	0x0000 0000
DMA_MUXC6CTRL	0x118	0x0000 0000
DMA_MUXC7CTRL	0x11c	0x0000 0000
DMA_MUXG1CTRL	0x120	0x0000 0000
DMA_MUXG2CTRL	0x124	0x0000 0000
DMA_MUXG3CTRL	0x128	0x0000 0000
DMA_MUXG4CTRL	0x12c	0x0000 0000
DMA_MUXSYNCSTS	0x130	0x0000 0000
DMA_MUXSYNCCLR	0x134	0x0000 0000
DMA_MUXGSTS	0x138	0x0000 0000
DMA_MUXGCLR	0x13c	0x0000 0000

9.5.1 DMA interrupt status register (DMA_STS)

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

Bit	Register	Reset value	Type	Description
Bit 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	Channel 7 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	Channel 7 global event flag 0: No transfer error, half transfer or transfer complete event occurred 1: Transfer error, half transfer or transfer complete event
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 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

9.5.2 DMA interrupt flag clear register (DMA_CLR)

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

Bit	Register	Reset value	Type	Description
Bit 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 DTERRF7 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 flag clear 0: No effect 1: Clear DTERRF7, HDTF7, FDTF7 and GF7 flags 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 flag clear 0: No effect 1: Clear DTERRF6, HDTF6, FDTF6 and GF6 flags 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 flag clear 0: No effect 1: Clear DTERRF5, HDTF5, FDTF5 and GF5 flags 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 flag clear 0: No effect 1: Clear DTERRF4, HDTF4, FDTF4 and GF4 flags in the DMA_STS register
Bit 11	DTERRFC3	0x0	rw1c	Channel 3 data transfer error flag clear 0: No effect 1: Clear the DTERRF3 flag in the DMA_STS register
Bit 10	HDTFC3	0x0	rw1c	Channel 3 half transfer flag clear 0: No effect 1: Clear the HDTF3 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 flag clear 0: No effect 1: Clear DTERRF3, HDTF3, FDTF3 and GF3 flags 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 flag clear 0: No effect 1: Clear DTERRF2, HDTF2, FDTF2 and GF2 flags 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	Chanel 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 flag clear 0: No effect 1: Clear DTERRF1, HDTF1, FDTF1 and GF1 flags in the DMA_ISTS register

9.5.3 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 11: 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	Loop 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

9.5.4 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 to 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.

9.5.5 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

9.5.6 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.

9.5.7 DMAMUX select register (DMA_MUXSEL)

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

Bit	Register	Reset value	Type	Description
31:1	Reserved	0x0000 0000	resd	Kept at its default value.
Bit 0	TBL_SEL	0x0	rw	Multiplexer table select 0x1: Flexible mapping table

9.5.8 DMAMUX channel-x control register (DMA_MUXCxCTRL) (x = 1…7)

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 28:24	SYNCSEL	0x00	rw	Synchronization select
Bit 23:19	REQCNT	0x00	rw	DMA request count This field is used to define the DMA request number after synchronization event and/or the DMA request nmber before the generation of output event. These bits can only be written when both SYNCEN and EVTGEN bits are LOW.
Bit 18:17	SYNCPOL	0x0	rw	Synchronization polarity It is used to define the polarity of the selected synchronization input. 0x0: No event 0x1: Rising edge 0x2: Falling edge 0x3: Rising edge and falling edge
Bit 16	SYNCEN	0x0	rw	Synchronization enable 0: Disabled 1: Enabled
Bit 15:10	Reserved	0x00	resd	Kept at its default value.

Bit 9	EVTGEN	0x0		Event generate enable 0: Disabled 1: Enabled
Bit 8	SYNCOVIEN	0x0		Synchronization overrun interrupt enable 0: Interrupt disabled 1: Interrupt enabled
Bit 7	Reserved	0x0	resd	Kept at its default value.
Bit 6:0	REQSEL	0x00		DMA request select flag It is used to select the input DMA request. Refer to the DMAMUX table for the configuration of multiplexer input.

9.5.9 DMAMUX generator-x control register (DMA_MUXGxCTRL) (x = 1…4)

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

Bit	Register	Reset value	Type	Description
Bit 31:24	Reserved	0x00	resd	Kept at its default value.
Bit 23:19	GREQCNT	0x00	rw	DMA request generation count It is used to define the number of DMA request (GNBREQ + 1) to be generated after trigger event These bits can be written n when the GEN bit is disabled.
Bit 18:17	GPOL	0x0	rw	DMA request generator trigger polarity It is used to define the polarity of the selected trigger input. 0x0: No event 0x1: Rising edge 0x2: Falling edge 0x3: Rising edge and falling edge
Bit 16	GEN	0x0	rw	DMA request generator enable 0: Disabled 1: Enabled
Bit 15:9	Reserved	0x00	resd	Kept at its default value.
Bit 8	TRGOVIEN	0x0	rw	Trigger overrun interrupt enable 0: Disabled 1: Enabled
Bit 7:5	Reserved	0x0	resd	Kept at its default value.
Bit 4:0	SIGSEL	0x00	rw	Signal select It is used to select the DMA trigger input for DMA request generator.

9.5.10 DMAMUX channel synchronization status register (DMA_MUXSYNCSTS)

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

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7:0	SYNCOVF	0x00	ro	Synchronization overrun interrupt flag The synchronization overrun interrupt occurs when the DMA request counter is below REQCNT. This flag is set when a new synchronization event occurs.

9.5.11 DMAMUX channel interrupt clear register (DMA_MUXSYNCCLR)

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

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x0000 00	resd	Kept at its default value.
Bit 7:0	SYNCOVFC	0x00	rw1c	Synchronization overrun interrupt flag clear Write “1” to each bit to clear the corresponding overrun flag SYNCOVF in the MUXSYNCSTS register.

9.5.12 DMAMUX generator interrupt status register (DMA_MUXGSTS)

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

Bit	Register	Reset value	Type	Description
Bit 31:4	Reserved	0x0000 000	resd	Kept at its default value.
Bit 3:0	TRGOVF	0x00	ro	Trigger overrun interrupt flag When the DMA request counter is below GREQCNT, this flag is set if a new trigger event occurs.

9.5.13 DMAMUX generator interrupt clear register (DMA_MUXGCLR)

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

Bit	Register	Reset value	Type	Description
Bit 31:4	Reserved	0x0000 000	resd	Kept at its default value.
Bit 3:0	TRGOVFC	0x00	rw1c	Trigger overrun interrupt flag clear Write “1” to each bit to cleat the corresponding overrun flag TRGOVF in the DMA_MUXGSTS register.

10 CRC calculation unit (CRC)

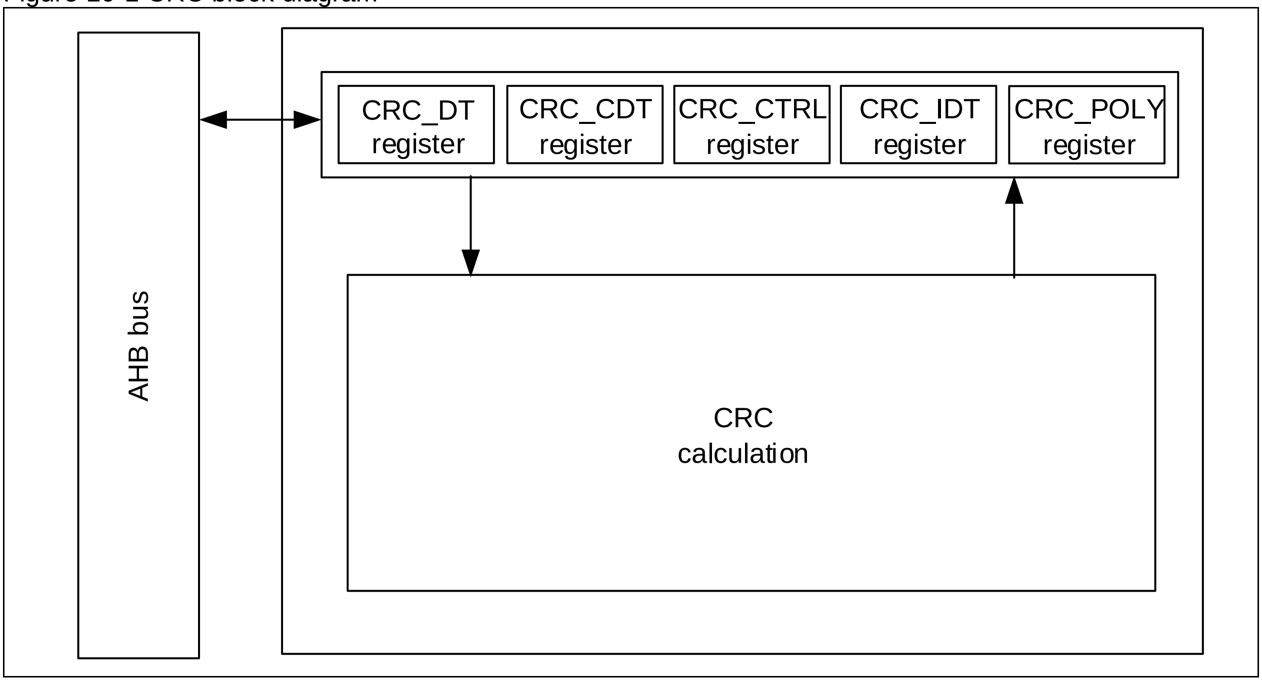
10.1 CRC introduction

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 toggle (word, REVOD=1) or input data toggle (byte, REVID=01; half-word, REVID=0; word: REVID=11). CRC calculation unit is also equipped with initialization function. After each CRC 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 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 function overview

In CRC calculation, the input data is used as the dividend and the generating polynomial as the divisor for Modulo-2 Division, and the remainder obtained is the CRC value.

CRC calculation process

- Toggle input, that is, toggle the input data according to the REVID value in the CRC_CTRL register.
- Initialize: perform XOR with the initial value set in the CRC_IDT for the first time of calculation (if it is not the first time, the initial value should be the previously calculated result).

- CRC calculation: perform Modulo-2 Division with the generating polynomial (0x4C11DB7), and the remainder obtained is the CRC value.
- Toggle output: determine whether to perform toggle (word) according to the REVOD value in the CRC_CTRL register before output.
- XOR calculation. The XOR-ed result is fixed at 0x0000 0000

CRC-32/MPEG-2 parameters:

- Generating polynomial: 0x4C11DB7,

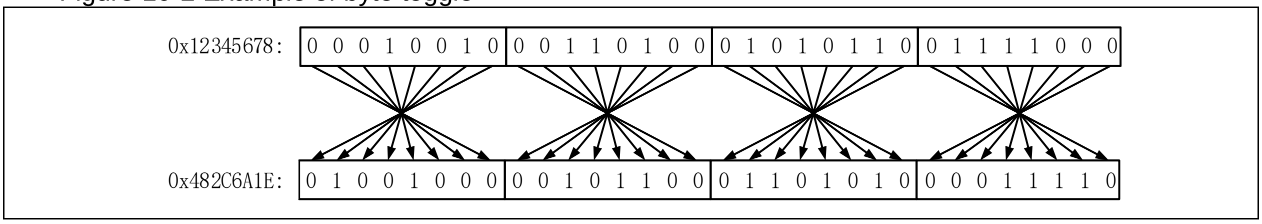
$$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$$

- Initial value is 0xFFFF FFFF, to avoid obtaining the same calculation result for 1 byte 0x00 and multibyte 0x00.
- XOR value: 0x0000 0000, indicating the CRC result does not require an additional XOR calculation.

Toggle function

- Select byte toggle, 8 bits as a group and in reverse order. As shown in the figure below, if the original data is 0x12345678, the toggled data is 0x482C6A1E.
- Select half-word toggle, 16 bits as a group and in reverse order.
- Select word toggle, 32 bits as a group and in reverse order.

Figure 10-2 Example of byte toggle



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 It is used to temporarily store 1 byte of data. 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	rw	Reset CRC calculation unit It is set by software and 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 1 Mbit/s of communication speed (enhanced edition).

11.2 I²C main features

- I²C bus
  - Master and slave modes
  - Multimaster capability
  - Standard mode (100 kHz), fast mode (400 kHz) and enhanced fast mode (1 MHz)
  - 7-bit and 10-bit address modes
  - Two 7-bit slave addresses (2 addresses, one of them can be masked)
  - Broadcast call mode
  - Programmable data setup and hold time
  - Clock stretching capability
- Support DMA transfer
- Programmable digital noise filter
- Wake up from Deepsleep mode through address match event
- Support SMBus 2.0 protocol
  - PEC generation and verification
  - Acknowledgement control for command and data
  - ARP(address resolution protocol)
  - Master capability
  - Device capability
  - SMBus reminder capability
  - Timeout detection
  - Idle detection
- PMBus

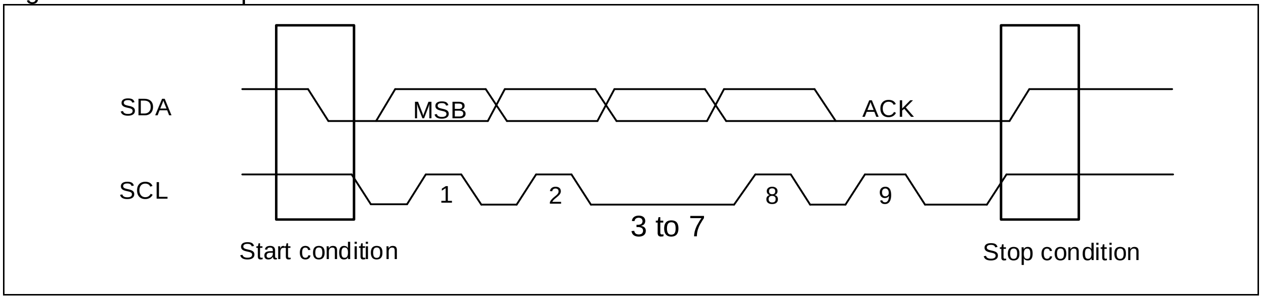
11.3 I²C function overview

I²C bus consists of a data line (SDA) and clock line (SCL). It can achieve a maximum of 100 kHz speed in standard mode, up to 400 kHz in fast mode, and 1 MHz in enhanced 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 the Start condition, and becomes idle as long as it receives the 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.

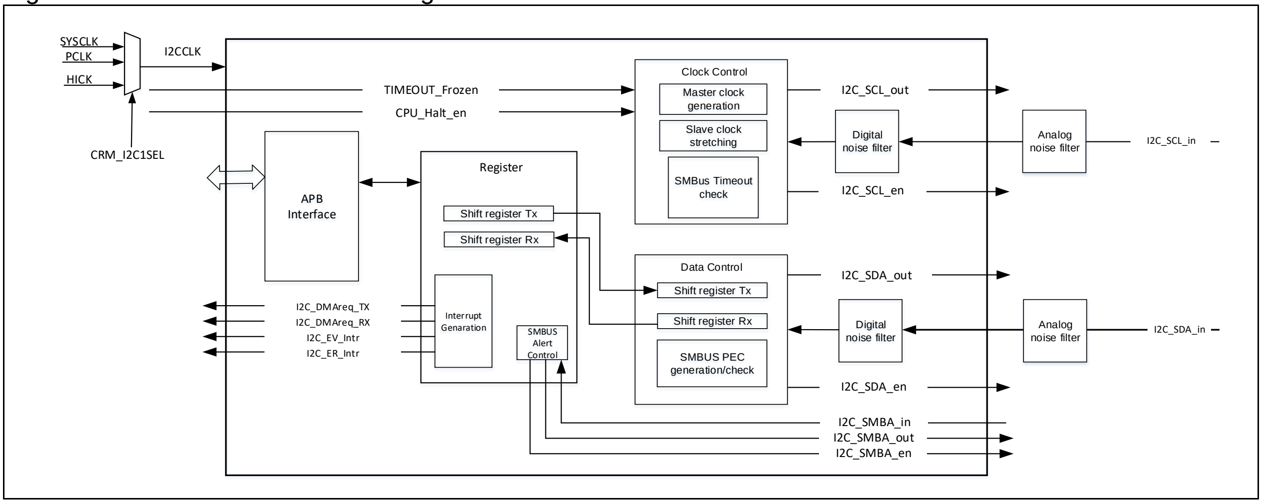
Figure 11-1 I²C bus protocol



11.4 I²C interface

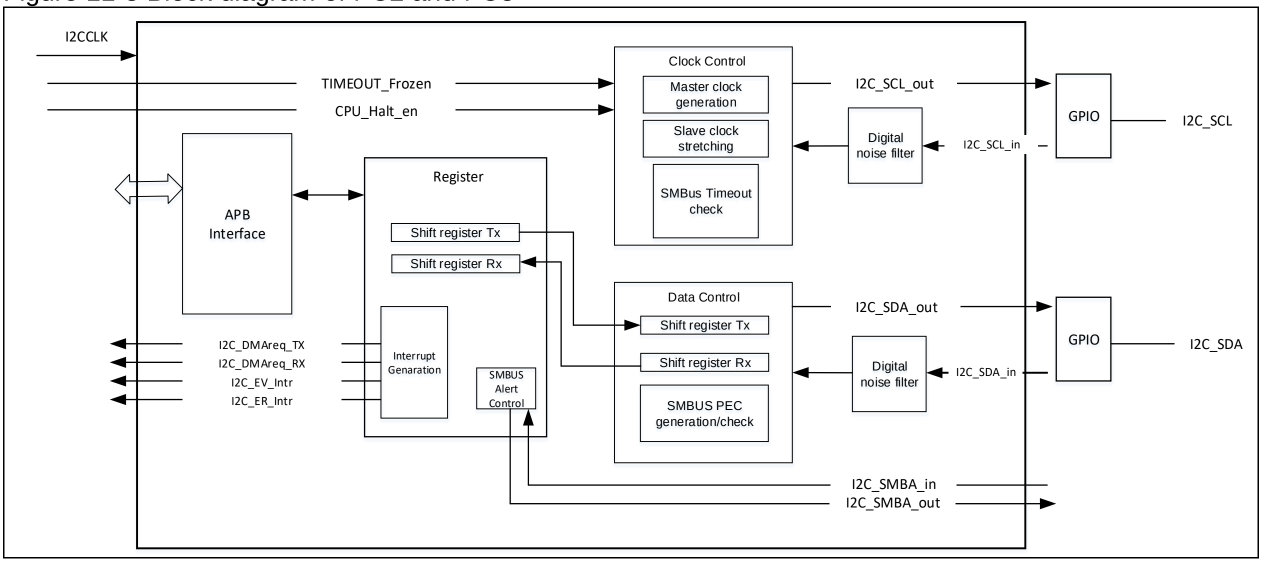
The figure below shows the block diagram of I²C interface.

Figure 11-2 I²C1 interface block diagram



For I²C1, it is possible to select SYSCLK, PCLK or HICK as clock source through the I2C1SEL bit in the CRM_PICLKS register. Waking up from Deepsleep mode is supported. The I²C1 has an analog filter able to shield noise of 50ns.

Figure 11-3 Block diagram of I²C2 and I²C3



The PCLK is used as the clock source of I²C2 and I²C3. The I²C2 and I²C3 do not support to wake up from Deepsleep mode, and have no analog filter.

1. 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 the 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).

2. 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

3. Digital filter capability

The digital filter is available on both SCL and SDA lines. It is enabled by setting the DFLT[3: 0] bit (0~15) in the I2C_CTRL1 register to reduce noise on bus on a large scale. The filter time is DLFT x t_{I2C_CLK}. The digital filter is not allowed to be altered when the I²C is enabled.

4. Address control

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

Slave address mode:

- In 7-bit mode (ADDR1MODE=0)
  - ADDR1EN=1, ADDR2EN=0 stands for a single address mode: only matches OADDR1
  - ADDR1EN=1, ADDR2EN=1 stands for dual address mode: matches OADDR1 and OADDR2
- In 10-bit mode (ADDR1MODE=1)
  - Only supports a single address mode (ADDR1EN=1, ADDR2EN=0), matches OADDR1

Slave address masking capability

The Slave address 2 (OADDR2) is maskable, which is done by setting the ADDR2MASK[2: 0].

- 0: Address bit [7:1]
- 1: Address bit [7:2]
- 2: Address bit [7:3]
- 3: Address bit [7:4]
- 4: Address bit [7:5]
- 5: Address bit [7:6]
- 6: Address bit [7]
- 7: All addresses, excluding those reserved by I²C

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 (DEVADDREN = 1).
- SMBus master default address (0b0001000x): This address is enabled for SMBus master notification protocol in SMBus master mode (HADDREN = 1).
- 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
- ADDRF=1, with SDIR indicating the transmission direction
  - When SDIR=0, slave enters receiver mode, starting receiving data.
  - When SDIR=1, slave enters transmitter mode, starting transmitting data.

5. Clock stretching capability

Clock stretching is enabled by default (STRETCH=0 in the I2C_CTRL1 register). The slave can hold the SCL line low for software operation. If the clock stretching capability is not supported by master, then the STRETCH must be set to 1 in the I2C_CTRL register. It should be noted that the clock stretching capability of I²C slave must be configured before the I²C peripherals are enabled.

Clock stretching capability enabled

I²C slave stretches the SCL clock in one of the following conditions:

- Address reception: When the address received by slave matches the local address enabled (ADDRF=1 in the I2C_STS), the SCL line is pulled down until the ADDRF is cleared by setting the ADDRC in the I2C_CLR

- Data reception: the shift register receives a new data before the data in the I2C_RXDT register has been read. In this case, the SCL line is pulled low until the data of the I2C_RXDT register is read.
- Data transmission: If no data is written when the ADDRF is cleared, TDBE= 1 in the I2C_STS, then the SCL line will be pulled down until the data is written to the I2C_TXDT.
- Data transmission: If no data is written to the I2C_TXDT after the completion of the previous data transfer, the SCL line will be pulled low until data is written to the I2C_TXDT
- Data reception: When the shift register has received another new byte before the data in the I2C_RXDT register is read, the I2C will hold the SCL bus low to wait for the software to read I2C_RXDT register
- When slave byte control mode is selected (SCTRL=1 in the I2C_CTRL1) and RLDEN=1 in the I2C_CTRL2 register, if TCRLD = 1, indicating the completion of the last data transfer, then the TCRLD will be cleared by hardware so as to release the SCL line after a non-zero value is written to the CNT bit in the I2C_CTRL2 register

Clock stretching capability disabled

The SCL clock is disabled when STRETCH=1 in the I2C_CTRL1 register, with the following conditions worth our notice:

- Address reception: The SCL clock is not stretched when the address received by slave matches the local address enabled
- Data reception: If there is data to be read in the I2C_RXDT register before the next ACK signal, an overflow will occur, and the OUF bit will also be set to 1 in the I2C_STS register
- Data transmission: If no data is written to the I2C_TXDT register after the completion of the previous data transfer, an underflow will occur, and the OUF will also be set to 1 in the I2C_STS register

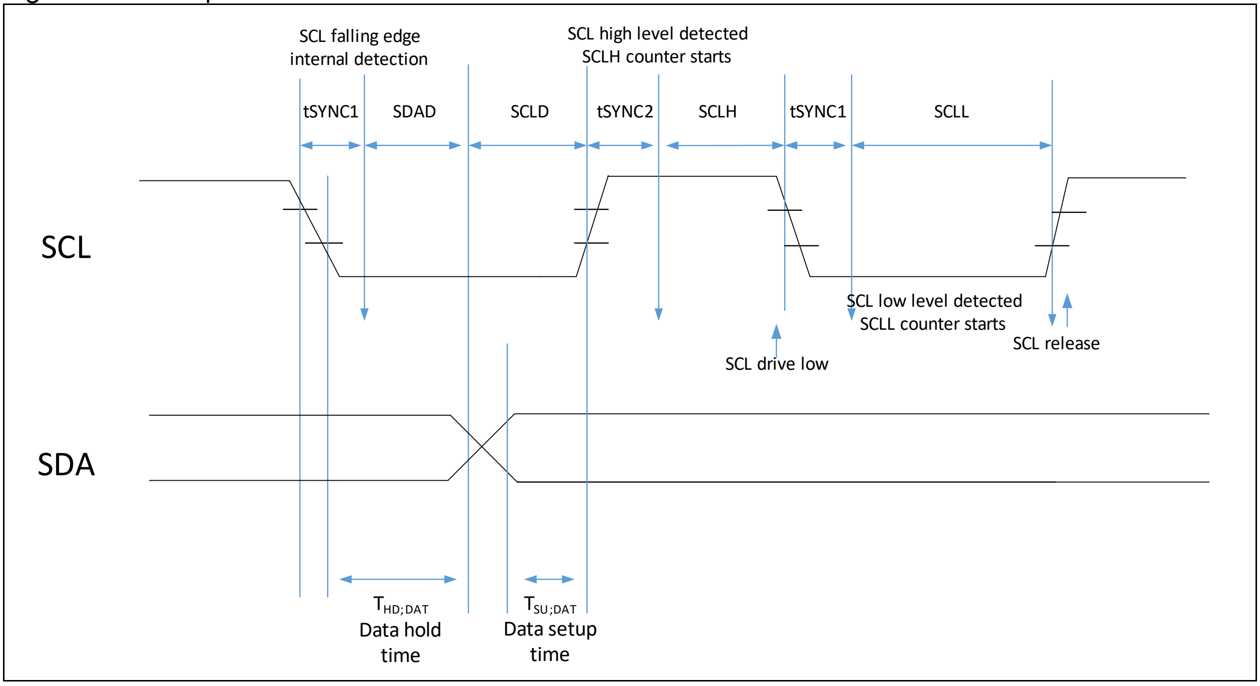
11.4.1 I²C timing control

I²C core is clocked by I2C_CLK whereas the I2C_CLK is clocked by PCLK1. The PCLK1 should be set to be less than 4/3 SCL cycles. The corresponding bits in the I2C_CLKCTR register are used for timing configuration.

— DIV[7:0]: I²C clock divider
— SDAD[3:0]: Data hold time ($t_{HD;DAT}$)
— SCLD[3:0]: Data setup time ($t_{SU;DAT}$)
— SCLH[7:0]: SCL high
— SCLL[7:0]: SCL low

Note: Timing configuration cannot be modified once the I²C is enabled.

Figure 11-4 Setup and hold time



It is possible to configure data hold time ($t_{HD;DAT}$) and data setup time ($t_{SU;DAT}$) freely by setting the DIV[7:0], SDAD[3:0] and SCLD[3:0] in the I2C_CLKCTRL register.

- Data hold time ($t_{HD;DAT}$): refers to the duration from SCL falling edge to SDA output

  $t_{HD;DAT} = t_{SDAD} + t_{SYNC}$

  t_{SDAD} = SDAD x (DIV + 1) x t_{I2C_CLK}

  t_{SYNC} = (DLFT + 3) x t_{I2C_CLK} - t_f

  t_{SYNC} consists of three parts:

  — SCL falling edge time (t_f)
  — Digital filter input latency (DLFT x t_{I2C_CLK})
  — Synchronization delay between SCL and I2C_CLK (2~3 I2C_CLK cycles)

- Data setup time ($t_{SU;DAT}$): refers to the duration from SDA output to SCL rising edge

  $t_{SU;DAT}$ = SCLD x (DIV+1) x t_{I2C_CLK} – t_r

In master mode, the width of SCL signals (high and low) can be configured freely by setting the DIV[7: 0], SCLH[7: 0] and SCLL[7: 0] in the I2C_CLKCTRL register.

SCL low: When the SCL low signal is detected, the internal SCLL counter starts counting until it reaches the SCLL value. At this point, the SCL line is released and become high.

SCL high: When the SCL high signal is detected, the internal SCLH counter starts counting. When the counter value reaches the SCLH value, the SCL line is pulled low. In the process of SCL remaining high, if it is pulled low by external bus, the internal SCLH counter will stop counting and start counting in SCL low mode, laying the foundation for clock synchronization

- SCL high signal width:

  t_{HIGH} = (SCLH + 1) x (DIV + 1) x t_{I2C_CLK}

- SCL low signal width:

  T_{Low} = (SCLL + 1) x (DIV + 1) x t_{I2C_CLK}

Table 11-1 I²C timing specifications

Parameter		Standard mode		Fast mode		Fast mode plus		SMBus	
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
f_{SCL} (kHz)	SCL clock frequency		100		400		1000		100
t_{LOW} (us)	SCL clock low	4.7		1.3		0.5		4.7	
t_{HIGH} (us)	SCL clock high	4.0		0.6		0.26		4.0	50
$t_{HD;DAT}$ (us)	Data hold time	0		0	0.9	0	0.45	300	
$t_{SU;DAT}$ (ns)	Data setup time	250		100		50		250	
tr (ns)	SCL/SDA rising edge		1000		300		120		1000
tf (ns)	SCL/SDA falling edge		300		300		120		300

11.4.2 Data transfer management

Data transfer counter is available in the I²C interface to control communication flow. It is mainly used for:

— NACK transmission: master reception mode
— STOP transmission: master reception/transmission modes
— RESTART generation: master reception/transmission modes
— ACK control: slave mode (SMBus)
— PEC transmission/reception: master/slave modes

Generally, the data transfer management counter (by setting the CNT[7:0] in the I2C_CTRL2) is applicable to master mode. It is disabled in slave mode. This counter is used only in SMBus mode for the ACK control and PEC reception of each byte by the slave. In SMBus mode, the slave enables data counter with the SCTRL bit in the I2C_CTRL2 register.

Byte control through master

The CNT[7:0] bit in the I2C_CTRL2 register is used to configure the number of bytes to be transferred, ranging from 1 to 255. If the number of data to be transferred is greater than 255, then the RLDEN bit has to be set to 1 in the I2C_CTRL2 register to enable reload mode. The following configuration processes are described in two aspects:

- ≤255 bytes, for example, the number of data to be transferred is 100 bytes
  — Step 1: Disable reload mode by setting RLDEN=0;
  — Step 2: Set CNT[7:0]=100.
- >255 bytes, for example, the number of data to be transferred is 600 bytes
  — Step 1: Enable reload mode by setting RLDEN=1;
  — Step 2: Set CNT[7:0]=255, the remaining bytes are 600-255=345 bytes;
  — Step 3: After the completion of 255-byte data transfer, the TCRLD is set in the I2C_STS register, and then set CNT[7:0]=255 for continuous transfer, and the remaining bytes are 345-255=90;
  — Step 4: After the completion of the second 255-byte data transfer, the TCRLD is set in the I2C_STS register, and then set RLDEN=0 to disable reload mode before setting CNT[7:0]=90 for continuous transfer.

There are two ways to stop the last data transfer (RLDEN=0, reload mode is disabled)

- Stop data transfer automatically (ASTOPEN=1 in the I2C_CTRL2)
  — When the number of data programmed in the CNT[7:0] bit has been fully transferred, the master will automatically send a STOP condition.
- Stop data transfer by software (ASTOPEN=0 in the I2C_CTRL2)
  — When the number of data programmed in the CNT[7:0] has been fully transferred, the TDC will be set in the I2C_STS register, and the SCL, at this point, will be pulled low, an interrupt generated if TDCIEN is enabled. In this case, it is possible to send a STOP condition by setting GENSTOP=1 in the I2C_CTRL2 register, or send a RESTART condition by setting GENSTART=1 in the I2C_CTRL2 register, before clearing TDC flag by software.

Byte control through slave

This feature is enabled by setting the SCTRL bit in the I2C_CTRL2 register so that the slave is able to control ACK/NACK signals of each byte independently.

- Proceed as below:
  - Set SCTRL=1 to enable Byte Control Through Slave
  - After the slave address is matched (ADDRF=1), enable reload mode by setting RLDEN=1, and then set CNT[7:0]=1
  - When a byte is received, the TCRLD is set in the I2C_STS register, and the slave will pull the SCL bus low between the 8th and 9th clock edges. At this point, the user can read the RXDT register and generate an ACK or NACK signal through the NACKEN bit in the I2C_CTRL2 register
  - When an NACK signal is generated, it indicates the end of communication
  - When an ACK signal is generated, the communication flow keeps going on. At this point, set CNT[7:0]=1, the TCRLD flag is cleared automatically by hardware, and the SCL bus is released for the reception of the next byte

As we know, the value in the CNT[7:0] bit is not limited to 1. If you want to receive 8 data, for example, but just want to control the ACK/NACK signals of the 8th data. Proceed as below: set CNT[7:0]=8, the slave will receive 7 consecutive data, with ACK signals sent. Once the 8th data reception is completed, the SCL bus is pulled low, and then proceed as above to select whether to send an ACK or NACK.

It should be noted that the clock stretching capability must be enabled (STRETCH=0 in the I2C_CTRL1 register) before selecting Byte Control Mode Through Slave.

Table 11-2 I^2C configuration table

Description	RLDEN	ASTOPEN	SCTRL
Master transmit/receive RESTART	0	0	×
Master transmit/receive STOP	0	1	×
Slave receive (control ACK/NACK of each byte)	1	×	1
Slave transmit/receive (ACK response to all bytes)	×	×	0

11.4.3 I^2C master communication flow

1. **I^2C clock initialization (by setting the I2C_CLKCTRL register)**
   - I^2C clock divider: DIV[7:0]
   - Data hold time ($t_{HD;DAT}$): SDAD[3:0]
   - Date setup time ($t_{SU;DAT}$): SCLD[3:0]
   - SCL high duration: SCLH[7:0]
   - SCL low duration: SCLL[7:0]

   The register can be configured by means of Artery_I2C_Timing_Configuration tool.
2. **Set the number of bytes to be transferred**
   - ≤255 bytes

     Disable reload mode by setting RLDEN=0 in the I2C_CTRL2 register

     Set CNT[7:0]=N in the I2C_CTRL2 register
   - >255 bytes

     Enable reload mode by setting RLDEN=1 in the I2C_CTRL2 register

     Set CNT[7:0]=255 in the I2C_CTRL2 register

     Remaining bytes N=N-255
3. **End of data transfer**
   - ASTOPEN=0: stop data transfer by software. After the completion of data transfer, the TDC is set in the I2C_STS register, and GENSTOP=1 or GENSTART=1 is written by software to send a STOP or START condition
   - ASTOPEN=1: data transfer is stopped automatically. A STOP condition is sent at the end of data transfer

4. **Set slave address**
   - Set slave address value (by setting the SADDR bit in the I2C_CTRL2 register)
   - Set slave address mode (by setting the ADDR10 bit in the I2C_CTRL2 register)

     ADDR10=0: 7-bit address mode

     ADDR10=1: 10-bit address mode
5. **Set transfer direction (by setting the DIR bit in the I2C_CTRL2 register)**
   - DIR=0: Master reception
   - DIR=1: Master transmission
6. **Start data transfer**

   When GENSTART=1 in the I2C_CTRL2 register, the master starts sending a START condition and slave address. After receiving the ACK from the slave, ADDRF=1 is asserted in the I2C_STS register. The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.
7. **Master transmit**
   1. I2C_TXDT data register is empty, the shift register is empty, TDIS=1 in the I2C_STS register
   2. Writing 1 to the TXDT register, and data is immediately moved to the shift register
   3. TXDT register becomes empty, TDIS=1 again
   4. Writing 2 to the TXDT register, TDIS is cleared
   5. Repeat step 2 and 3 until the data in the CNT[7:0] is sent
   6. If TCRLD=1 (reload mode) in the I2C_STS register, the following two circumstances should be noted:

      Remaining bytes N>255: write 255 to the CNT bit, N=N-255, TCRLD is cleared, and data transfer continues

      Remaining bytes N≤255: Disable reload mode (RLDEN=0), write N to the CTN bit, TCRLD is cleared, and data transfer continues
8. **Master receive**
   1. After the slave address is matched, ADDRF=1 in the I2C_STS register, clear ADDRF flag by setting ADDRC=1 in the I2C_CLR register, and then it starts sending data
   2. After the reception of data, RDBF=1, read the RXDT register will clear the RDBF automatically
   3. Repeat step 2 until the reception of data programmed in the CNT[7:0] bit
   4. If TCRLD=1 (reload mode) in the I2C_STS, the following two circumstances should be noted:

      Remaining bytes N>255: write 255 to the CNT bit, N=N-255, TCRLD is cleared, and data transfer continues

      Remaining bytes N≤255: Disable reload mode (RLDEN=0), write N to the CTN bit, TCRLD is cleared, and data transfer continues
   5. After the reception of the last data, an NACK signal will be sent by master
9. **Stop condition**
   - STOP condition generation:

     ASTOPEN=0: TDC=1 in the I2C_STS register, set GENSTOP=1 to generate a STOP condition

     ASTOPEN=1: A STOP condition is generated automatically
   - Wait for the generation of a STOP condition, when a STOP condition is generated, STOPF=1 is asserted in the I2C_STS register. The STOPF flag can be cleared by setting STOPC=1 in the I2C_CLR register, and then transfer stops.

When the host receives an NACK signal during transmission, then ACKFAIL is set in the I2C_STS register, and a STOP condition is sent to stop communication, whatever mode (either ASTOPEN=0 or ASTOPEN=1).

Master transmitter

Figure 11-5 I²C master transmission flow

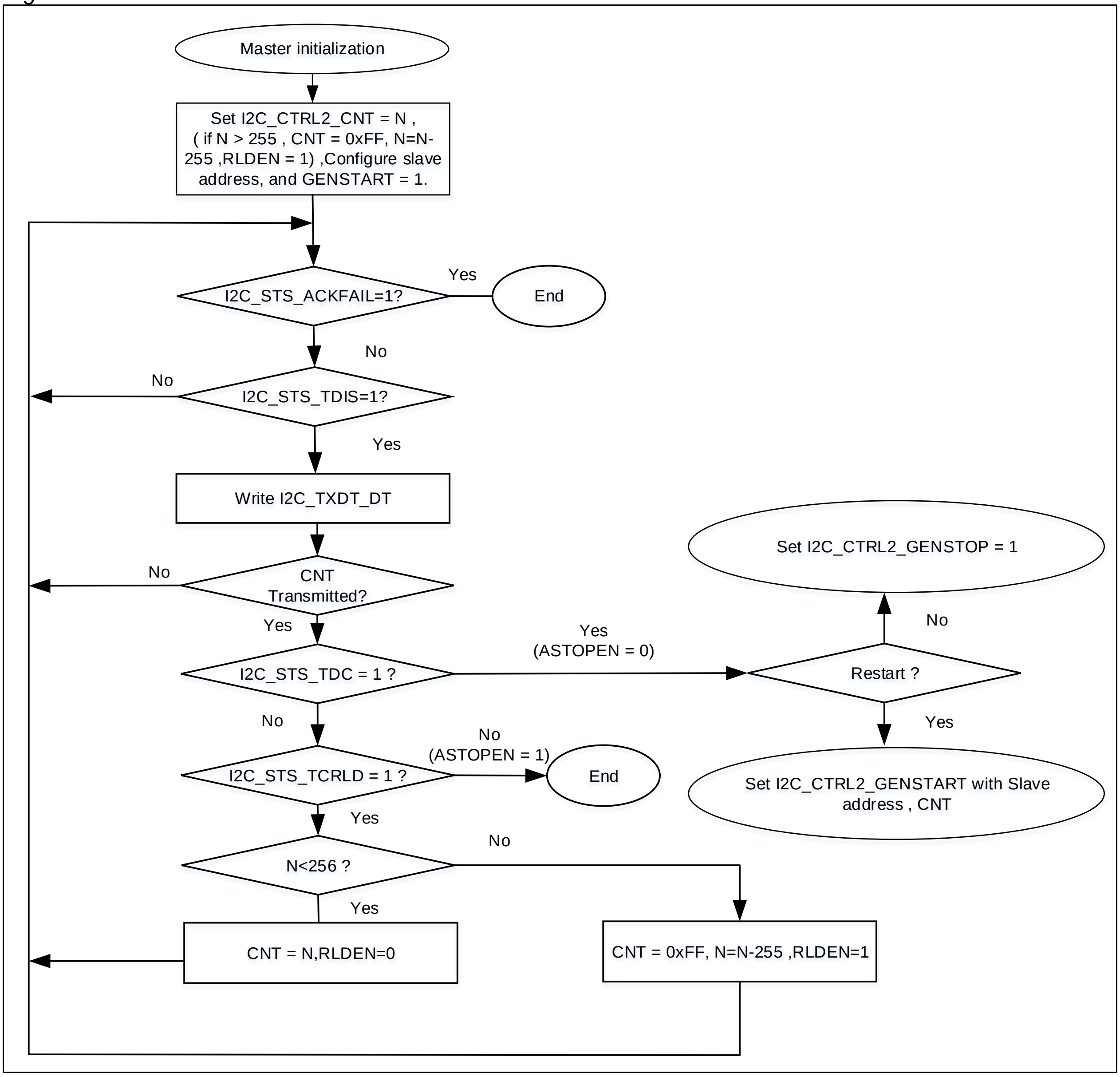
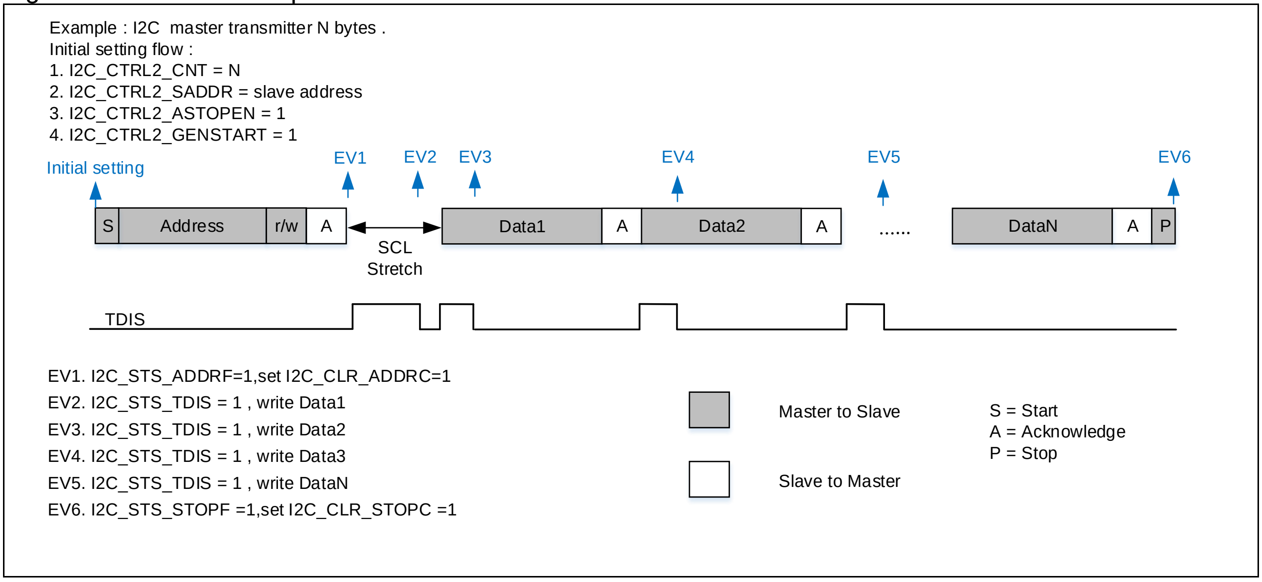


Figure 11-6 Transfer sequence of I²C master transmitter



Master receiver

Figure 11-7 I²C master receive flow

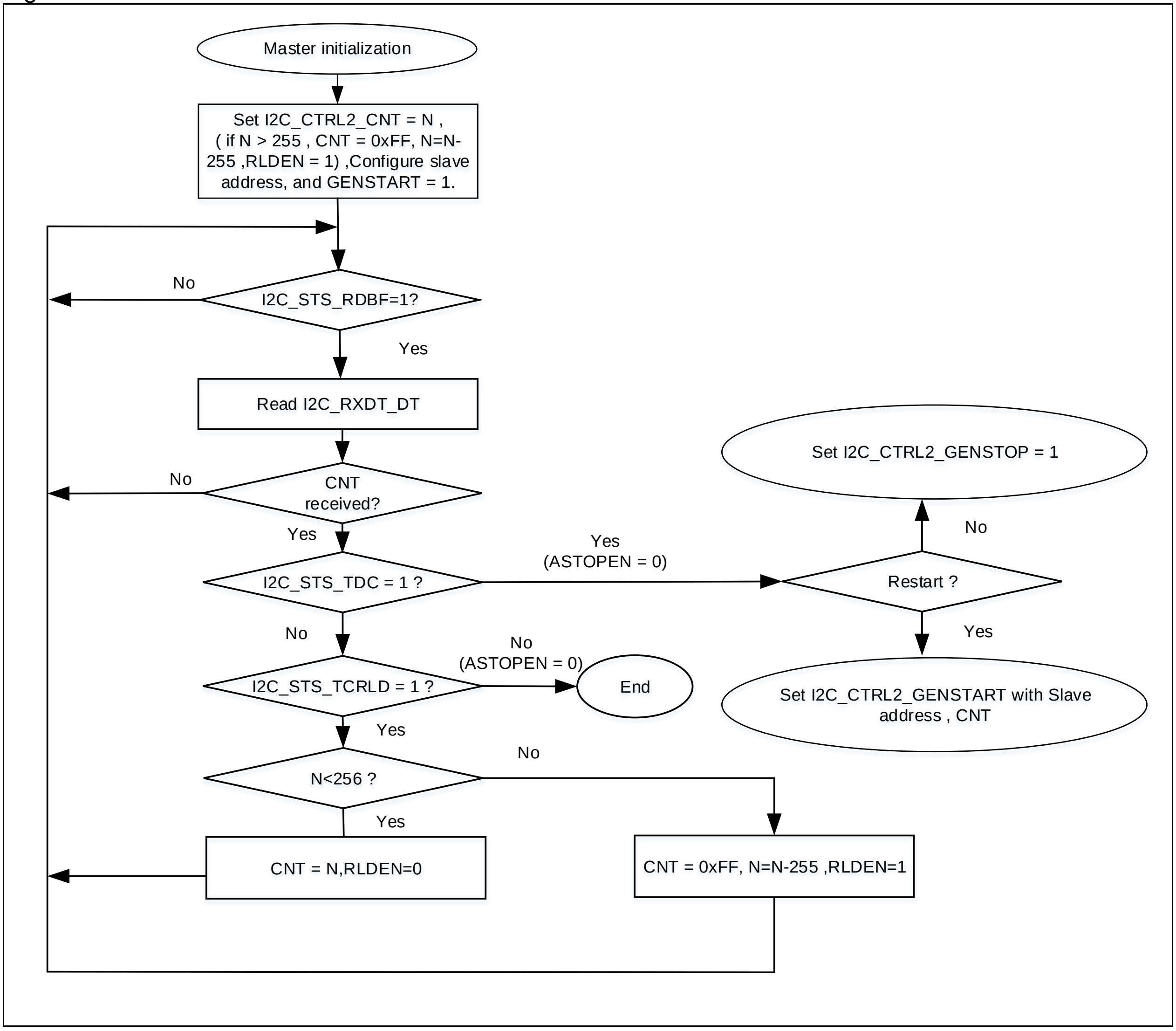
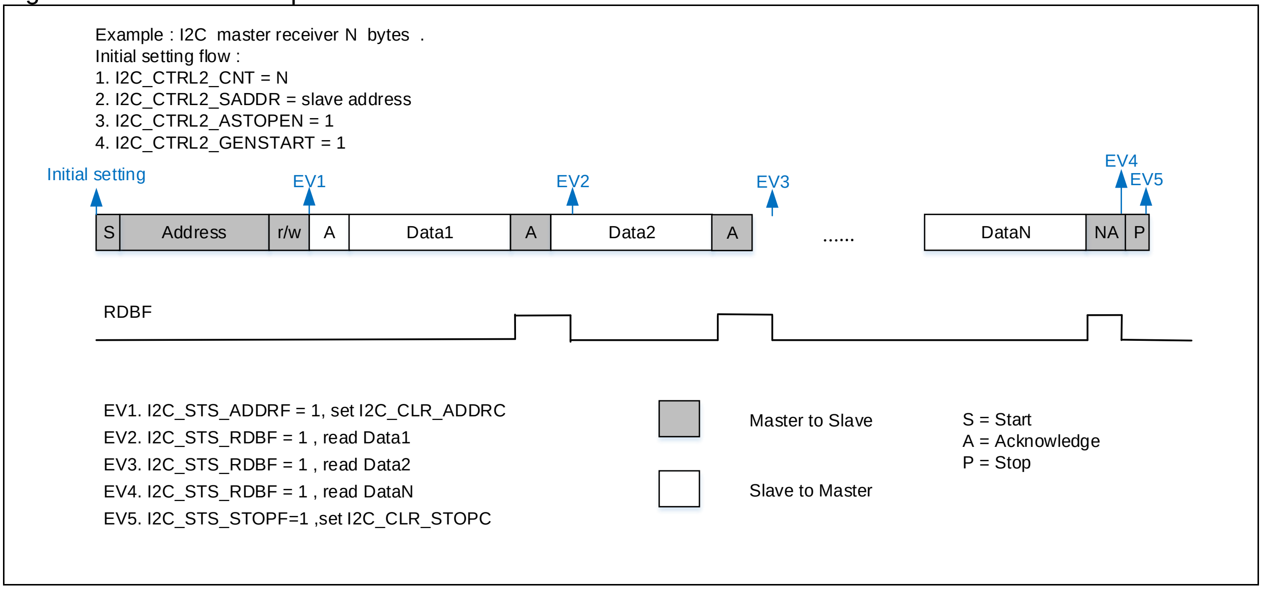


Figure 11-8 Transfer sequence of I²C master receiver



Master special transfer sequence

In 10-bit addressing mode, the READH10 bit of the I2C_CTRL2 register is used to generate a special timing. When READH10=1, the master sends data to the slave before read access to the slave, as shown in the figure below:

Operating method:

When ASTOPEN=0, data is transferred from the master to the slave. At the end of data transfer, READH10=0 is asserted, and then the master starts receiving data from the slave.

Figure 11-9 10-bit address read access when READH10=1

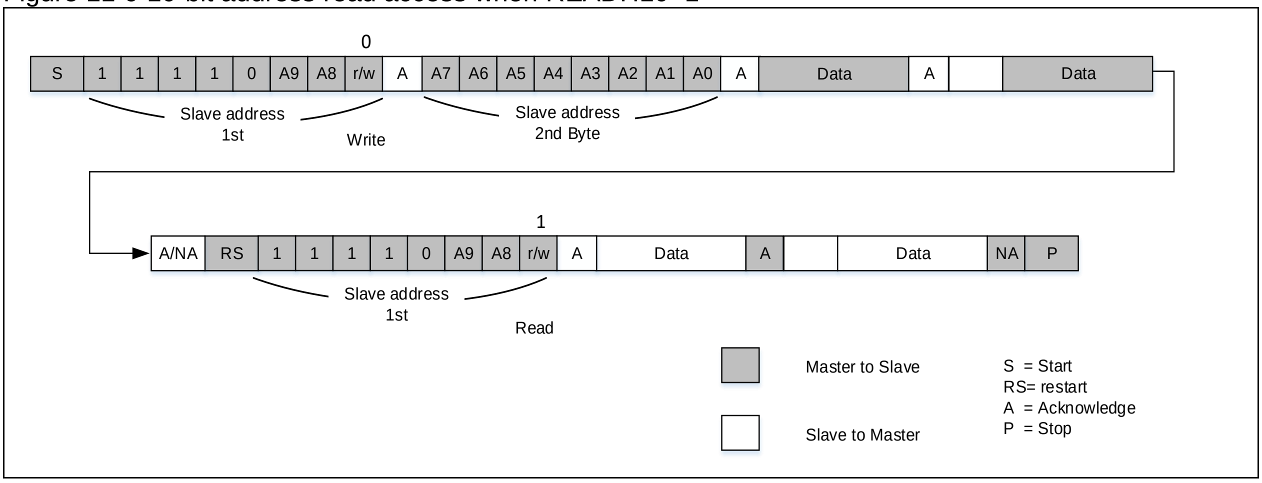
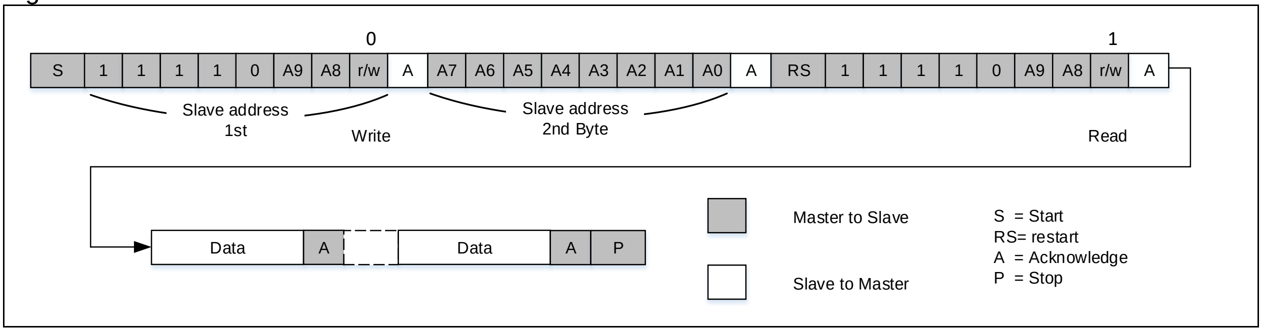


Figure 11-10 10-bit address read access when READH10=0



11.4.4 I²C slave communication flow

1. **I²C clock initialization (by setting the I2C_CLKCTRL register)**
   - I²C clock division: DIV[7:0]
   - Data hold time (tHD;DAT): SDAD[3:0]
   - Data setup time (tSU;DAT): SCLD[3:0]

   The register can be configured by means of Artery_I2C_Timing_Configuration tool.

2. **Set local address 1**
   - Set address mode:

     7-bit address: by setting ADDR1MODE=0 in the I2C_OADDR1 register

     10-bit address: by setting ADDR1MODE=1 in the I2C_OADDR1 register
   - Set address 1: by setting the ADDR1 bit in the I2C_OADDR1 register
   - Enable address 1: by setting ADDR1EN=1 in the I2C_OADDR1 register

3. **Set local address 2**
   - Set address 2: by setting the ADDR2 bit in the I2C_OADDR2 register
   - Set address 2 mask bit: by setting the ADDR2MASK bit in the I2C_OADDR2 register
   - Enable address 2: by setting ADDR2EN=1 in the I2C_OADDR2 register

   Note: Only 7-bit address mode is available in the address 2 mode. The ADDR2MASK bit is used to mask some address bits freely so that the slave can respond to some specific addresses. Refer to Section 14.2 for more information about the ADDR2MASK bit.

   In the case of using only one address, only address 1 needs to be configured, without the need of address 2 mode.

4. **Wait for address matching**

   When the local address is received, the ADDRF bit is set in the I2C_STS register. The data transfer direction can be obtained by read access to the SDIR bit in the I2C_STS register. When SDIR=0, it indicates that the slave is receiving data, whereas SDIR=1 indicates that the slave is sending data. The ADDR[6:0] bit of the I2C_STS register indicates what kind of address has been received, which is particularly helpful in the case when the dual address mode is used and the address 2 mode mask bit is set.

   Data transfer starts when the ADDRF is cleared by setting ADDRC=1 of the I2C_CLR register

5. **Data transfer (slave transmission, clock stretching enabled, STRETCH=0)**

   After address matching:
   1. I2C_TXDT data register becomes empty, the shift register becomes empty, and TDIS=1 in the I2C_STS register
   2. Data is then transferred to the shift register after writing 1 to the TXDT register
   3. The TXDT register then becomes empty, and the TDIS is set again
   4. TDIS is cleared by writing 2 to the TXDT register
   5. Repeat step 3 and 4 until the completion of data transfer
   6. Wait for the generation of an NACK signal. Once received, the ACKFAILF is set in the I2C_STS register. The ACKFAILF flag is cleared by writing 1 to the ACKFAILC
   7. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS

register. At the end of data transfer, the STOPF is cleared by writing 1 to the STOPC, transmission ends.

In the case of the clock stretching being disabled (STRETCH=1), if data has not yet been written to the TXDT register before the transmission of the first bit of the to-be-transferred data (that is, before the generation of SDA edge), an underrun error may occur, and the OUF bit is set in the I2C_STS register, sending 0xFF to the bus.

In order to write data in time, data must be written to the DT register first before communication, in two different ways:

— Write operation through software: Clear the TXDT register by setting TDBE=1 through software, and then write the first data to the TXDT register, the TDBE is cleared
— Write operation through interrupts or DMA: Clear the TXDT register by setting TDBE=1 through software, then set the TDIS bit to generate a TDIS event, which generates an interrupt or DMA request. At this point, data is written to the TXDT register using DMA or interrupt functions

6. Data transfer (slave receive, clock stretching enabled, STRETCH=0)

After address matching:

1. I2C_RXDT register becomes empty, the shift register becomes empty, and RDBF=0 in the I2C_STS register
2. Upon the receipt of data, RDBF=1; The RDBF is cleared by read operation to the RXDT register
3. Repeat step 2 until the completion of all data transfer
4. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. The STOPF can be cleared by writing 1 to the STOPC bit in the I2C_CLR register, transfer ends.

In slave receive mode, the slave byte control mode (SCTRL=1) can be used for data reception. This mode allows to control ACK/NACK signals of each byte received. This mode is typically available in SMBus protocol. Refer to 11.4.2 Data transfer management for more information about this mode.

Note that the slave must read the received data in the case of the clock stretching being disabled (STRETCH=1). If one-byte data has been received and data is not read yet before the end of the next data reception, an overrun error occurs, setting the OUF bit in the I2C_STS register, and sending NCAK.

An interrupt will be generated if the corresponding interrupt enable bit is enabled. For more information about interrupt generation, refer to the interrupt chapter.

Slave transmitter

Figure 11-11 I²C slave transmission flow

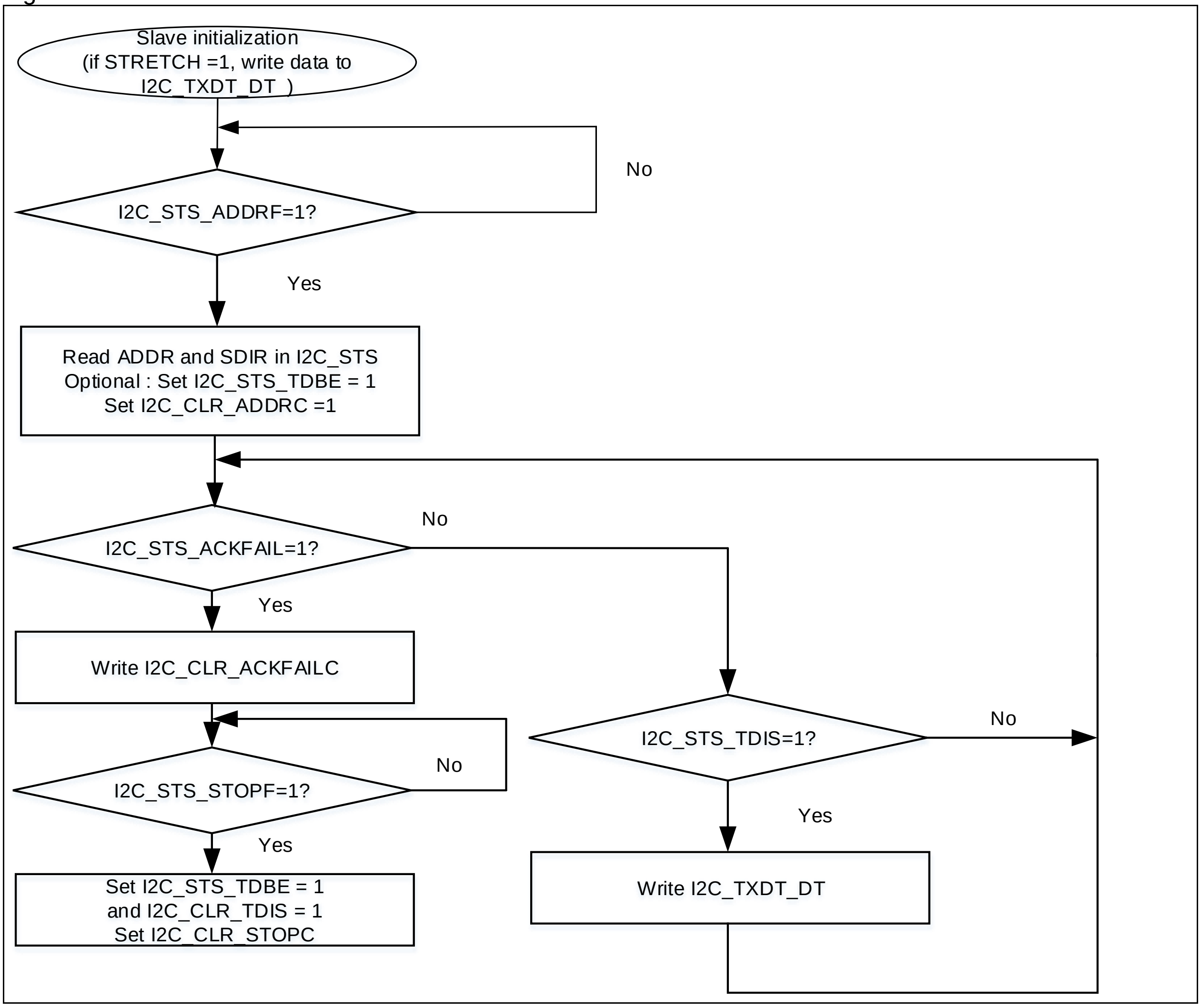
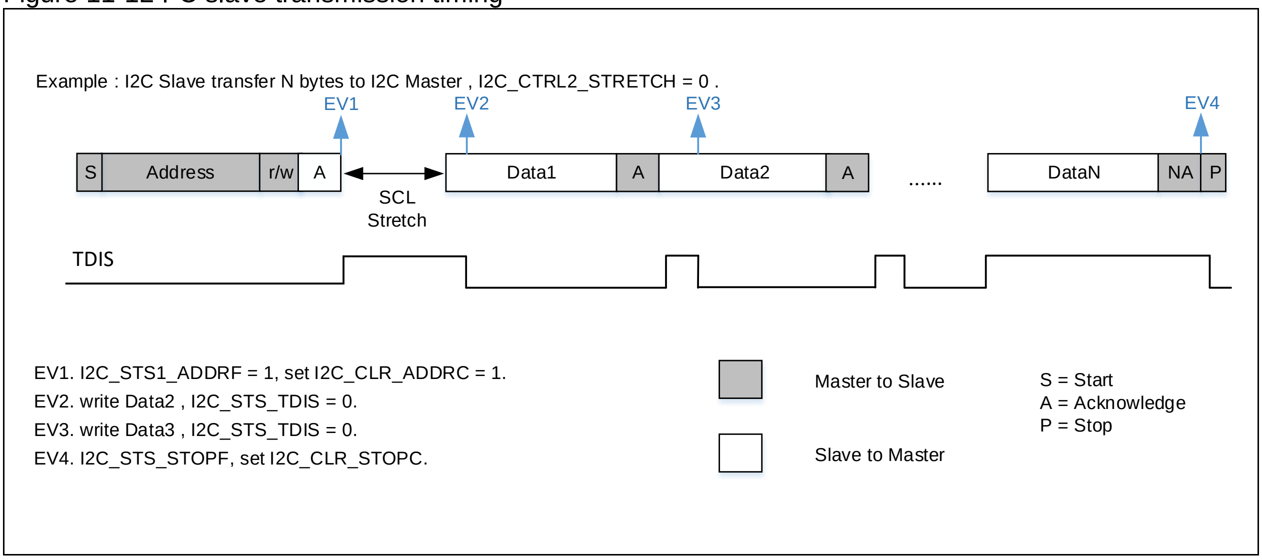


Figure 11-12 I²C slave transmission timing



Slave receiver

Figure 11-13 I²C slave receive flow

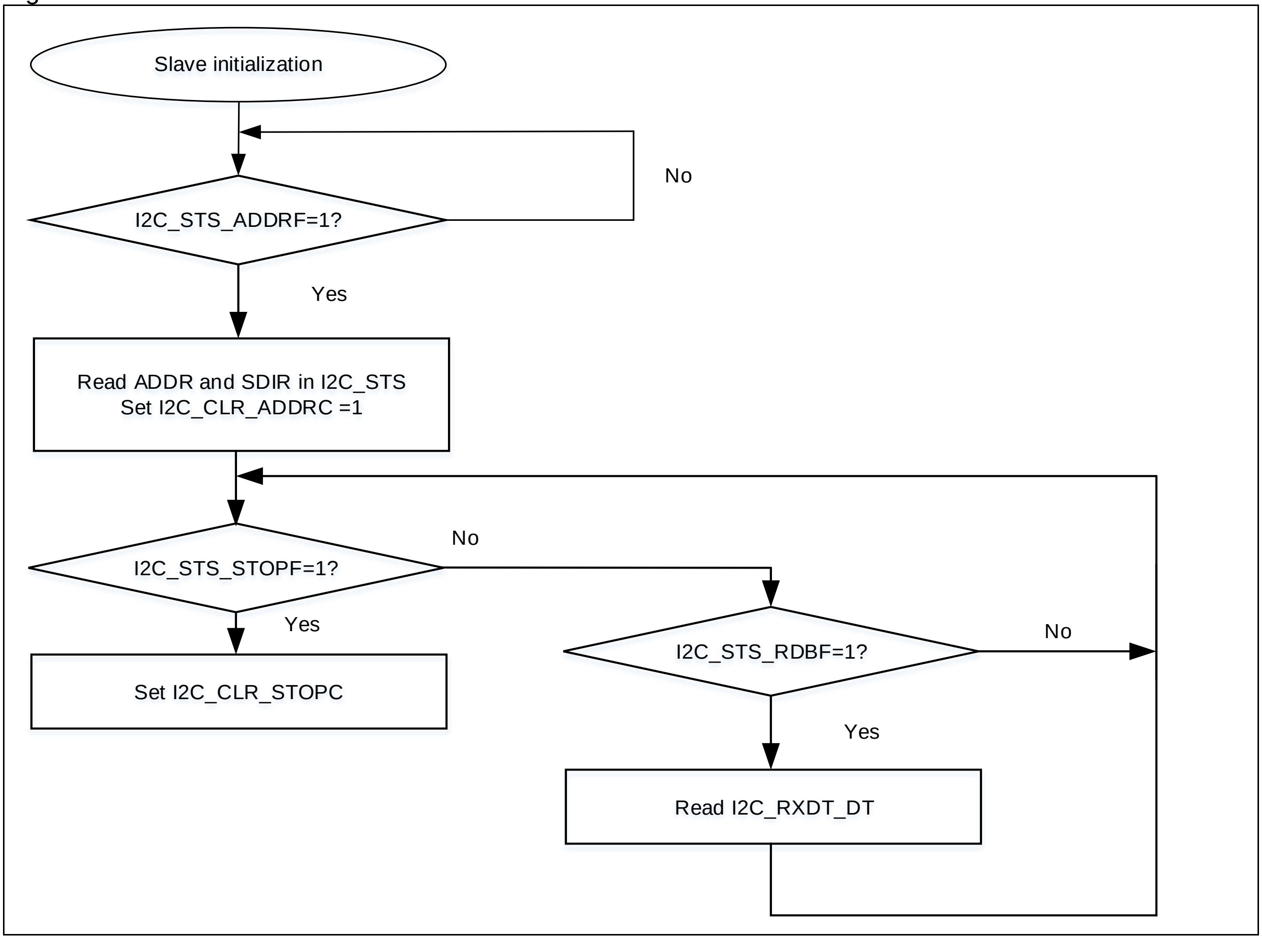
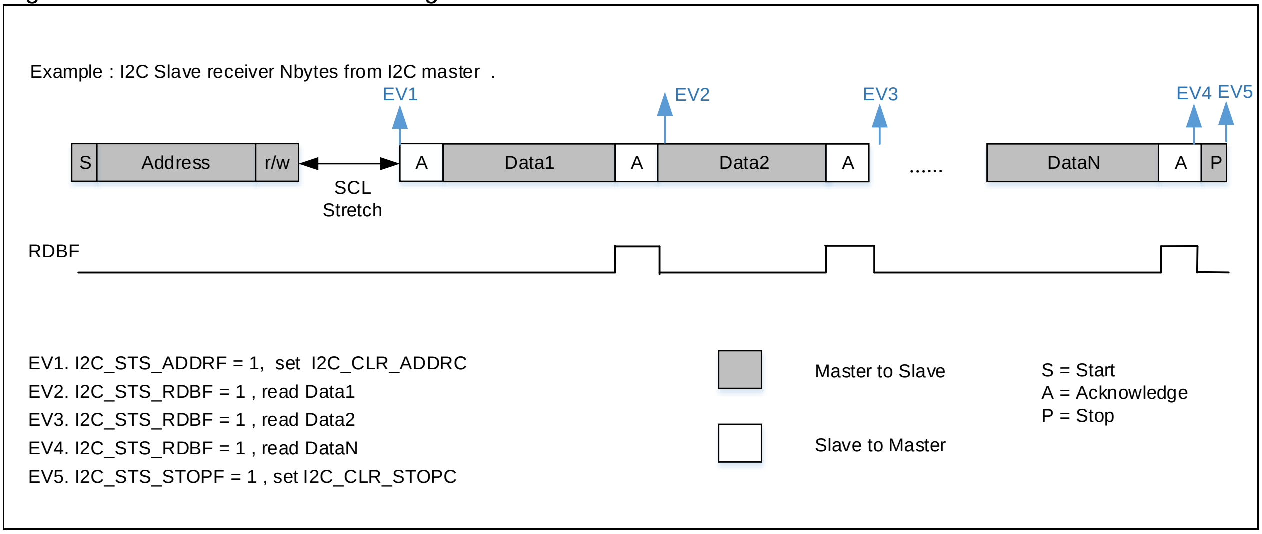


Figure 11-14 I²C slave receive timing



11.4.5 SMBus

The System Management Bus (SMBus) is a two-wire interface through which various devices can communicate with each other. It is based on I^2C. 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.

Difference between SMBus and I^2C

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 I^2C.
2. SMBus transmission speed ranges from 10 kHz to 100 kHz. In contrast, I^2C 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, but there is no limit for I^2C in this regard.

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 DEVADDREN bit in the I2C_CTRL1 register can enable the I^2C 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.

In host mode (HADDREN =1), the I^2C 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. Wait until the host gets the slave addresses through ARA
3. Report its own address, but it continues to wait if the arbitration is lost
4. 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 in the I2C_CTRL1 register to check address and data.

PEC transfer:

- Host: PEC transfer is enabled by setting PECTEN=1 in the I2C_CTRL2 register. The host sends a PEC as soon as the number of data transfer reaches N-1 (CNT=N)
- Slave: PEC transfer is enabled by setting PECTEN=1 in the I2C_CTRL2 register. When the number of data transfer reaches N-1 (CNT=N), the slave will consider the Nth data as a PEC and check it. A NACK will be sent if the PEC checking result is not correct, setting the PECERR flag in the I2C_STS register. In case of slave transmission mode, a NACK must follow the PEC whatever the checking result.

SMBus timeout

The SMBus protocol specifies three timeout detection modes:

- Low level timeout ($t_{TIMEOUT}$): The time duration when the SCL is kept low in a single mode (taking into account master/slave device, however actively or passively pulled low)
- Cumulative timeout for a slave device at low level ($t_{LOW:SEXT}$): The cumulative time duration when the SCL is pulled low by a slave device during the period from a START condition to a STOP condition
- Cumulative timeout for a master device at low level ($t_{LOW:MEXT}$): The cumulative time duration when the SCL is pulled low by a master device during the period from the ACK of the last byte to the 8th bit of the next byte (a single byte)

It should be noted that both $t_{LOW:SEXT}$ and $t_{LOW:SEXT}$ only deal with the time when they set themselves low level, excluding the time when they are pulled low by external sources. In contrast, both of these cases are considered in the calculation of $t_{TIMEOUT}$.

Table 11-3 SMBus timeout specification

Type of timeout	Min	Max	Unit
$t_{TIMEOUT}$	25	35	ms
$t_{LOW:SEXT}$	-	25	ms
$t_{LOW:MEXT}$	-	10	ms

The I²C peripherals embeds two counters for timeout detection, which can be configured through the I2C_TIMEOUT register. When a timeout event occurs, the TMOUT is set in the I2C_STS register. The TMOUT bit can be cleared by writing 1 to the TMOUTC bit in the I2C_CLR register

- EXTTIME: This is used to the cumulative timeout detection for master/slave devices at low level Timeout duration=(EXTTIME + 1) x 2048 x T_{I2C_CLK}
- TOTIME: This is used for clock level timeout detection, selected through the TOMODE bit.

  TOMODE=0: Low level timeout detection, timeout duration=(TOTIME + 1) x 2048 x T_{I2C_CLK}

  TOMODE=1: High level timeout detection, timeout duration=(TOTIME + 1) x 4 x T_{I2C_CLK}

Table 11-4 SMBus timeout detection configuration

Type of timeout	Other configuration	Enable bit	Timeout calculation
$t_{TIMEOUT}$	TOMODE=0	TOEN=1	(TOTIME + 1) x 2048 x T_{I2C_CLK}
$t_{LOW:SEXT}$	-	EXTEN=1	(EXTTIME + 1) x 2048 x T_{I2C_CLK}
$t_{LOW:MEXT}$	-	EXTEN=1	(EXTTIME + 1) x 2048 x T_{I2C_CLK}

Slave receive byte control

In slave receive mode, the slave receive byte control mode (SCTRL=1) can be used to control ACK/NACK signals of each received byte. Refer to the 11.4.2 Data transfer management for more information.

Table 11-5 SMBus mode configuration

Transfer mode	PECEN	PECTEN	RLDEN	ASTOPEN	SCTRL
Master transmit/receive +STOP	1	1	0	1	-
Master transmit/receive +RESTART	1	1	0	0	-
Slave receive	1	1	1	-	1
Slave transmit	1	1	0	-	-

How to use the interface in SMBus mode

1. Set SMBus default address acknowledgement:

   HADDREN=1: Master default address acknowledged (0b0001000x)

   DEVADDREN=1: Device default address acknowledged (0b1100001x)
2. Configure PEC
3. Slave receive byte control mode can be enabled (with SCTRL bit in the I2C_CTRL1) in slave mode, if necessary
4. Other configurations follow the I^2C

However, the detailed SMBus protocol implementation should be handled by software, since the I^2C interface is only enabled to recognize the addresses of SMBus protocols.

11.4.6 SMBus master communication flow

The SMBus is similar to the I^2C in terms of master communication flow.

1. **I^2C clock initialization (by setting the I2C_CLKCTRL register)**
   - I^2C clock divider: DIV[7:0]
   - Data hold time ($t_{HD;DAT}$): SDAD[3:0]
   - Data setup time ($t_{SU;DAT}$): SCLD[3:0]
   - SCL high duration: SCLH[7:0]
   - SCL low duration: SCLL[7:0]

   The register can be configured by means of Artery_I2C_Timing_Configuration tool.
2. **SMBus-related initialization**
   - Select SMBus host: host default address acknowledged (0b0001000x) by setting HADDREN=1
   - Enable PEC calculation: set PECEN=1 in the I2C_CTRL register
   - Enable PEC transfer: set PECTEN=1 in the I2C_CTRL2 register
3. **Set the number of bytes to be transferred**
   - Disable reload mode by setting RLDEN=0 in the I2C_CTRL2 register
   - Set CNT[7:0]=N in the I2C_CTRL2 register

   The number of bytes to be transferred is <255 in SMBus mode at one time.
4. **End of data transfer**
   - ASTOPEN=0: stop data transfer by software. After the completion of data transfer, the TDC is set in the I2C_STS register, and GENSTOP=1 or GENSTART=1 is written by software to send a STOP or START condition
   - ASTOPEN=1: data transfer is stopped automatically. A STOP condition is sent at the end of data transfer
5. **Set slave address**
   - Set slave address value (by setting the SADDR bit in the I2C_CTRL2 register)
   - Set 7-bit slave address mode (by setting the ADDR10=0 in the I2C_CTRL2 register)
6. **Set transfer direction (by setting the DIR bit in the I2C_CTRL2 register)**
   - DIR=0: Master reception
   - DIR=1: Master transmission
7. **Start data transfer**

   In case of GENSTART=1 in the I2C_CTRL2 register, the master starts sending a START condition and slave address. After receiving the ACK from the slave, ADDRF=1 is asserted in the I2C_STS register. The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.

8. **Master transmit**
   1. I2C_TXDT data register is empty, the shift register is empty, TDIS=1 in the I2C_STS register
   2. Writing 1 to the TXDT register, and data is immediately moved to the shift register
   3. TXDT register becomes empty, TDIS=1 again
   4. Writing 2 to the TXDT register, TDIS is cleared
   5. Repeat step 2 and 3 until the specified data (N-1) is sent
   6. The master will automatically transmit the Nth data, that is, PEC.
9. **Master receive**
   1. After the reception of data, RDBF=1, read the RXDT register will clear the RDBF automatically
   2. Repeat step 1 until the reception of the specified data (N). The Nth data is set as PEC. A NACK is automatically sent after the receipt of the Nth data (PEC) whatever the PEC result.
10. **STOP condition**
   - STOP condition generation:

     ASTOPEN=0: TDC=1 in the I2C_STS register, set GENSTOP=1 to generate a STOP condition

     ASTOPEN=1: A STOP condition is generated automatically
   - Wait for the generation of a STOP condition, when a STOP condition is generated, STOPF=1 is asserted in the I2C_STS register. The STOPF flag can be cleared by setting STOPC=1 in the I2C_CLR register, and then transfer stops

SMBus master transmitter

Figure 11-15 SMBus master transmission flow

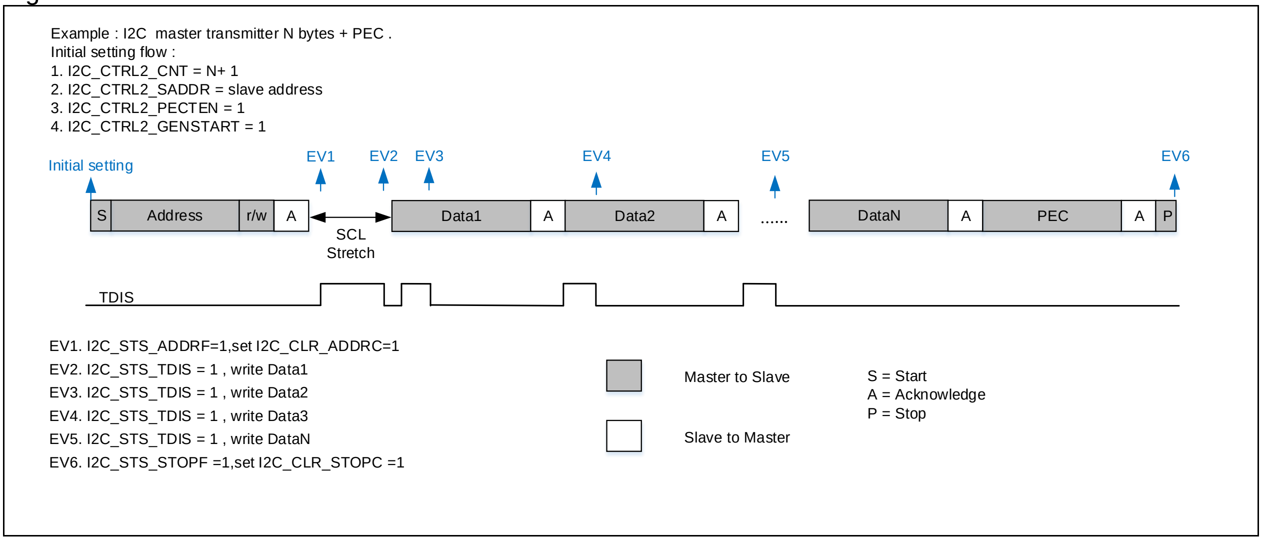
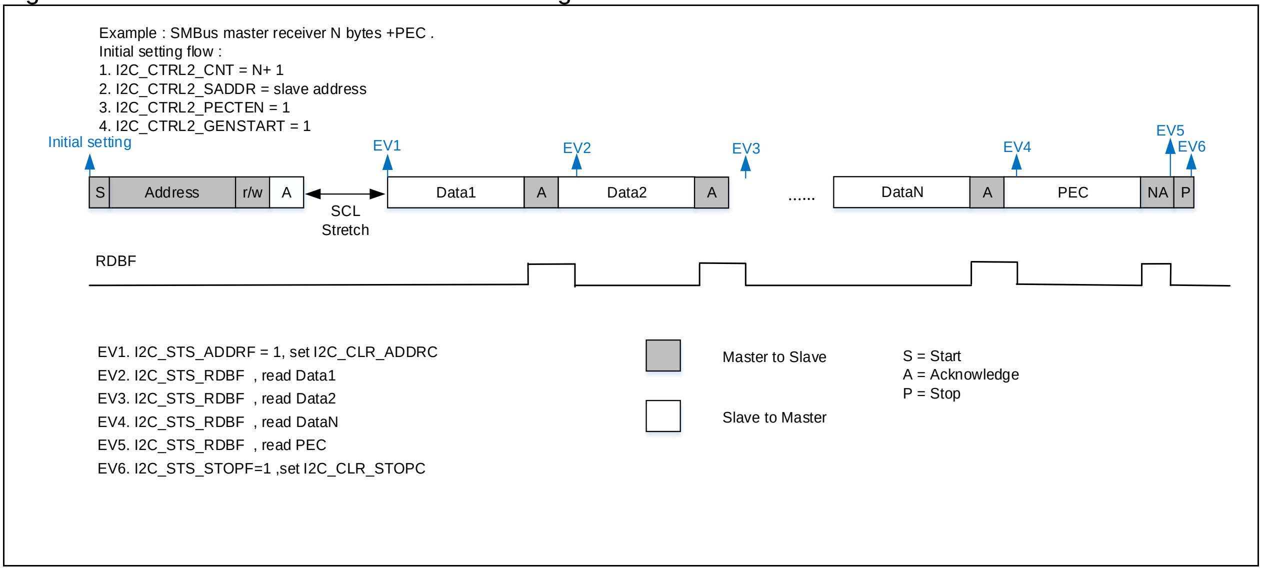


Figure 11-16 SMBus master transmission timing



SMBus master receiver

Figure 11-17 SMBus master receive flow

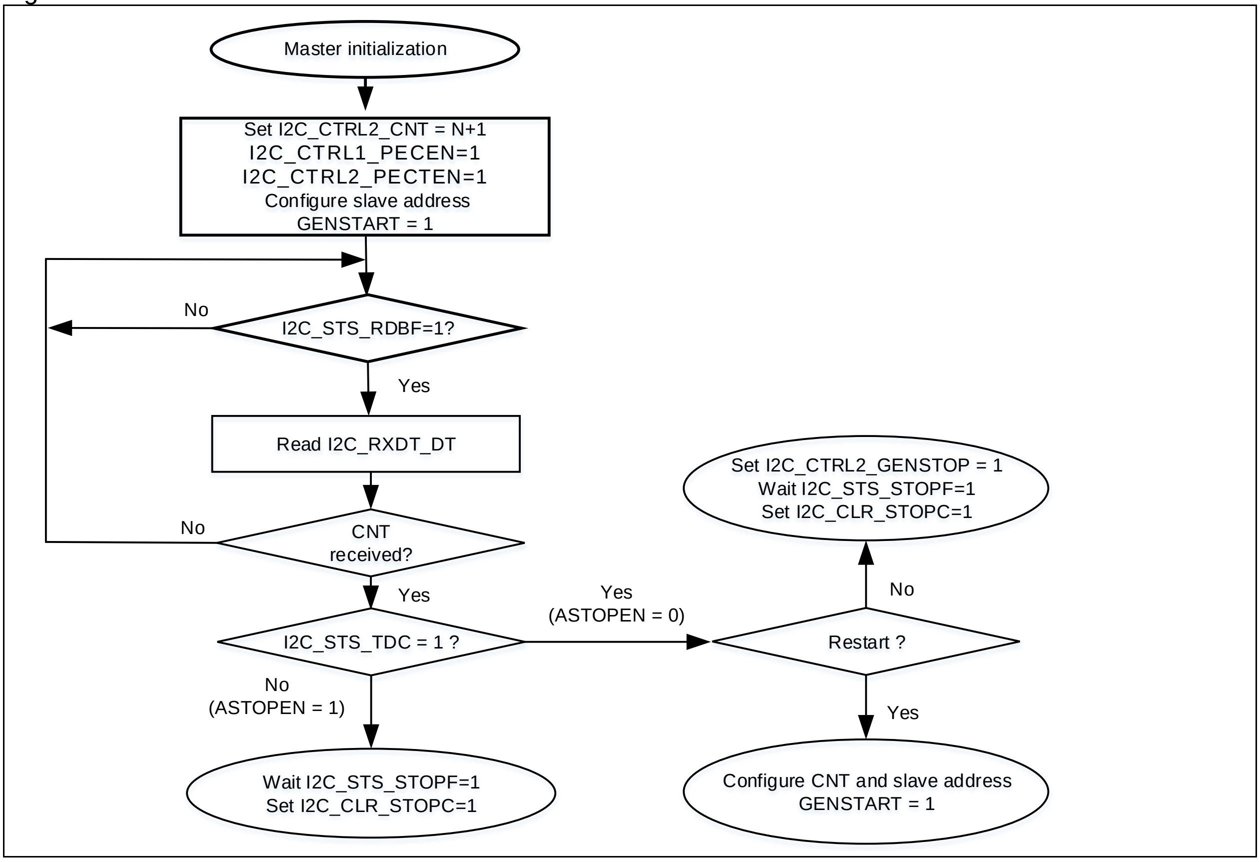
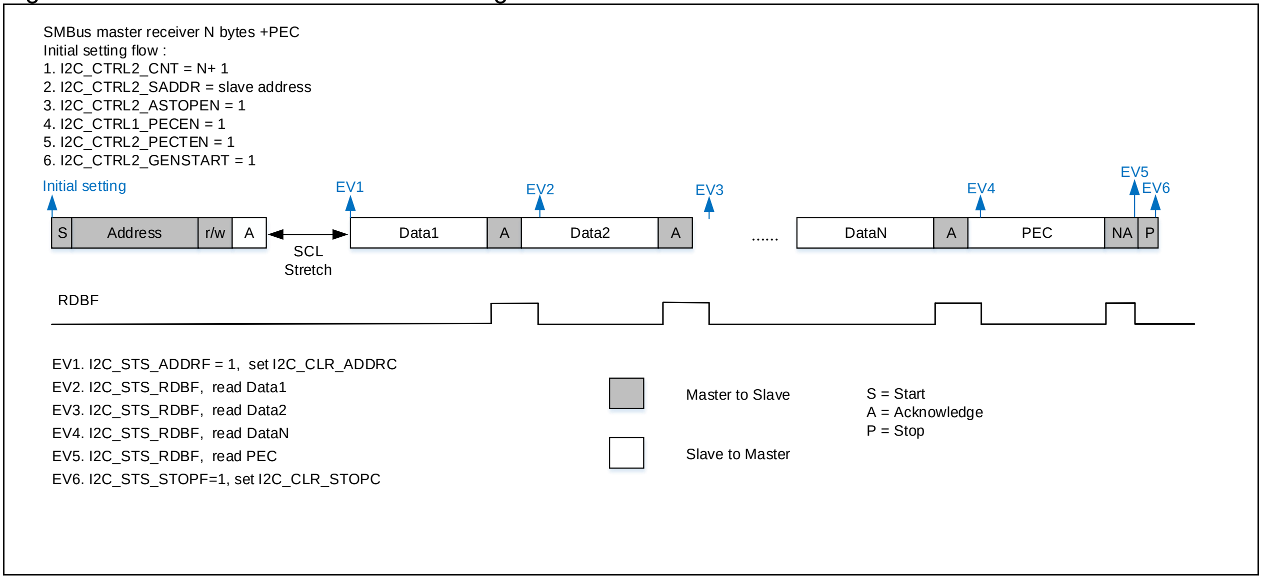


Figure 11-18 SMBus master receive timing



11.4.7 SMBus slave communication flow

The SMBus is similar to the I²C in terms of slave communication flow.

1. **I²C clock initialization (by setting the I2C_CLKCTRL register)**
   - I²C clock division: DIV[7:0]
   - Data hold time (tHD;DAT): SDAD[3:0]
   - data setup time (tSU;DAT): SCLD[3:0]

   The register can be configured by means of Artery_I2C_Timing_Configuration tool.
2. **Set local address**
   - Set 7-bit address mode: by setting ADDR1MODE = 0 in the I2C_OADDR register
   - Set address 1: by setting the ADDR1 bit in the I2C_OADDR1 register
   - Enable address 1: by setting ADDR1EN=1 in the I2C_OADDR1 register
3. **SMBus-related initialization**
   - Select SMBus host: device default address acknowledged (0b1100001x) by setting DEVADDREN=1
   - Enable PEC calculation: Set PECEN=1 in the I2C_CTRL1 register
   - Set slave byte control mode:

     Slave transmit: disable byte control mode by setting SCTRL=0 in the I2C_CTRL1 register

     Slave receive: enable byte control mode by setting SCTRL=1 in the I2C_CTRL1 register
4. **Wait for address matching**

   When the local address is received, the ADDRF bit is set in the I2C_STS register. The data transfer direction can be obtained by read access to the SDIR bit in the I2C_STS register. When SDIR=0, it indicates that the slave is receiving data, whereas SDIR=1 indicates that the slave is sending data. The ADDR[6:0 bit of the I2C_STS register indicates what kind of address has been received, which is particularly helpful in the case when the dual address mode is used and the address 2 mode mask bit is set.

   Enable PEC transfer: by setting PECTEN=1 in the I2C_CTRL2 register

   Set the number of data to be transferred:
   - Slave transmit: by setting CNT=N in the I2C_CTRL2 register
   - Slave receive: by setting CNT=1 in the I2C_CTRL2 register

   Set reload mode:
   - Slave transmit: by setting RLDEN=0 in the I2C_CTRL2 register

— Slave receive: by setting RLDEN=1 in the I2C_CTRL2 register

The ADDRF flag can be cleared by setting ADDRC=1 in the I2C_CLR register, and then data transfer starts.

5. Data transfer (slave transmission, clock stretching enabled, STRETCH=0)

After address matching:

1. I2C_TXDT data register becomes empty, the shift register becomes empty, and TDIS=1 in the I2C_STS register
2. Data is then transferred to the shift register after writing 1 to the TXDT register
3. The TXDT register then becomes empty, and the TDIS is set again
4. TDIS is cleared by writing 2 to the TXDT register
5. Repeat step 3 and 4 until data (N-1) is sent
6. The slave will automatically transmit the Nth data, that is, PEC
7. Wait for the generation of an NACK signal. Once received, the ACKFAILF is set in the I2C_STS register. The ACKFAILF flag is cleared by writing 1 to the ACKFAILC
8. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. At the end of data transfer, the STOPF is cleared by writing 1 to the STOPC, transmission ends.

6. Data transfer (slave receive, clock stretching enabled, STRETCH=0)

After address matching:

1. I2C_RXDT register becomes empty, the shift register becomes empty, and RDBF=0 in the I2C_STS register
2. Upon the receipt of one-byte data, RDBF=1 and TCRLD=1, then the SCL is pulled low by the slave
3. The RDBF is cleared by read operation to the RXDT register
4. NACKEN bit of the I2C_CTRL register can be configured to generate an ACK or NACK, if needed

   If a NACK is detected, it indicates the completion of communication

   If an ACK is detected, communication continues. Writing CNT=1 will automatically clear the TCRLD flag by hardware, and the SCL is released by the slave for the reception of the next data
5. Repeat step 2/3/4 until the completion of data reception (N-1)
6. Set RLDEN=0 of the I2C_CTRL2 register to disable reload mode. Set CNT=1 to repeat step 2/3 to receive a PEC. The PECERR bit will be set if a PEC error occurs
7. Wait for the generation of a STOP condition. Once received, the STOPF is set in the I2C_STS register. The STOPF can be cleared by writing 1 to the STOPC bit in the I2C_CLR register, transfer ends.

SMBus slave transmitter

Figure 11-19 SMBus slave transmission flow

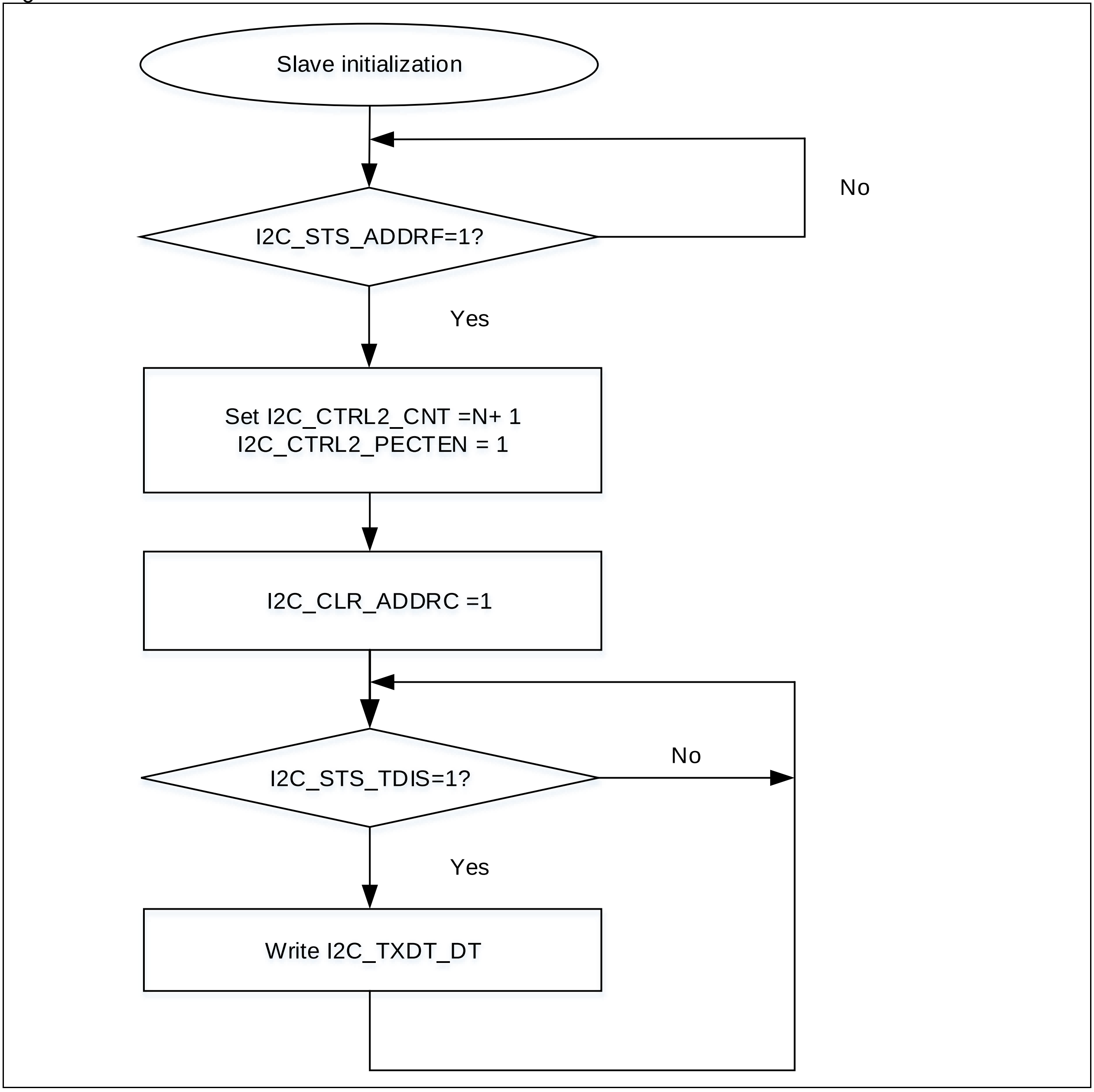
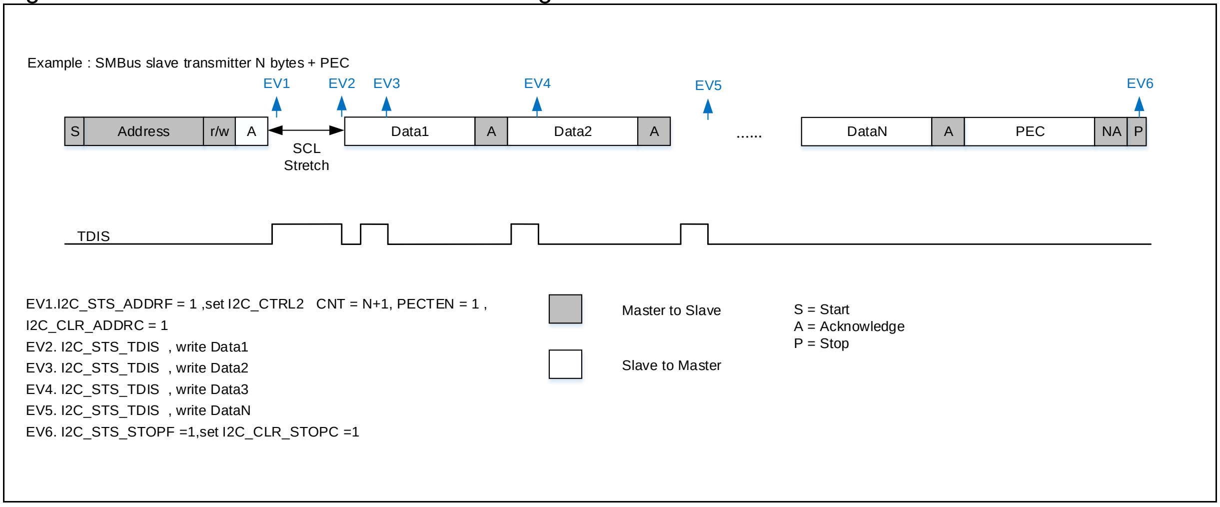


Figure 11-20 SMBus slave transmission timing



SMBus slave receiver

Figure 11-21 SMBus slave transmission timing

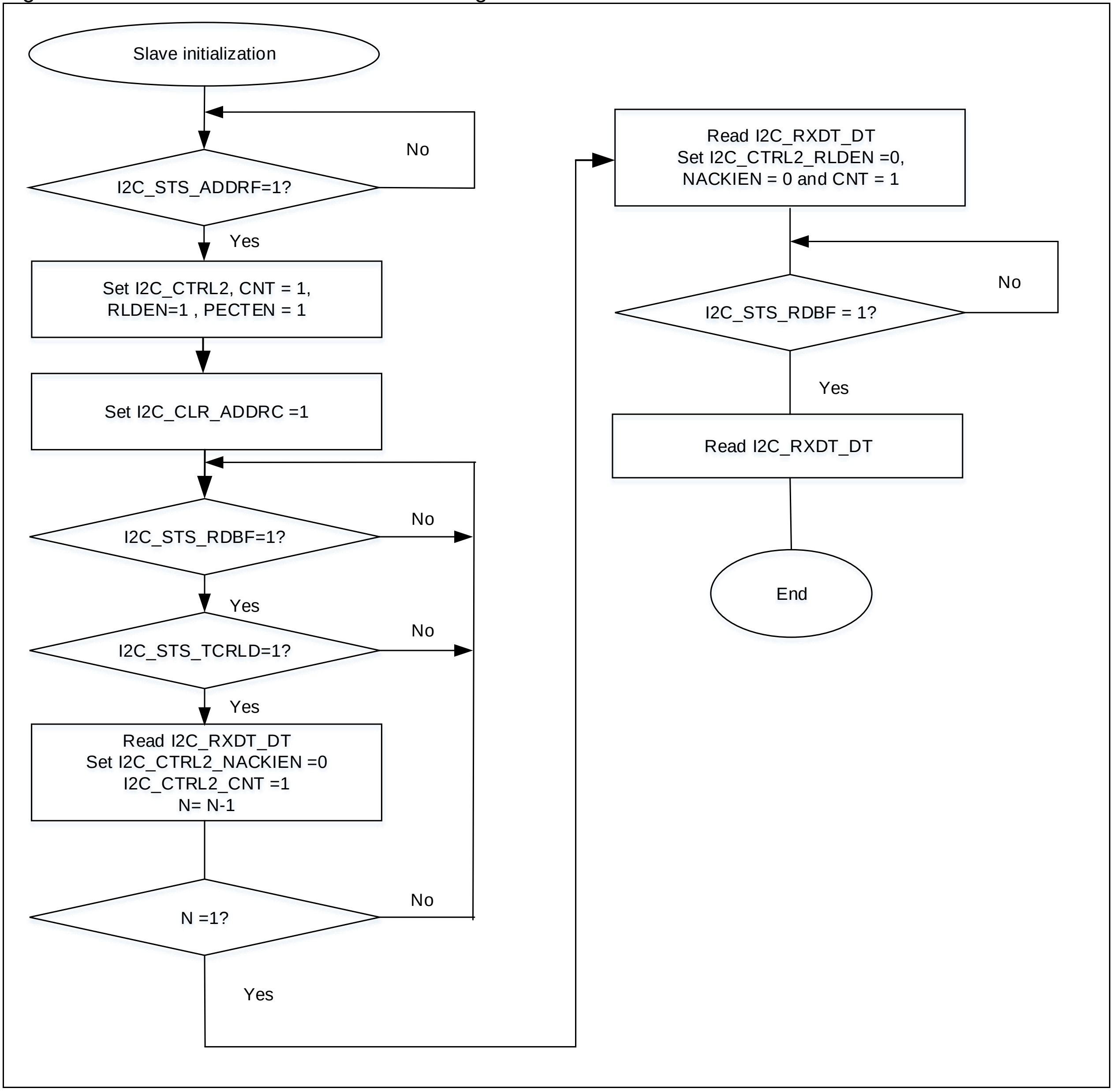
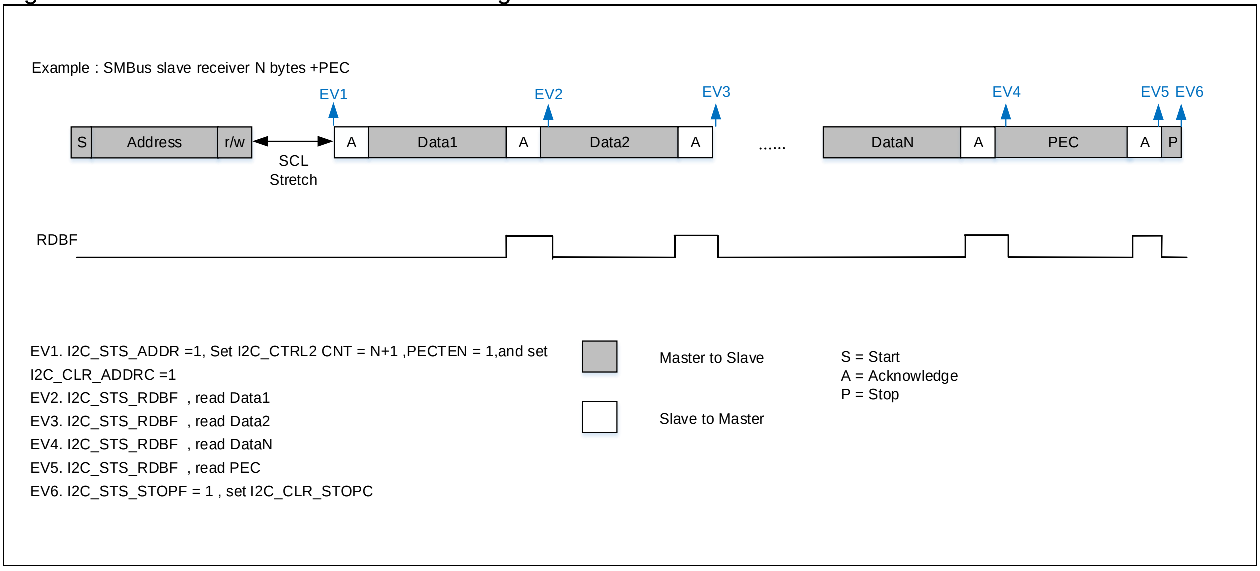


Figure 11-22 SMBus slave receive timing



11.4.8 Data transfer using DMA

I^2C data transfer can be done using DMA controller so as to reduce the burden on the CPU. The TDIEN and RDIEN must be set 0 when using DMA for data transfer.

Transmission using DMA (DMATEN=1)

1. Set the peripheral address (DMA_CxPADDR= I2C_TXDT 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 (DMAREN=1)

1. Set the peripheral address (DMA_CxPADDR = I2C_RXDT 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 I 2C 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: refer to the I^2C master communication flow section for STOP condition

   Slave receiver: refer to the I^2C slave communication flow section for STOP condition

11.4.9 Error management

The error management feature included in the I²C provides a guarantee for the reliability of communication. The manageable error events are listed below:

Table 11-6 I²C error event

Error event	Event flag	Enable control bit	Clear bit
SMBus Alert	ALERTF	ERRIEN	ALERTC
Timeout error	TMOUT	ERRIEN	TMOUTC
PEC error	PECERR	ERRIEN	PECERRC
Overrun/underrun	OUF	ERRIEN	OUFC
Arbitration lost	ARLOST	ERRIEN	ARLOSTC
Bus error	BUSERR	ERRIEN	BUSERRC

Overrun/Underrun (OUF)

In slave mode, an underrun/overrun may appear if the clock stretching feature is disabled (STRETCH=1 in the I2C_CTRL1 register).

In slave transmit mode: if data has not yet been written to the TXDT register before the transmission of the first bit of the to-be-transferred data (that is, before the generation of SDA edge), an underrun error may occur, and the OUF bit is set in the I2C_STS register, sending 0xFF to the bus.

In slave receive mode: The slave must read the received data in the case of the clock stretching being disabled (STRETCH=1). If one-byte data has been received and data is not read yet before the end of the next data reception, an overrun error occurs, setting OUF=1 in the I2C_STS register, and sending NCAK.

Arbitration lost (ARLOST)

An arbitration lost may occur when the device controls the SDA line to output high level but the actual bus output is low.

— Master transmit: An arbitration may occur during an address transfer and a data transfer
— Master receive: An arbitration may occur during an address transfer and an ACK response
— Slave transmit: An arbitration may occur during a data transfer
— Slave receive: An arbitration may occur during an ACK response

Once an arbitration lost is detected, the ARLOST is set by hardware in the I2C_STS register. The SCL and SDA buses will be released and go automatically back to slave mode.

Bus error (BUSERR)

The SDA line, during a data transfer, must be kept in a stable state when the SCL is in high level. The SDA can be changed only when the SCL signal becomes low, otherwise, a bus error may appear.

When the SCL is high:

— SDA changes from 1 to 0: a misplaced START condition
— SDA changes from 0 to 1: a misplaced STOP condition

Both of these conditions above may trigger a bus error. Once it occurs, the BUSERR is set by hardware in the I2C_STS register.

Packet error checking (PECERR)

The PEC is available only in SMBus mode. In master receive and slave receive modes, a PEC error may appear if the received PEC is not equal to the internally calculated PEC. In this case, the PECERR bit is set by hardware in the I2C_STS register.

In slave receive mode, an NACK is sent when a PEC error is detected.

In master receive mode, an NACK is always sent, whatever the PEC check result.

SMBus alert (ALERTF)

The SMBus alert feature is present when HADDREN=1 (SMBus master mode) and SMBALERT=1 (SMBus alert mode).Once an alert event is detected on the ALERT pin (ALERT pin changes from high to low), the ALERTF bit is set by hardware in the I2C_STS register.

Timeout error (TMOUT)

SMBus defines a timeout mechanism for the improvement of the system stability, preventing the bus from being pulled down in the case of a master or slave failure. Once a timeout event (defined in SMBus chapter) is detected, the TMOUT is set by hardware in the I2C_STS register. If a timeout error occurs in slave mode, the SCL and SDA buses are immediately released; if a timeout error occurs in master mode, a STOP condition is automatically by host to abort the communication

11.4.10 Wakeup from Deepsleep mode at address matching event

I2C1 supports to wake up from Deepsleep mode when the address matches. To enable this function, configure internal registers as below before entering Deepsleep mode:

- Set WAKEUPEN=1 in the I2C1_CTRL1 register
- Set DFLT=0 in the I2C1_CTRL1 register
- Set STRETCH=0 in the I2C1_CTRL1 register
- Select HICK by setting the I2C1SEL bit in the CRM_PICLKS register

Deepsleep mode wakeup procedure:

1. After completion of the above mentioned configurations, the system enters Deepsleep mode, and at this point, HICK is disabled
2. When the I2C bus enabling condition is detected, I2C interface enable HICK and pull the SCL bus low
3. After the HICK is enabled, start receiving address
   - Address matches: wake up the system, and during the wakeup process, I2C interface keeps pulling SCL bus low until the address match interrupt is handled and ADDRF flag is cleared
   - Address does not match: disable HICK, and the system does not wake up
4. SCL bus is released and enters normal transmission status
   - The I2C is not allowed to enter Deepsleep mode after it is accessed as master transfer data or slave

11.5 I²C interrupt requests

The following table lists all the I²C interrupt requests.

Table 11-7 I²C interrupt requests

Interrupt event	Event flag	Enable control bit
Address matched	ADDRF	ADDRIEN
Acknowledge failure	ACKFAIL	ACKFAILIEN
Stop condition received	STOPF	STOPIEN
Transmit interrupt state	TDIS	TDIEN
Receive data buffer full	RDBF	RDIEN
Transfer complete, wait for loading data	TCRLD	TDCIEN
Data transfer complete	TDC	
SMBus alert	ALERTF	ERRIEN
Timeout error	TMOUT	
PEC error	PECERR	

Overrun/underrun	OUF
Arbitration lost	ARLOST
Bus error	BUSERR

11.6 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.

11.7 I²C registers

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

Table 11-8 I²C register map and reset values

Register	Offset	Reset value
I2C_CTRL1	0x00	0x00000000
I2C_CTRL2	0x04	0x00000000
I2C_OADDR1	0x08	0x00000000
I2C_OADDR2	0x0C	0x00000000
I2C_CLKCTRL	0x10	0x00000000
I2C_TIMEOUT	0x14	0x00000000
I2C_STS	0x18	0x00000000
I2C_CLR	0x1C	0x00000000
I2C_PEC	0x20	0x00000000
I2C_RXDT	0x24	0x00000000
I2C_TXDT	0x28	0x00000000

11.7.1 Control register 1 (I2C_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31:24	Reserved	0x00	res	Kept at its default value.
Bit 23	PECEN	0x0	rw	PEC calculation enable 0: PEC calculation disabled 1: PEC calculation enabled
Bit 22	SMBALERT	0x0	rw	SMBus alert enable / pin set To enable SMBus master alert feature: 0: SMBus alert disabled 1: SMBus alert enabled To enable SMBus slave alert feature: 0: Pin high 1: Pin low, response address 0001100x
Bit 21	DEVADDREN	0x0	rw	SMBus device default address enable 0: SMBus device default address disabled 1: SMBus device default address enabled, response device default address 1100001x.
Bit 20	HADDREN	0x0	rw	SMBus host address enable 0: SMBus host address disabled 1: SMBus host address enabled, response host address 0001000x
Bit 19	GCAEN	0x0	rw	General call address enable 0: General call address disabled 1: General call address enabled, response address 0000000x
Bit 18	WAKEUPEN	0x0	rw	Deepsleep mode wakeup enable 0: Disabled 1: Enabled
Bit 17	STRETCH	0x0	rw	Clock stretching mode

				0: Clock stretching mode enabled 1: Clock stretching mode disabled Note: It is valid in slave mode only.
Bit 16	SCTRL	0x0	rw	Slave receiving data control 0: Slave receiving data control disabled 1: Slave receiving data control enabled
Bit 15	DMAREN	0x0	rw	DMA receive data request enable 0: DMA receive data request disabled 1: DMA receive data request enabled
Bit 14	DMATEN	0x0	rw	DMA transmit data request enable 0: DMA transmit data request disabled 1: DMA transmit data request enabled
Bit 13	Reserved	0x0	res	Kept at its default value.
Bit 12	ANGNFOFF	0x0	rw	Analog filter off 0: Enabled 1: Disabled
Bit 11:8	DFLT	0x0	rw	Digital filter value The glitches less than the filter time on the SCL bus will be filtered; filter time= DFLT x T_{I2C_CLK}.
Bit 7	ERRIEN	0x0	rw	Error interrupt enable 0: Error interrupt disabled 1: Error interrupt enabled
Bit 6	TDCIEN	0x0	rw	Transfer data complete interrupt enable 0: Transfer data complete interrupt disabled 1: Transfer data complete interrupt enabled
Bit 5	STOPIEN	0x0	rw	Stop generation complete interrupt enable 0: Stop generation complete interrupt disabled 1: Stop generation complete interrupt enabled
Bit 4	ACKFAILIEN	0x0	rw	Acknowledge fail interrupt enable 0: Acknowledge fail interrupt disabled 1: Acknowledge fail interrupt enabled
Bit 3	ADDRIEN	0x0	rw	Address match interrupt enable 0: Address match interrupt disabled 1: Address match interrupt enabled
Bit 2	RDIEN	0x0	rw	Receive data interrupt enable 0: Receive data interrupt disabled 1: Receive data interrupt enabled
Bit 1	TDIEN	0x0	rw	Transmit data interrupt enable 0: Transmit data interrupt disabled 1: Transmit data interrupt enabled
Bit 0	I2CEN	0x0	rw	I^2C peripheral enable 0: Disabled 1: Enabled

11.7.2 Control register 2 (I2C_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31:27	Reserved	0x00	res	Kept at its default value.
Bit 26	PECTEN	0x0	rw	Request PEC transmission enable 0: Transmission disabled 1: Transmission enabled
Bit 25	ASTOPEN	0x0	rw	Automatically send stop condition enable 0: Disabled (Software sends STOP condition) 1: Enabled (Automatically send STOP condition)
Bit 24	RLDEN	0x0	rw	Send data reload mode enable 0: Send data reload mode disabled 1: Send data reload mode enabled
Bit 23:16	CNT[7:0]	0x00	rw	Transmit data counter
Bit 15	NACKEN	0x0	rw	Not acknowledge enable 0: Acknowledge enabled 1: Acknowledge disabled
Bit 14	GENSTOP	0x0	rw	Generate stop condition 0: No stop generation 1: Stop generation
Bit 13	GENSTART	0x0	rw	Generate start condition 0: No start generation 1: Start generation

Bit 12	READH10	0x0	rw	10-bit address header read enable 0: 10-bit address header read disabled 1: 10-bit address header read enabled
Bit 11	ADDR10	0x0	rw	Host send 10-bit address mode enable 0: 7-bit address mode 1: 10-bit address mode
Bit 10	DIR	0x0	rw	Master data transmission direction 0: Transmit 1: Receive
Bit 9:0	SADDR[9:0]	0x000	rw	The slave address sent by the master In 7-bit address mode, BIT0 and BIT[9:8] don't care.

11.7.3 Own address register 1 (I2C_OADDR1)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	res	Kept at its default value.
Bit 15	ADDR1EN	0x0	rw	Own Address 1 enable 0: Own Address 1 disabled 1: Own Address 1 enabled
Bit 14:11	Reserved	0x0	res	Kept at its default value.
Bit 10	ADDR1MODE	0x0	rw	Own Address mode 0: 7-bit address mode 1: 10-bit address mode
Bit 9:0	ADDR1[9:0]	0x000	rw	Own address 1 In 7-bit address mode, bit 0 and bit [9:8] don't care

11.7.4 Own address register 2 (I2C_OADDR2)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	res	Kept at its default value.
Bit 15	ADDR2EN	0x0	rw	Own address 2 enable 0: Own address 2 disabled 1: Own address 2 enabled
Bit 14:11	Reserved	0x0	res	Kept at its default value.
Bit 10:8	ADDR2MASK[2:0]	0x0	rw	Own address 2-bit mask 000: Match address bit [7:1] 001: Match address bit [7:2] 010: Match address bit [7:3] 011: Match address bit [7:4] 100: Match address bit [7:5] 101: Match address bit [7:6] 110: Match address bit [7] 111: Response all addresses other than those reserved for I^2C
Bit 7:1	ADDR2[7:1]	0x00	rw	Own address 2 7-bit address mode
Bit 0	Reserved	0x0	res	Kept at its default value.

11.7.5 Timing register (I2C_CLKCTRL)

Bit	Register	Reset value	Type	Description
Bit 31:28	DIVL[3:0]	0x0	rw	Low 4 bits of clock divider value
Bit 27:24	DIVH[7:4]	0x0	rw	High 4 bits of clock divider value DIV = (DIVH << 4) + DIVL
Bit 23:20	SCLD[3:0]	0x0	rw	SCL output delay T_{SCLD} = (SCLD + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 19:16	SDAD[3:0]	0x0	rw	SDA output delay T_{SDAD} = (SDAD + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 15:8	SCLH[7:0]	0x00	rw	SCL high level T_{SCLH} = (SCLH + 1) x (DIV + 1) x T_{I2C_CLK}
Bit 7:0	SCLL[7:0]	0x00	rw	SCL low level T_{SCLL} = (SCLL + 1) x (DIV + 1) x T_{I2C_CLK}

11.7.6 Timeout register (I2C_TIMEOUT)

Bit	Register	Reset value	Type	Description
Bit 31	EXTEN	0x0	rw	Cumulative clock low extend timeout enable 0: Cumulative clock low extend timeout disabled 1: Cumulative clock low extend timeout enabled Corresponds to $T_{LOW:SEXT}$ / $T_{LOW:MEXT}$ in SMBus
Bit 30:28	Reserved	0x0	res	Kept at its default value.
Bit 27:16	EXTTIME[11:0]	0x000	rw	Cumulative clock low extend timeout value Timeout duration = (EXTTIME + 1) x 2048 x T_{I2C_CLK}
Bit 15	TOEN	0x0	rw	Detect clock low/high timeout enable 0: Clock low/high timeout detection disabled 1: clock low/high timeout detection enabled Corresponds to $T_{TIMEOUT}$ in SMBus
Bit 14:13	Reserved	0x0	res	Kept at its default value.
Bit 12	TOMODE	0x0	rw	Clock timeout detection mode 0: Clock low level detection 1: Clock high level detection
Bit 11:0	TOTIME[11:0]	0x000	rw	Clock timeout detection time For clock low level detection (TOMODE = 0): Timeout duration = (TOTIME + 1) x 2048 x T_{I2C_CLK} For clock high level detection (TOMODE = 1): Timeout duration = (TOTIME + 1) x 4 x T_{I2C_CLK}

11.7.7 Status register (I2C_STS)

Bit	Register	Reset value	Type	Description
Bit 31:24	Reserved	0x00	res	Kept at its default value.
Bit 23:17	ADDR[6:0]	0x00	r	Slave address matching value In 7-bit address mode: Slave address received In 10-bit address mode: 10-bit slave address header received
Bit 16	SDIR	0x0	r	Slave data transmit direction 0: Receive data 1 Transmit data
Bit 15	BUSYF	0x0	r	Bus busy flag transmission mode 0: Bus idle 1: Bus busy Once a START condition is detected, this bit is set; Once a STOP condition is detected, this bit is automatically cleared.
Bit 14	Reserved	0x00	res	Kept at its default value.
Bit 13	ALERTF	0x0	r	SMBus alert flag SMBus host: This bit indicates the reception of an alert signal (ALERT pin changes from high to low) 0: No alert signal received 1: Alert signal received SMBus slave: This bit indicates the device default address reception 0001100x 0: No alert signal received 1: Alert signal received
Bit 12	TMOUT	0x0	r	SMBus timeout flag 0: No timeout 1: Timeout
Bit 11	PECERR	0x0	r	PEC receive error flag 0: No PEC error 1: PEC error
Bit 10	OUF	0x0	r	Overflow or underflow flag In transmission mode: 0: No overrun or underrun 1: Underrun In reception mode: 0: No overrun or underrun 1: Overrun
Bit 9	ARLOST	0x0	r	Arbitration lost flag 0: No arbitration lost detected. 1: Arbitration lost detected.
Bit 8	BUSERR	0x0	r	Bus error flag

				0: No Bus error occurred 1: Bus error occurred
Bit 7	TCRLD	0x0	r	Transmission is complete, waiting to load data 0: Data transfer is not complete yet 1: Data transfer is complete This bit is set when data transfer is complete (CNT=1) and reload mode is enabled (RLDEN=1). It is automatically cleared when writing a CNT value. This bit is applicable in master mode or when SCTRL=1 in slave mode
Bit 6	TDC	0x0	r	Data transfer complete flag 0: Data transfer is not completed yet (the shift register still holds data) 1: Data transfer is completed (shift register become empty and all data has been sent to the bus) This bit is set when ASTOPEN = 0, RLDEN = 0 and CNT = 0. It is automatically cleared after a START or a STOP condition is received.
Bit 5	STOPF	0x0	r	Stop condition generation complete flag 0: No Stop condition detected. 1: Stop condition detected.
Bit 4	ACKFAILF	0x0	r	Acknowledge failure flag 0: No acknowledge failure 1: Acknowledge failure
Bit 3	ADDRF	0x0	r	0~7 bit address match flag 0: 0~7 bit address mismatch 1: 0~7 bit address match
Bit 2	RDBF	0x0	r	Receive data buffer full flag 0: Data register has not received data yet 1: Data register has received data
Bit 1	TDIS	0x0	rw1s	Transmit data interrupt status 0: Data has been written to the I2C_TXDT 1: Data has been sent from the I2C_TXDT to the shift register. I2C_TXDT become empty, and thus the to-be transferred data must be written to the I2C_TXDT. When the clock stretching mode is disabled, a TDIS event is generated by writing 1 so that data is written to the I2C_TXDT register in advance.
Bit 0	TDBE	0x0	rw1s	Transmit data buffer empty flag 0: I2C_TXDT holds data 1: I2C_TXDT is empty This bit is only used to indicate the current status of the I2C_TXDT register. The I2C_TXDT register can be cleared by writing 1 through software.

11.7.8 Status clear register (I2C_CLR)

Bit	Register	Reset value	Type	Description
Bit 31:14	Reserved	0x00000	res	Kept at its default value.
Bit 13	ALERTC	0x0	w	Clear SMBus alert flag SMBus alert flag is cleared by writing 1.
Bit 12	TMOUTC	0x0	w	Clear SMBus timeout flag SMBus timeout flag is cleared by writing 1.
Bit 11	PECERRC	0x0	w	Clear PEC receive error flag PEC receive error flag is cleared by writing 1.
Bit 10	OUFC	0x0	w	Clear overload / underload flag The overload / underload flag is cleared by writing 1.
Bit 9	ARLOSTC	0x0	w	Clear arbitration lost flag The arbitration lost flag is cleared by writing 1.
Bit 8	BUSERRC	0x0	w	Clear bus error flag The bus error flag is cleared by writing 1.
Bit 7:6	Reserved	0x0	res	Kept at its default value.
Bit 5	STOPC	0x0	w	Clear stop condition generation complete flag The stop condition generation complete flag is cleared by writing 1.
Bit 4	ACKFAILC	0x0	w	Clear acknowledge failure flag The acknowledge failure flag is cleared by writing 1.

Bit 3	ADDRC	0x0	w	Clear 0~7 bit address match flag The 0~7 bit address match flag is cleared by writing 1.
Bit 2:0	Reserved	0x0	res	Kept at its default value.

11.7.9 PEC register (I2C_PEC)

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	res	Kept at its default value.
Bit 7:0	PECVAL[7:0]	0x00	r	PEC value

11.7.10 Receive data register (I2C_RXDT)

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	res	Kept at its default value.
Bit 7:0	DT[7:0]	0x00	r	Receive data register

11.7.11 Transmit data register (I2C_TXDT)

Bit	Register	Reset value	Type	Description
Bit 31:8	Reserved	0x000000	res	Kept at its default value.
Bit 7:0	DT[7:0]	0x00	rw	Transmit data register

12 Universal synchronous/asynchronous receiver/transmitter (USART)

12.1 USART introduction

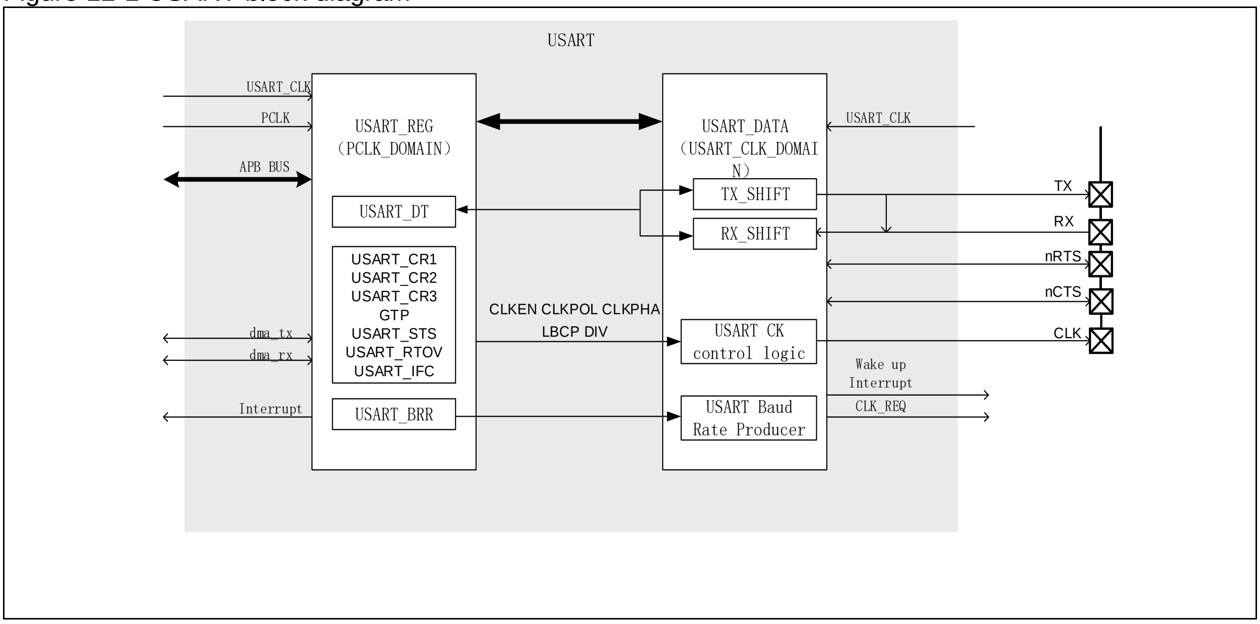
The universal synchronous/asynchronous receiver/transmitter (USART) serves an interface for communication by means of various configurations and peripherals with different data formats. It supports asynchronous full-duplex and half-duplex as well as synchronous transfer. With a programmable baud rate generator, USART offers up to 9.375 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, CTS/RTS (Clear To Send/Request To Send) hardware flow operation, RS485 and Modbus.

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.

It supports dual clock domain. The PCLK is sourced by divided system clock, and the USART_CLK is clocked by PCLK, HICK or LEXT, which allows USART to work in Deepsleep mode and support low-power mode wakeup.

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
  - RS-485
  - 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 9.375 Mbits/s
- Programmable frame format
  - Programmable data word length (7 bits, 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 data transmission order (MSB/LSB)
  - Programmable Tx/Rx pin polarity
  - Programmable DT polarity
- 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 13 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
  - Receiver timeout detection
  - Byte match detection
  - Low-power mode wakeup

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

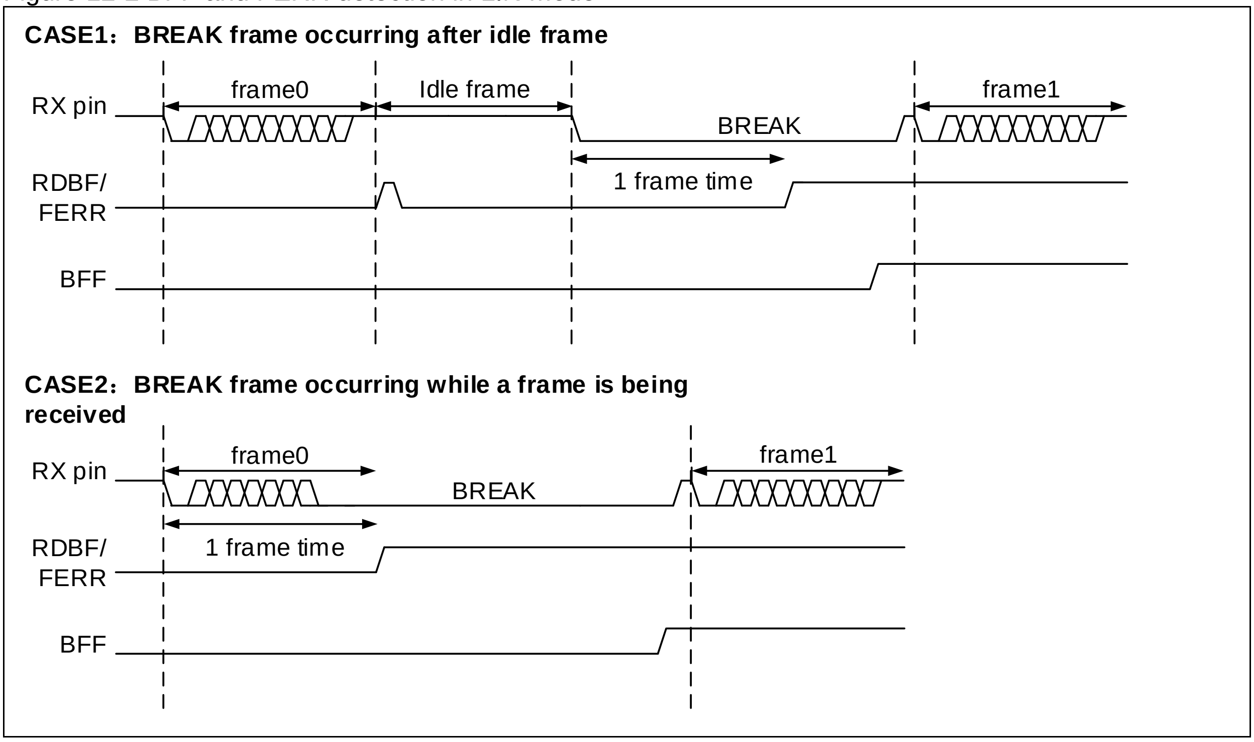
Selection of operation mode is done by following the configuration process listed below. In addition, such configuration method, along with those of receiver and transmitter described in the subsequent sections, are used to make USART initialization configuration

1. **LIN mode**

Set LINEN=1, CLKEN=0, STOPBN[1:0]=0, SCMEN=0, SLHDEN=0, IRDAEN=0, DBN[1:0]=00.

LIN master has break generation capability, and can transmit 13-bt low-level LIN synchronous break frame by setting SBF=1. The LIN slave has break detection capability, and can select 11-bit or 10-bit break detection by setting BFBN=1 or BFBN=0.

Figure 12-2 BFF and FERR detection in LIN mode



2. **Smartcard mode**

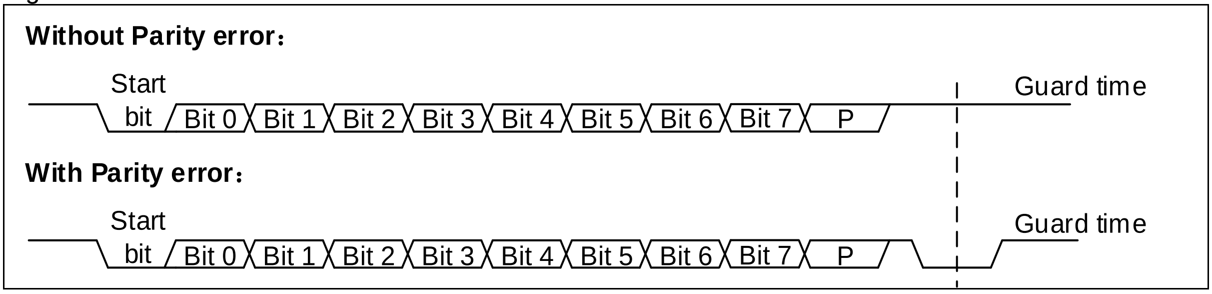
Set SCMEN=1, LINEN=0, SLHDEN=0, IRDAEN=0, CLKEN=1, DBN[1:0]=01, PEN=1 and STOPBN[1:0]=11.

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 Smartcard that the data has not been correctly received.

Figure 12-3 Smartcard frame format

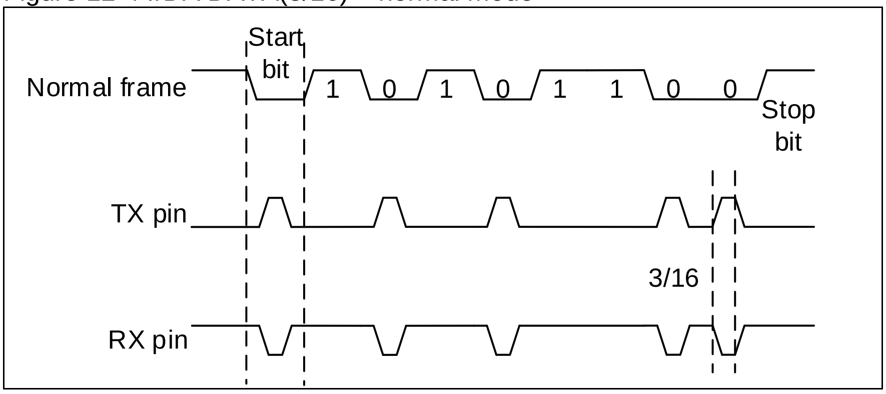


3. **Infrared mode**

Set 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. **Modbus**

USART only supports basic hardware required by Modbus/RTU and Modbus/ASCII implementation, which means that the control must be done by software (USART provides EOB (end of block) detection only).

In Modbus/RTU, the EOB detection is implemented by the programmable timeout recognizing the receive line idle time being larger than 2 bytes. Users can configure the RTOV register to set the required timeout value (unit: 1 bit width), and enable timeout detection by setting RTODEN=1. When the receive line idle time detected by USART receiver is equal to the programmed timeout value, the USART will set RTODF. An interrupt is generated when RTODIE=1, and the RTODF bit can be cleared by writing 1 to the RTODCF bit.

In Modbus/ASCII, the EOB detection is implemented by the byte match feature recognizing the special byte sequence (CR/LF). Write LF ASCII code to the ID[7:0], and set CMDIE=1 to enable byte match feature. When the data received by USART matches ID[7:0], the USART will set CMDF. An interrupt is generated when CMDIE=1, and the CMDCF bit can be cleared by writing 1 to the CMDF bit.

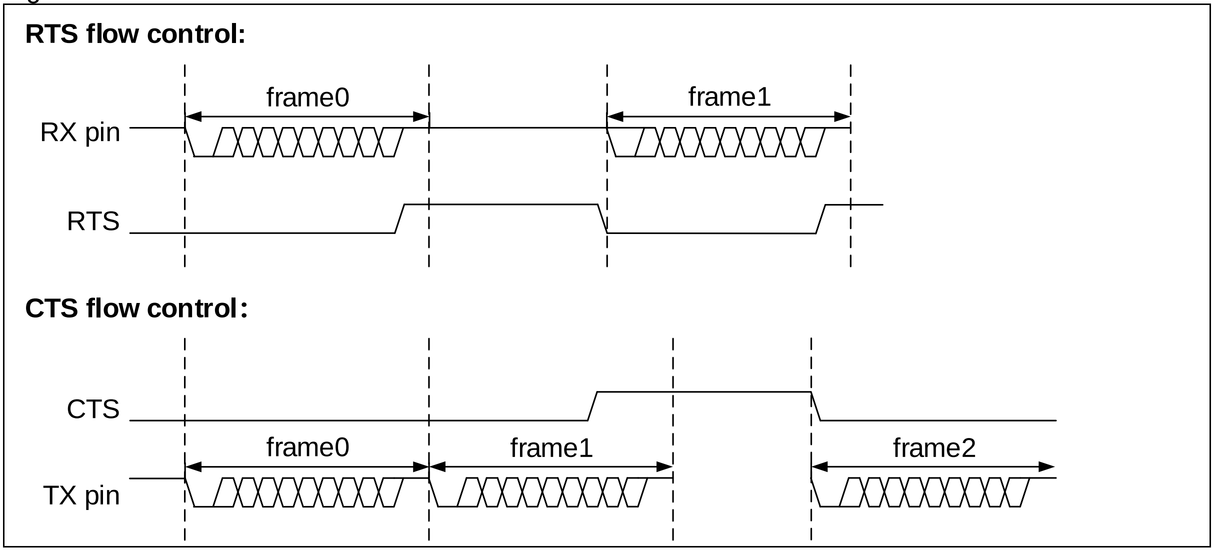
5. **Hardware flow control mode**

Setting the RTSEN and CTSEN bit will enable RTS and CTS flow control, respectively.

RTS flow control: When the USART receiver is ready to receive new data, the RTS becomes effective (pull down low). When the data is received in the receiver (at the beginning of each stop bit), the RTS bit is set, indicating the data transmission is to be stopped at the end of the current frame.

CTS flow control: USART transmitter checks CTS input before transmitting the next frame. If CTS is effective (that is, CTS is low), the next data is to be transmitted. If CTS becomes invalid (CTS is high) during transmission, the data transmission will stop after the completion of the current transmission.

Figure 12-5 Hardware flow control



6. RS485 mode

This mode is enabled by setting RS485EN=1. The enable signal is output on the RTS pin. The DEP bit is used to select the polarity of the DE signal. The TSDT[4: 0] bit is used to define the latency before the transmission of the start bit on the transmitter side, while the TCDT[4: 0] is used to define the latency before the TC flag is set following the stop bit at the end of the last data.

7. Silent mode

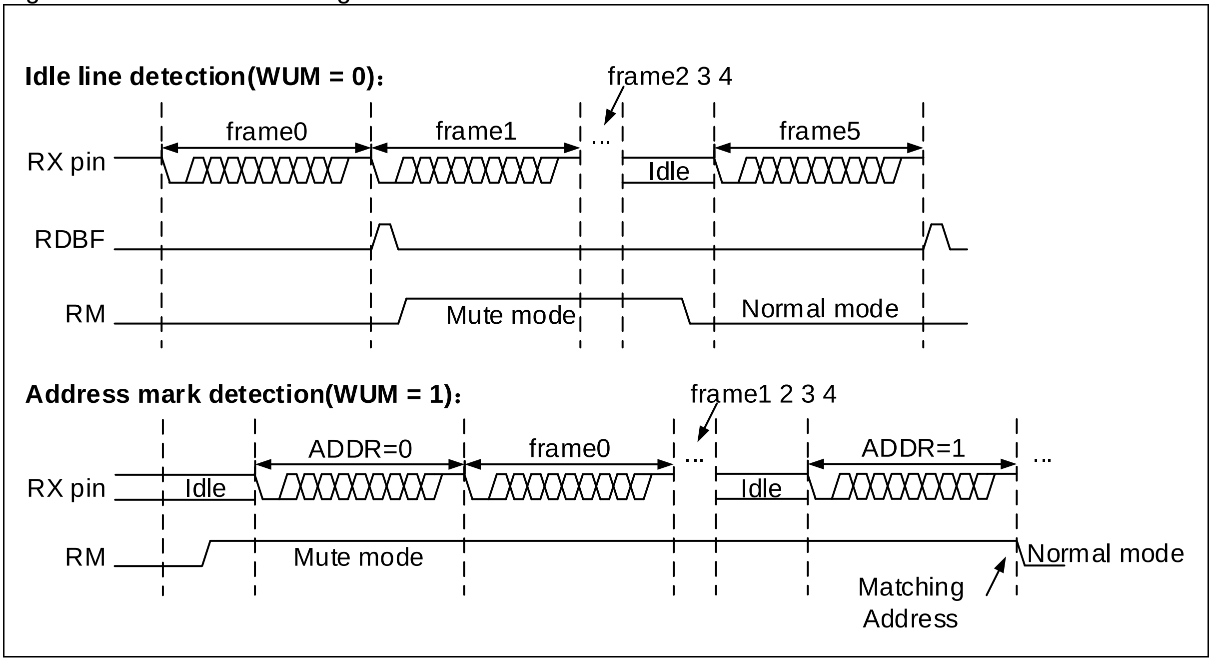
This mode is enabled by setting RM=1. When the WUM bit is set 1 or 0, it wakes up from silent mode through ID match and idle bus, respectively. The ID[7: 0] is configurable. Select ID[7: 0] or ID[3: 0] by setting the IDBN bit. When ID match is selected, if the MSB of data bit is set, it indicates that the current data stands for ID.

When parity check is disabled, if DBN[1:0]=10, the MSB refers to the USART_DT[6]; if DBN[1:0]=00, MSB refers to the USART_DT[7]; if DBN[1:0]=01, MSB refers to the USART_DT[8].

When parity check is enabled, if DBN[1:0]=10, the MSB refers to the USART_DT[5]; if DBN[1:0]=00, MSB refers to the USART_DT[6]; if DBN[1:0]=01, MSB refers to the USART_DT[7].

When the ID[3: 0] bit is selected, the four LSB bits indicate the ID value; When the ID[7: 0] bit is selected, all of the LSB bits indicates the ID value, except for the above parity check bits and MSB bits.

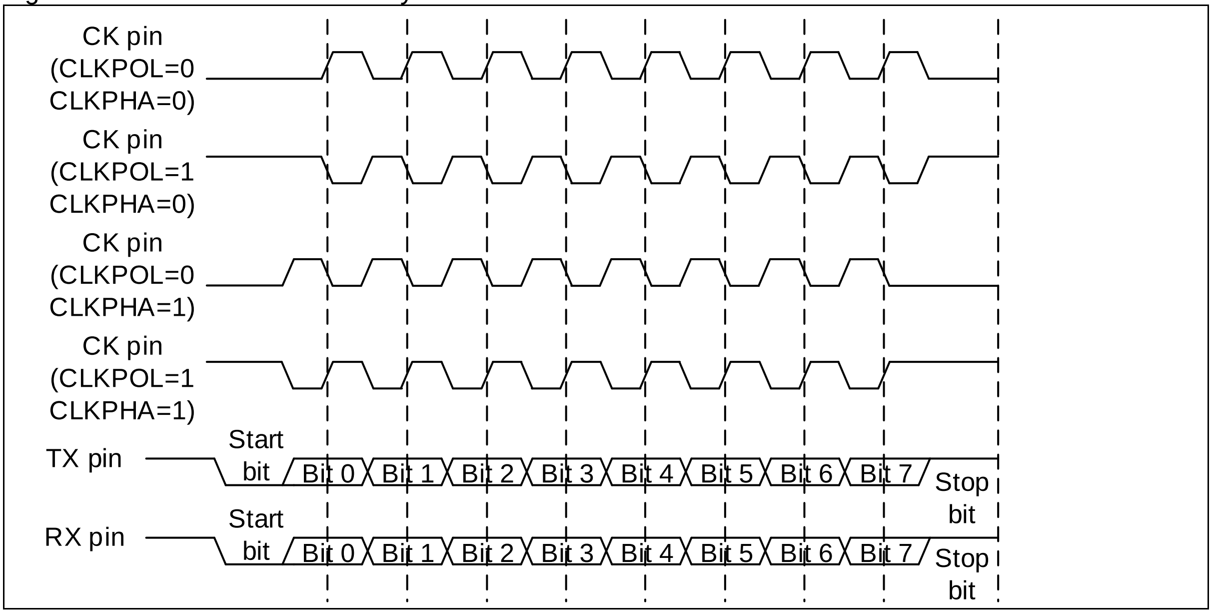
Figure 12-6 Mute mode using Idle line or Address mark detection



8. Synchronous mode

Setting the CLKEN bit enables synchronous mode and clock pin output. 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 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

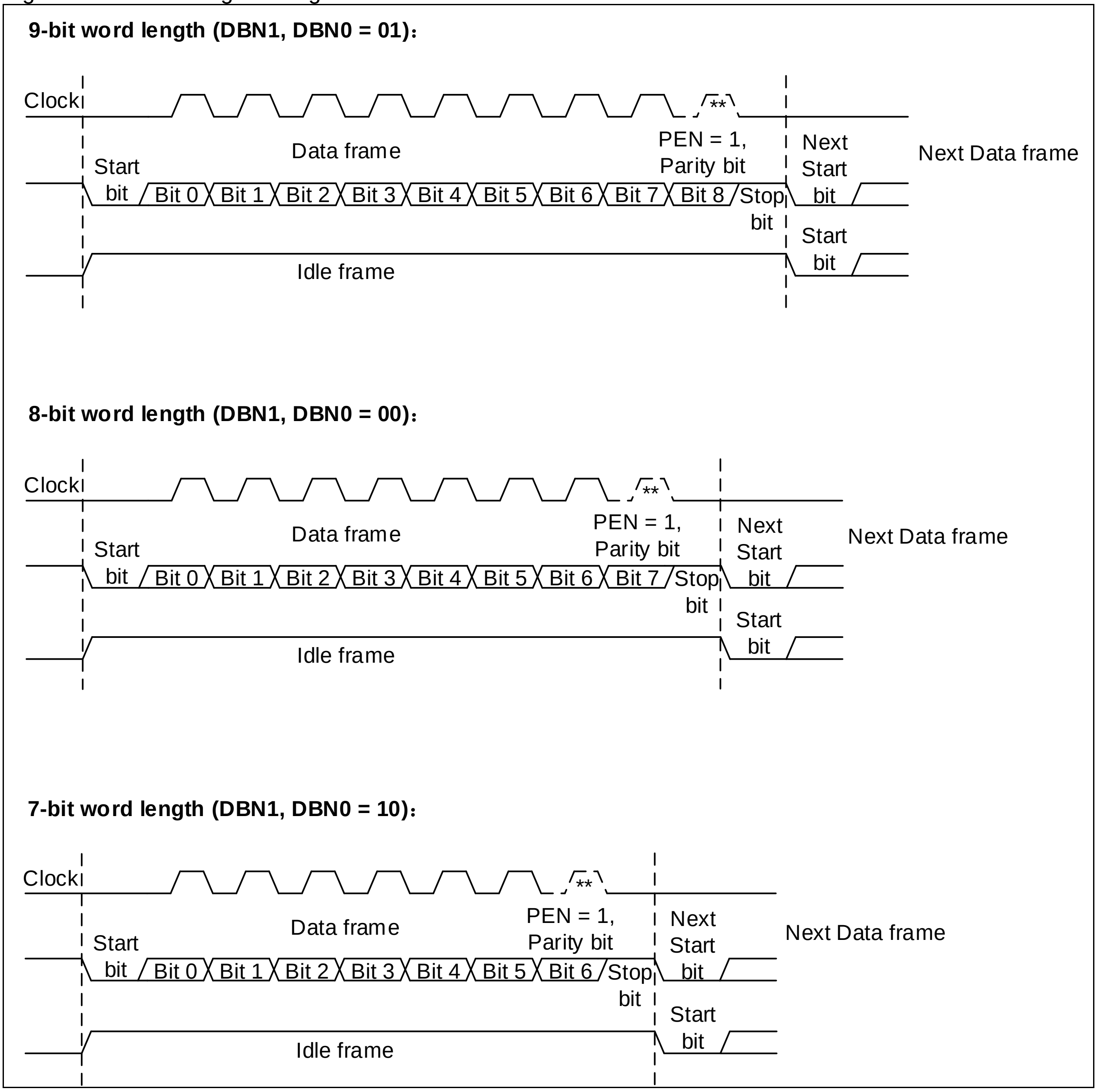
USART data frame consists of start bit, data bit and stop bit, with the last data bit being as a parity bit.

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.

In non-LIN mode, a break frame transmission and detection must be in line with this rule. For instance, if DBN[1:0]=00, the break frame size for transmission and detection should be 10-bit low level plus its stop bit. In LIN mode, refer to Mode selector and configuration process for more details. The DBN1 and DBN0 bits are used to program 7-bit (DBN[1:0]=10), 8-bit (DBN[1:0]=00) or 9-bit (DBN[1:0]=01) data bits.

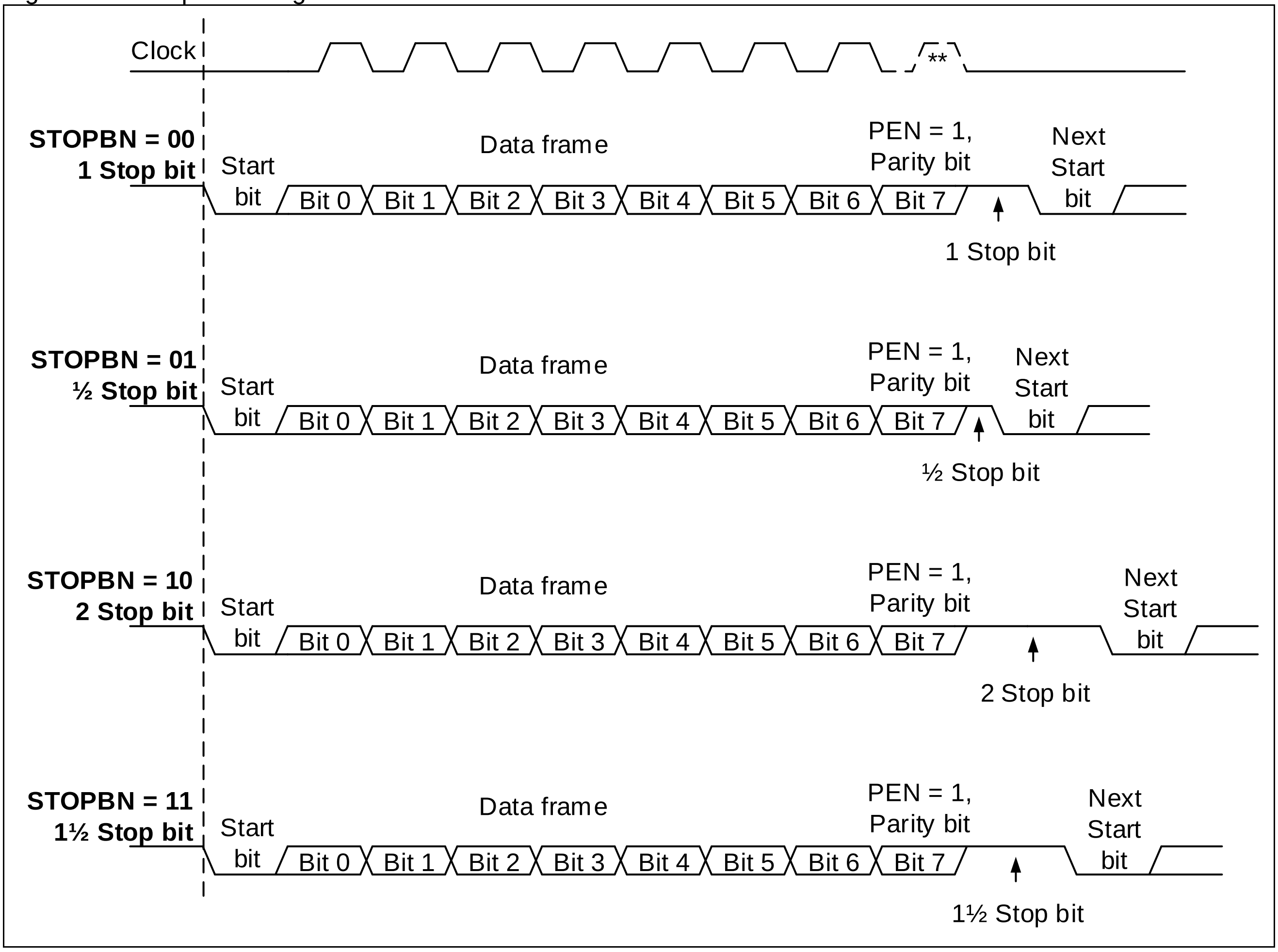
Figure 12-8 Word length configuration



The STOPBN bit is used to program one-bit (STOPBN=00), 0.5-bit (STOPBN=01), two-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



Set the MTF bit to determine whether MSB (MTF=1) first or LSB (MTF=0) first.

Set USART_DT as 1=L,0=H (DTREV=1) or 0=L,1=H (DTREV=0) for transmission and reception by setting the DTREV bit.

Set the TXREV bit to select VDD=0/mark,Gnd=1/idle (TXREV=1) or VDD=1/idle,Gnd=0/mark (TXREV=0) for signal transmission on USART_TX pin.

Set the RXREV bit to select VDD=0/mark,Gnd=1/idle (RXREV=1) or VDD=1/idle,Gnd=0/mark (RXREV=0) for signal transmission on USART_RX pin.

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.
5. Configure the channel priority of DMA transfer in the DMA control register.
6. Configure DMA interrupt generation after half or full transfer in the DMA control register.
7. Enable DMA transfer channel in the DMA control register.

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.
5. Configure the channel priority of DMA transfer in the DMA control register.
6. Configure DMA interrupt generation after half or full transfer in the DMA control register.
7. Enable a DMA transfer channel in the DMA control register.

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 greater than or equal to 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:

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

Where, f_{CK} refers to the system clock of USART (PCLK1 for USART2, 3 and USART4, 5, 7, 8, and PCLK2 for USART1, 6).

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 rate		**fPCLK=36 MHz**			**fPCLK=72 MHz**		
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%

If the baud rate is 115.2 Kbps, when fPCLK=36MHz, the baud register should be set as 312(0x138), and the calculated result is 36000000 / 312 = 115384 = 115.384Kbps.

Error: (actual value – theoretical value) / theoretical value * 100%:(115.384 - 115.2) / 115.2 * 100% = 0.15%.

12.7 Transmitter

12.7.1 Transmitter introduction

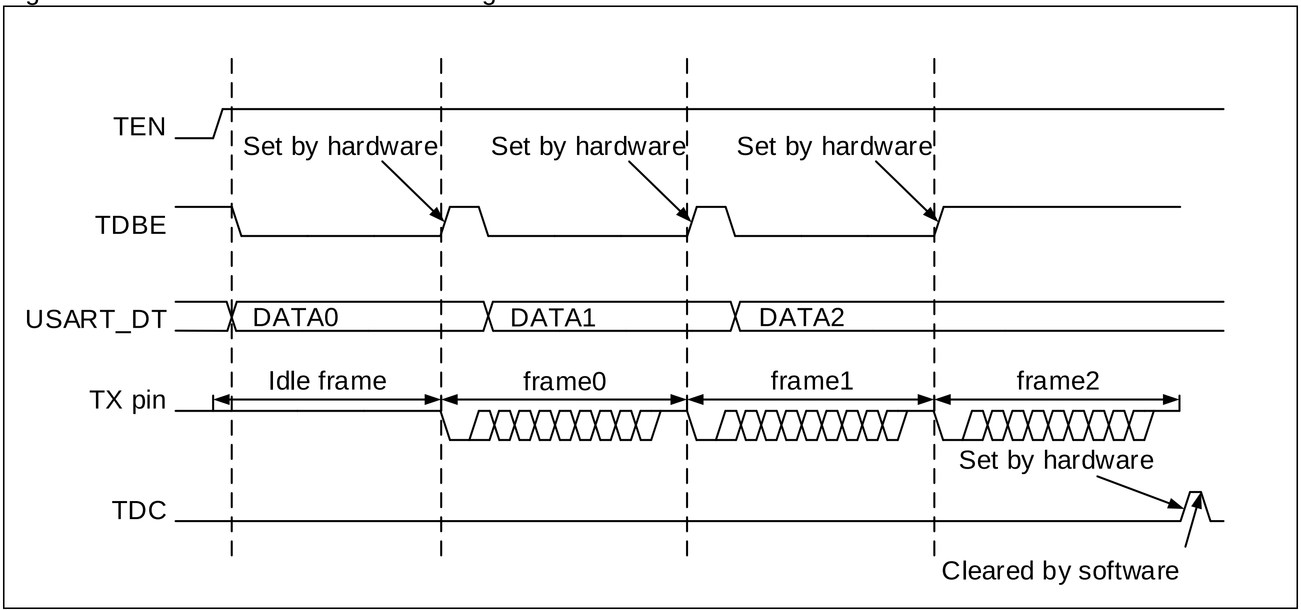
USART transmitter has its individual TEN control bit. The transmitter and receiver share the same baud rate that is programmable. There is a transmit data buffer (TDR) and a transmit shift register in the USART. The TDBE bit is set whenever the TDR is empty, and an interrupt is generated if the TDBEIEN 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 full-duplex/half-duplex selector for more information (Section 12.2).
3. Mode configuration: Refer to mode selector for more information (Section 12.3).
4. Frame format configuration: Refer to frame format for more information (Section 12.4).
5. Interrupt configuration: Refer to interrupt generation for more information (Section 12.11).
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 baud rate generation for details (Section 12.6).
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 12-10 Variations when transmitting TDC/TDBE



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 full-duplex/half-duplex selector for more information (Section 12.2).
3. Mode configuration: Refer to mode selector for more information (Section 12.3).
4. Frame format configuration: Refer to frame format for more information (Section 12.4).
5. Interrupt configuration: Refer to interrupt generation for more information (Section 12.11).
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 baud rate generation for details (Section 12.6).
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 RDNE 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.
- USART receiver: It is handled as a data frame, and the IDLEF bit is set. An interrupt is generated if the IDLEIEN is set.

Idle frame reception:

- USART receiver processes this idle frame as a data frame, and the IDLEF bit is set (an interrupt is generated if the IDLEIEN bit 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 is set.

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.

The table below 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	Valid
Any value	111/110/101/011	0	Valid

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[1:0]	DIV[3:0] = 0	DIV[3:0] != 0
00	3.75%	3.33%
01	3.41%	3.03%

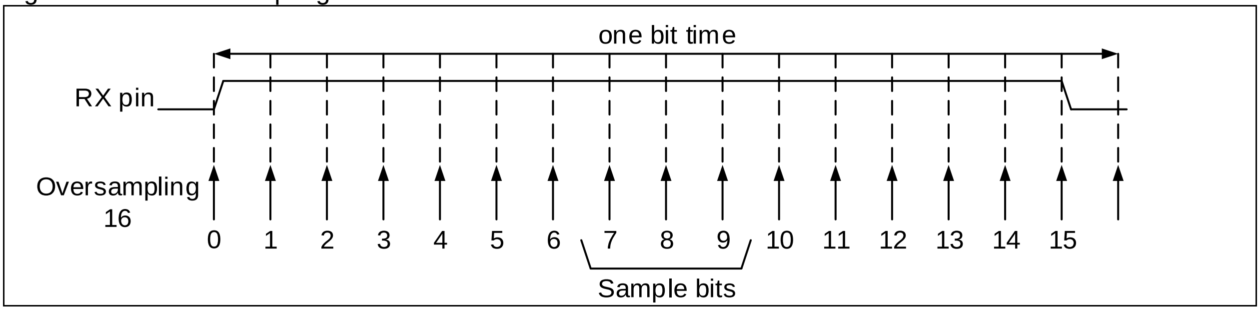
10	4.16%	3.7%

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 12-11 Data sampling for noise detection



12.9 Low-power wakeup

USART supports low-power wakeup. Before entering DEEPSLEEP mode, the software should guarantee that the USART_CLK is clocked by HICK and LEXT, confirm no transmission by checking the OCCUPY bit, verify the completion of USART receiver initialization by checking the RXON bit, and finally set SMUSEN=1 to enable USART in DEEPSLEEP mode.

After entering the DEEPSLEEP mode, the USART_CLK is disabled, and USART detects the falling edge on the receiver line. Once a falling edge is detected, USART will request MCU to enable USART_CLK (in enabled state until the USART returns to Idle state). If a wakeup source is detected in this process, USART will generate an interrupt to wake up MCU; if no wakeup source is detected, USART will request MCU to disable USART_CLK and wait for the next falling edge).

USART supports three wakeup modes depending on the LPWUM[1:0], i.e., ID match (LPWUM=00), start bit (LPWUM=10) and RDBF flag (LPWUM=11). If the programmed wakeup source is detected in DEEPSLEEP mode, the LPWUF bit is set. Setting the LPWUFIE bit generates an interrupt. It should be noted that this interrupt is only valid in DEEPSLEEP mode. In addition,, if the RDBF flag is selected for wakeup, setting RDBFIE can enable an interrupt.

The system clock is disabled after entering the DEEPSLEEP; therefore, the software needs to configure the wakeup mode and set the corresponding interrupt enable bits in advance.

When USART in silent mode enters DEEPSLEEP mode:

1. *Do not use idle bus for silent mode wakeup.*
2. *If the ID match is selected for silent mode wakeup, the MCU low-power mode wakeup mode should be ID match mode. If the RDBF bit is set before entering DEEPSLEEP mode, for the ID match mode, even if MCU exits DEEPSLEEP mode, the USART is still in silent mode.*
3. *If the Start bit is selected to wake up MCU to exit* DEEPSLEEP *mode, the LPWUF bit is set, while the RDBF bit is not.*

Note: For USART to wake up Deepsleep mode, it is necessary to clear USART flag bits and EXINT's pending flag by software.
When using USART to wake up Deepsleep mode, it is recommended to configure rising edge trigger for EXINT, because if rising edge and falling edge (both edges) are configured for EXINT, it is necessary to clear EXINT pending flag individually.

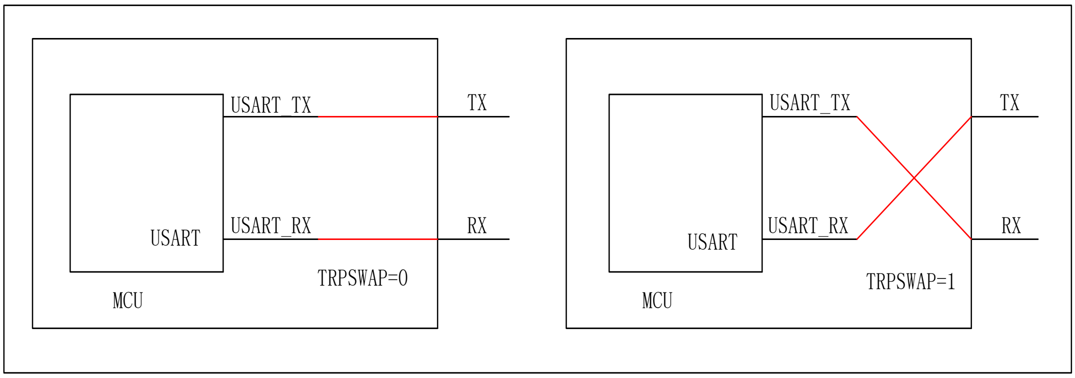
12.10 Tx/Rx swap

When the TRPSWAP bit (USART_CTRL2[15]) is set, Tx/Rx pin can be swapped. Two common scenes are listed below:

- If the Tx/Rx were reversed while the user attempts to connect the device externally to a RS-232 chip, they can be swapped through the TRPSWAP bit, without the need of hardware intervention.

- If the user only connected the master Tx to the slave Rx in full-duplex mode, the Tx/Rx can be interchangeable with the TRPSWAP bit , after the master and slave are swapped, without the need of hardware intervention.

Figure 12-12 Tx/Rx swap



Note: The SWAP (USART_CTRL2[15]) can be modified only when the USART is disabled (UEN=0)

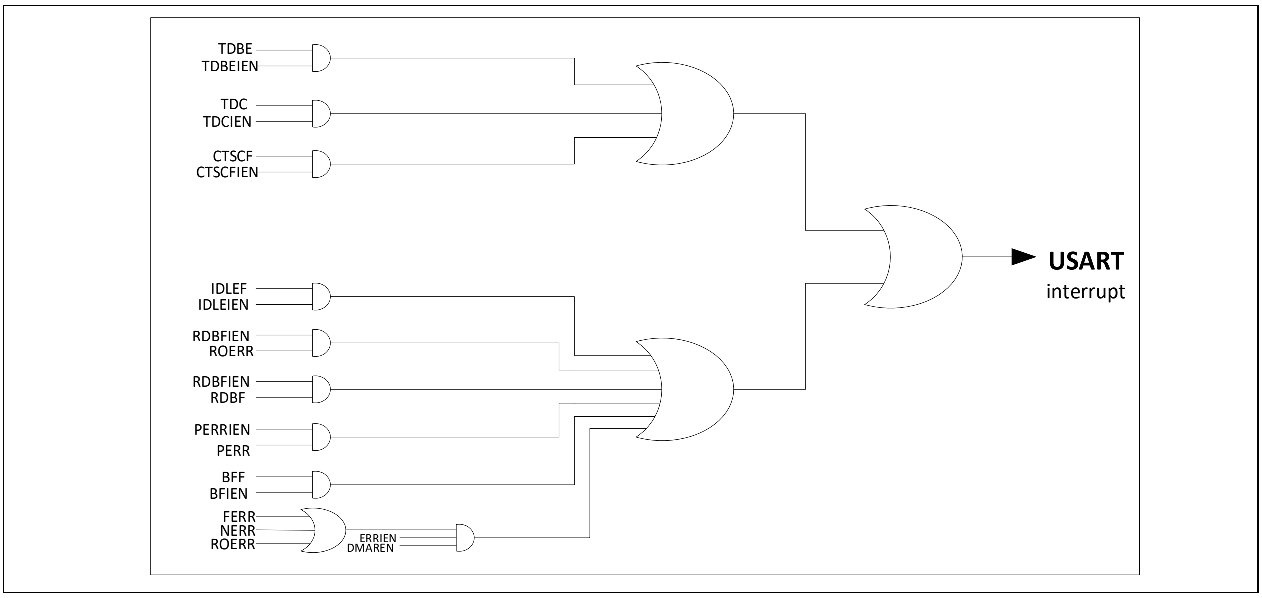
12.11 Interrupt requests

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. The table below 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 requests

Interrupt event	Event flag	Enable bit
Transmit data register empty	TDBE	TDBEIEN
CTS flag	CTSCF	CTSCFIEN
Transmit complete	TDC	TDCIEN
Receive data buffer full	RDBF	RDBFIEN
Receive 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 12-13 USART interrupt map diagram



12.12 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.

12.13 USART registers

Table 12-6 USART register map and reset value

Register	Offset	Reset value
USART_STS	0x00	0x00C0
USART_DT	0x04	0x0000
USART_BAUDR	0x08	0x0000
USART_CTRL1	0x0C	0x0000
USART_CTRL2	0x10	0x0000
USART_CTRL3	0x14	0x0000
USART_GDIV	0x18	0x0000
USART_RTOV	0x1C	0x0000
USART_IFC	0x20	0x0000

12.13.1 Status register (USART_STS)

Bit	Register	Reset value	Type	Description
Bit 31:23 Bit 19:18 Bit 15:12 Bit 10	Reserved	0x000000	resd	Forced 0 by hardware.
Bit 22	RXON	0		Receiver enable flag 0: Disabled 1: Enabled Note: USART4 to USART8 do not support this bit (0 by default)

Bit 21	TXON	0		Transmitter enable flag 0: Disabled 1: Enabled Note: USART4 to USART8 do not support this bit (0 by default)
Bit 20	LPWUF	0	r	Low-power mode wakeup flag When a wakeup event is detected, this bit is set. It is cleared by software. 0: No wakeup event detected 1: Wakeup event detected Note: USART4 to USART8 do not support this bit (0 by default)
Bit 17	CMDF	0	r	Byte match detection flag This bit is set by hardware when the byte defined by ID[7:0] is received. It is cleared by software. 0: No byte received 1: Byte received
Bit 16	OCCUPY	0		Receiver occupied flag 0: Not occupied 1: Occupied Note: USART4 to USART8 do not support this bit (0 by default)
Bit 11	RTODF	0	r	Receiver timeout detection flag This bit is set by hardware when the timeout value reaches the programmed value in RTOV register and without any communication. It is cleared by software. 0: No timeout detected 1: Timeout detected
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

12.13.2 Data register (USART_DT)

Bit	Register	Reset value	Type	Description
Bit 31:9	Reserved	0x000000	resd	Forced to 0 by hardware.
Bit 8:0	DT	0x000	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.

12.13.3 Baud rate register (USART_BAUDR)

Note: If the TE or RE is disabled respectively, the baud counter stops counting.

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Forced to 0 by hardware.
Bit 15:0	DIV	0x0000	rw	Divider This field defines the USART divider.

12.13.4 Control register 1 (USART_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31:29	Reserved	0x00000	resd	Forced to 0 by hardware.
Bit 28	DBN1	0x0	rw	Data bit num This bit, along with the DBN0 bit, is used to program the number of data bits. 10: 7 data bits 00: 8 data bits 01: 9 data bits 11: Write operation forbidden.
Bit 27	RTODEN	0	rw	Receiver time out detection enable 0: Disabled 1: Enabled
Bit 26	RETODIE	0	rw	Receiver time out detection interrupt enable 0: Disabled 1: Enabled
Bit 25:21	TSDT	0x00	rw	Transmit start delay time In RS485 mode, the first data (in sequential transmit mode) is transmitted after a period of time of being written so as to ensure that the transfer direction of the external transmitter/receiver to switch back to transmit. This time depends on the TSDT value, in unit of 1/16 baud rate.
Bit 20:16	TCDT	0x00	rw	Transmit complete delay time

				In RS485 mode, a period of time (delay) is needed before the last data transfer is complete even if the last STOP bit has been transferred. This time duration allows the transfer direction of the external receiver/transmitter to switch back to receive. This time depends on the TCDT value, in unit of 1/16 baud rate.
Bit 15	Reserved	0	resd	Kept at its default value.
Bit 14	CMDIE	0	rw	Character match detection interrupt enable 0: Disabled 1: Enabled
Bit 13	UEN	0x0	rw	USART enable 0: Disabled 1: Enabled
Bit 12	DBN0	0x0	rw	Data bit num This bit, along with DBN1, is used to program the number of data bits 10: 7 data bits 00: 8 data bits 01: 9 data bits 11: Write operation forbidden.
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.

12.13.5 Control register 2 (USART_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31:28	IDH	0x0	rw	USART identification This field holds the upper four bits of USART ID. It is configurable
Bit 27:20	Reserved	0x000	resd	Kept at its default value.
Bit 19	MTF	0	rw	MSB transmit first This bit is used to select MSB transmit first or LSB transmit first. 0: LSB first 1: MSB first
Bit 18	DTREV	0	rw	DT register polarity reverse 0: 1=H, 0=L 1: 1=L, 0=H
Bit 17	TXREV	0	rw	TX polarity reverse 0: VDD=1/idle,Gnd=0/mark 1: VDD=0/mark,Gnd=1/idle
Bit 16	RXREV	0	rw	RX polarity reverse 0: VDD=1/idle,Gnd=0/mark 1: VDD=0/mark,Gnd=1/idle
Bit 15	TRPSWAP	0x0	rw	Transmit/receive pin swap 0: Transmit/receive pin is not swappable 1: Transmit/receive pin is swappable
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	Kept 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	IDBN	0	rw	Identification bit num This bit is used to select ID bit number. 0: 4 bit

				1: Data bit - 1 bit Note: When this bit is set, in 7, 8 or 9-bit data mode, the ID bit number is the lower 6, 7 or 8 bit, respectively.
Bit 3:0	IDL	0x0	rw	USART identification This field holds the lower four bits of USART ID. It is configurable.

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

12.13.6 Control register 3 (USART_CTRL3)

Bit	Register	Reset value	Type	Description
Bit 31:18 Bit 12	Reserved	0x000000	resd	Forced to 0 by hardware.
Bit 17:16	LPWUM	0x0	rw	Low power wakeup method 00: ID match 01: Reserved 10: Start bit 11: RDBF Note: USART4 to USART8 do not support this bit (0 by default)
Bit 15	DEP	0	rw	DE polarity selection 0: High level active 1: Low level active
Bit 14	RS485EN	0	rw	RS485 enable This bit is used to enable RS485 mode. In RS485 mode, the USART controls the transfer direction of the external receiver/transmitter through the DE signal. 0: RS485 mode disabled. The control signal DE output is disabled. RTS pin is used in RS232 mode. 1: RS485 mode enabled. The control signal DE outputs on the RTS pin
Bit 13	LPWUFIE	0	rw	Low power wakeup flag interrupt enable 0: Disabled 1: Enabled Note: USART4 to USART8 do not support this bit (0 by default)
Bit 11	SMUSEN	0	rw	Deepsleep mode USART enable 0: Disabled 1: Enabled Note: USART4 to USART8 do not support this bit (0 by default)
Bit 10	CTSCFIEN	0x0	rw	CTSCF interrupt enable 0: CTSCF interrupt disabled 1: CTSCF interrupt 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 line 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.

12.13.7 Guard time and divider register (GDIV)

Bit	Register	Reset value	Type	Description
Bit 31:16	Reserved	0x0000	resd	Forced to 0 by hardware.
Bit 15:8	SCGT	0x00	rw	Smart card guard time 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 invalid 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 ……

12.13.8 Receiver timeout detection register (RTOV)

Bit	Register	Reset value	Type	Description
Bit 31:24	Reserved	0x00	resd	Forced to 0 by hardware.
Bit 23:0	RTOV	0x00	rw	Receiver time out value The unit is 1 bit width.

12.13.9 Interrupt flag clear register (IFC)

Bit	Register	Reset value	Type	Description
Bit 31:21 Bit 19:18 Bit 16:12 Bit 10:0	Reserved	0x00	resd	Forced to 0 by hardware.
Bit 20	LPWUFC	0	w1	Low power wake up flag clear Note: USART4 to USART8 do not support this bit (0 by default)
Bit 17	CMDFC	0	w1	Character match detection flag clear
Bit 11	RTODFC	0	w1	Receiver time out detection flag clear

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 configuration procedure of SPI used as SPI or I²S.

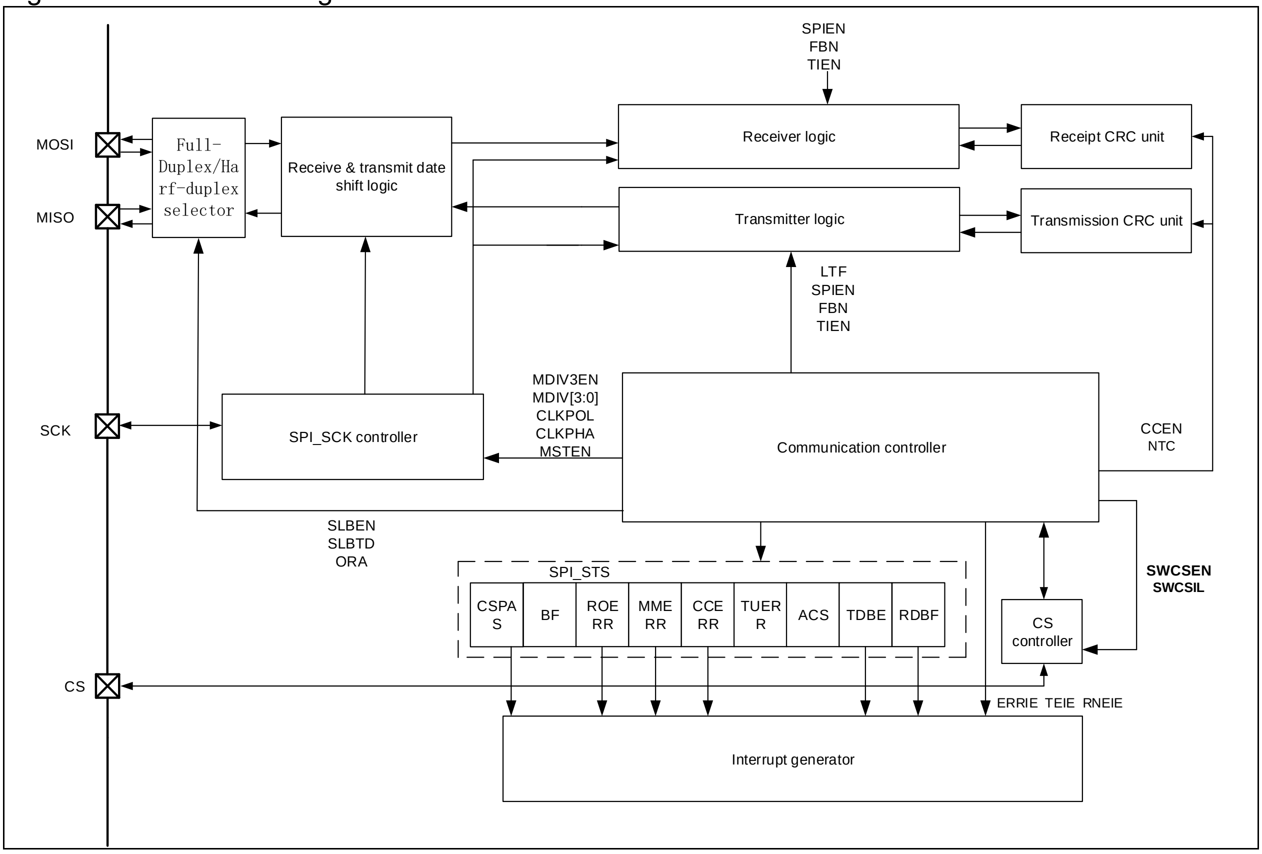
13.2 Functional overview

13.2.1 SPI description

The SPI can be configured in host or slave mode depending on software configuration. It is capable of operating in full-duplex, full-duplex receive-only, and half-duplex transmit-only/receive-only modes, with DMA capability and automatic CRC calculation and check functions. The SPI interface can be configured by software to be compatible with TI protocol.

SPI block diagram:

Figure 13-1 SPI block diagram



Main features as SPI:

- Programmable full-duplex or half-duplex communication
  — Full-duplex synchronous communication (supporting receive-only mode to free the transmit IO for other purposes)
  — Half-duplex synchronous communication (transfer direction is configurable by software: receive or transmit)
- Programmable master or slave mode
- Programmable CS signal handling
  — CS signal handling by hardware
  — CS signal handling by software

- Programmable 8-bit or 16-bit frame format
- Programmable communication frequency and prescaler (prescaler up to $f_{PCLK}/2$)
- Programmable clock polarity and phase
- Programmable data transfer order (MSB-first or LSB-first)
- Programmable error interrupt flags (CS pulse error, 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 check
- Busy status flag
- Compatible with the TI protocol

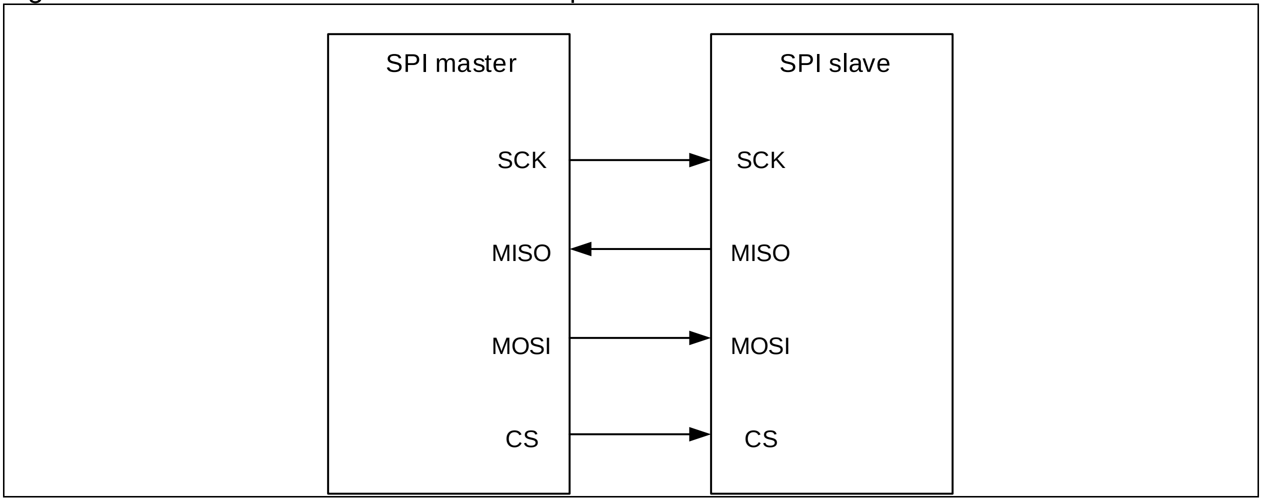
13.2.2 Full-duplex/half-duplex selector

When used as an SPI interface (through software configuration), it is capable of operating in 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=0 and the ORA=0. In this case, the SPI supports simultaneous data transmission and reception. IOs are connected as follows:

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



In both master and 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=0 and ORA=1. In this case, the SPI can be used as data receiver (transmission is not available). In master mode, the MISO pin receives data and the IO mapped onto MOSI is released. In slave mode, the MOSI pin receives data and the IO mapped onto MISO is released.

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

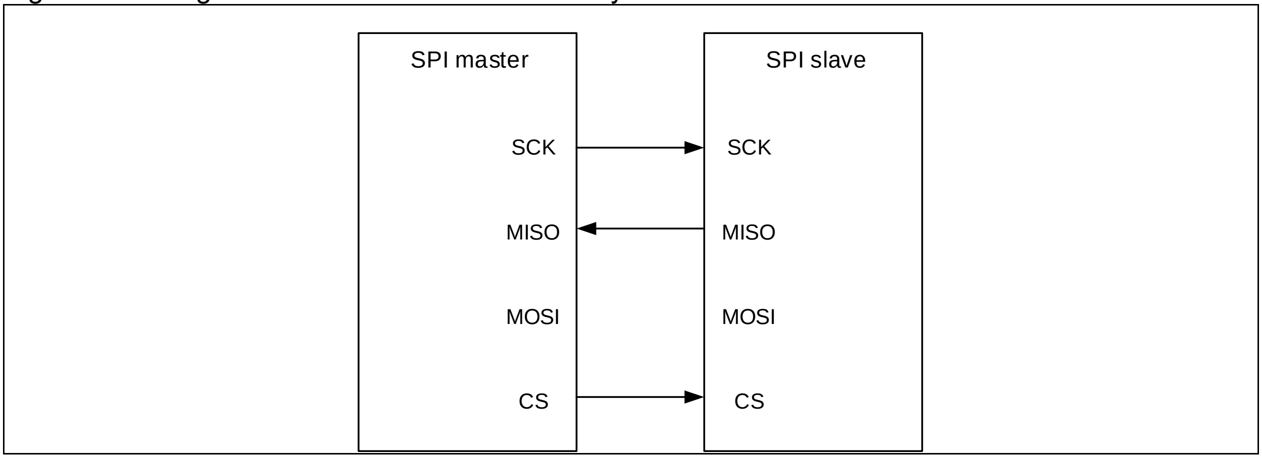
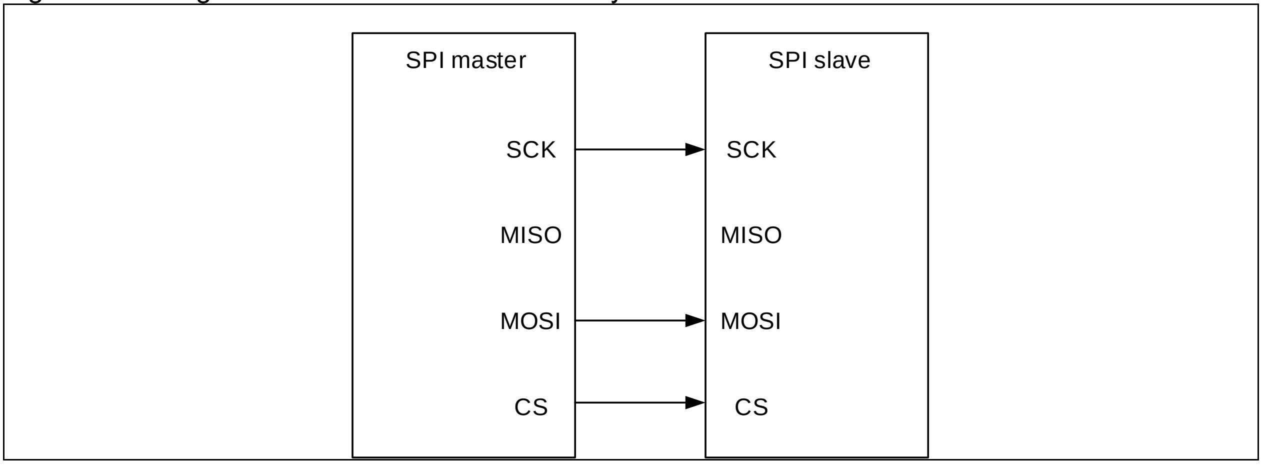


Figure 13-4 Single-wire unidirectional receive only in SPI slave mode



In master mode, it is required to wait until the second to last RDBF bit is set and then one SPI_CPK clock before disabling the SPI. The last RDBF must be set to 1 before entering power-saving mode (or disabling SPI system clock).

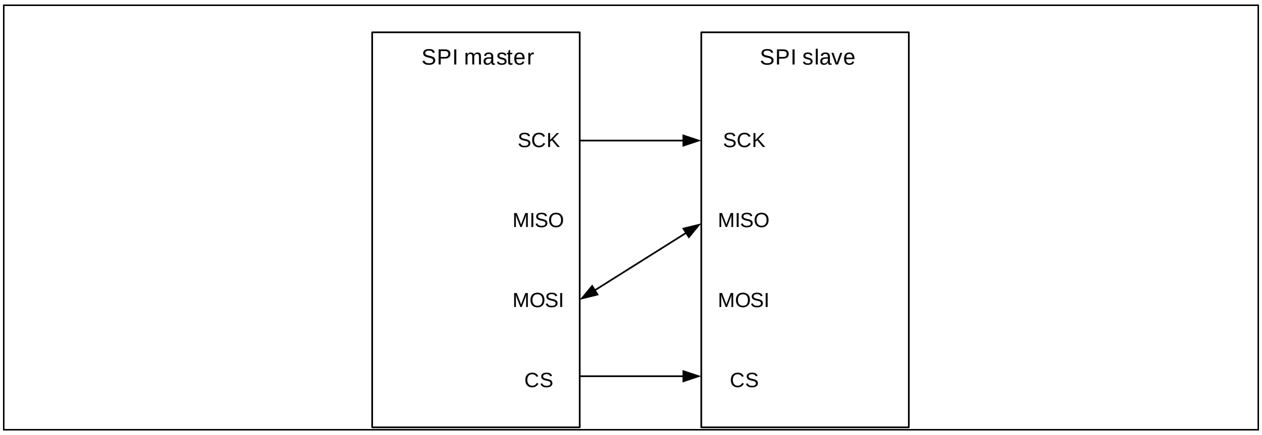
In slave mode, there is no need to check any flag before disabling the SPI. However, it is required to wait until the BF becomes 0 before entering power-saving mode.

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

When SLBEN=1, 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/ receives data, and IO mapped onto MISO pin is released. In slave mode, the MISO pin transmits/receives data, and IO mapped onto MOSI pin is released.

The SLBTD bit is used by software to configure transfer direction. When SLBTD=1, the SPI can be used only for data transmission; when SLBTD=0, the SPI is used for data reception only.

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



In both master and slave mode, when the SPI is selected for data transmission in single-wire bidirectional half-duplex mode, 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 one SPI_SCK cycle before disabling the SPI. And the last RDBF must be set to 1 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 flag 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 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, the CS hardware control is enabled by setting HWCSOE=1 and SWCSEN=0. 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, the CS hardware control is enabled by setting HWCSOE=0 and SWCSEN=0. 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 accordingly. An interrupt is generated if ERRIE=1. During the period of MMERR being 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, the CS hardware control is enabled by setting HWCSOE=0 and SWCSEN=0. The slave determines 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 accordingly. 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 via the SPI interface, 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 configuration of SPI_SCK, with the configuration procedure detailed as follows:

SPI_SCK controller configuration procedure:

- Clock polarity and clock phase selection: by setting the CLKPOL and CLKPHA bit.
- Clock prescaler selection: Select the desired PCLK frequency by setting the CRM bit, and 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 there until when the SPI is disabled and the reception is complete.

13.2.5 CRC

The SPI interface provides separate CRC calculation unit for transmission and reception. When used as SPI through software configuration, the automatic CRC calculation and check is performed while the user is reading or writing through DMA or CPU. During 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 CRC value is 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, the CRC calculation and check in CPU reception mode will be completed automatically by following CPU transmission mode to operate the NTC bit,

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 already 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 plus 1. 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 continue to perform CRC automatic check. Note that the received CRC data will be moved into the SPI_DT register by hardware, the RDBF is set, and DMA read request will be issued if then DAM transfer feature is enabled. Hence, it is recommended to read the SPI_DT register by software 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 TI mode

The SPI interface is compatible with the TI protocol. The TI mode is enabled by setting the TIEN bit to 1.

In this mode, the SPI interface will generates a communication clock SPI_CLK in accordance with the TI protocol. This means that the SPI_CLK polarity and phase are forced to conform to the TI protocol requirements, without the need of the intervention of CLKPOL and CLKPHA bits. Thus the CLKPOL and CLKPHA bits cannot be used to change the polarity and phase of the SPI_CLK either.

In this mode, the SPI interface will generate a CS signal in accordance with the TI protocol, meaning that the CS input and output are forced to conform to the TI protocol requirements, without the need of the intervention of SWCSEN, SWCSIL and HWCSOE bits. Thus, the SWCSEN, SWCSIL and HWCSOE bits cannot be used for CS signal management either.

In slave mode, once the TI mode is enabled, the SPI slave controls the MISO pin only during data transmission, meaning that the MISO pin state remains Hi-Z in idle state.

In slave mode, once the TI mode is enabled, the SPI interface is capable of detecting CS pulse errors during data transmission, and setting the CSPAS bit (It is cleared by reading the SPI_STS) as soon as a CS pulse error is detected. At this point, the detected erroneous pulse will be ignored by the SPI. However, since there is something wrong with the CS signal, the software should disable the SPI slave and re-configure the SPI master before re-enabling the SPI slave for communication.

13.2.8 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 which 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

13.2.9 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 while the previously received data has not been read yet (RDBF=1), then the data overflow occurs. The previously receive data are 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

13.2.10 Motorola mode

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

Full-duplex communication – master mode

For host side, configured as follows:

MSTEN=1: Master enable

SLBEN=0: Full-duplex mode

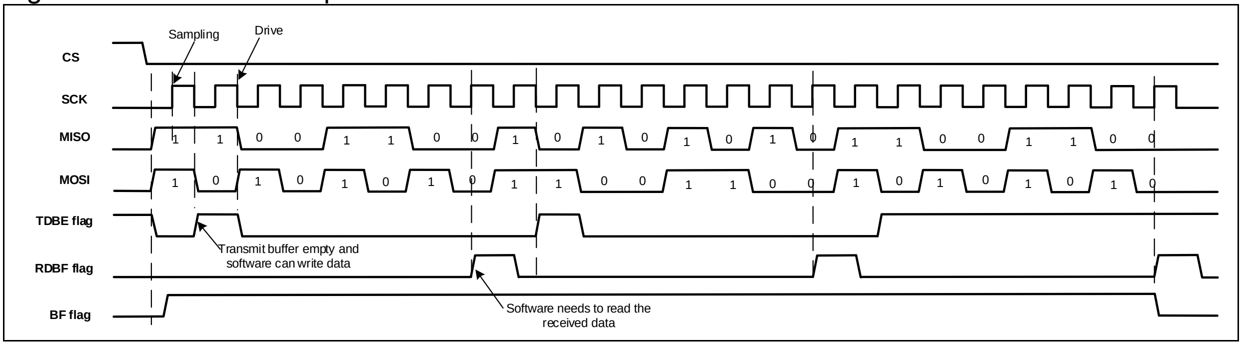
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

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

For slave side, configured as follows:

MSTEN=0: Slave enable

SLBEN=0: Full-duplex mode

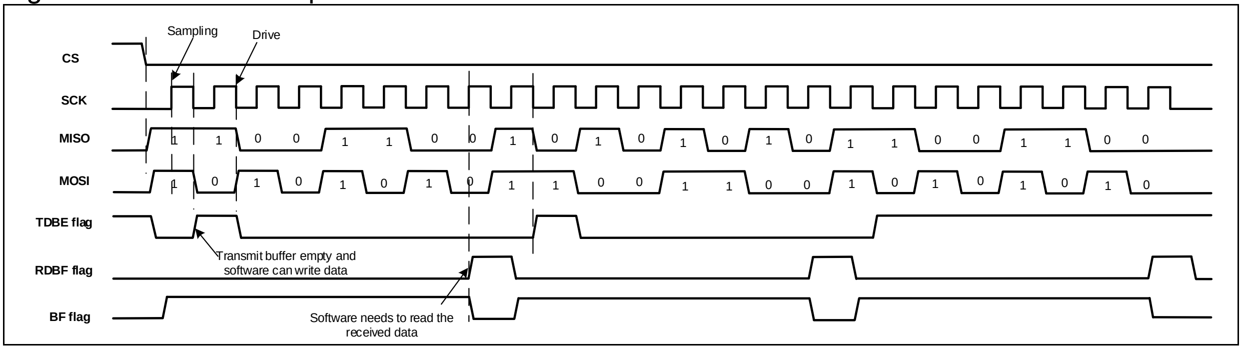
CLKPOL=0, CLKPHA=0: SCK idle output low, use the first edge for sampling

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 timing

Configured as follows:

MSTEN=1: Master enable

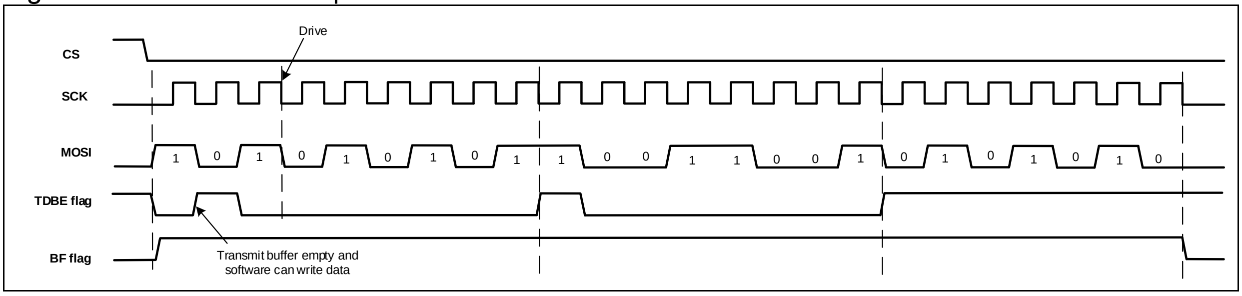
SLBEN=1: Single line bidirectional mode

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

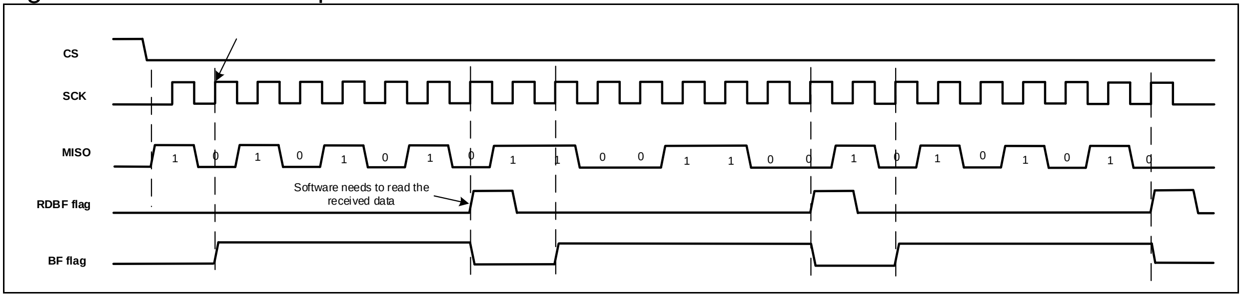
SLBTD=0: Receive mode

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

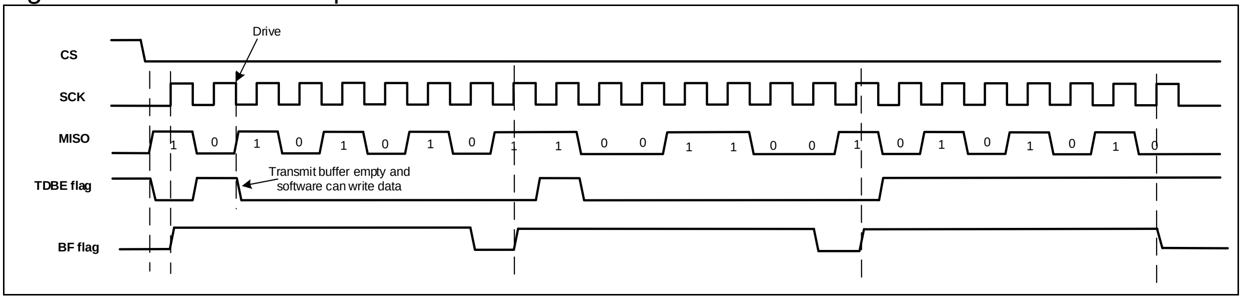
SLBTD=1: Transmit enable

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

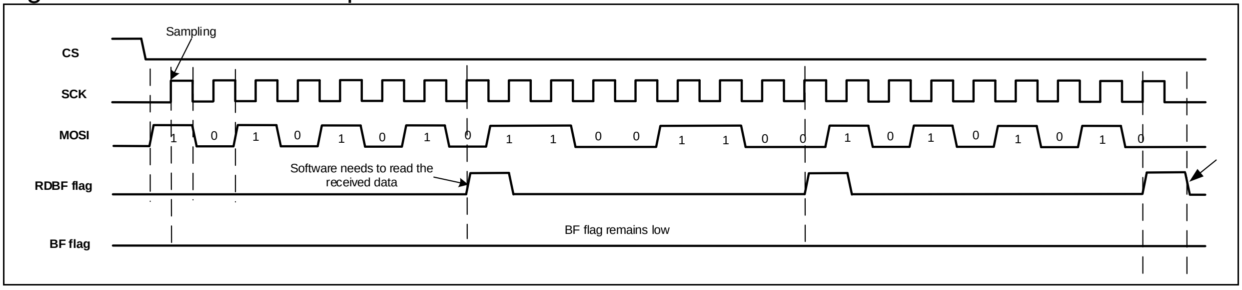
SLBTD=0: Receive enable

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 Master half-duplex receive

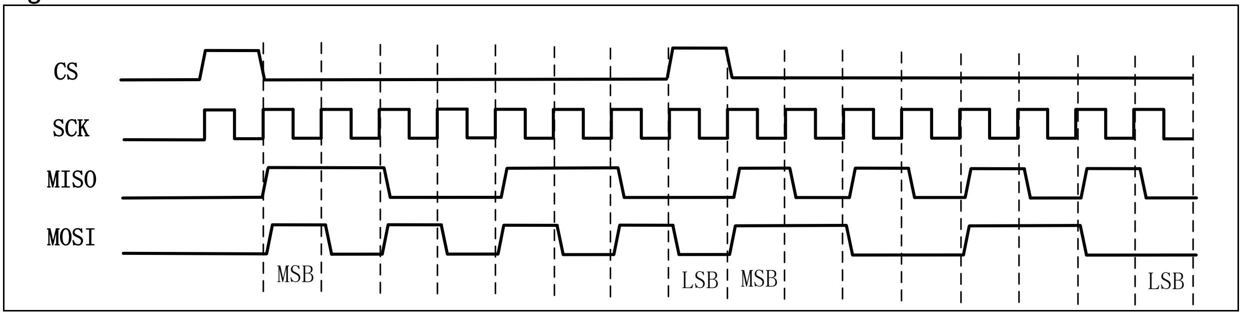


13.2.11 TI mode

The SPI interface supports TI mode. This mode is enabled by setting TIEN=1.

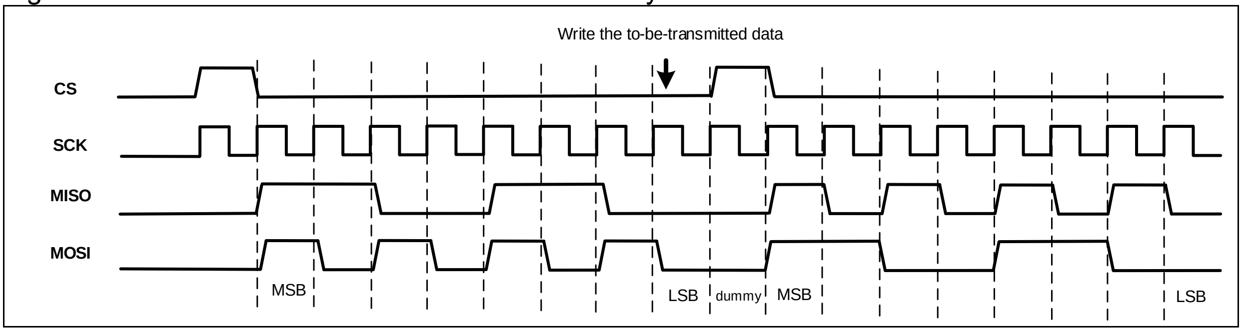
In TI mode, a slight difference is present between continuous and discontinuous communication timings. When the to-be transmitted data is written before the rising SCK edge corresponding to the last data of the current transmit frame, it is a continuous communication, without dummy CLK between data, and the host sends a valid CS pulse while transmitting the last data of the current frame.

Figure 13-12 TI mode continous transfer



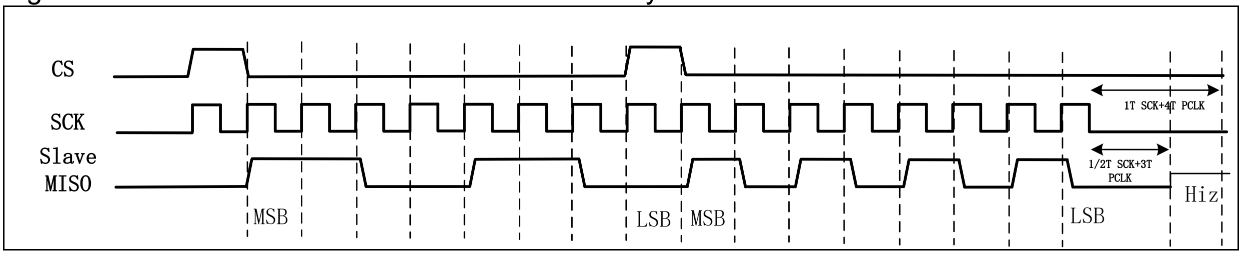
In TI mode, when the to-be-transmitted data is written between the rising and falling SCK edge corresponding to the last data of the current transmit frame, a dummy CLK exists between data.

Figure 13-13 TI mode continous transfer with dummy CLK



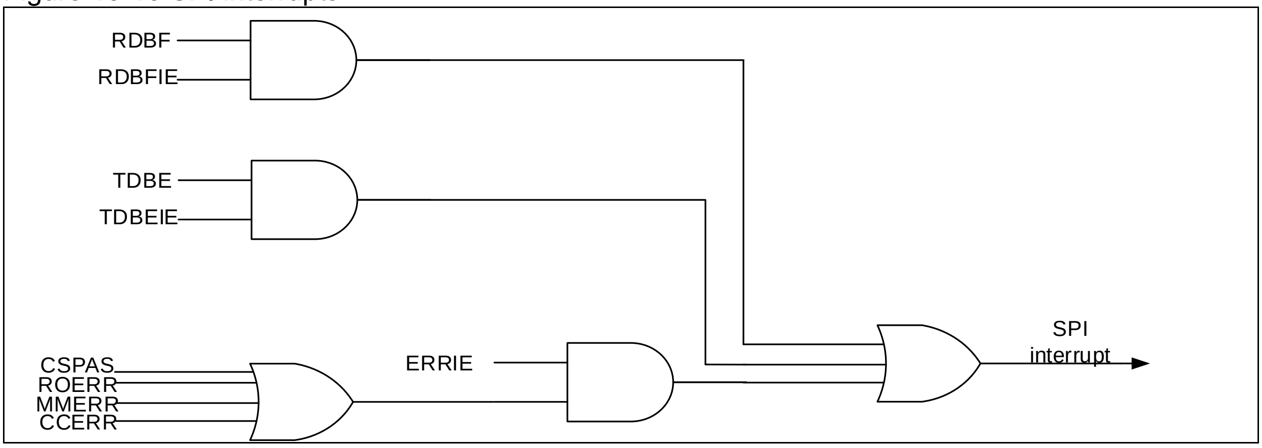
In TI mode, when the to-be-transmitted data is written after the falling SCK edge corresponding to the last data of the current transmit frame, the host always issues a valid SCK clock after 1T SCK + 4T PCLK cycles. If the slave still does not detect a valid CS pulse at the end of the current data reception, it disables MISO output after 1/2T SCK + 3T PCLK cycles to control MISO floating.

Figure 13-14 TI mode continous transfer with dummy CLK



13.2.12 Interrupts

Figure 13-15 SPI interrupts



13.2.13 IO pin control

When used as SPI, the SPI interface is connected to external devices through four pins.

- MISO: Master In/Slave Out. The pin is used to receive data from slave in SPI master mode, and transmit data in slave mode.
- MOSI: Master Out/Slave In. The pin is used to transmit data in SPI master mode, and receive data in slave mode.
- SCK: SPI communication clock. The pin serves as output (communication clock is sent to peripheral via this pin) in SPI master mode, and as input (communication clock from master is input to SPI via this pin) in SPI slave mode.
- CS: Chip Select. This is an optional pin which is used to select master/slave mode. Refer to CS section for more information.

13.2.14 Precautions

- CRC value should be read by software reading DT register at the end of CRC reception

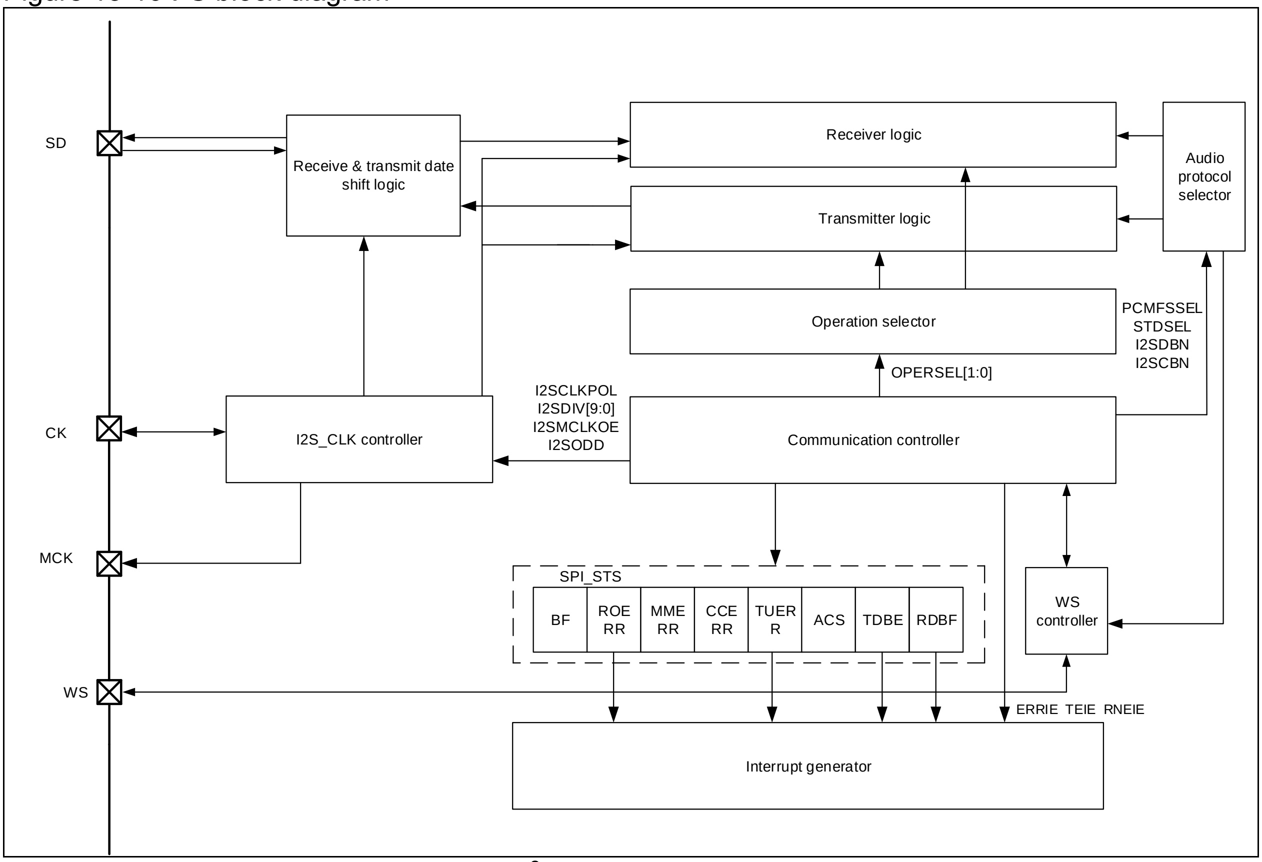
13.3 I²S functional description

13.3.1 I²S introduction

The I²S is capable of operating in master receive, master transmit, and slave receive and slave transmit, depending on software configuration. These four operating modes support four audio protocols including Philips standard, MSB-aligned standard, LSB-aligned standard and PCM standard, respectively. The DMA transfer is also supported.

The combination of two I²S interfaces can be used to support I²S full-duplex mode. Refer to I²S full-duplex section for more information.

Figure 13-16 I²S block diagram



Main features when the SPI is used as I²S:

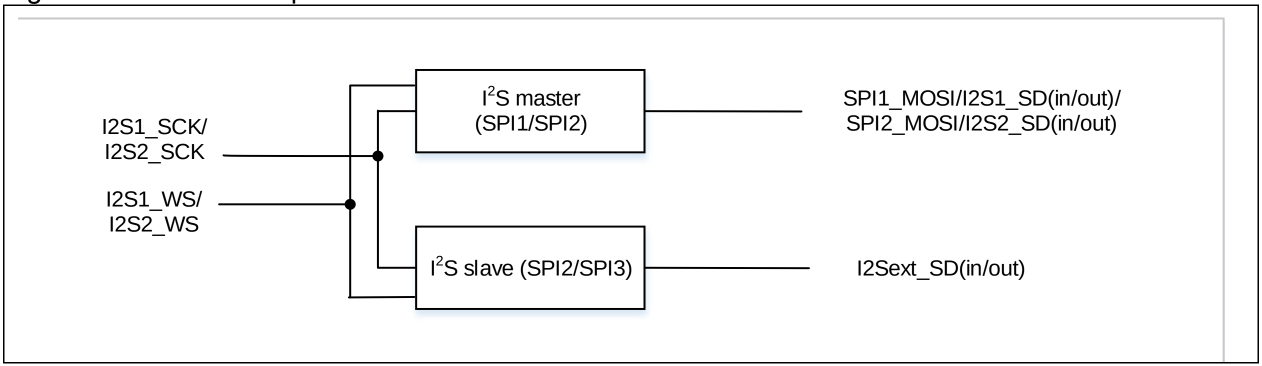
- Programmable operating modes
  - Slave device transmission
  - Slave device reception
  - Master device transmission
  - Master device reception
- Programmable clock polarity
- Programmable clock frequency (8 KHz to 192 KHz)
- Programmable data bits (16 bit, 24 bit, 32 bit)
- Programmable channel bits (16 bit, 32 bit)
- Programmable audio protocol
  - I²S Philips standard
  - MSB-aligned standard (left-aligned)
  - LSB-aligned standard (right-aligned)
  - PCM standard (channel frame with short and long frame synchronization)
- I²S full-duplex
- DMA transfer

- Main peripheral clock with a fixed frequency of 256x Fs (audio sampling frequency)

13.3.2 I²S full-duplex

Two SPIs can be combined to support I²S full-duplex mode through the SCFG_CFG2[31:30] bit in the SCFG register. Of the three SPIs, either SPI1 or SPI2 can be configured as full-duplex master, while the SPI2 or SPI3 can be set as full-duplex slave, which is selected through the SCFG_CFG2[31:30] bit in the SCFG register. Once selected (combining two SPIs to achieve I²S full-duplex mode), the IO remap relations of the master remains unchanged, while the SCK and WS of the slave are connected to the SCK and WS of the master internally, with the SD line of the slave remapped onto the I2S_SDEXT. The slave's original IO remap relations become invalid, keeping the corresponding IOs free for other purposes.

Figure 13-17 I²S full-duplex structure



I²S full-duplex master side:

It supports master or slave mode. It can programmed as a receiver or transmitter.

- I2Sx_WS takes part in communication for actual WS signal interaction
- I2Sx_SCK takes part in communication for actual clock signal interaction
- I2Sx_SD takes part in communication for data and information interaction of the master side

I²S full-duplex slave side:

It supports slave mode only. It can be programmed as a transmitter or receiver.

- I2Sy_WS does not take part in communication, releasing the corresponding IOs for other purposes
- I2Sy_SCK does not take part in communication, releasing the corresponding IOs for other purposes
- I2Sy_SD does not take part in communication, releasing the corresponding IOs for other purposes
- I2S_SDEXT takes part in communication for data and information interaction of the slave side

Note: x can be 1 or 2, whereas y can be 2 or 3.

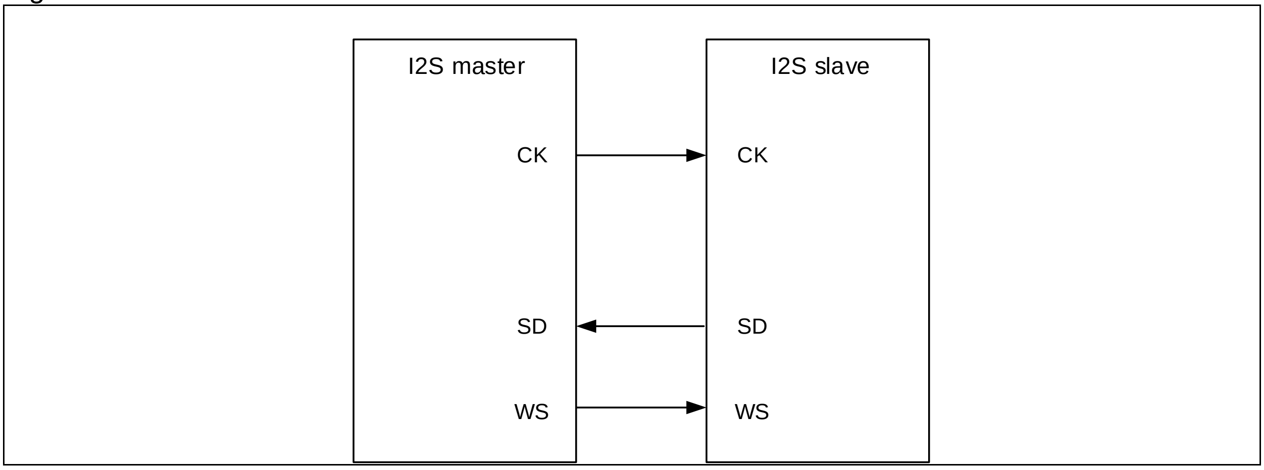
13.3.3 Operating mode selection

The SPI, used as I²S selector, offers multiple operating modes for selection, namely, slave device transmit, slave device receive, master device transmit and master device receive. 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.

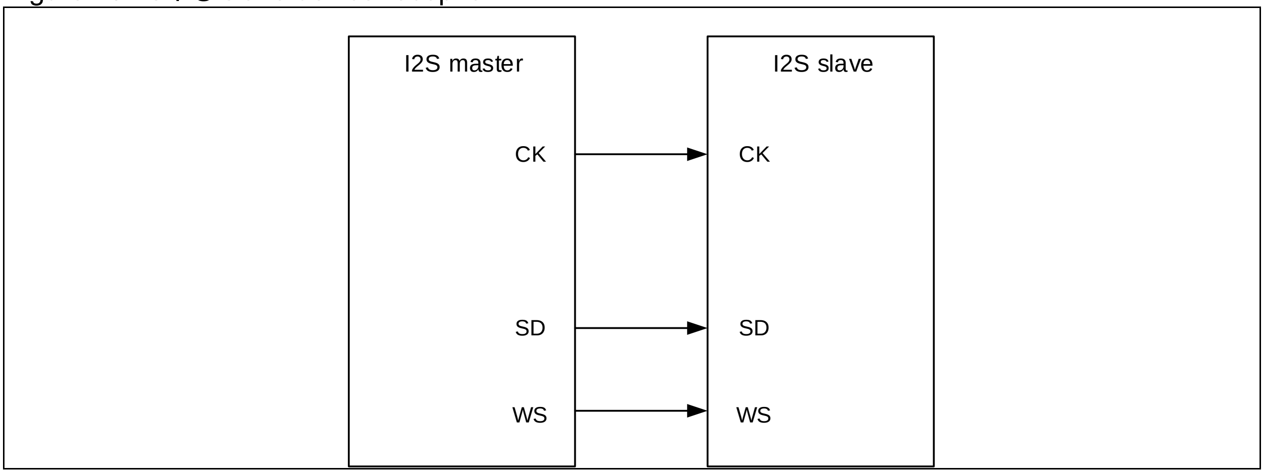
Figure 13-18 I²S slave device transmission



Slave device reception:

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

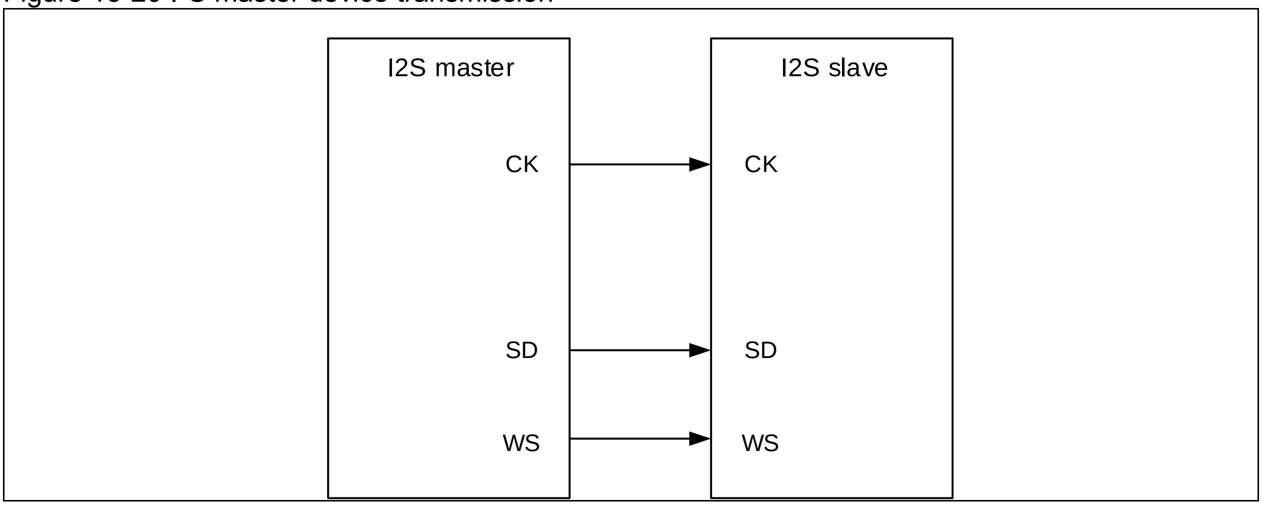
Figure 13-19 I²S slave device reception



Master device transmission:

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

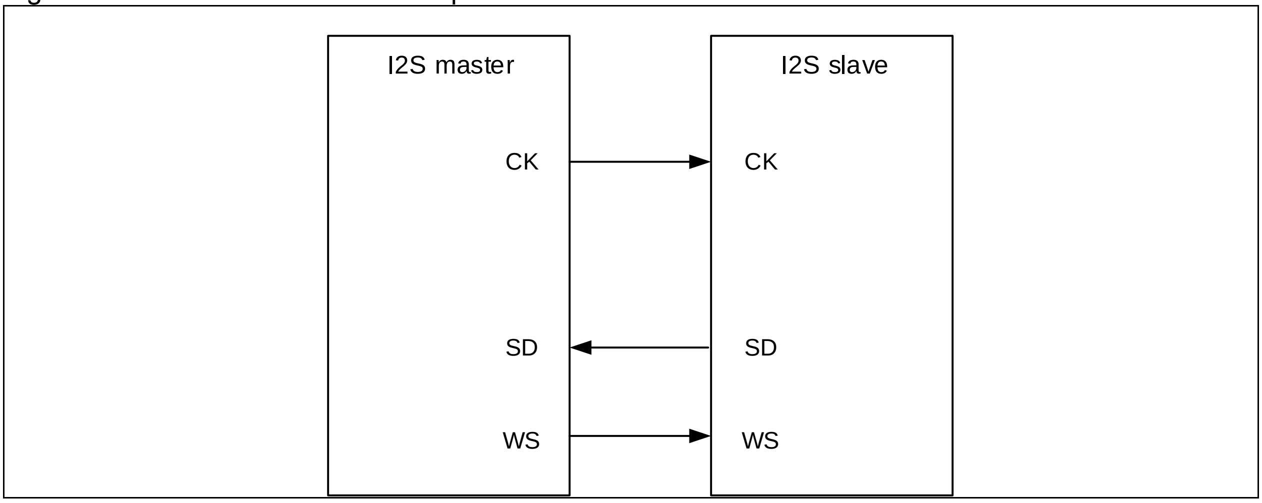
Figure 13-20 I²S master device transmission



Master device reception:

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

Figure 13-21 I²S master device reception



13.3.4 Audio protocol selector

As I²S interface, the SPI supports multiple audio protocols. The user is able to select the desired audio protocol, the number of data bits and of channel bits through the audio protocol selector by software. By controlling the WS controller automatically, the audio protocol selector outputs or detects WS signals that conform to 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 the number of data bits by setting the I2SDBN bit
  I2SDBN=00: 16 bit
  I2SDBN =01: 24 bit
  I2SDBN =10: 32 bit
- Select the number of 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 protocol, 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 bits are the same as the channel bits. Each channel requires only 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 bits are different from the channel bits. Each channel requires only one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The first 16 bits are valid data, and the remaining 16-bit are forced to 0 by hardware.
- **Philips standard, PCM standard or MSB-aligned standard, 24-bit data and 32-bit channel**
  The data bits are different from the change bits. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. The first 16-bit channel transmits and receives the first 16-bit data, while the last 16-bit channel transmits and receives the upper 8-bit data, and the lower 8-bit data are forced to 0 by hardware.
- **Philips standard, PCM standard, MSB-aligned or LSB-aligned standard, 32-bit data and 32-bit channel**

The data bits are the same as the channel bits. 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 (transmit and reception) in two times, with 16-bit data each time.

- **LSB-aligned standard, 16-bit data and 32-bit channel**

  The data bits are different from the channel bits. Each channel requires only one read/write operation from/to the SPI_DT register, and the number of DMA transfer is 1. The last 16 bits (LSB) are valid data, 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 bits are different from the channel bits. Each channel requires two read/write operations from/to the SPI_DT register, and the number of DMA transfer is 2. Of the first 16-bit data, its lower 8 bits are valid data, and the upper 8 bits are forced to 0 by software; the last 16 bits transmit and receive the second 16-bit data.

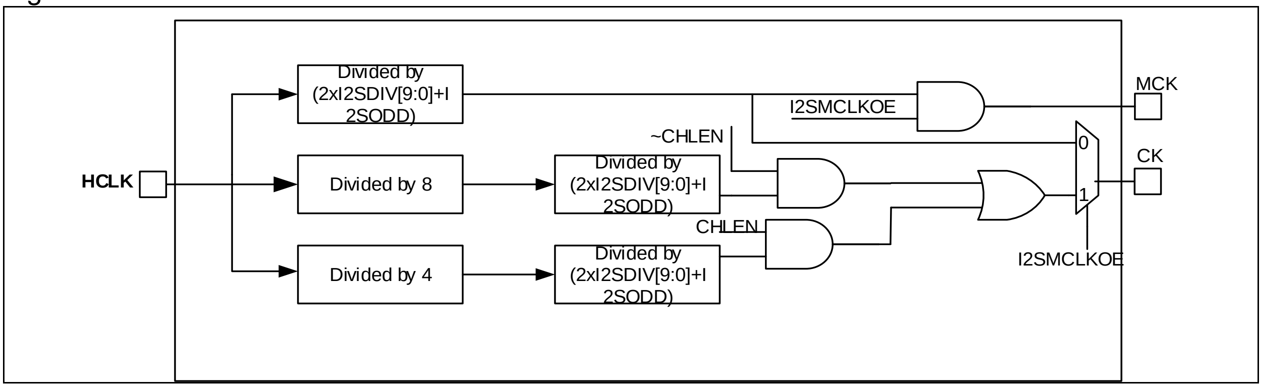
13.3.5 I2S_CLK controller

As I^2S, 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 configuration of I2S_SCK, with the configuration procedure detailed as follows:

In I^2S master mode, the SPI provides communication clock (CK) and main peripheral clock (MCK) shown in *Figure 13-22*. The CK and MCK are generated by HCLK divider, and the MCK frequency division factor depends on the I2SDIV and I2SODD. The calculation formula is seen in *Figure 13-22*.

The CK frequency division factor depends on whether to provide the main clock for peripherals. To ensure that the main clock is always 256 times the audio sampling frequency, the provision of main clock and the number of 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 frequency division factor as that of the MCK, that is the final communication clock; When the main clock is not needed, the CK frequency division factor is determined by I2SDIV and I2SODD, shown in *Figure 13-22*.

Figure 13-22 CK & MCK source in master mode



In addition to the above 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 I2S DIV	16bit I2S_ODD	16bit RealFs	16bit Error	32bit I2S DIV	32bit I2S_ODD	32bit RealFs	32bit Error
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%

13.3.6 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 a 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.

13.3.7 Transmitter/Receiver

Whether 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 I^2S. Any operations related to CRC, including CCERR flag and corresponding interrupts, are not supported.
- I^2S 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 I^2S 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 cycles before disabling the I^2S.
  — I2SDBN=00, I2SCBN=1, STDSLE=00 or STDSLE=01 or STDSLE=11: wait for the last RDBF=1 and one CK cycles before the I^2S.
  — I2SDBN, I2SCBN, STDSLE combination: wait for the second-to-last RDBF=1 and one CK cycles before disabling the I^2S.

I^2S transmitter configuration procedure:

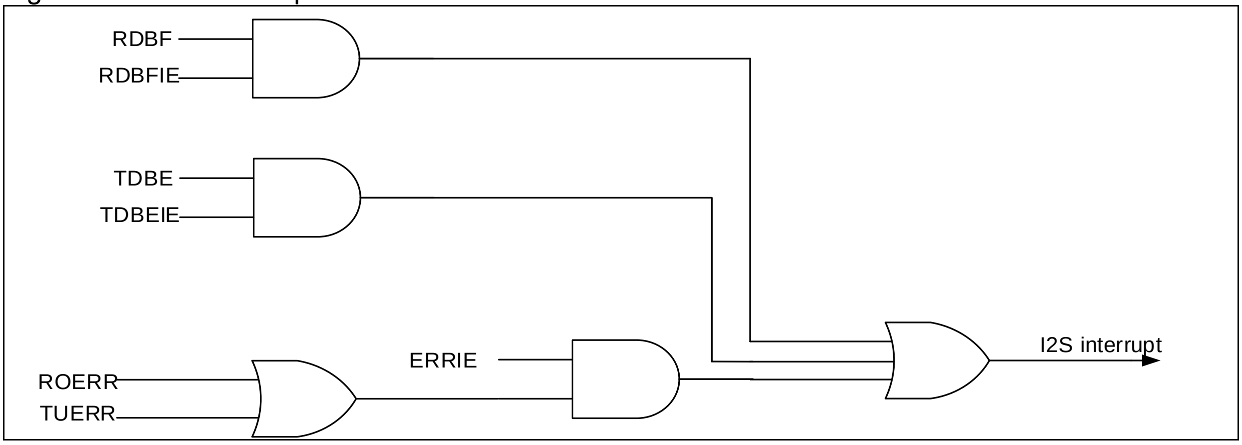
- Configure operation mode selector
- Configure audio protocol selector
- Configure I2S_SCK controller
- Configure DMA transfer (if necessary)
- Set the I2SEN bit to enable I^2S
- Follow above steps to configure the I^2SxEXT (For I^2S full-duplex mode)

I^2S 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 I^2S
- Follow above steps to configure the I^2SxEXT (For I^2S full-duplex mode)

13.3.8 Interrupts

Figure 13-23 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 there is a need to provide main clock for peripherals. Considering the SPI interface cannot be used as I²S and SPI at the same time, the I²S shares some pins with 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 or words (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 I²S mode.

13.4.2 SPI control register2 (SPI_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Forced to 0 by hardware.
Bit 9	MDIV3EN	0x0	rw	Master clock frequency divided by 3 enable 0: Disabled 1: Enabled Note: When this bit is set, the MDIV[3: 0] becomes invalid, and the SPI clock is forced to be PCLK/3.
Bit 8	MDIV[3]	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	TIEN	0x0	rw	TI mode enable 0: TI mode disabled (Motorola mode) 1: TI mode enabled (TI mode) Note: This mode is not used in I2S mode. It must be 0 in I2S mode.
Bit 3	Reserved	0x0	resd	Kept at 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: 9	Reserved	0x00	resd	Forced to 0 by hardware
Bit 8	CSPAS	0x0	ro	CS pulse abnormal setting flag 0: CS pulse flag normal 1: CS pulse flag is set abnormally Note: This bit is used for TI slave mode. It is cleared by reading the STS register.
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^{2}S 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^{2}S 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) (Not used in I^{2}S mode)

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) (Not used in I^2S mode)

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 register (SPI_I2SCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 12	Reserved	0x0	resd	Forced to 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 default value
Bit 5: 4	STDSEL	0x0	rw	I^2S 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	I^2S 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	I^2S 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	I^2S channel bit num This bit can be configured only when the I^2S 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 0 by hardware.
Bit 9	I2SMCLKOE	0x0	rw	I^2S Master clock output enable 0: Disabled 1: Enabled
Bit 8	I2SODD	0x0	rw	Odd factor for I^2S division 0: Actual divider factor =I2SDIV*2 1: Actual divider factor =(I2SDIV*2)+1
Bit 11: 10 Bit 7: 0	I2SDIV	0x02	rw	I^2S division It is not allowed to configure I2SDIV[9: 0]=0 or I2SDIV[9: 0]=1

14 Timer

AT32F423 timers include basic timers, general-purpose timers, and advanced timers.

Please refer to *Section 14.1* ~ *Section 14.5* for 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	ETR input	Break input
Advanced-control timer	TMR1	16	Up Down Up/Down	16-bit	1~65536	O	4	O	O	O
General-purpose timer	TMR2	16/32	Up Down Up/Down	X	1~65536	O	4	O	O	X
	TMR3 TMR4	16	Up Down Up/Down	X	1~65536	O	4	O	O	X
	TMR9 TMR12	16	Up Down Up/Down	8-bit	1~65536	O	2	O	X	O
	TMR10 TMR11 TMR13 TMR14	16	Up Down Up/Down	8-bit	1~65536	O	1	X	X	O
Basic timer	TMR6 TMR7	16	Up	X	1~65536	O	X	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 time	TMR2	16/32	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization /ADC/DAC
	TMR3 TMR4	16	Up Down Up/Down	O	O	X	X	O	O	Timer synchronization /ADC/DAC
	TMR9 TMR12	16	Up Down Up/Down	O	O	O	O	O	X	Timer synchronization ADC/DAC
	TMR10 TMR11 TMR13 TMR14	16	Up Down Up/Down	O	O	O	O	X	X	Timer synchronization
Basic timer	TMR6 TMR7	16	Up	X	X	X	X	X	X	ADC/DAC

14.1 Basic timer (TMR6 and TMR7)

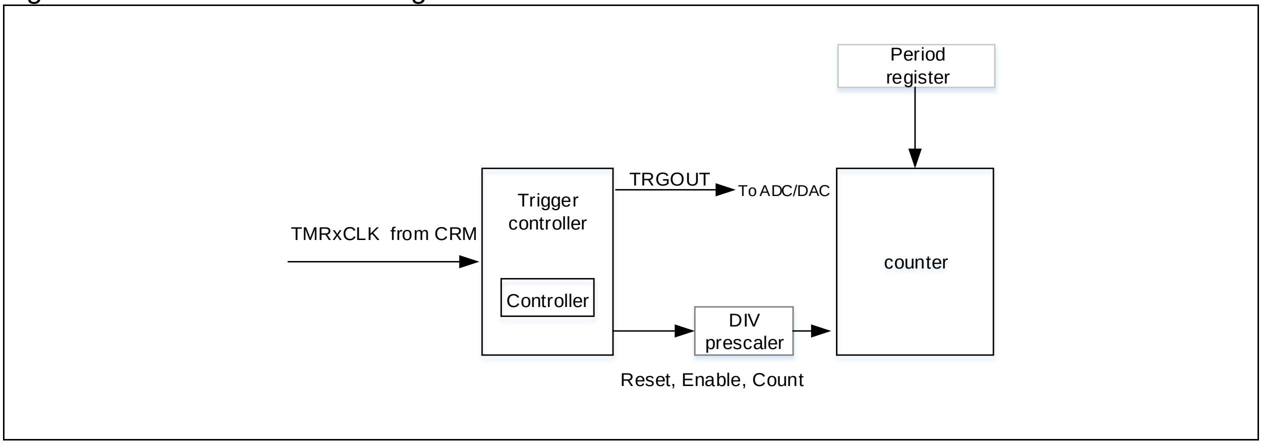
14.1.1 TMR6 and TMR7 introduction

Basic timers (TMR6 and TMR7) include a 16-bit up counter and the corresponding control logic, without being connected to external I/Os. They can be used for basic timing function.

14.1.2 TMR6 and TMR7 main features

- 16-bit auto reload upcounter
- 16-bit prescaler used to divide the TMR_CLK clock frequency by any factor between 1 and 65536
- Synchronization circuit to trigger the DAC (unique features of TMR6 and TMR7)

Figure 14-1 Basic timer block diagram

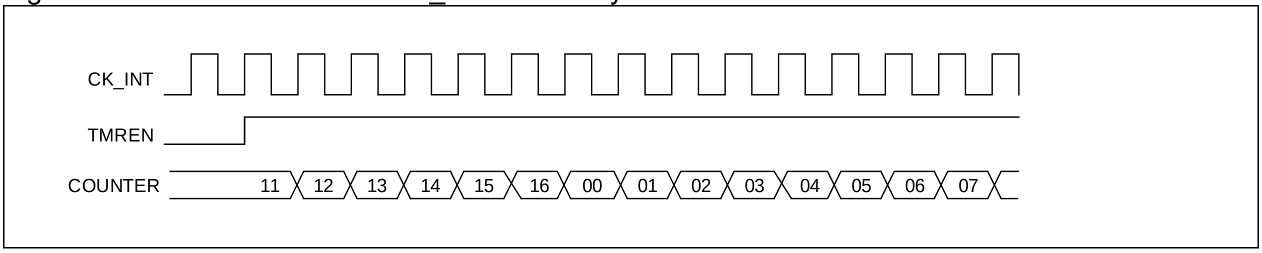


14.1.3 TMR6 and TMR7 function overview

14.1.3.1 Counting clock

The counter clock of TMR6 and TMR7 is provided by the internal clock source (CK_INT) divided by prescaler. 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.

Figure 14-2 Control circuit with CK_INT divided by 1



14.1.3.2 Counting mode

The basic timer only supports upcounting mode. It has an internal 16-bit counter.

The TMRx_PR register is used to set counting period of the counter. 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.

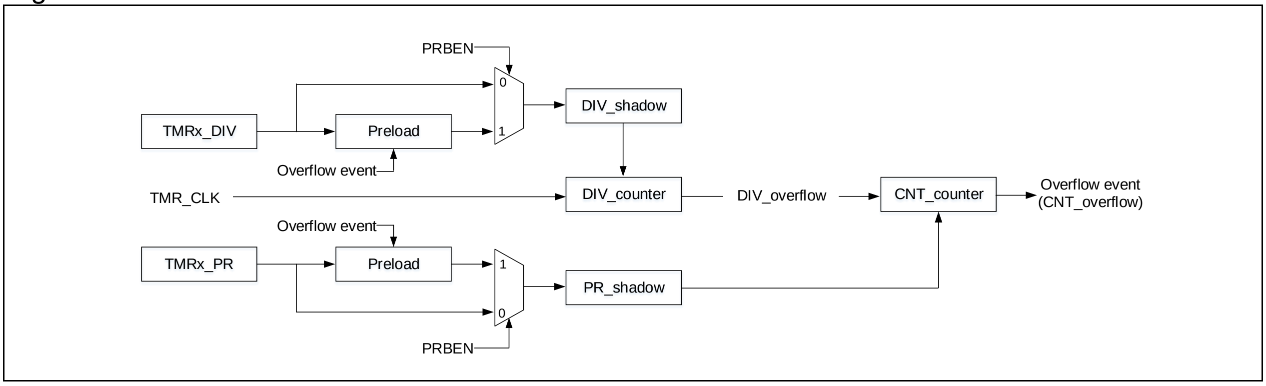
TMRx_DIV register is used to define the counting frequency of the counter. The counter counts once every DIV[15:0]+1 clock cycle. Similar to TMRx_PR register, after periodic buffer is enabled, the value of the TMRx_DIV register is transferred into the shadow register upon an overflow event.

Reading the TMRx_CNT register returns the current counter value. Writing the TMRx_CNT register will update the current counter value.

An overflow event is enabled by default. It can be disabled by setting OVFEN=1 in the TMRx_CTRL1 register. The OVFS bit in the TMRx_CTRL1 register is used to select the source of an overflow event, which is, by default, counter overflow or underflow, setting OVFSWTR, reset signal generated by slave mode timer controller in reset mode. Once the OVFS is set, an overflow event is generated only when overflow or underflow occurs.

Setting the TMREN bit (TMREN=1) enables the timer to start counting. Base on synchronization logic, however, the actual counter enable signal TMR_EN is set 1 clock cycle after the TMREN is set.

Figure 14-3 Basic structure of a counter



Upcounting mode

In upcounting 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 setting OVFIF=1 at the same time. If the overflow event is disabled, the counter is no longer reloaded with a prescaler value and a periodic value when a counter overflow event occurs, otherwise, the counter is updated with prescaler and periodic values at an overflow event.

Figure 14-4 Overflow event when PRBEN=0

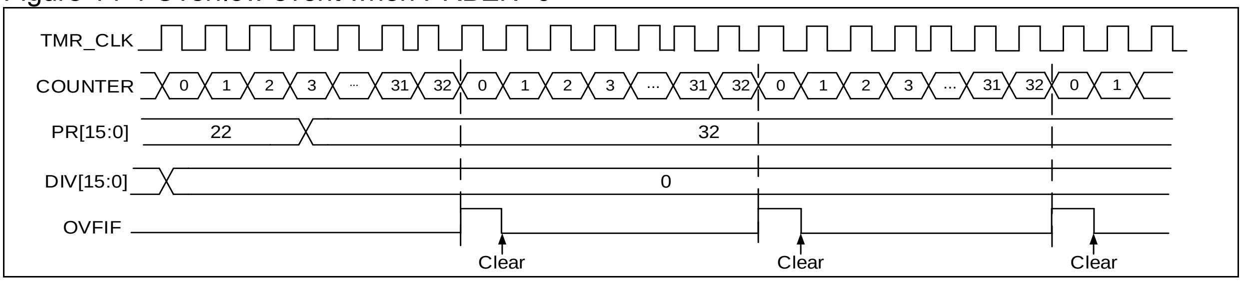


Figure 14-5 Overflow event when PRBEN=1

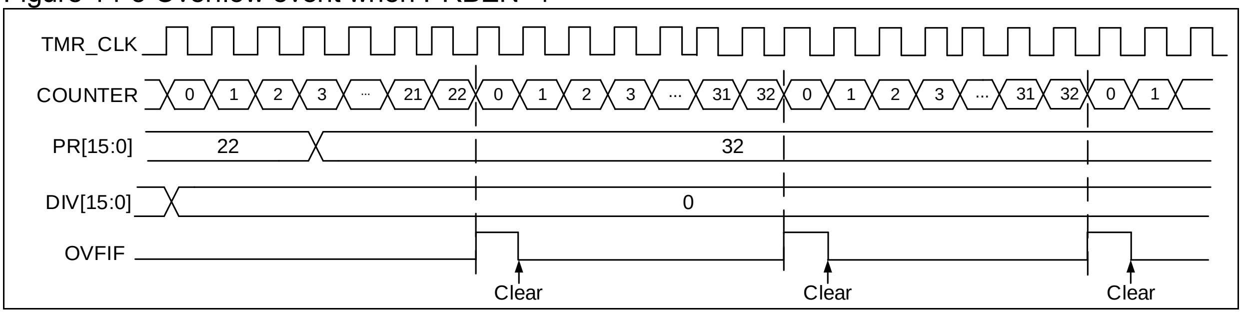
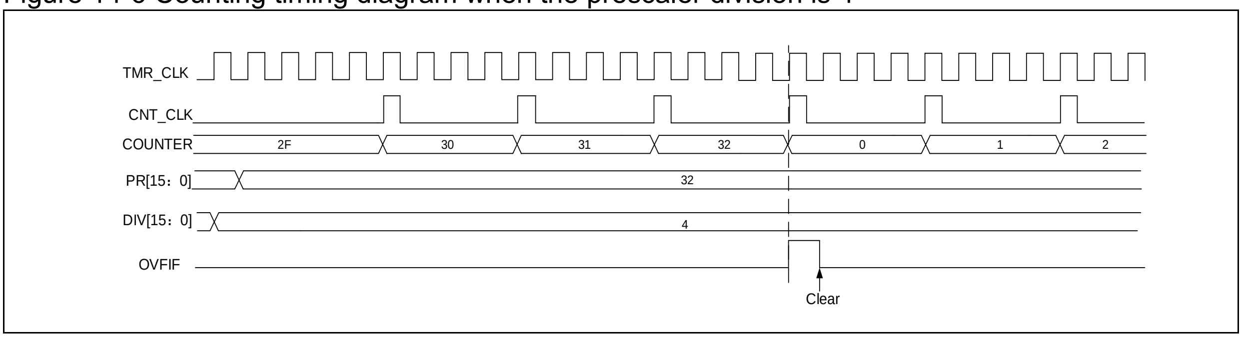


Figure 14-6 Counting timing diagram when the prescaler division is 4



14.1.3.3 Debug mode

When the microcontroller enters debug mode (Cortex ®-M4F core halted), the TMRx counter stops counting when the TMRx_PAUSE bit is set to 1.

14.1.4 TMR6 and TMR7 registers

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

In Table 14-2, all the TMR6 and TMR7 registers are mapped to a 16-bit addressable space.

Table 14-2 TMR6 and TMR7 register table and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	0x0000
TMRx_IDEN	0x0C	0x0000
TMRx_ISTS	0x10	0x0000
TMRx_SWEVT	0x14	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000

14.1.4.1 TMR6 and TMR7 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at default value.
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 default value.
Bit 3	OCMEN	0x0	rw	One cycle mode enable This bit is used to select whether to stop the counter at overflow event. 0: Disabled 1: Enabled
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 from the slave controller 1: Only counter overflow generates an overflow event.
Bit 1	OVFEN	0x0	rw	Overflow event enable This bit is used to enable or disable OEV event generation. 0: OEV event is enabled. An overflow event is generated by any of the following events: - Counter overflow - Setting the OVFSWTR bit to 1 - Overflow event generated from the slave controller 1: OEV event is disabled. If the OVFSWTR bit is set to 1, or if a hardware reset is generated from the slave mode controller, the counter and the prescaler are reinitialized. Note: This bit is set and cleared by software.
Bit 0	TMREN	0x0	rw	TMR enable 0: Disabled 1: Enabled

14.1.4.2 TMR6 and TMR7 control register2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at default value.
Bit 6: 4	PTOS	0x0	rw	Master TMR output selection This field is used to select the signals in master mode to be sent to slave timers. 000: Reset 001: Enable 010: Update
Bit 3: 0	Reserved	0x0	resd	Kept at default value.

14.1.4.3 TMR6 and TMR7 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15: 9	Reserved	0x00	resd	Kept at default value.
Bit 8	OVFDEN	0x0	rw	Overflow event DMA request enable 0: Disabled 1: Enabled
Bit 7: 1	Reserved	0x00	resd	Kept at default value.
Bit 0	OVFIEN	0x0	rw	Overflow interrupt enable 0: Disabled 1: Enabled

14.1.4.4 TMR6 and TMR7 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at default value.
Bit 0	OVFIF	0x0	rw0c	Overflow interrupt flag This bit is set by hardware at an overflow event. It is cleared by software. 0: No overflow event occurred. 1: Overflow event occurred, and if OVFEN=0, and OVFS=0 in the TMRx_CTRL1 register: − An overflow event occurred when OVFG=1 in the TMRx_SWEVE register − An overflow event occurred when the counter value (CVAL) is reinitialized by a trigger event.

14.1.4.5 TMR6 and TMR7 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 1	Reserved	0x0000	resd	Kept at default value.
Bit 0	OVFSWTR	0x0	rw0c	Overflow event triggered by software An overflow event is trigged by software. 0: No effect 1: Generate an overflow event by software write operation.

14.1.4.6 TMR6 and TMR7 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.1.4.7 TMR6 and TMR7 division (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). At each overflow event, DIV value is written to the DIV register.

14.1.4.8 TMR6 and TMR7 period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 15: 0	PR	0x0000	rw	Period value This indicates the period value of the TMRx counter. The timer stops working when the period value is 0.

14.2 General-purpose timer (TMR2 to TMR4)

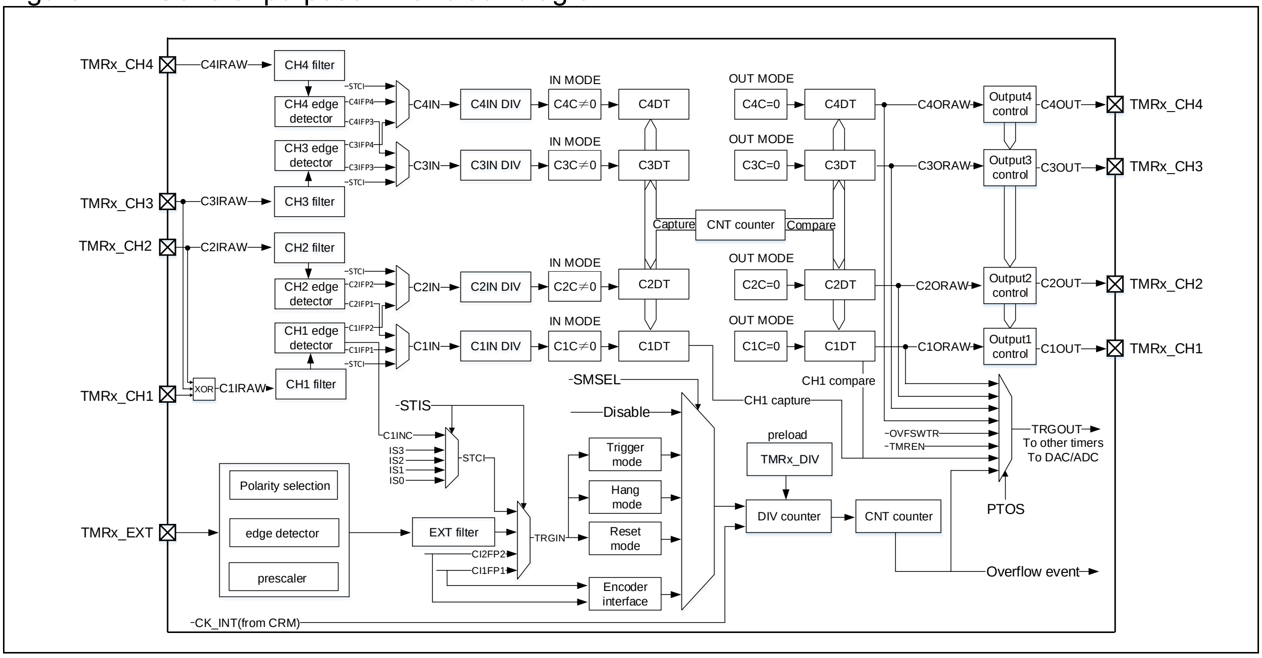
14.2.1 TMR2 to TMR4 introduction

The general-purpose timers (TMR2 to TMR4) consist of a 16-bit counter supporting up, down, up/down (TMR2 can be extended to 32 bits) counting modes, four capture/compare registers, and four independent channels. They can be used for input capture and programmable PWM output.

14.2.2 TMR2 to TMR4 main features

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

Figure 14-7 General-purpose timer block diagram

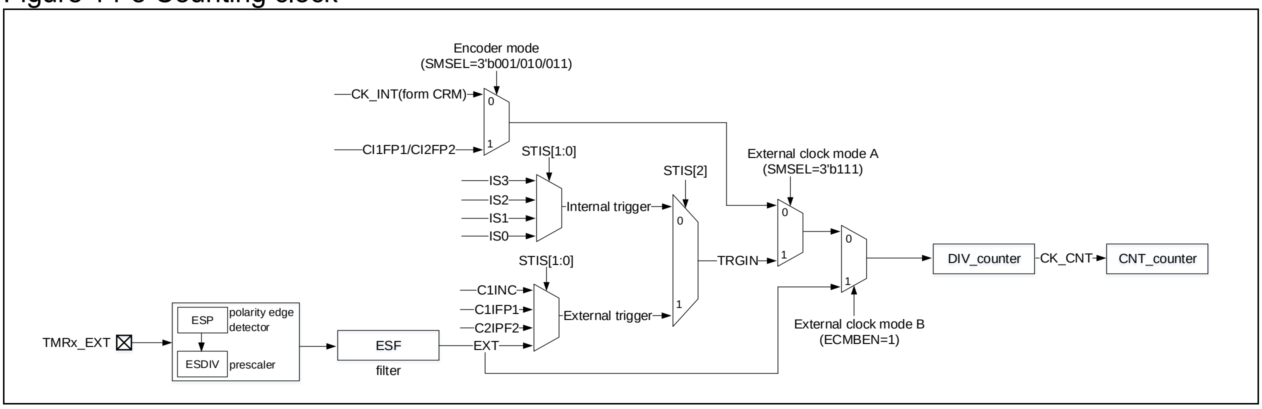


14.2.3 TMR2 to TMR4 functional overview

14.2.3.1 Counting clock

The counter clock of TMR2 to TMR4 can be provided by the internal clock (CK_INT), external clock (external clock mode A and B) and internal trigger input (ISx)

Figure 14-8 Counting clock



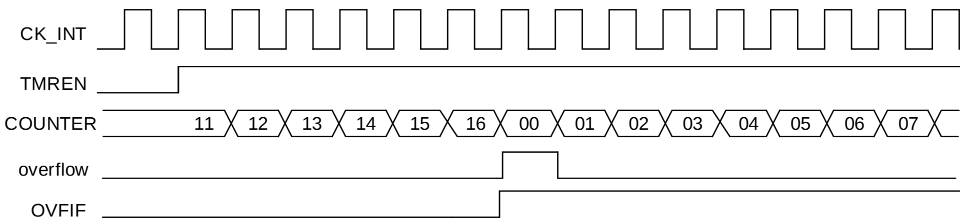
Internal clock (CK_INT)

By default, the CK_INT, which is divided by a prescaler, is used to drive the counter to count. 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.

Follow the procedures below:

- Select a counting mode by setting the TWCMSEL[1:0] in TMRx_CTRL1 register. If a unidirectional aligned counting mode is selected, it is necessary to select a counting direction through the OWCDIR bit in TMRx_CTRL1 register.
- Set counting frequency through TMRx_DIV register
- Set counting cycles through TMRx_PR register
- Enable the counter by setting the TMREN bit in the TMRx_CTRL1 register

Figure 14-9 Control circuit with CK_INT, 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)

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 2 input filter (C2DF[3:0] in TMRx_CM1 register) and channel 2 input polarity (C2P/C2CP in TMRx_CCTR register);

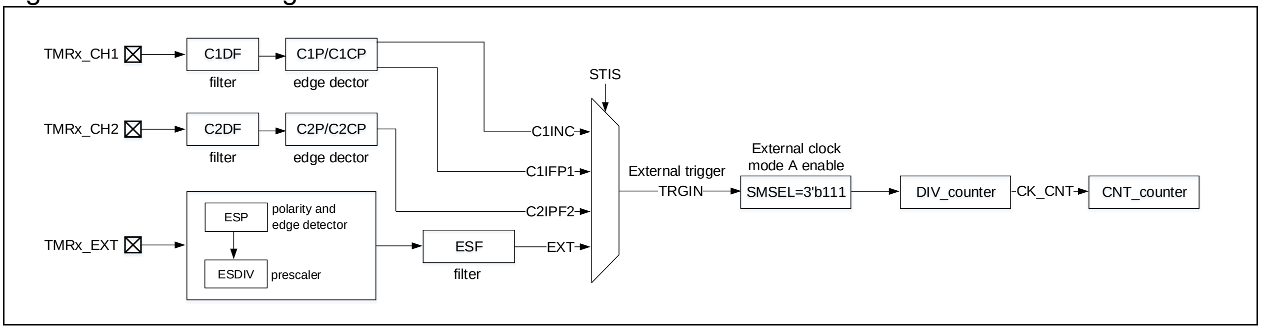
  If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).
- Set TRGIN signal source using the STIS[1:0] 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 14-10 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 14-11 Counting in external clock mode A, PR=0x32 and DIV=0x0

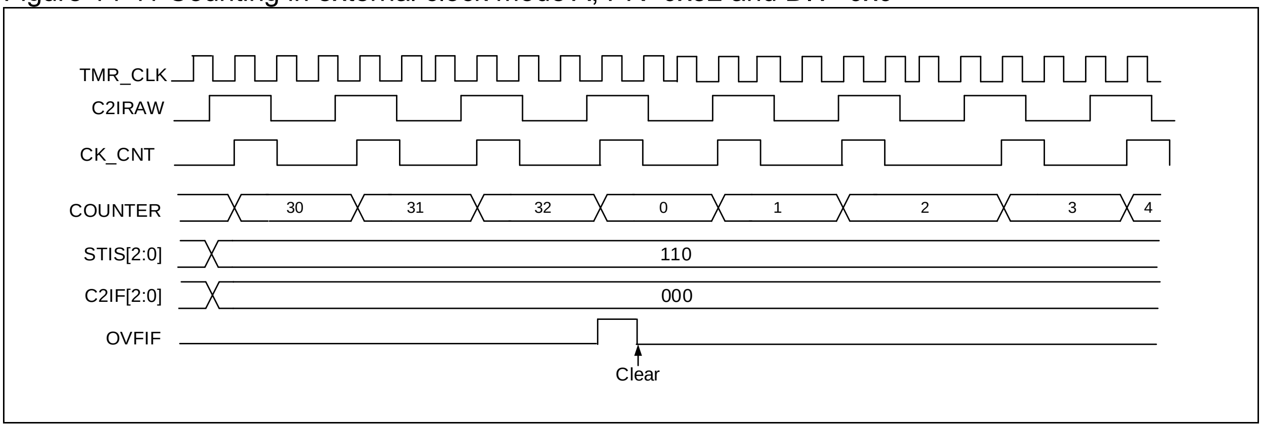
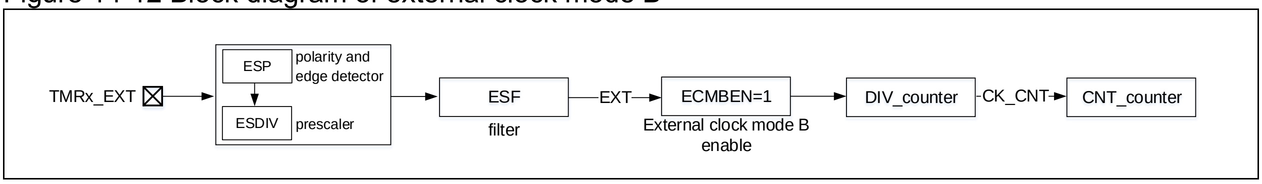
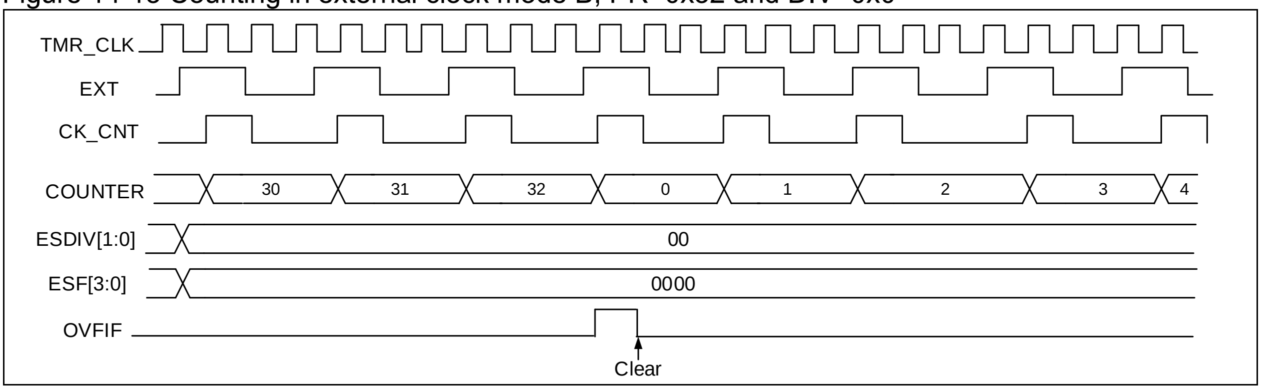


Figure 14-12 Block diagram of external clock mode B



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 14-13 Counting in external clock mode B, PR=0x32 and 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 from another timer. The internal trigger signal is selected by setting the STIS[2: 0] bit to enable counting.

TMR2 to TMR4 consist 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 the new prescaler value is taken into account when the next overflow event occurs.

Below is the configuration procedure for internal trigger input:

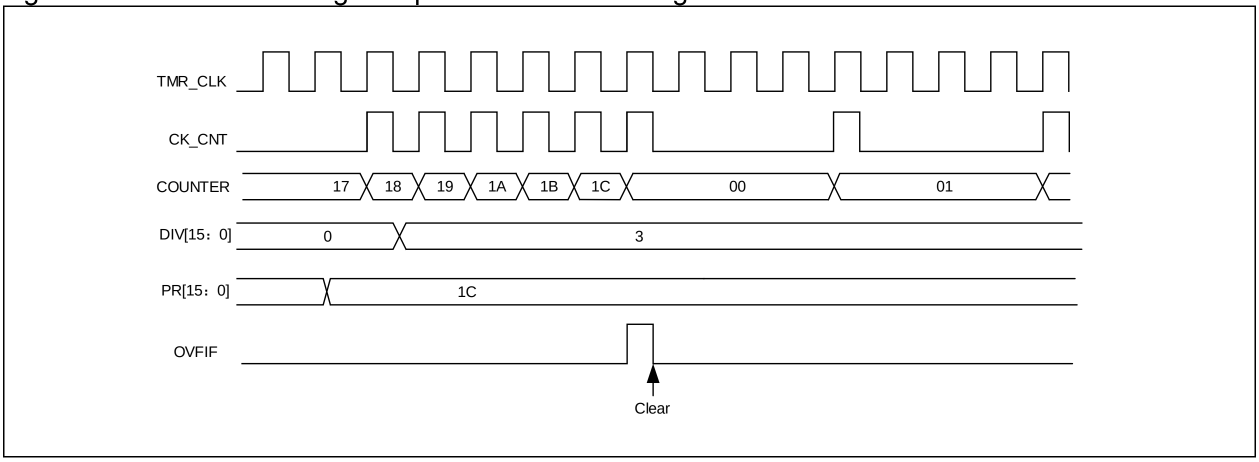
- Set counting cycles through TMRx_PR register
- Set counting frequency through TMRx_DIV register
- Set counting modes through the TWCMSEL[1:0] in TMRx_CTRL1 register
- Select internal trigger by setting STIS[2:0]= 3'b000~3'b011 in TMRx_STCTRL register
- Select external clock mode A by setting SMSEL[2:0]=3'b111 in TMRx_STCTRL register
- Enable TMRx to start counting through the TMREN in TMRx_CTRL1 register

Table 14-3 TMRx internal trigger connection

Slave controller	**IS0 (STIS = 000)**	**IS1 (STIS = 001)**	**IS2 (STIS = 010)**	**IS3 (STIS = 011)**
TMR2	TMR1	TMR9	TMR3	OTGFS_SOF
TMR3	TMR1	TMR2	TMR9	TMR4
TMR4	TMR1	TMR2	TMR3	TMR9

Note 1: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

Figure 14-14 Counter timing with prescaler value chang from 1 to 4



14.2.3.2 Counting mode

The timers (TMR2 to TMR4) support several counting modes to meet different application scenarios. They have an internal 16-bit up, down, up/down counter. TMR2 can be extended to 32 bits by setting the PMEN bit to 1.

The TMRx_PR register is used to set counting period of the counter. 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.

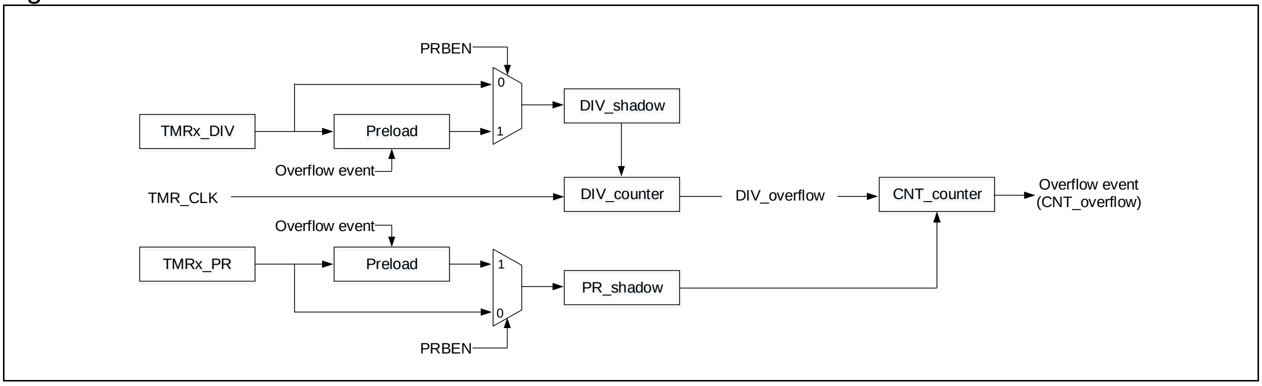
TMRx_DIV register is used to define the counter frequency of the counter. The counter counts once every DIV[15:0]+1 clock cycle. Similar to TMRx_PR register, after enabling periodic buffer, the value of the TMRx_DIV register are transferred into the shadow register at each overflow event.

Reading the TMRx_CNT register returns the current counter value. Writing the TMRx_CNT register will update the current counter value.

An overflow event is enabled by default. It can be disabled by setting OVFEN=1 in the TMRx_CTRL1 register. The OVFS bit in the TMRx_CTRL1 register is used to select the source of an overflow event, which is, by default, counter overflow or underflow, setting OVFSWTR, reset signal generated by slave mode timer controller in reset mode. Once the OVFS is set, an overflow event is generated only when overflow or underflow occurs.

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 14-15 Basic structure of a counter



Upcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register.

In upcounting 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 setting OVFIF=1. If the overflow event is disabled, the register is no longer reloaded with the prescaler and periodic value after counter overflow occurs, otherwise, the prescaler and periodic value will be updated at an overflow event.

Figure 14-16 Overflow event when PRBEN=0

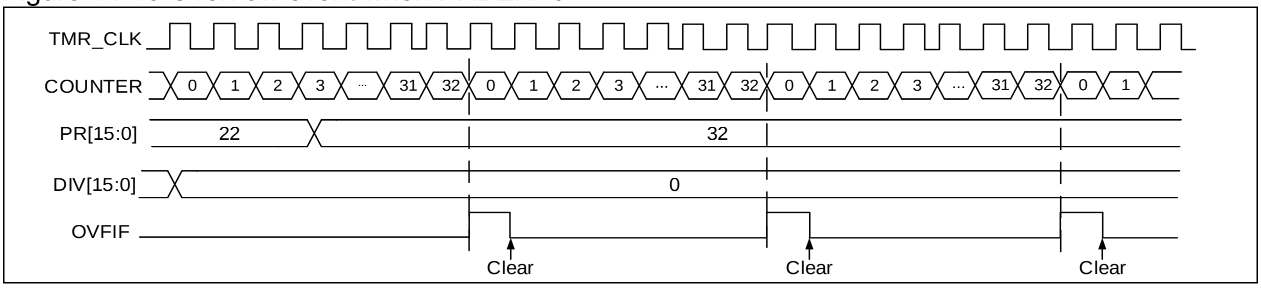
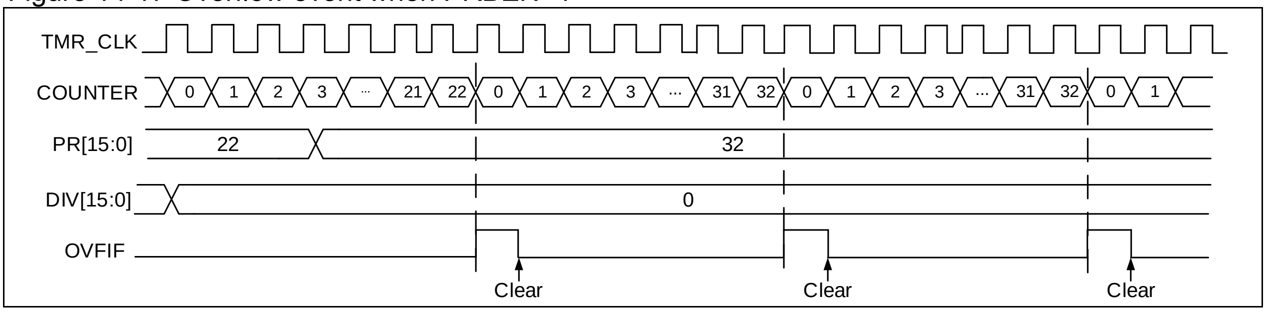


Figure 14-17 Overflow event when PRBEN=1

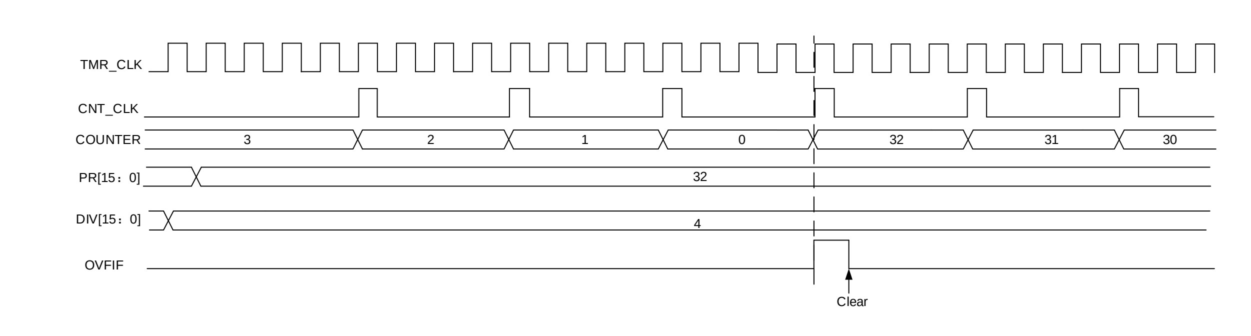


Downcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register.

In downcounting 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 14-18 Counter timing diagram with internal clock divided by 4



Up/down counting mode (center-aligned mode)

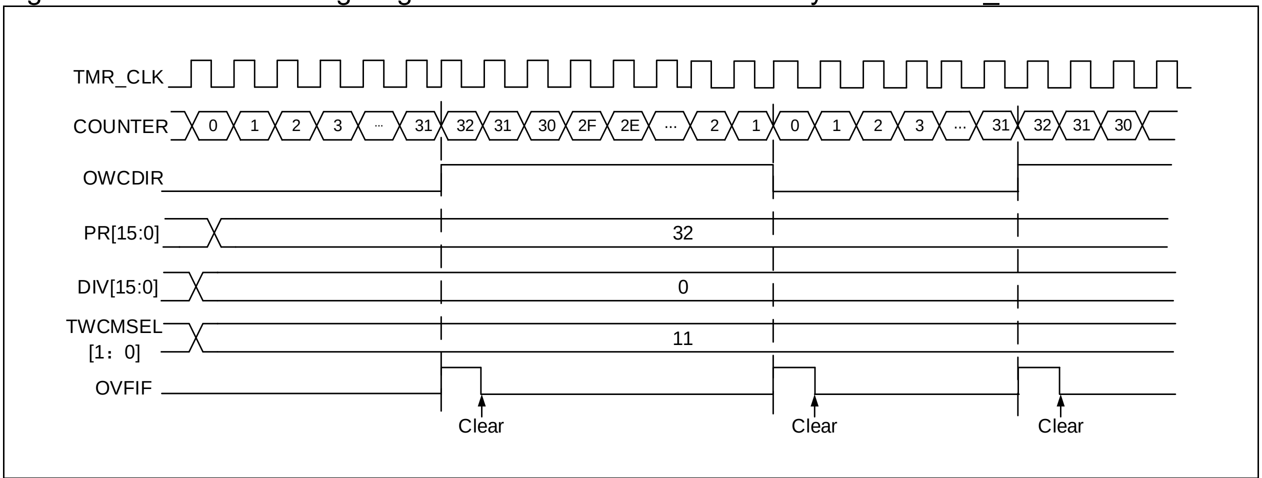
This mode is selected by setting CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register.

In up/down counting 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 used to select the condition under which the CxIF flag is set in two-way counting mode. In other words, when TWCMSEL[1:0]=2'b01 (counting mode 1) is selected, the CxIF flag is set only when the counter counts down; when TWCMSEL[1:0]=2'b10 (counting mode 2) is selected, the CxIF flag is set only when the counter counts up; when TWCMSEL[1:0]=2'b11 (counting mode 3) is selected, the CxIF flag is set when the counter counts up and down.

Note: The OWCDIR is ready-only in up/down counting mode.

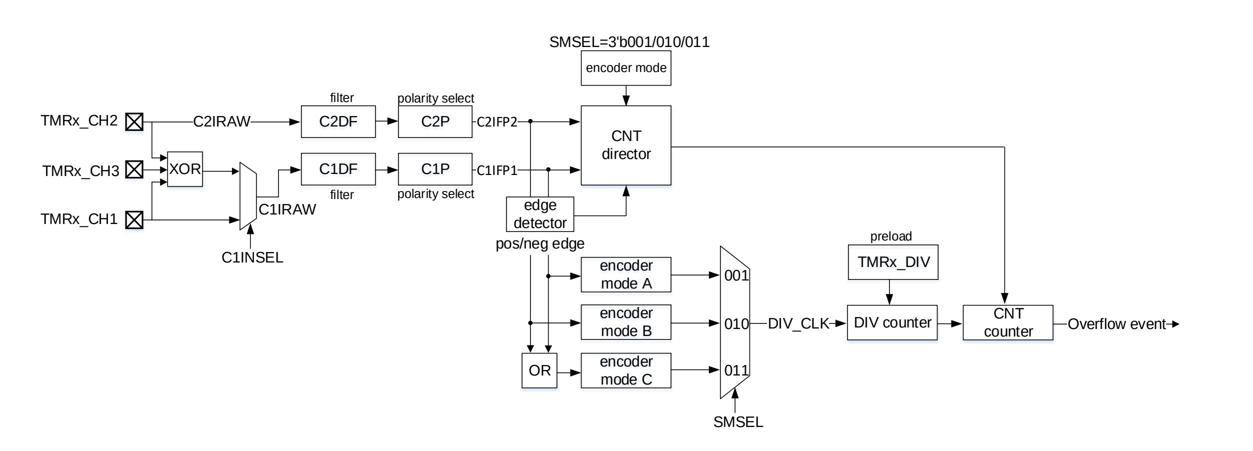
Figure 14-19 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Encoder interface mode

In this mode, the two input (TMRx_CH1 and TMRx_CH2) signals are required. Depending on the level on one input signal, 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 14-20 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.

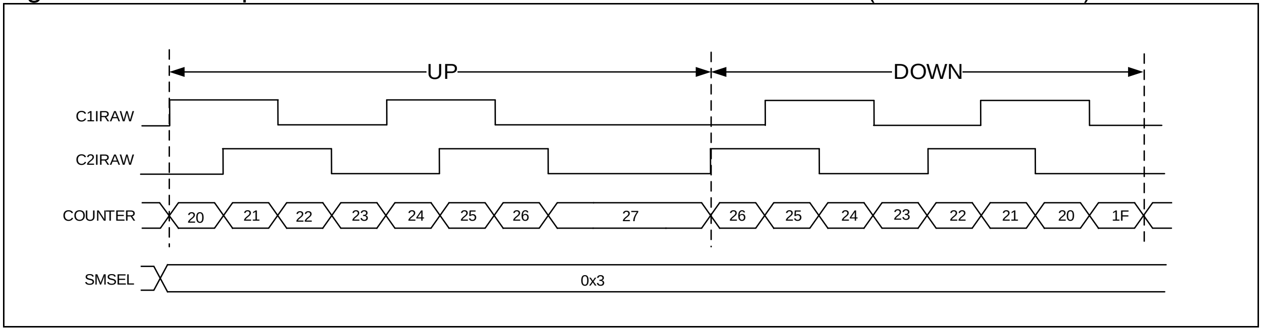
To use encoder mode, follow the procedures below:

- 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 14-4 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 14-21 Example of counter behavior in encoder interface mode (encoder mode C)



14.2.3.3 TMR input function

Each of timers (TMR2 to TMR4) has four independent channels, with each channel being configured as input or output. As input, each channel input signal is processed as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSE bit is used to select the source of C1IRAW from TMRx_CH1, or 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 goes through edge detector, and generates 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 goes through capture signal selector, and generates the CxIN signal after capture signal 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, C1IFP2 is the C1IFP1 signal that is from channel 1 and passes through channel 2 edge detector). The STCI comes from slave timer controller, and its source is selected by STIS bit.
- CxIN goes through input divider and generates the CxIPS signal. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit.

Figure 14-22 Input/output channel 1 main circuit

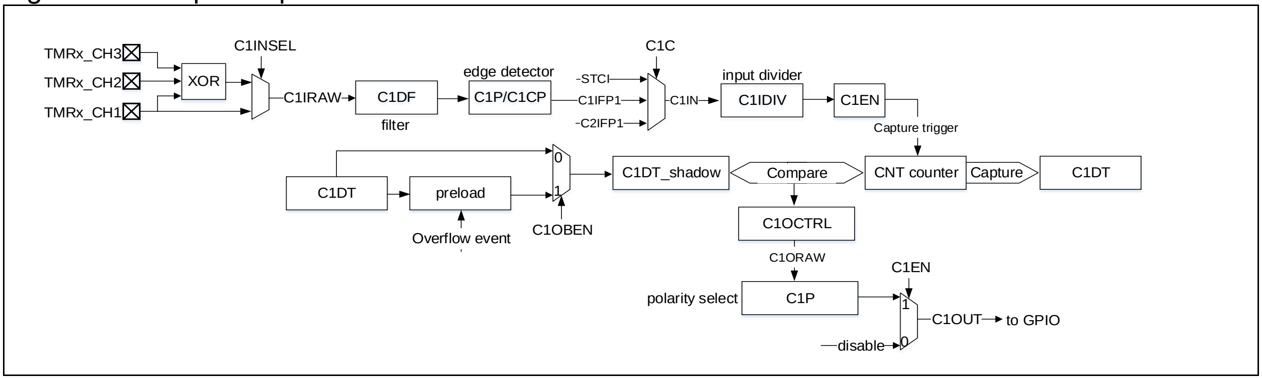
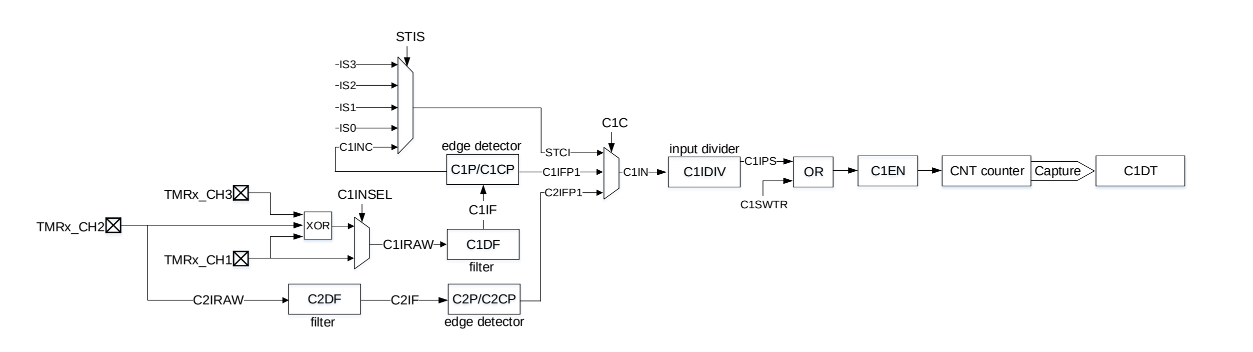


Figure 14-23 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT register latches 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 is generated, and the previous counter value will be overwritten by the current counter value, with setting CxRF to 1.

To capture the rising edge of C1IN input, follow the procedure below

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set C1IN signal filter bandwidth (CxDF[3: 0])
- Set the active edge of C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTRL register
- Program C1IN signal capture frequency divider (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 through an XOR gate. This function is selected by setting the C1INSE in the TMRx_CTRL2 register.

The XOR gate can be used to connect Hall sensors. For example, connect the three XORed 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

PWM input mode is applied to channel 1 and 2. To use this mode, both C1IN and C2IN have to be mapped to the same TMRx_CHx, and the CxIFPx of either channel 1 or channel 2 must be configured as trigger input and slave mode controller is configured in reset mode.

The PWM input mode can be used to measure the period and duty cycle of the PWM input signal. For example, the user can measure the period and duty cycle of the PWM applied on channel 1 using the following procedures:

- Set C1C=2‘b01: select C1IN for C1IFP1
- Set C1P=1’b0, select C1IFP1 rising edge active
- Set C2C=2‘b10, select C2IN for C1IFP2
- Set C2P=1’b1, select C1IFP2 falling edge active
- Set STIS=3’b101, select the slave mode timer trigger signal as C1IFP1
- Set SMSEL=3‘b100: configure the slave mode controller in reset mode
- Set C1EN=1’b1 and C2EN=1’b1. Enable channel 1 and input capture

After above configuration, the rising edge of channel 1 input signal will trigger the capture and stores the capture value into C1DT register, and it will reset the counter at the same time. The falling edge of the channel 1 input signal triggers the capture and stores the capture value into C2DT register. The period of the channel 1 input signal is calculated through C1DT, and its duty cycle through C2DT.

Figure 14-24 PWM input mode configuration example

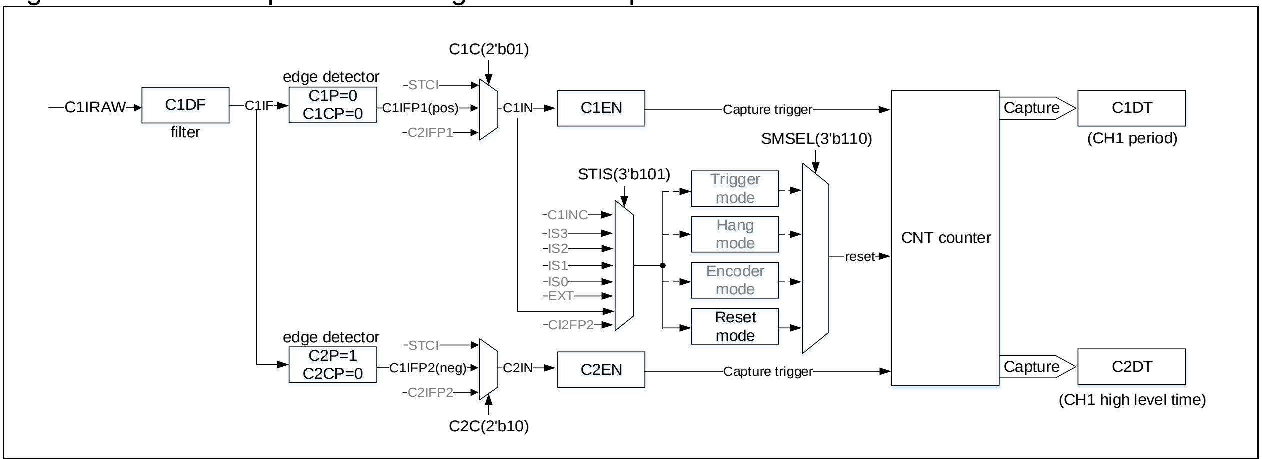
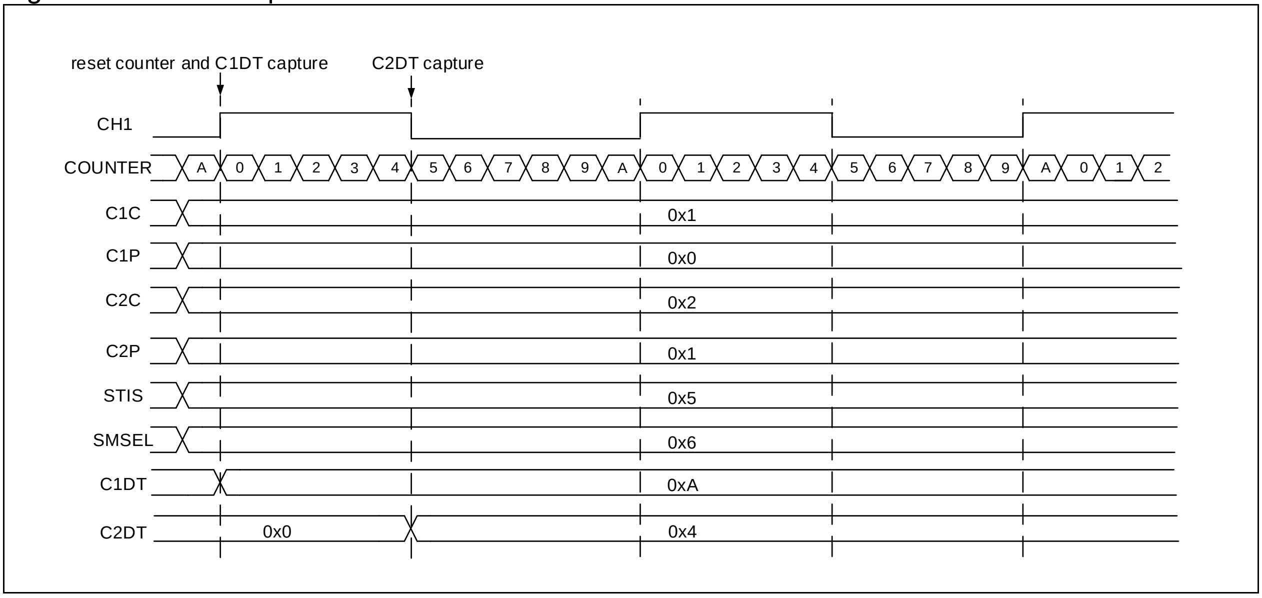


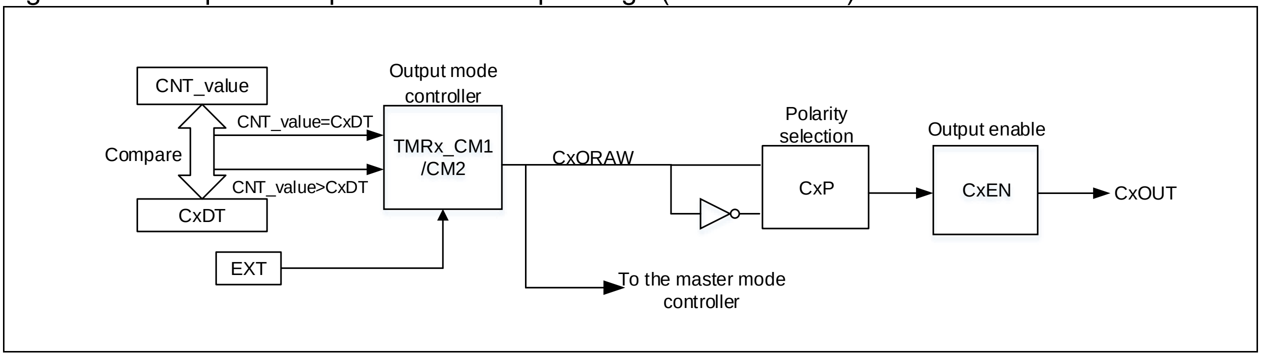
Figure 14-25 PWM input mode



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 14-26 Capture/compare channel output stage (channel 1 to 4)



Output mode

Write CxC[1: 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:

Enable PWM mode A by setting CxOCTRL=3'b110. In upcounting mode, C1ORAW outputs high when TMRx_C1DT>TMRx_CVAL, otherwise, it is low; in downcounting mode, C1ORAW outputs low when TMRx_C1DT<TMRx_CVAL, otherwise, it is high.

To use PWM mode A, the following procedures are recommended:

- Set PWM period through TMRx_PR register
- Set PWM duty cycles through TMRx_CxDT
- Select PWM mode A by setting CxOCTRL=3'b110 in the TMRx_CM1/CM2 register
- Set counting frequency through TMRx_DIV register
- Select counting mode by setting the TWCMSEL[1:0] bit in the TMRx_CTRL1 register
- Select output polarity through the CxP and CxCP bits in the TMRx_CCTRL register
- Enable channel output through the CxEN and CxCEN bits in the TMRx_CCTRL register
- Enable TMRx output through the OEN bit in the TMRx_BRK register

- Configure GPIOs corresponding to TMR output channels as multiplexed mode
- Enable TMRx to start counting through the TMREN bit in the TMRx_CTRL1 register.

PWM mode B:

Enable PWM mode B by setting CxOCTRL=3'b111. In upcounting mode, C1ORAW outputs low when TMRx_C1DT>TMRx_CVAL, otherwise, it is high; in downcounting mode, C1ORAW outputs high when TMRx_C1DT<TMRx_CVAL, otherwise, it is low.

Forced output mode:

Enable forced output mode by setting CxOCTRL=3'b100/101. In this case, the CxORAW is forced to be the programmed level, regardless of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.

Output compare mode:

Enable output compare mode by setting CxOCTRL=3'b001/010/011. 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. Enable one-pulse by setting OCMEN=1. 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 in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.

Fast output mode:

Enable this mode by setting CxOIEN=1. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but change 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 14-27 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-28 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 14-29 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 14-30 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 14-27 C1ORAW toggles when counter value matches the C1DT value

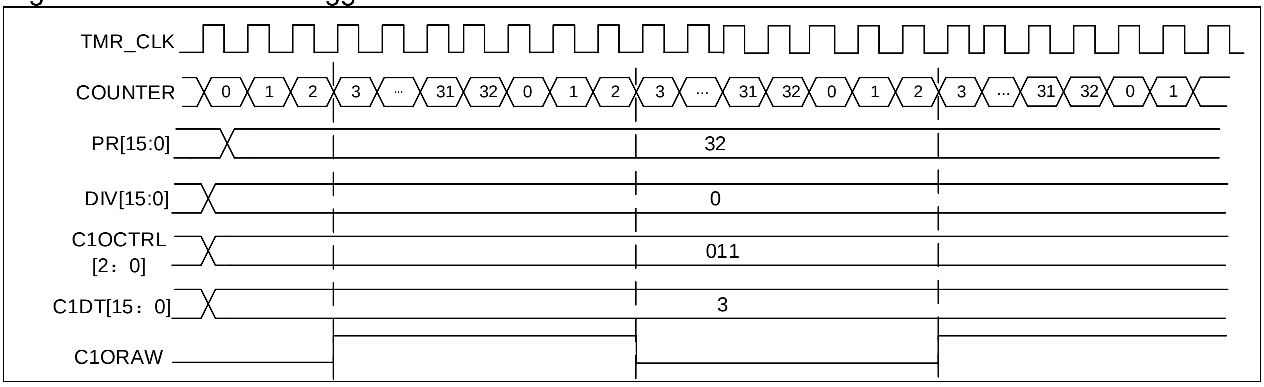


Figure 14-28 Upcounting mode and PWM mode A

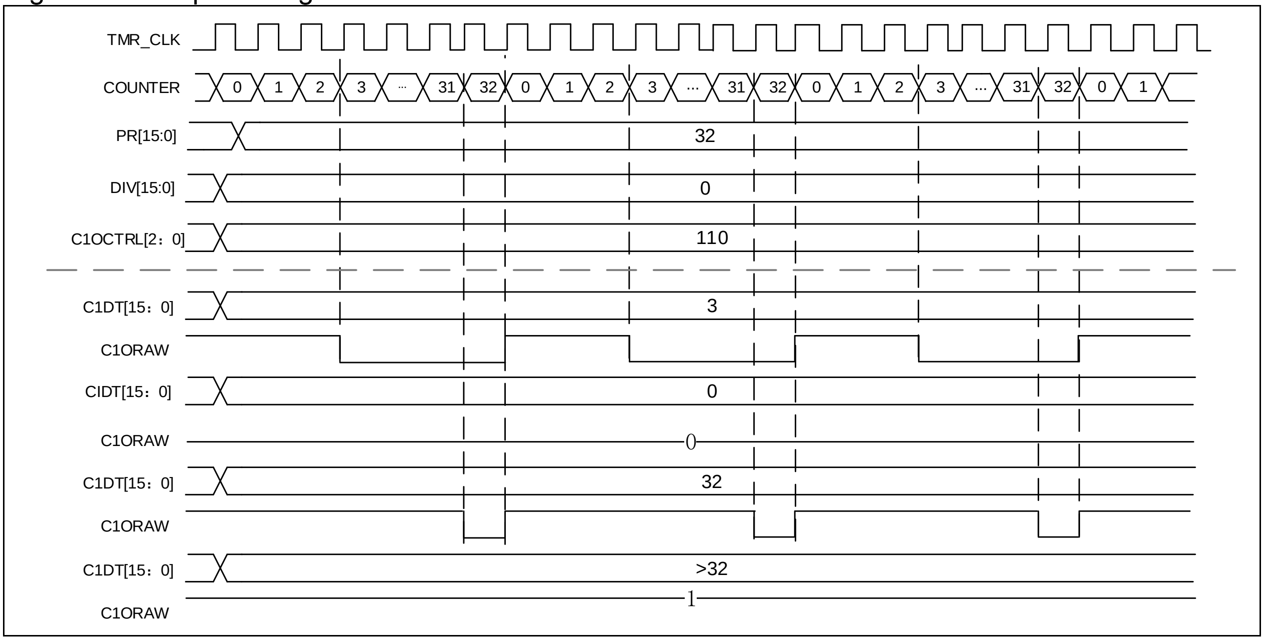


Figure 14-29 Up/down counting mode and PWM mode A

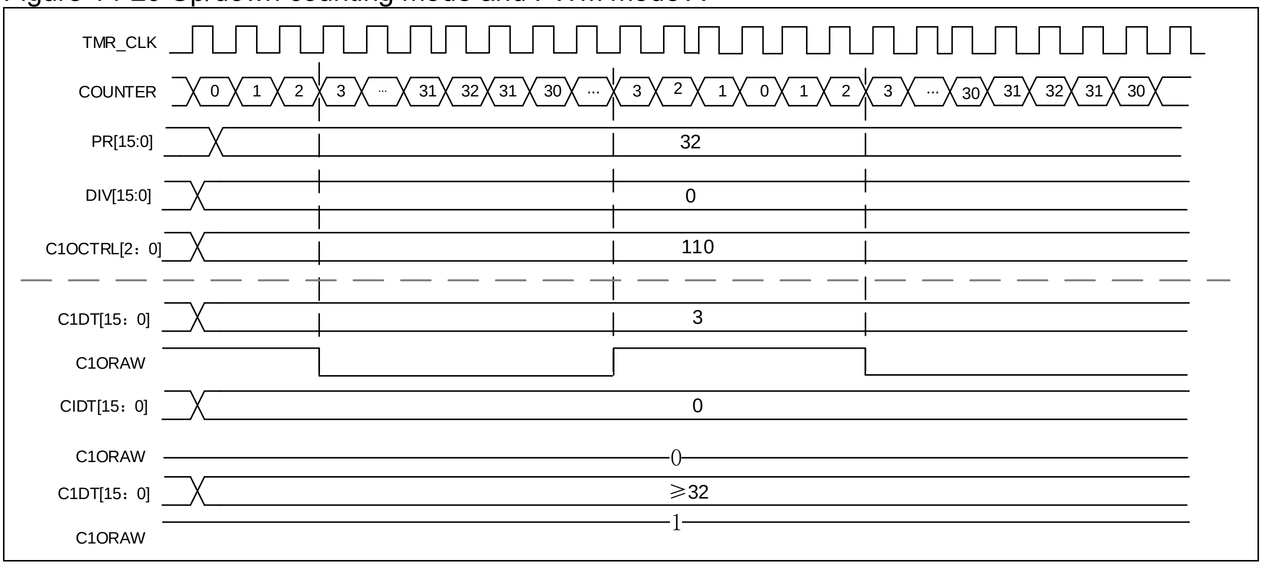
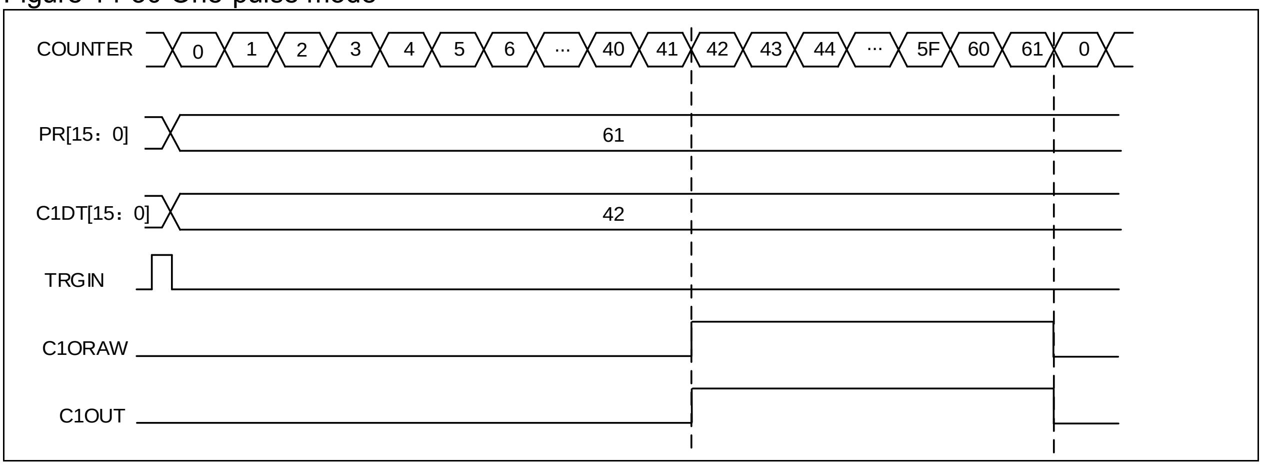


Figure 14-30 One-pulse mode



Master mode timer event output

When TMR is used as a master timer, one of the following source of signals can be selected as TRGOUT output to a slave mode timer. This is done by setting the PTOS bit in the TMRxCTRL2 register.

– PTOS=3'b000, TRGOUT output software overflow event (OVFSWTR bit in TMRx_SWEVT register)

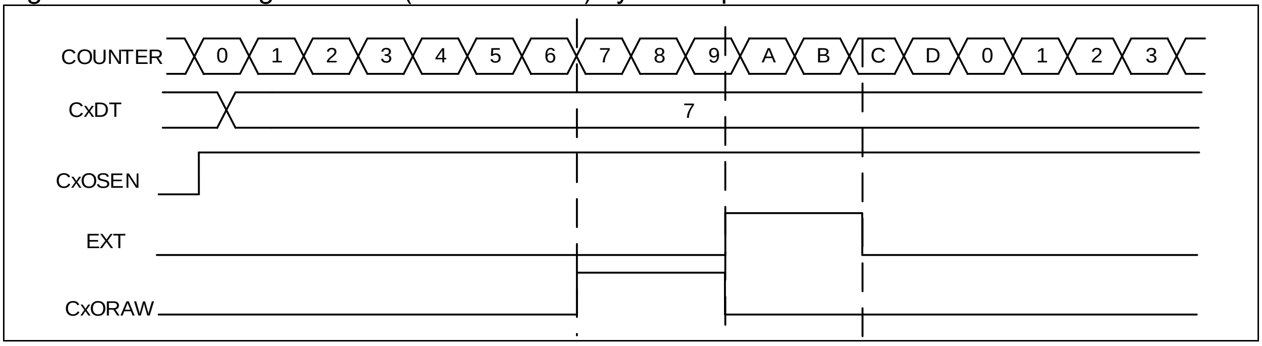
- PTOS=3’b001, TRGOUT output counter enable
- PTOS=3’b010, TRGOUT output counter overflow event
- PTOS=3’b011, TRGOUT output capture and compare event
- PTOS=3’b100, TRGOUT output C1ORAW
- PTOS=3’b101, TRGOUT output C2ORAW
- PTOS=3’b110, TRGOUT output C3ORAW
- PTOS=3’b111, TRGOUT output C4ORAW

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 14-31* 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 14-31 Clearing CxORAW(PWM mode A) by EXT input



14.2.3.5 TMR synchronization

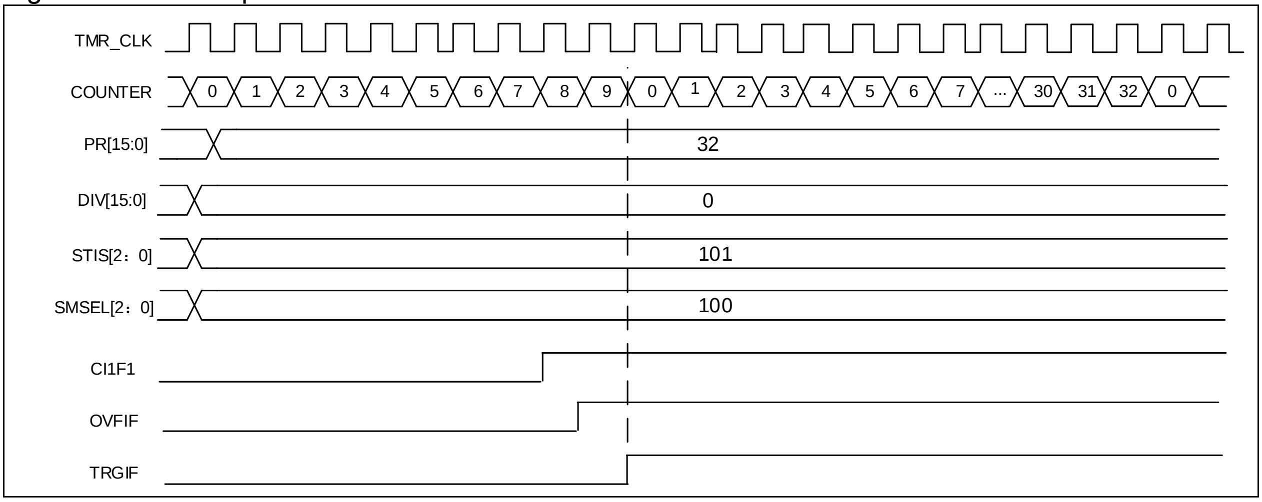
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

Slave 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.

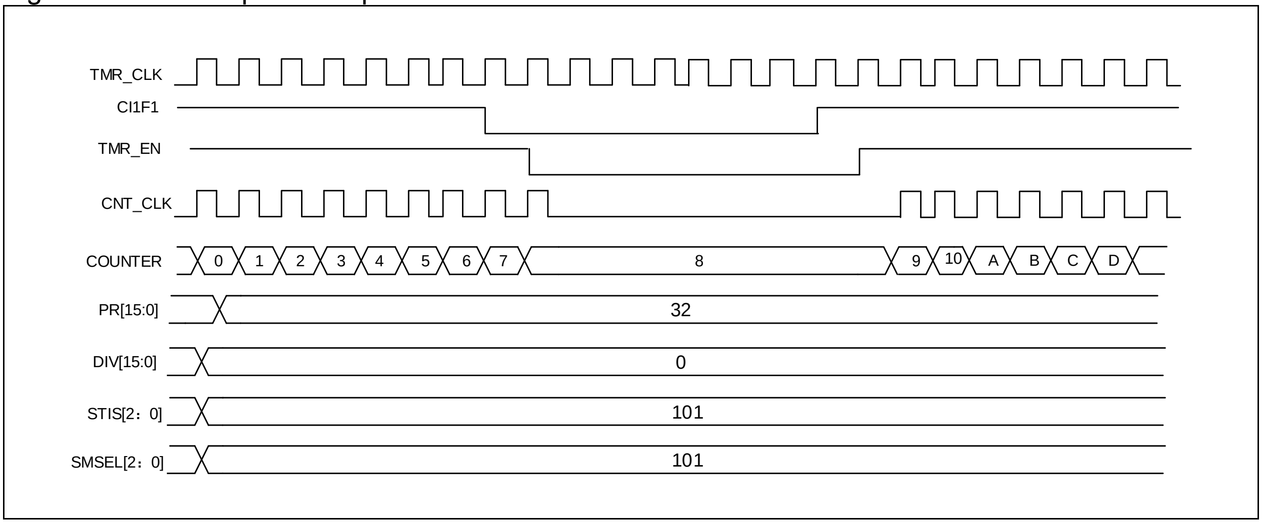
Figure 14-32 Example of reset mode



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.

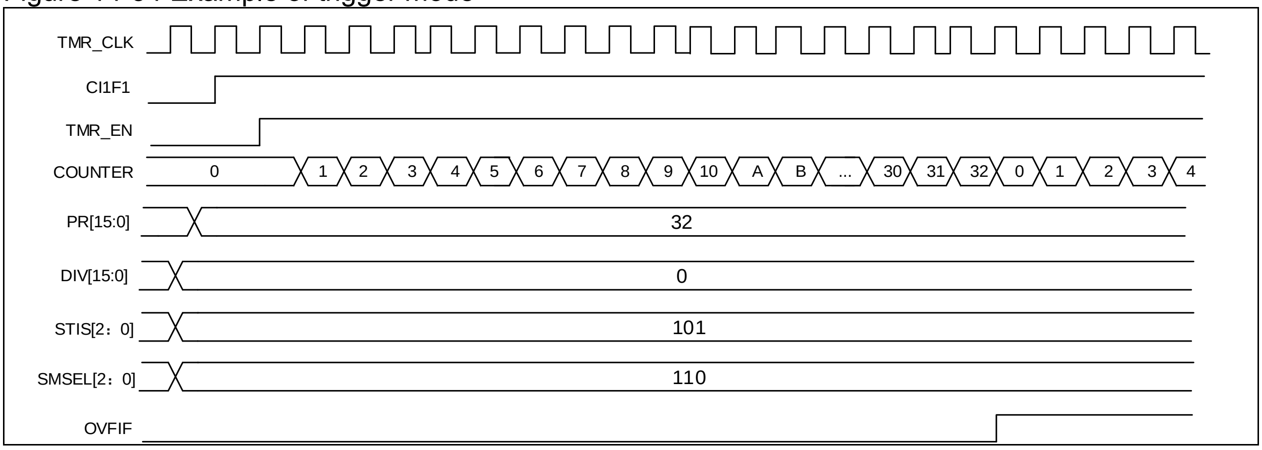
Figure 14-33 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

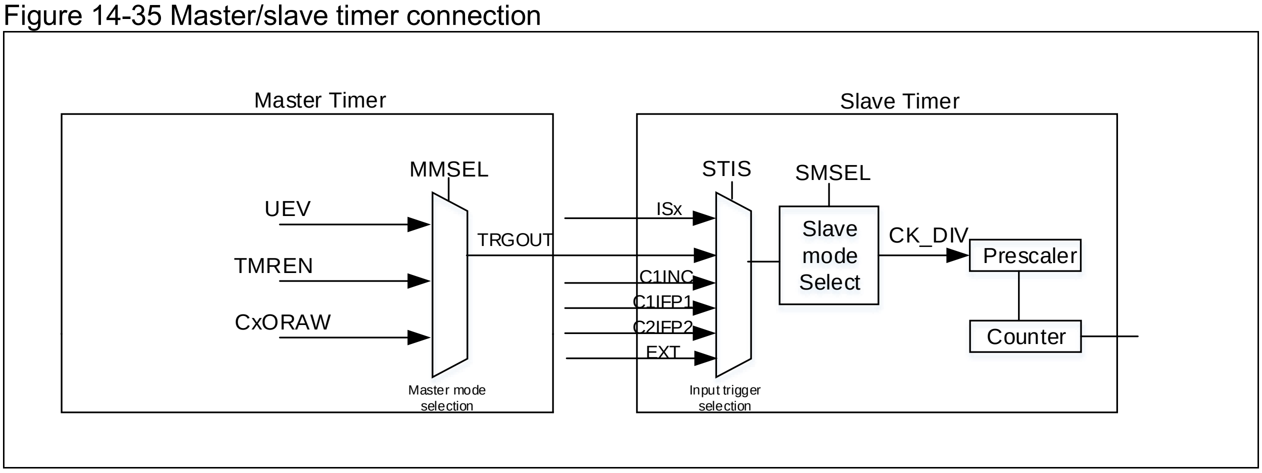
Figure 14-34 Example of trigger mode



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 14-35* provides an example of interconnection between master timer and slave timer.

Figure 14-35 Master/slave timer connection



Using master timer to clock the slave timer:

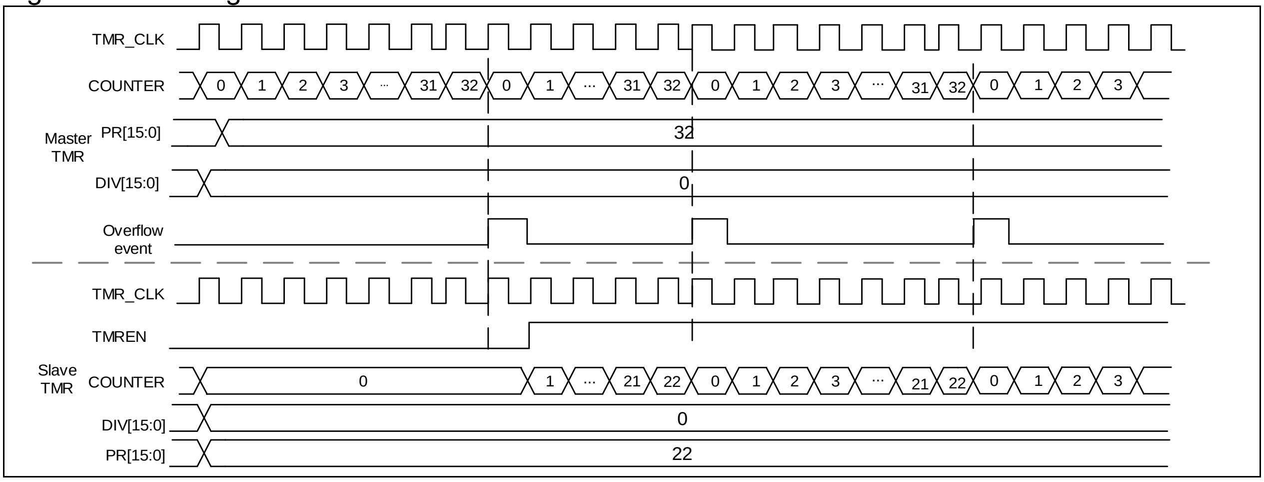
- Configure master timer output signal TRGOUT as an overflow event (PTOS[2: 0]=3'b010). The master timer outputs a pulse signal at each counter overflow event, which is used as the counting clock of the slave timer.

- 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 14-36 Using master timer to start slave timer

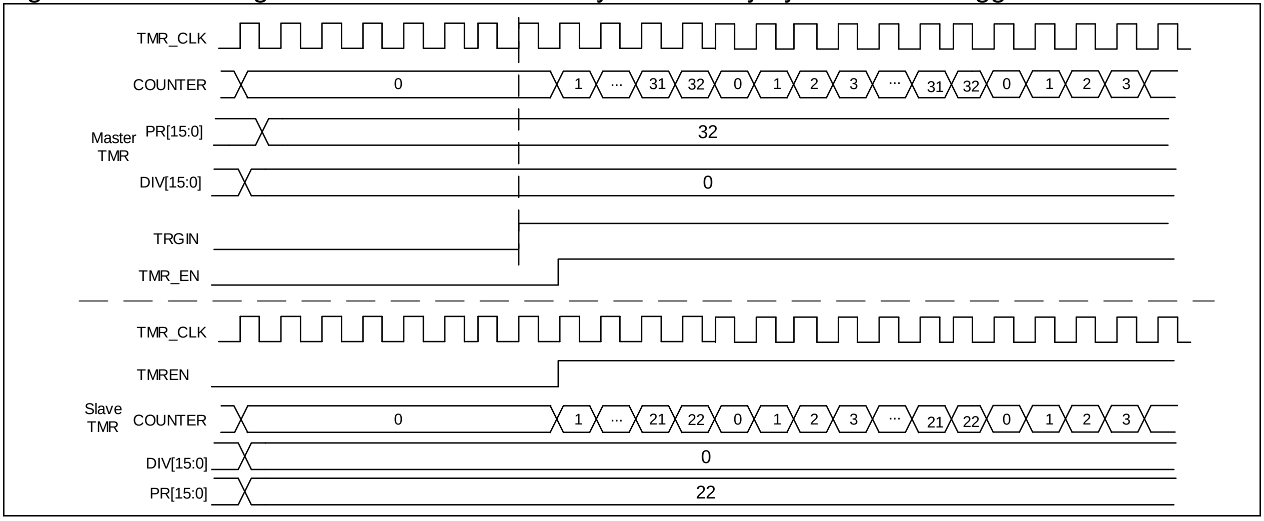


Starting master and slave timers synchronously by an external trigger:

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 14-37 Starting master and slave timers synchronously by an external trigger



14.2.3.6 Debug mode

When the microcontroller enters debug mode (Cortex^{TM}-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module.

14.2.4 TMR2 to TMR4 registers

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

TMR2 and TMR4 registers are mapped into a 16-bit addressable space.

Table 14-5 TMR2 to TMR4 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
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_C1DT	0x34	0x0000
TMRx_C2DT	0x38	0x0000
TMRx_C3DT	0x3C	0x0000
TMRx_C4DT	0x40	0x0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000

14.2.4.1 Control register 1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x0	resd	Kept at default value.
Bit 10	PMEN	0x0	rw	Plus Mode Enable This bit is used to enable TMR2 plus mode. In this mode, TMR2_CVAL, TMR2_PR and TMR2_CxDT are extended from 16-bit to 32-bit. 0: Disabled 1: Enabled Note: This function is only valid for TMR2. 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 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: 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

14.2.4.2 Control register 2 (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.

14.2.4.3 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 divider This field is used to select the frequency division factor 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 input of subordinate TMR. 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 14-3 for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at 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.

14.2.4.4 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at default value
Bit 14	TDEN	0x0	rw	Trigger DMA request enable 0: Disabled 1: Enabled
Bit 13	Reserved	0x0	resd	Kept at 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 default value
Bit 6	TIEN	0x0	rw	Trigger interrupt enable 0: Disabled 1: Enabled
Bit 5	Reserved	0x0	resd	Kept at 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

14.2.4.5 Interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at 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 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 occurred 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 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 occurred 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 occurred 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 occurred 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.

14.2.4.6 Software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 7	Reserved	0x000	resd	Kept at 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 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.

14.2.4.7 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.

14.2.4.8 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.

14.2.4.9 Channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept at 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 This bit defines the valid edge of input signals. Refer to C1P bit for details.
Bit 10	Reserved	0x0	resd	Kept at 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	C2CP	0x0	rw	Channel 2 complementary polarity This bit defines the valid edge of input signals. Refer to C1P bit for details.
Bit 6	Reserved	0x0	resd	Kept at 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	C1CP	0x0	rw	Channel 1 complementary polarity This bit defines the valid edge of input signals. Refer to C1P bit for details.
Bit 2	Reserved	0x0	resd	Kept at 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: C1CP/C1P are used to define the valid edge of input signals. 00: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 01: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted. 10: Reserved 11: C1IN rising and falling edges active. When used as external trigger, C1IN is not inverted.
Bit 0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-6 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.

14.2.4.10 Counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	CVAL	0x0000	rw	Counter value When TMR2 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the CVAL is extended to 32 bits.
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.2.4.11 Frequency 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.

14.2.4.12 Period register (TMRx_PR)

Bit	Register	Reset value	Type	Description
Bit 31: 16	PR	0x0000	rw	Period value When TMR2 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the PR is extended 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.

14.2.4.13 Channel 1 data register (TMRx_C1DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C1DT	0x0000	rw	Channel 1 data register When TMR2 enables plus mode (the PMEN bit in the TMR_CTRL1 register), the C1DT is extended 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.

14.2.4.14 Channel 2 data register (TMRx_C2DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C2DT	0x0000	rw	Channel 2 data register When TMR2 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.

14.2.4.15 Channel 3 data register (TMRx_C3DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C3DT	0x0000	rw	Channel 3 data register When TMR2 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.

14.2.4.16 Channel 4 data register (TMRx_C4DT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	C4DT	0x0000	rw	Channel 4 data register When TMR2 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.

14.2.4.17 DMA control register (TMRx_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at 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 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

......

14.2.4.18 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.

14.3 General-purpose timer (TMR9 and TMR12)

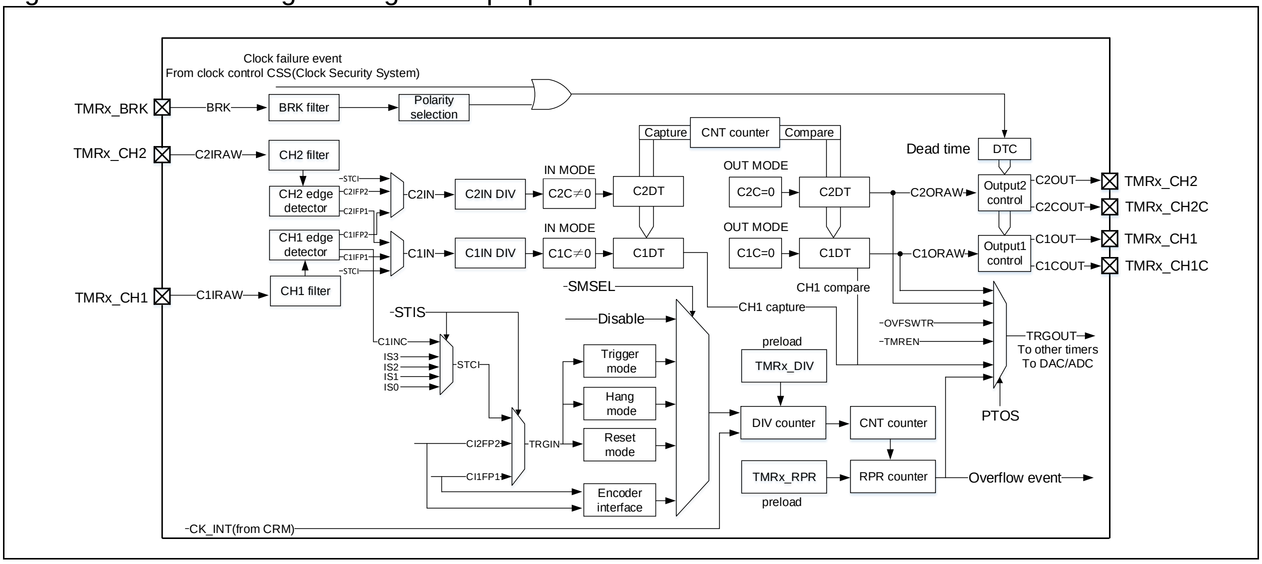
14.3.1 TMR9 and TMR12 introduction

The general-purpose timers (TMR9 and TMR12) consist of a 16-bit counter supporting upcounting mode. They have two capture/compare registers, and two independent channels for dead-time insertion, input capture and programmable PWM output.

14.3.2 TMR9 and TMR12 main features

- Counter clock source: internal clock, external input and internal trigger inputs
- 16-bit up counter, 8-bit repetition counter
- 2x independent channels for input capture, output compare, PWM generation, one-pulse mode and dead-timer insertion
- 2x independent channels for complementary channels
- TMR break feature
- Synchronization circuit to interconnect several timers together
- Interrupt/DMA generation on at overflow, trigger event, break input and channel events

Figure 14-38 Block diagram of general-purpose TMR9/12

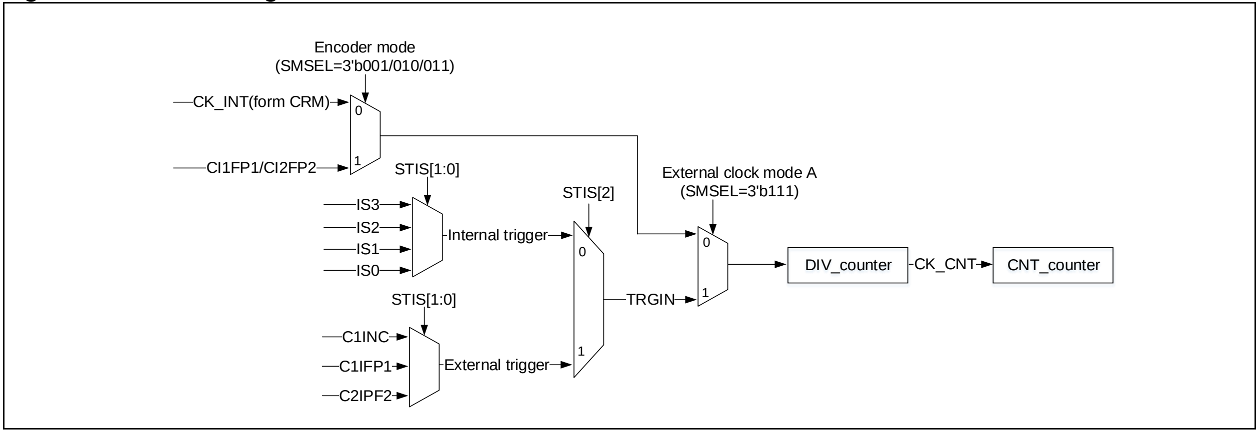


14.3.3 TMR9 and TMR12 functional overview

14.3.3.1 Counting clock

The counter clock of TMR9 and TMR12 can be provided by the internal clock (CK_INT), external clock (external clock mode A) and internal trigger input (ISx)

Figure 14-39 Counting clock



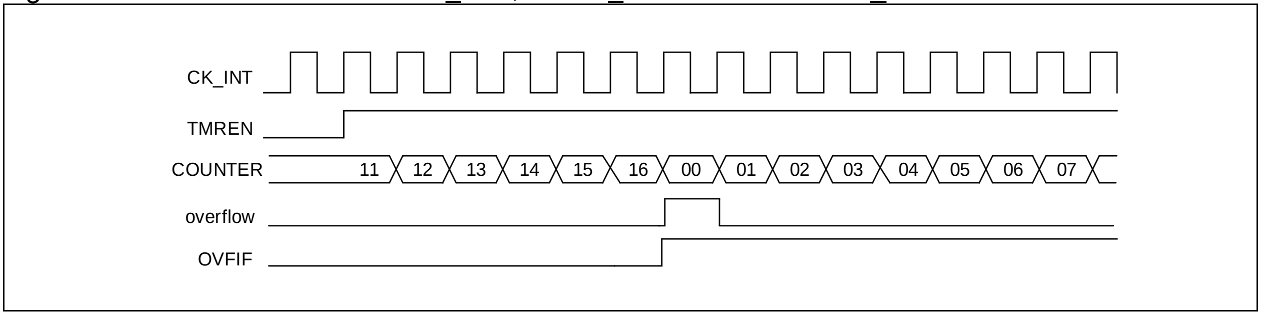
Internal clock (CK_INT)

By default, the CK_INT, which is divided by a prescaler, is used to drive the counter to count. 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.

Follow the procedures below:

- Select counting mode by setting the TWCMSEL[1:0] in TMRx_CTRL1 register. If a unidirectional aligned counting mode is selected, there is a need to select counting direction through the OWCDIR in TMRx_CTRL1 register.
- Set counting frequency through TMRx_DIV register
- Set counting cycles through TMRx_PR register
- Enable the counter by setting the TMREN bit in the TMRx_CTRL1 register

Figure 14-40 Control circuit with CK_INT, TMRx_DIV=0x0 and TMRx_PR=0x16



External clock (**TRGIN/EXT**)

The counter clock can be provided by the external clock source TRGIN.

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).

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

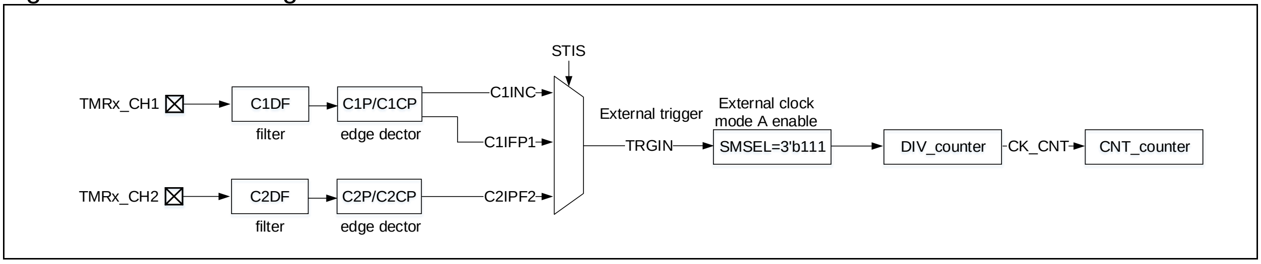
- Set external source TRGIN parameters

  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

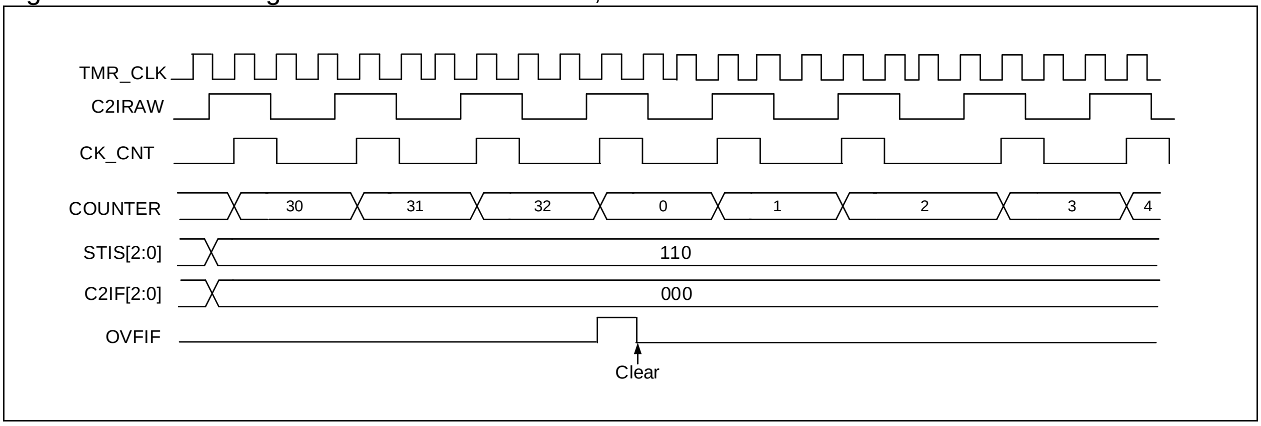
- Enable external clock mode A by setting SMSEL=3’b111 in TMRx_STCTR register
- Set counting frequency through the DIV[15:0] in TMRx_DIV register
- Set counting period through the PR[15:0] in TMRx_PR register
- Enable counter through the TMREN bit in TMRx_CTRL1 register

Figure 14-41 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 14-42 Counting in external clock mode A, PR=0x32 and 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 from another timer. The internal trigger signal is selected by setting the STIS[2: 0] bit to enable counting.

Each timer (TMR9 and TMR12) 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 the new prescaler value is taken into account when the next overflow event occurs.

Below is the configuration procedure for internal trigger input:

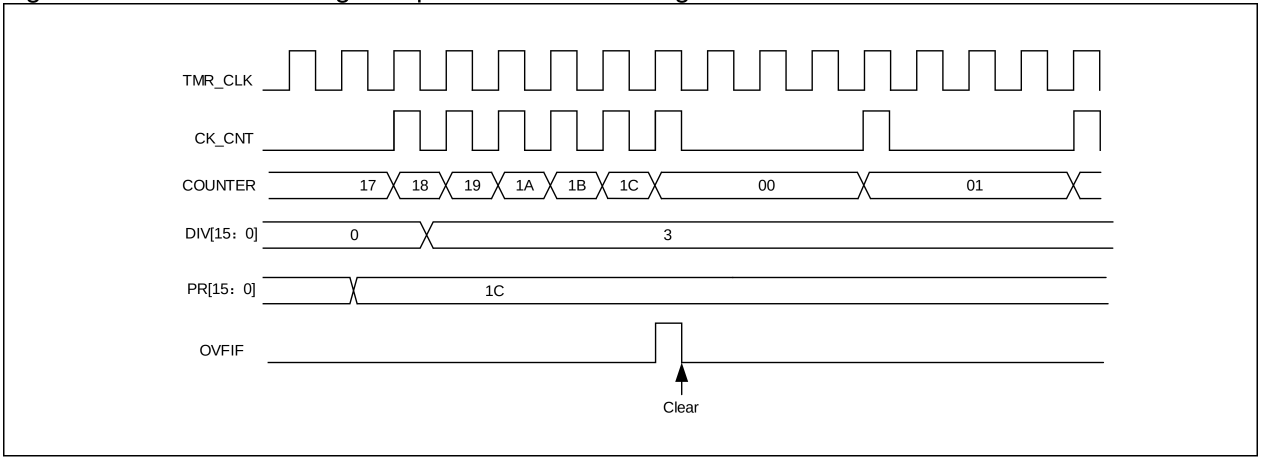
- Set counting cycles through TMRx_PR register
- Set counting frequency through TMRx_DIV register
- Set counting modes through the TWCMSEL[1:0] in TMRx_CTRL1 register
- Select internal trigger by setting STIS[2:0]= 3’b000~3’b011 in TMRx_STCTRL register
- Select external clock mode A by setting SMSEL[2:0]=3’b111 in TMRx_STCTRL register
- Enable TMRx to start counting through the TMREN in TMRx_CTRL1 register

Table 14-7 TMRx internal trigger connection

Slave controller	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR9	TMR2	TMR3	TMR10_C1OUT	TMR11_C1OUT
TMR12	TMR4	TMR2	TMR13_C1OUT	TMR14_C1OUT

Note 1: If there is no corresponding timer in a device, the corresponding trigger signal ISx is not present.

Figure 14-43 Counter timing with prescaler value chang from 1 to 4



14.3.3.2 Counting mode

The general-purpose timer (TMR9 and TMR12) consists of a 16-bit counter supporting multiple counting modes to meet different application scenarios.

The TMRx_PR register is used to define counting period of counter. 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.

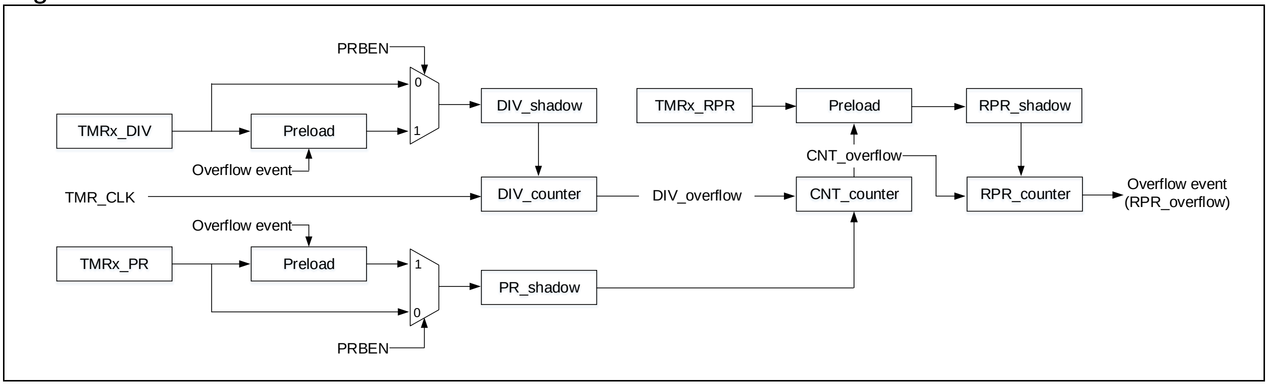
TMRx_DIV register is used to define the counter frequency of the counter. The counter counts once every DIV[15:0]+1 clock cycle. Similar to TMRx_PR register, after enabling periodic buffer, the value of the TMRx_DIV register are transferred into the shadow register at each overflow event.

Reading the TMRx_CNT register returns the current counter value. Writing the TMRx_CNT register will update the current counter value.

An overflow event is enabled by default. It can be disabled by setting OVFEN=1 in the TMRx_CTRL1 register. The OVFS bit in the TMRx_CTRL1 register is used to select the source of an overflow event, which is, by default, counter overflow or underflow, setting OVFSWTR, reset signal generated by slave mode timer controller in reset mode. Once the OVFS is set, an overflow event is generated only when overflow or underflow occurs.

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 14-44 Basic structure of a counter



Upcounting mode

This mode is enabled by setting CMSEL[1:0]=2’b00 and OWCDIR=1’b0 in the TMRx_CTRL1 register.

In upcounting 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 setting OVFIF bit to 1. If the overflow event is disabled, the counter is no longer reloaded with the prescaler and period value on counter overflow, otherwise, the prescaler and period value will be updated on an overflow event.

Figure 14-45 Overflow event when PRBEN=0

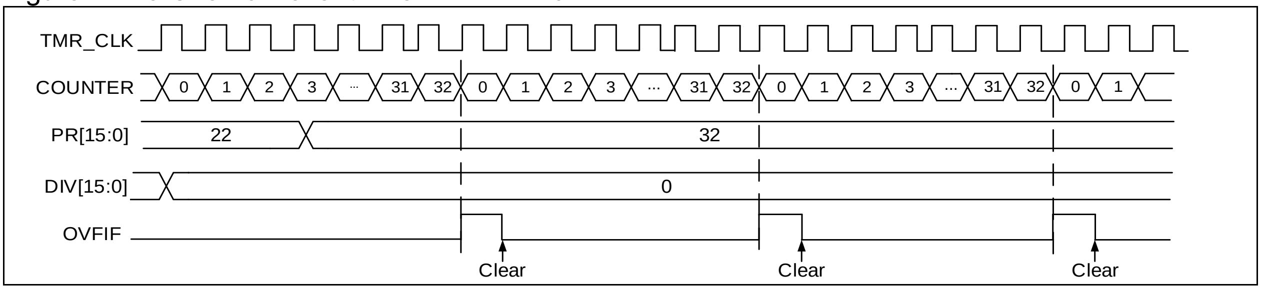
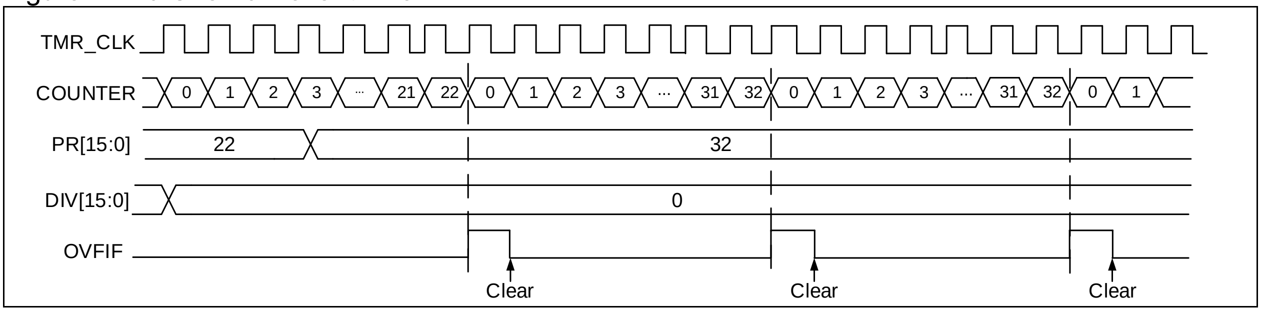


Figure 14-46 Overflow event when PRBEN=1

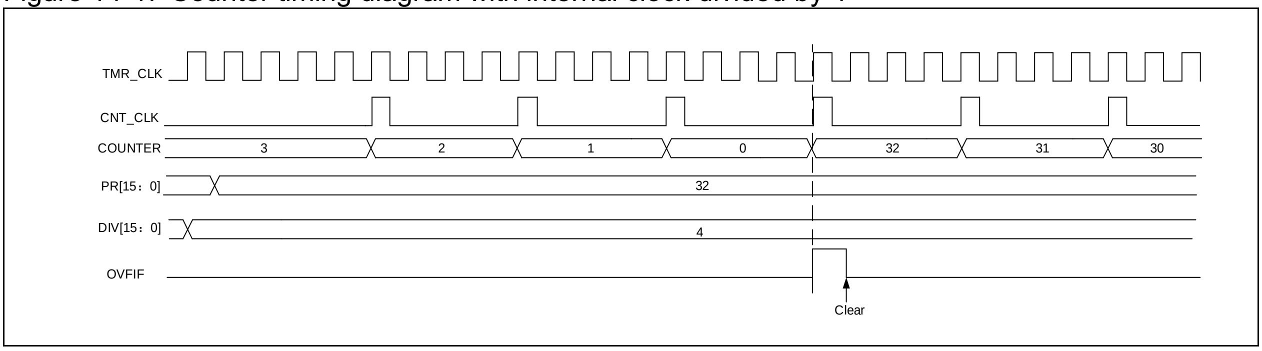


Downcounting mode

This mode is enabled by setting CMSEL[1:0]=2’b00 and OWCDIR=1’b1 in the TMRx_CTRL1 register.

In downcounting 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 14-47 Counter timing diagram with internal clock divided by 4



Up/down counting mode (center-aligned mode)

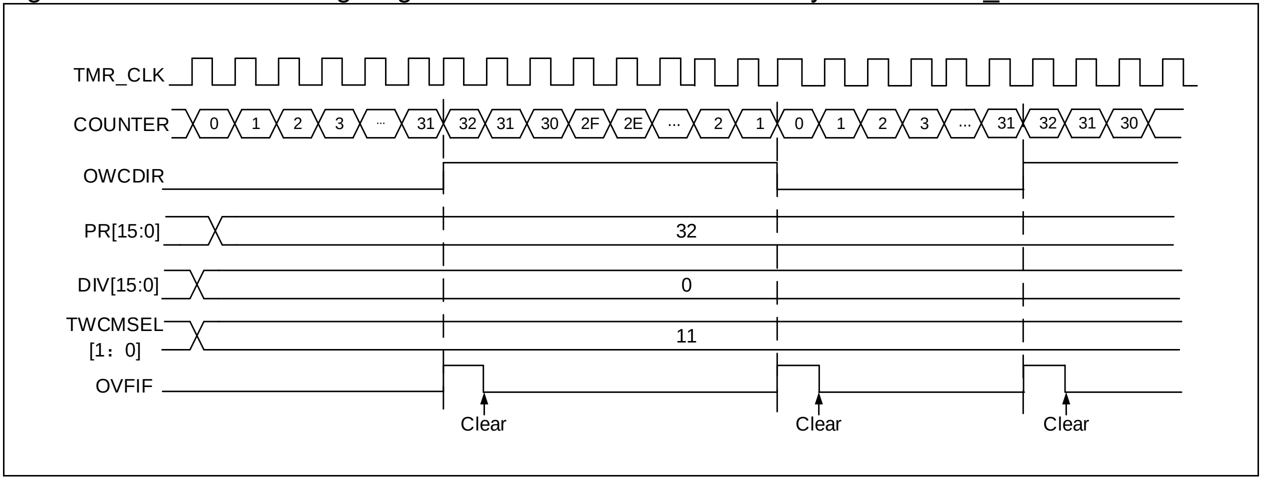
This mode is selected by setting CMSEL[1:0]≠2’b00 in the TMRx_CTRL1 register.

In up/down counting 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 used to select the condition under which the CxIF flag is set in two-way counting mode. In other words, when TWCMSEL[1:0]=2’b01 (counting mode 1) is selected, the CxIF flag is set only when the counter counts down; when TWCMSEL[1:0]=2’b10 (counting mode 2) is selected, the CxIF flag is set only when the counter counts up; when TWCMSEL[1:0]=2’b11 (counting mode 3) is selected, the CxIF flag is set when the counter counts up and down.

Note: The OWCDIR is ready-only in up/down counting mode.

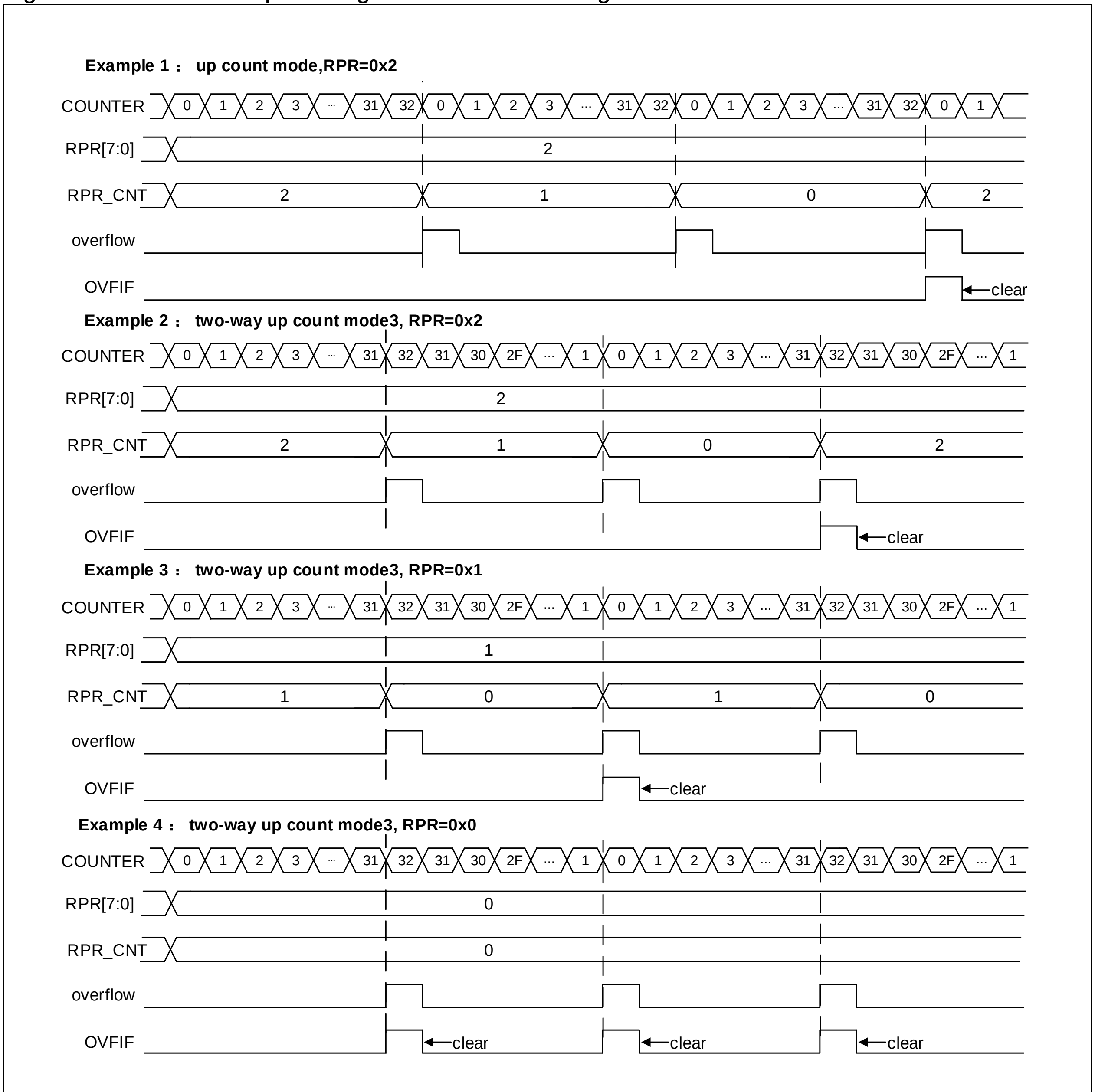
Figure 14-48 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Repetition counter mode:

The TMRx_RPR register is used to enable repetition counting mode. This mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event is generated when a counter overflow occurs (RPR[7:0]+1). The repetition counter is decremented at each counter overflow. An overflow event is generated when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

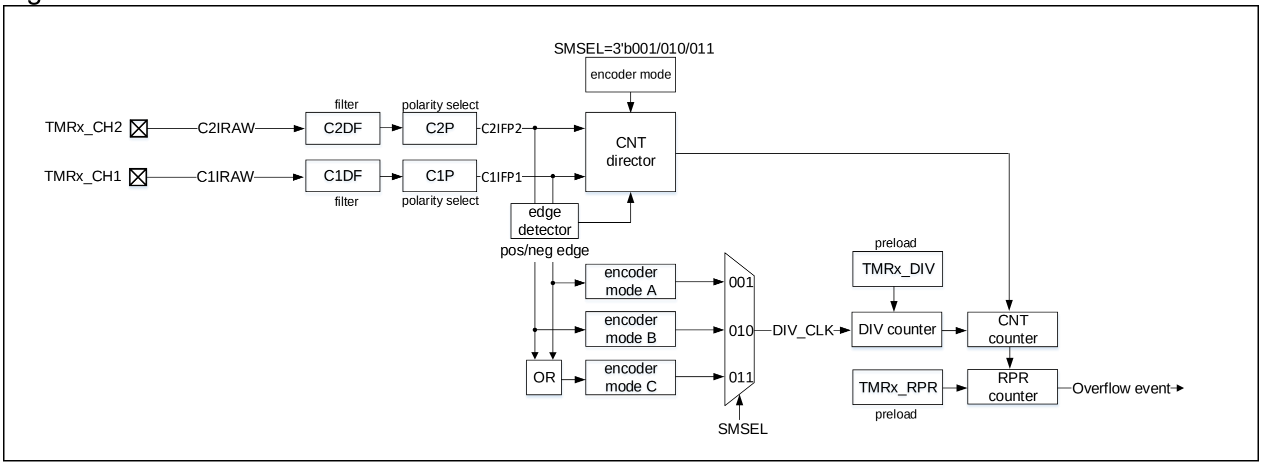
Figure 14-49 OVFIF in upcounting mode and central-aligned mode



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, as shown in the table below:

Figure 14-50 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.

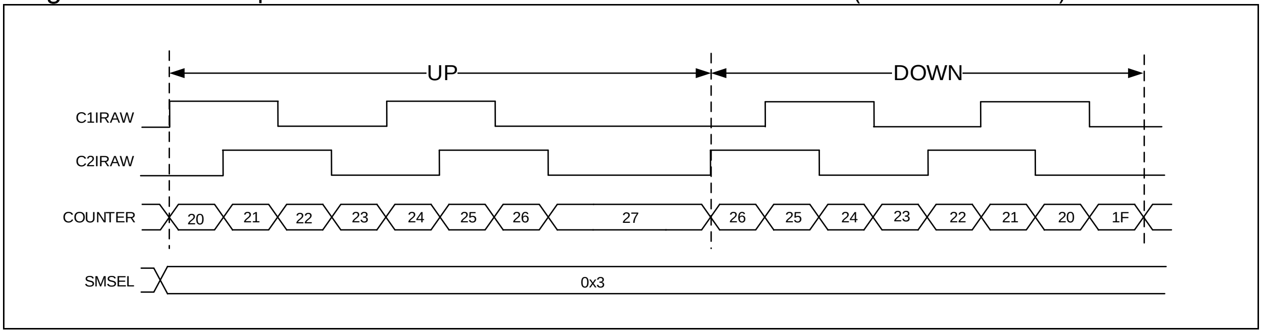
To use encoder mode, follow the procedures below:

– 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 14-8 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 14-51 Example of counter behavior in encoder interface mode (encoder mode C)



14.3.3.3 TMR input function

Each timer of TMR9 and TMR12 has two independent channels that can be configured as input or output each.

As input, each channel input signal is processed as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSE bit is used to select TMRx_CHx as the source of C1IRAW
- CxIRAW goes through digital filter and generates the filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF goes through edge detector, and generates 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 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, C1IFP2 is the C1IFP1 signal that is from channel 1 and processed by channel 2 edge detector). 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.

Figure 14-52 Input/output channel 1 main circuit

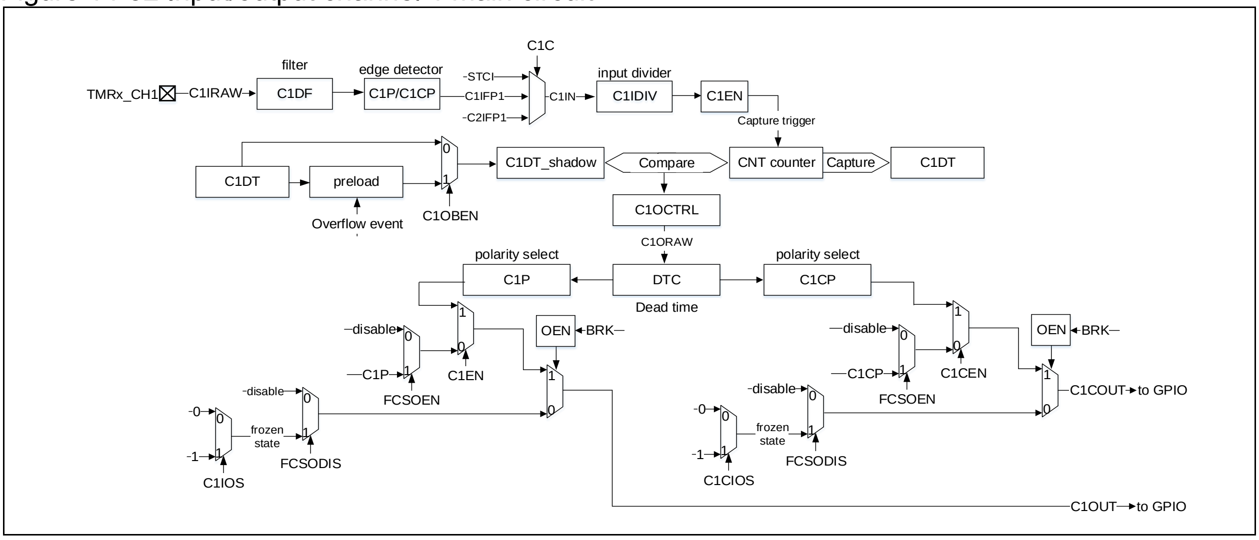
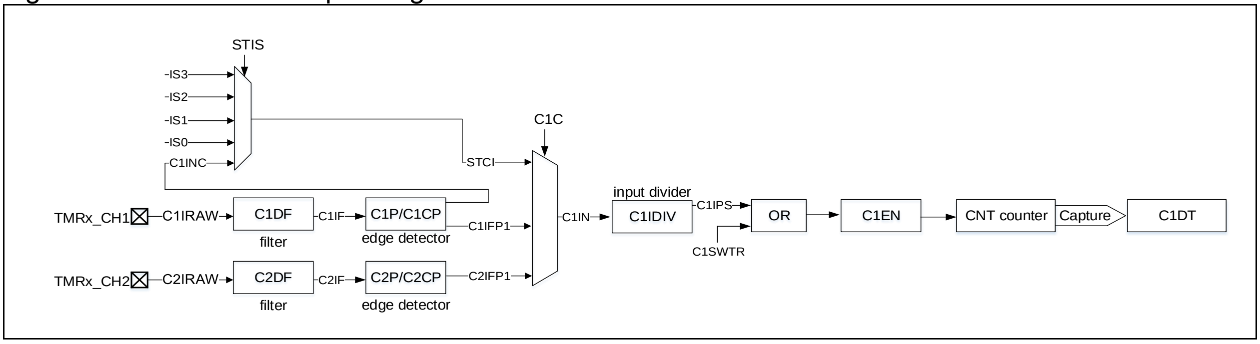


Figure 14-53 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT register latches 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 DMA request will be generated if the CxIEN bit or CxDEN bit is enabled. If the selected trigger signal is detected when CxIF=1, a capture overflow event is generated. The previous counter value will be overwritten by the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, follow the procedure below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set C1IN signal filter bandwidth (CxDF[3: 0])
- Set the active edge of C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTR register
- Program C1IN signal capture frequency divider (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

PWM input mode is applied to channel 1 and 2. To use this mode, both C1IN and C2IN are mapped on the same TMRx_CHx, and the CxIFPx of either channel 1 or channel 2 must be configured as trigger input and slave mode controller is configured in reset mode.

The PWM input mode can be used to measure the period and duty cycle of the PWM input signal. For example, the user can measure the period and duty cycle of the PWM applied on channel 1 using the following procedures:

- Set C1C=2‘b01: select C1IN for C1IFP1
- Set C1P=1’b0, select C1IFP1 rising edge active
- Set C2C=2‘b10, select C2IN for C1IFP2
- Set C2P=1’b1, select C1IFP2 falling edge active
- Set STIS=3’b101, select the slave mode timer trigger signal as C1IFP1
- Set SMSEL=3‘b100: configure the slave mode controller in reset mode
- Set C1EN=1’b1 and C2EN=1’b1. Enable channel 1 and input capture

After above configuration, the rising edge of channel 1 input signal will trigger the capture and stores the capture value into C1DT register, and it will reset the counter at the same time. The falling edge of the channel 1 input signal triggers the capture and stores the capture value into C2DT register. The period of the channel 1 input signal is calculated through C1DT, and its duty cycle through C2DT.

Figure 14-54 PWM input mode configuration example

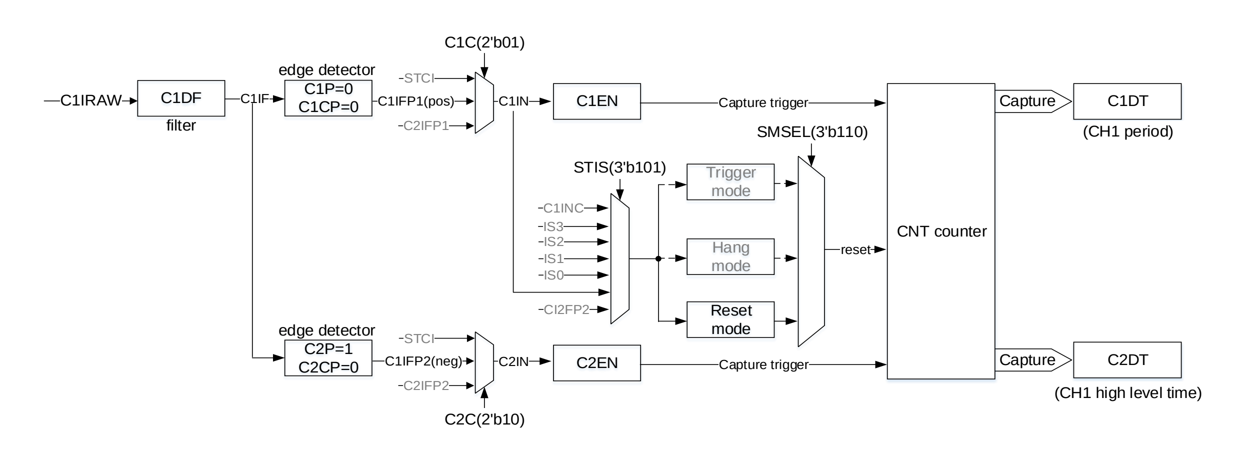
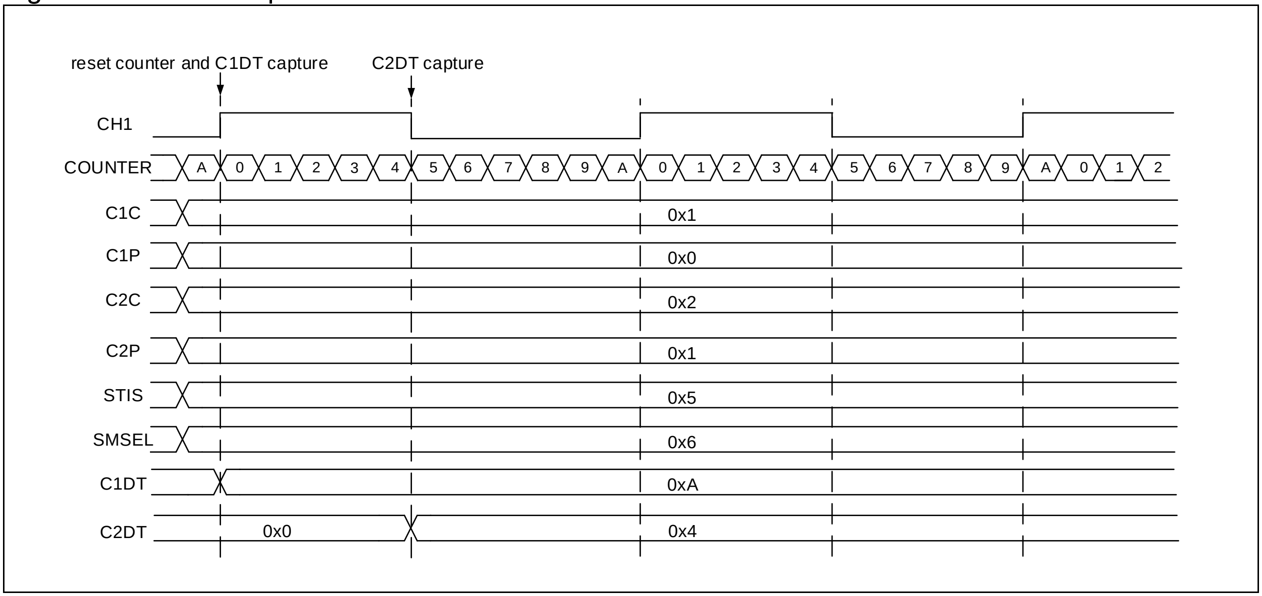


Figure 14-55 PWM input mode



14.3.3.4 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. TMR9 and TMR12 differ in output function on different channels.

Figure 14-56 Channel 1 output stage

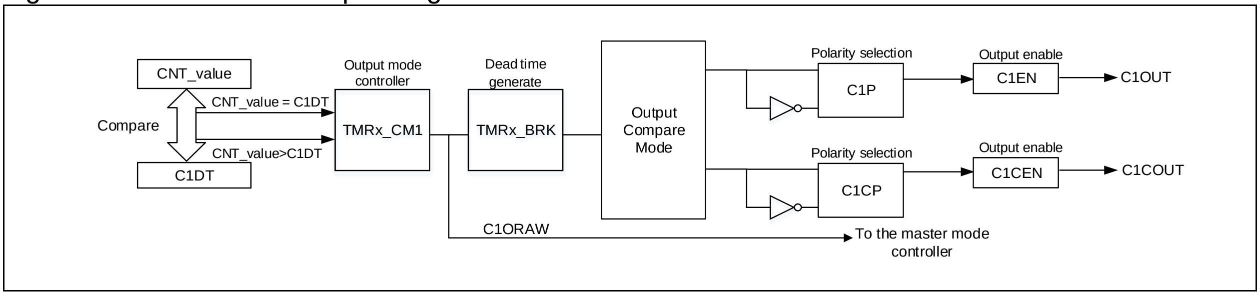
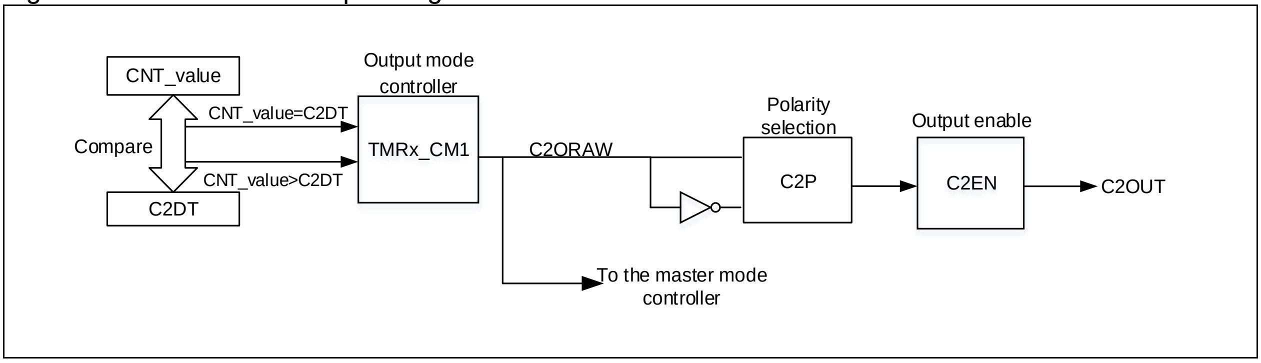


Figure 14-57 Channel 2 output stage



Output mode

Write CxC[1: 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:

Enable PWM mode A by setting CxOCTRL=3'b110. In upcounting mode, C1ORAW outputs high when TMRx_C1DT>TMRx_CVAL, otherwise, it is low; in downcounting mode, C1ORAW outputs low when TMRx_C1DT<TMRx_CVAL, otherwise, it is high.

To use PWM mode A, the following procedures are recommended:

- Set PWM period through TMRx_PR register
- Set PWM duty cycles through TMRx_CxD
- Select PWM mode A by setting CxOCTRL=3'b110 in the TMRx_CM1/CM2 register
- Set counting frequency through TMRx_DIV register
- Select counting mode by setting the TWCMSEL[1:0] bit in the TMRx_CTRL1 register
- Select output polarity through the CxP and CxCP bits in the TMRx_CCTRL register
- Enable channel output through the CxEN and CxCEN bits in the TMRx_CCTRL register
- Enable TMRx output through the OEN bit in the TMRx_BRK register
- Configure GPIOs corresponding to TMR output channels as multiplexed mode
- Enable TMRx to start counting through the TMREN bit in the TMRx_CTRL1 register.

PWM mode B:

Enable PWM mode B by setting CxOCTRL=3'b111. In upcounting mode, C1ORAW outputs low when TMRx_C1DT>TMRx_CVAL, otherwise, it is high; In downcounting mode, C1ORAW outputs high when TMRx_C1DT<TMRx_CVAL, otherwise, it is low.

Forced output mode:

Enable forced output mode by setting CxOCTRL=3'b100/101. In this case, the CxORAW is forced to be the programmed level, regardless of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.

Output compare mode:

Enable output compare mode by setting CxOCTRL=3'b001/010/011. 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. Enable one-pulse by setting OCMEN=1. 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 in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.

Fast output mode:

Enable this mode by setting CxOIEN=1. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but change 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 14-58 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-59 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 14-60 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 14-58 C1ORAW toggles when counter value matches the C1DT value

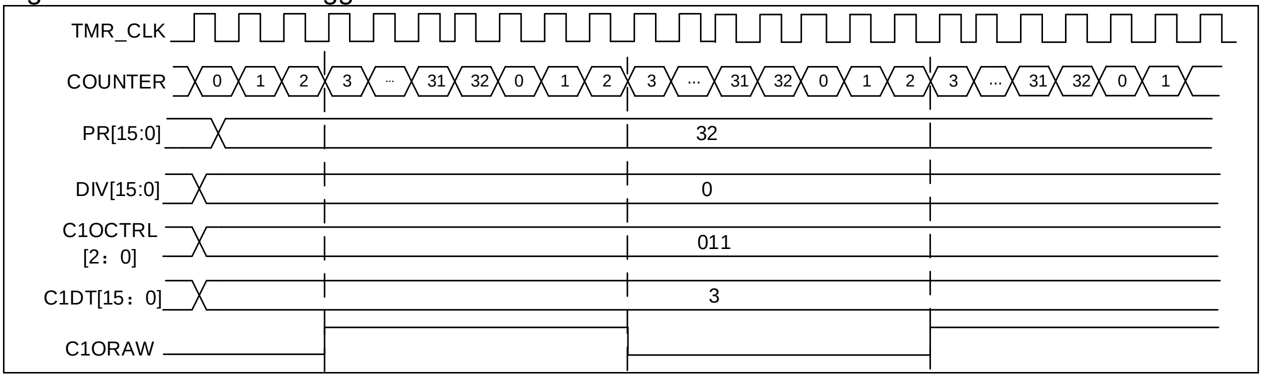


Figure 14-59 Upcounting mode and PWM mode A

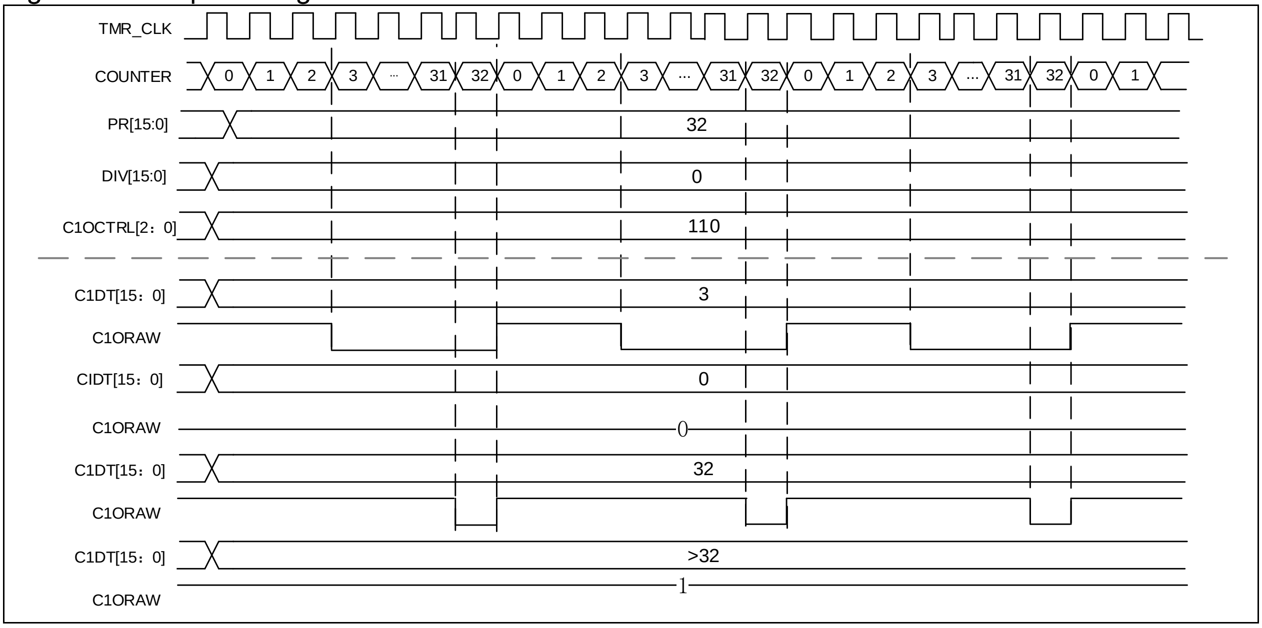
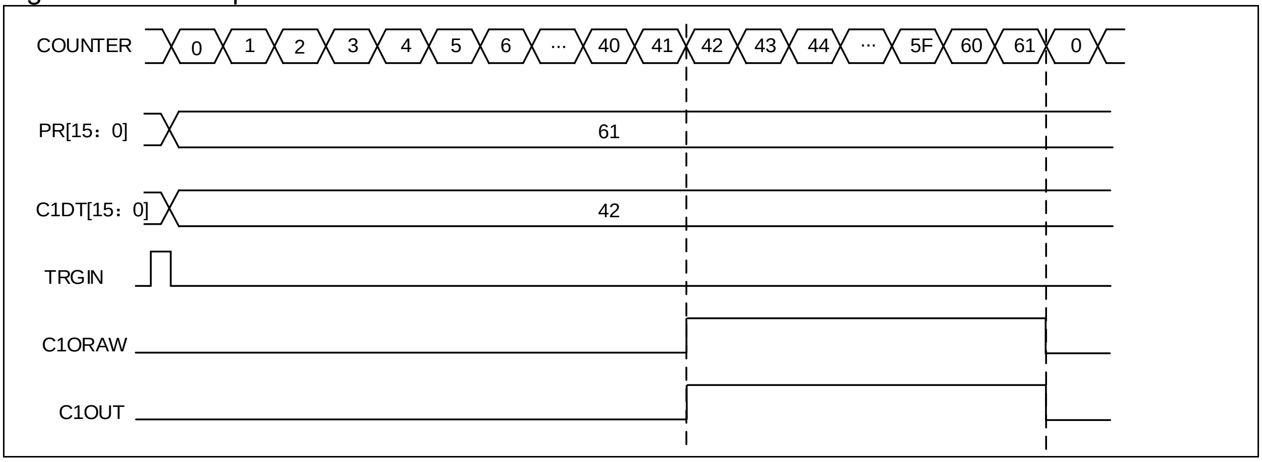


Figure 14-60 One-pulse mode



Master mode timer event output

When TMR is used as a master timer, one of the following source of signals can be selected as TRGOUT output to a slave mode timer. This is done by setting the PTOS bit in the TMRxCTRL2 register.

- PTOS=3’b000, TRGOUT output software overflow event (OVFSWTR bit in TMRx_SWEVT register)
- PTOS=3’b001, TRGOUT output counter enable
- PTOS=3’b010, TRGOUT output counter overflow event
- PTOS=3’b011, TRGOUT output capture and compare event
- PTOS=3’b100, TRGOUT output C1ORAW
- PTOS=3’b101, TRGOUT output C2ORAW

Dead-time insertion

The channels for TMR9 and TMR12 contain a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is selected by CxCP. Refer to Table 14-15 for more information about the output state of CxOUT and CxCOUT.

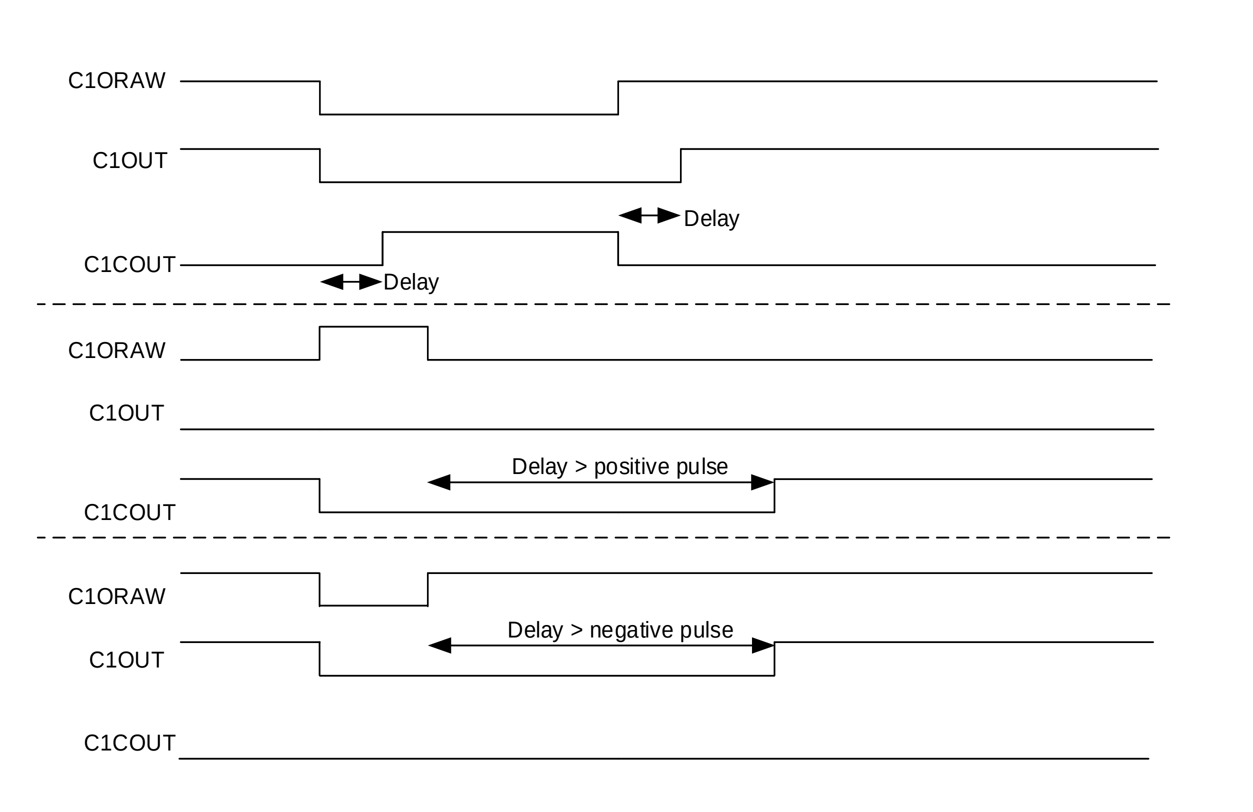
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

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 14-61 gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-61 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 14-11 for more details.

The break source can be a break input pin or a clock failure event. The polarity is controlled by 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.

  Note: The dead-time duration is usually longer than usual (around 2 clk_tmr clock cycles) due to OEN synchronization logic.

  - 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 14-62 TMR output control

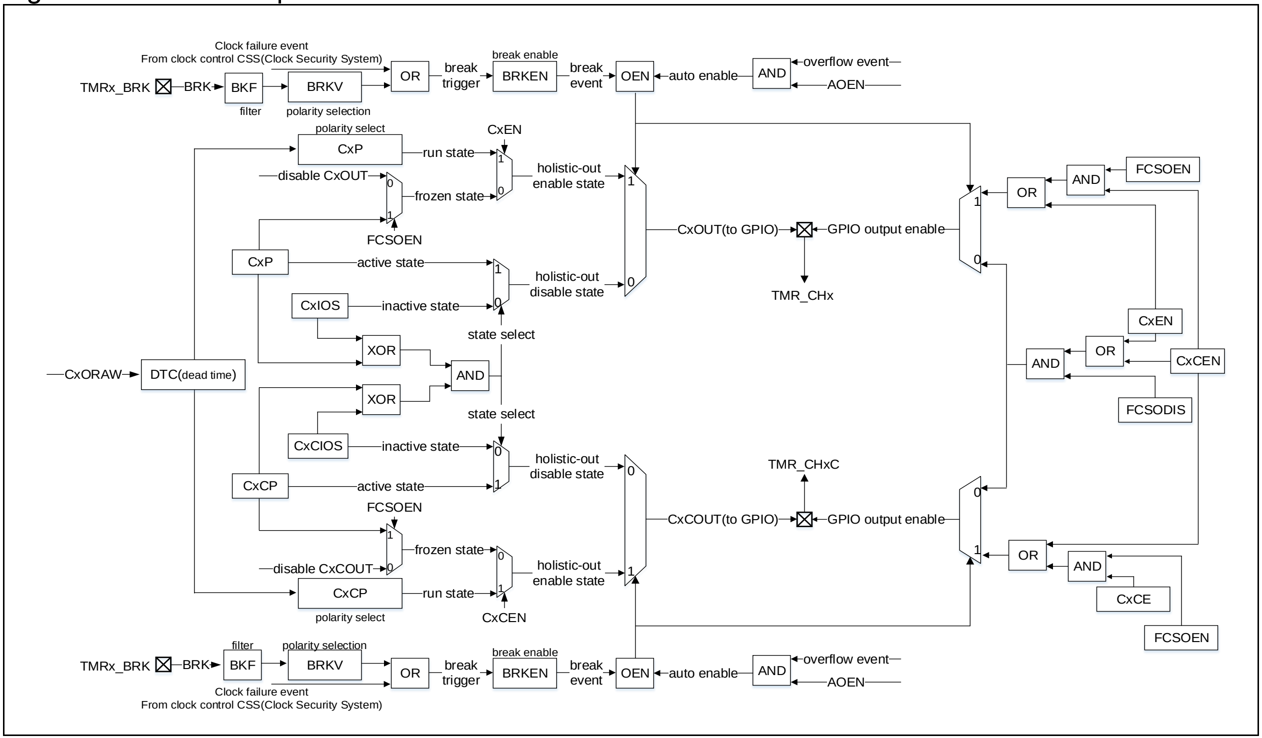
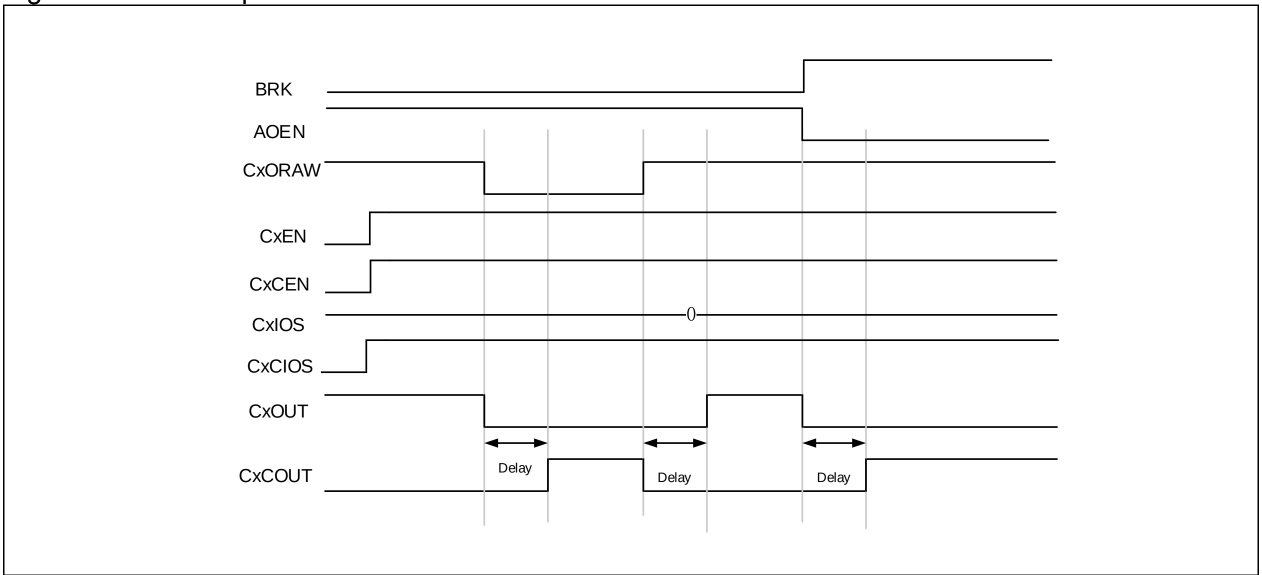


Figure 14-63 Example of TMR break function



14.3.3.6 TMR synchronization

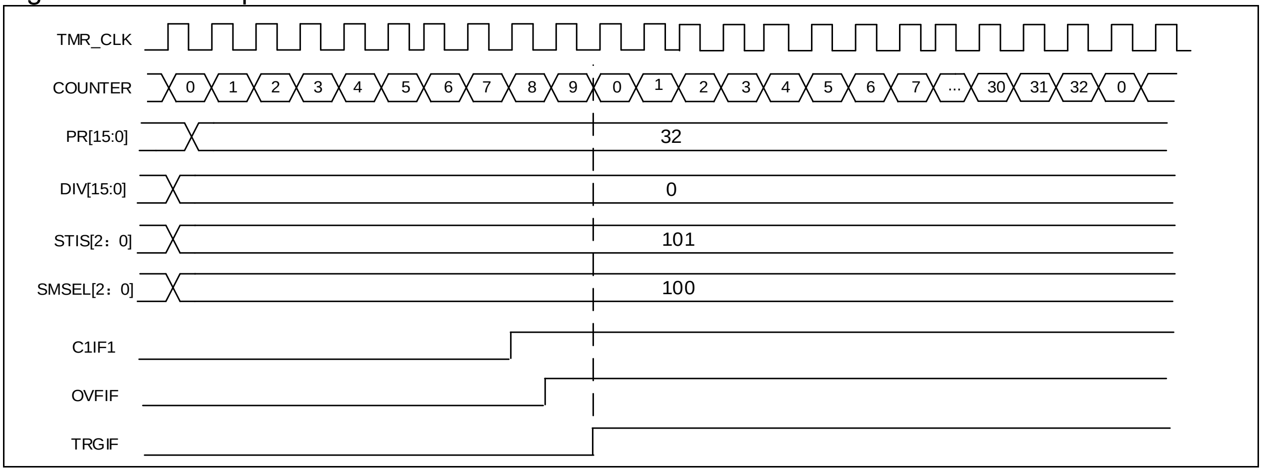
The master and slave timers are linked together internally for timer synchronization. Master mode timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

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.

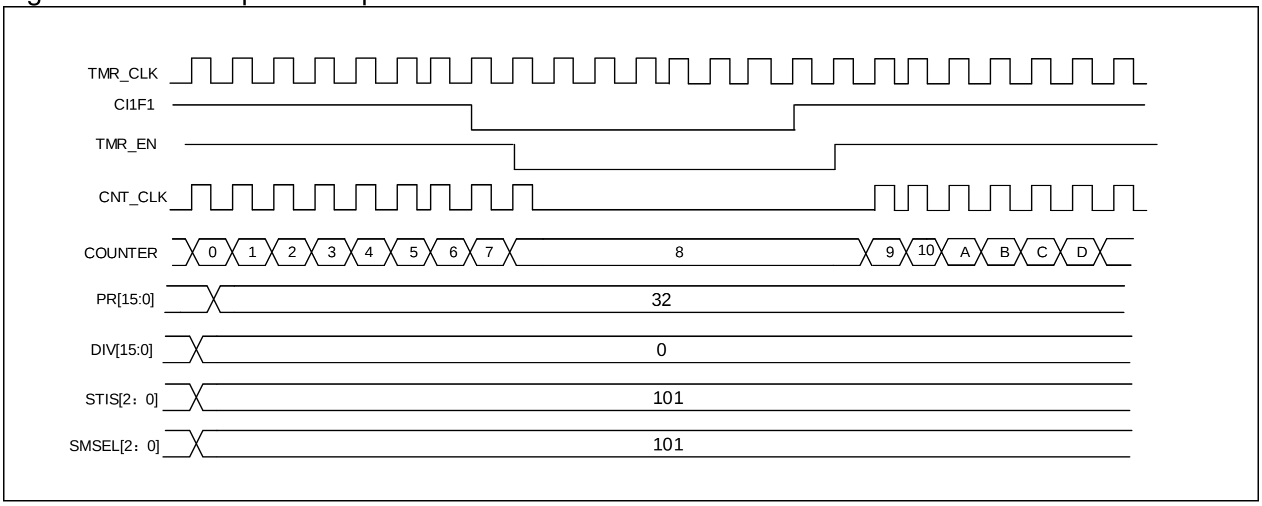
Figure 14-64 Example of reset mode



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.

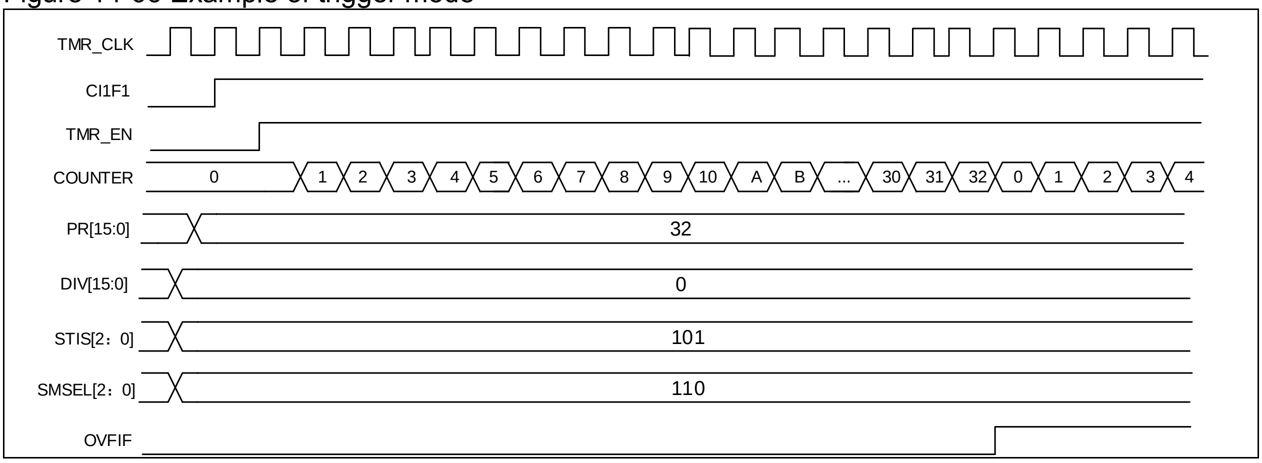
Figure 14-65 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-66 Example of trigger mode



See *Chapter 14.2.3.5* for more information about timer synchronization.

14.3.3.7 Debug mode

When the microcontroller enters debug mode (Cortex™-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module.

14.3.4 TMR9 and TMR12 registers

Table 14-9 TMR9 and TMR12 register map and reset value

Register name	Register	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_CCTRL	0x20	0x0000
TMRx_CVAL	0x24	0x0000
TMRx_DIV	0x28	0x0000
TMRx_PR	0x2C	0x0000
TMRx_RPR	0x30	0x0000
TMRx_C1DT	0x34	0x0000
TMRx_C2DT	0x38	0x0000
TMRx_BRK	0x44	0x0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000

14.3.4.1 TMR9 and TMR12 control register1 (TMRx_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at 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}). 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	resd	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: Enabled 1: Disabled

14.3.4.2 TMR9 and TMR12 control register 2 (TMRx_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 15: 2	Reserved	0x00	resd	Kept at default value.
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 Output disabled (OEN= 0) after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state Output disabled (OEN = 0) after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7	Reserved	0x0	resd	Kept at default value.
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: Overflow 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 that have 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 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 TMR9 and TMR12 slave timer control register (TMR1_STCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at default value.
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 14-7 for more information on ISx for each timer.
Bit 3	Reserved	0x0	resd	Kept at 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.

14.3.4.4 TMR9 and TMR12 DMA/interrupt enable register (TMRx_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at 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: 11	Reserved	0x00	resd	Kept at default value.
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: 3	Reserved	0x00	resd	Kept at 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

14.3.4.5 TMR9 and TMR12 interrupt status register (TMRx_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 11	Reserved	0x00	resd	Kept at 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	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: 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. 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 TMR9 and TMR12 software event register (TMRx_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at 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: 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.

14.3.4.7 TMR9 and TMR12 channel mode register 1 (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 bit. 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 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 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.

14.3.4.8 TMR9 and TMR12 Channel control register (TMRx_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at default value.
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: 00: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 01: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted. 10: Reserved 11: C1IN active edge is on its rising edge and falling edge. When used as external trigger, C1IN is not inverted.
Bit0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

14.3.4.9 TMR9 and TMR12 counter value (TMRx_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.3.4.10 TMR9 and TMR12 division value (TMRx_DIV)

Bit	Register	Reset value	Type	Description
Bit 15: 0	DIV	0x0000	rw	Divider value The counter clock frequency $f_{CK\_CNT} = f_{TMR\_CLK} / (DIV[15:0]+1)$. The value of this register is transferred to the actual prescaler register when an overflow event occurs.

14.3.4.11 TMR9 and TMR12 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.

14.3.4.12 TMR9 and TMR12 repetition period register (TMRx_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at 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.

14.3.4.13 TMR9 and TMR12 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.

14.3.4.14 TMR9 and TMR12 channel 2 data register (TMRx_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.

14.3.4.15 TMR9 and TMR12 break register (TMRx_BRK)

Bit	Register	Reset value	Type	Description
Bit 19: 16	BKF	0x0	rw	Break input filter This field is used to set the filter for break input. The filter number N indicates that the input edge can pass through filter only after N sampling events. 0000: $f_{SAMPLING}$=f_{DTS} (no filtering) 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: $f_{SAMPLING}$ = 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 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

14.3.4.16 TMR9 and TMR12 DMA control register (TMRx_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at 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 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 ......

14.3.4.17 TMR9 and TMR12 DMA data register (TMRx_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

14.4 General-purpose timer (TMR10/11/13/14)

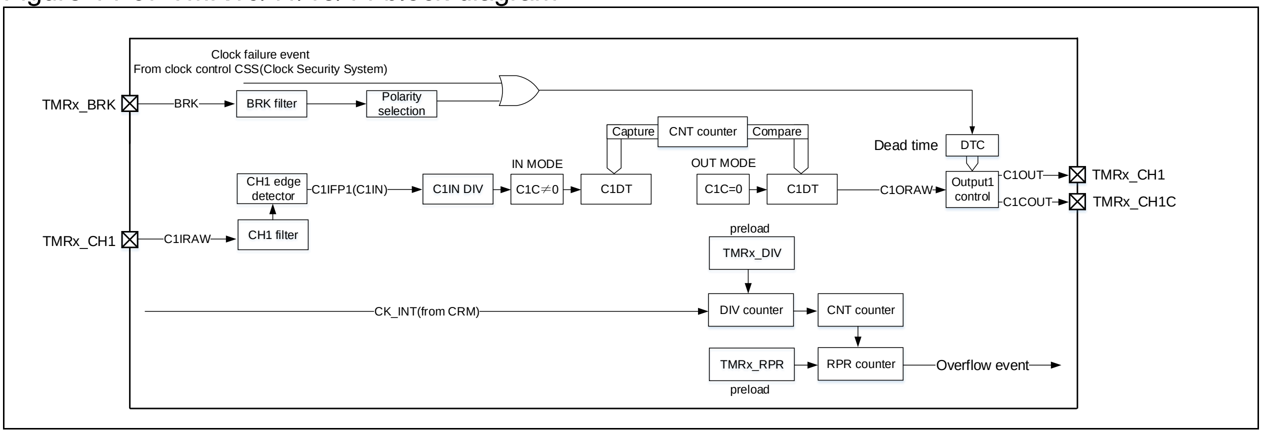
14.4.1 TMRx introduction

The general-purpose timers (TMR10/11/13/14) consist of a 16-bit upcounter, one capture/compare register, and one independent channel. They can be used for dead-time insertion, input capture and programmable PWM output.

14.4.2 TMRx main features

- Counter clock source: internal clock, external input and internal trigger inputs
- 16-bit upcounter, and 8-bit repetition counter
- One independent channel for input capture, output compare, PWM generation, one-pulse mode output and dead-time insertion
- One independent channel for complementary output
- TMR break function
- Synchronization control between master and slave timers
- Interrupt/DMA generation at overflow event, break signal input and channel events
- Support TMR burst DMA transfer

Figure 14-67 TMR10/11/13/14 block diagram

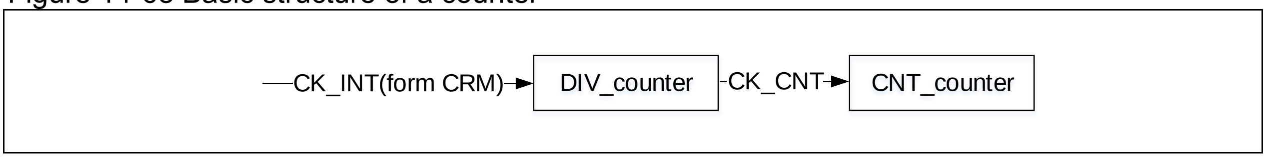


14.4.3 TMRx functional overview

14.4.3.1 Counting clock

The count clock of TMR10/11/13/14 can be provided by the internal clock (CK_INT).

Figure 14-68 Basic structure of a counter



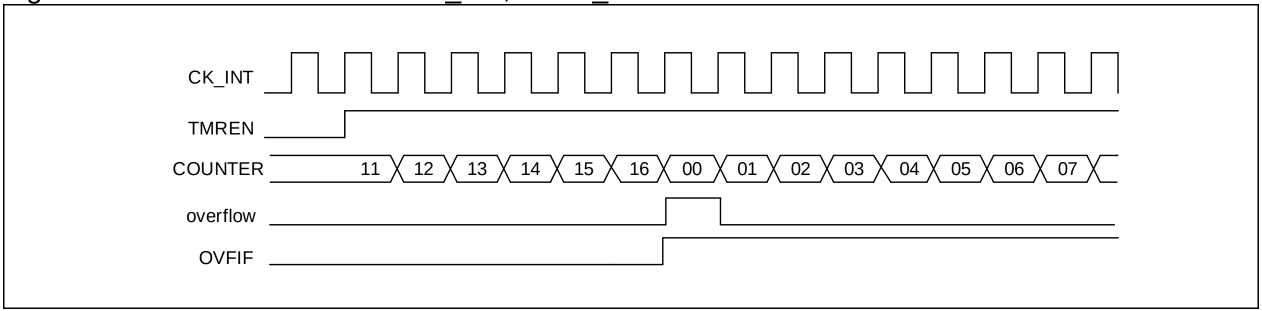
Internal clock (CK_INT)

By default, the CK_INT, which is 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.

Follow the configuration steps below:

- Select counting mode by setting the TWCMSEL[1:0] in TMRx_CTRL1 register. If a unidirectional aligned counting mode is selected, there is a need to select counting direction through the OWCDIR in TMRx_CTRL1 register.
- Set counting frequency through TMRx_DIV register
- Set counting cycles through TMRx_PR register
- Enable a counter by setting the TMREN bit in the TMRx_CTRL1 register

Figure 14-69 Control circuit with CK_INT, TMRx_DIV=0x0 and PR=0x16



14.4.3.2 Counting mode

The TMR10/11/13/14 supports multiple counting modes to meet various application scenarios. Each consists of a 16-bit upcounter.

The TMRx_PR register is used to define counting period of counter. 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.

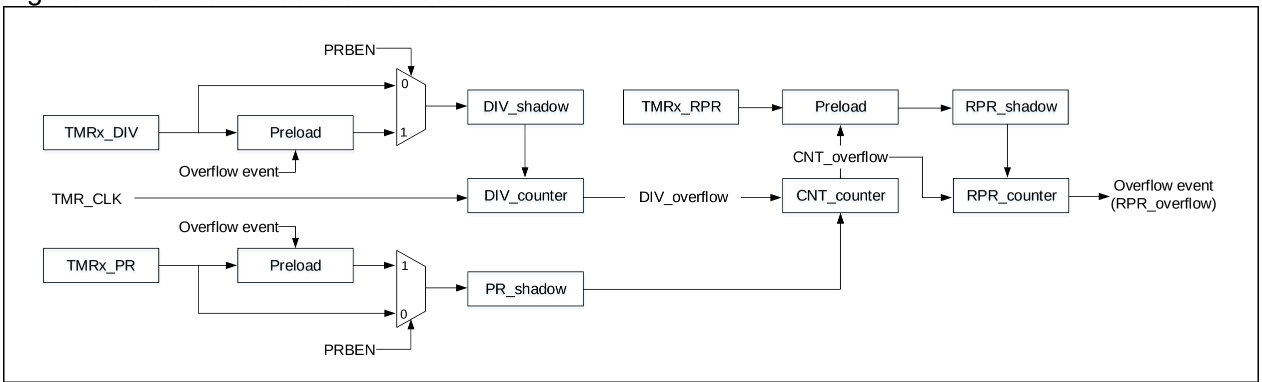
TMRx_DIV register is used to define the counter frequency of the counter. The counter counts once every DIV[15:0]+1 clock cycle. Similar to TMRx_PR register, after enabling periodic buffer, the value of the TMRx_DIV register are transferred into the shadow register at each overflow event.

Reading the TMRx_CNT register returns the current counter value. Writing the TMRx_CNT register will update the current counter value.

An overflow event is enabled by default. It can be disabled by setting OVFEN=1 in the TMRx_CTRL1 register. The OVFS bit in the TMRx_CTRL1 register is used to select the source of an overflow event, which is, by default, counter overflow or underflow, setting OVFSWTR, reset signal generated by slave mode timer controller in reset mode. Once the OVFS is set, an overflow event is generated only when overflow or underflow occurs.

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 14-70 Basic structure of a counter



Upcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register.

In upcounting 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 setting OVFIF bit to 1. If the overflow event is disabled, the counter is no longer reloaded with the prescaler and period value on counter overflow, otherwise, the prescaler and period value will be updated on an overflow event.

Figure 14-71 Overflow event when PRBEN=0

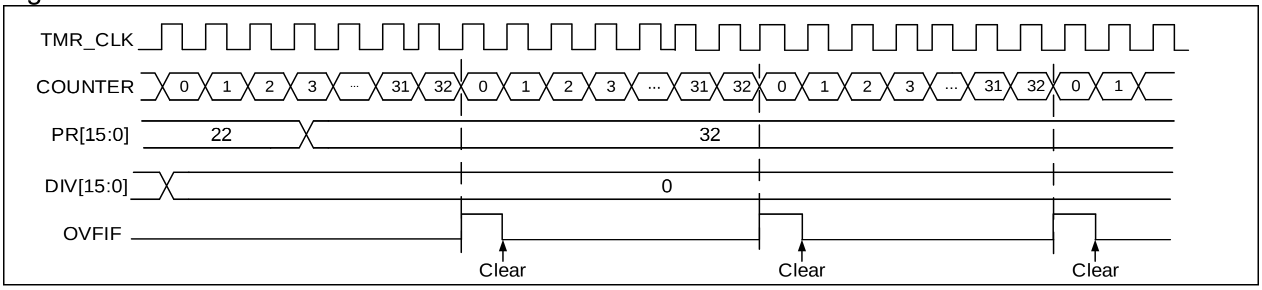
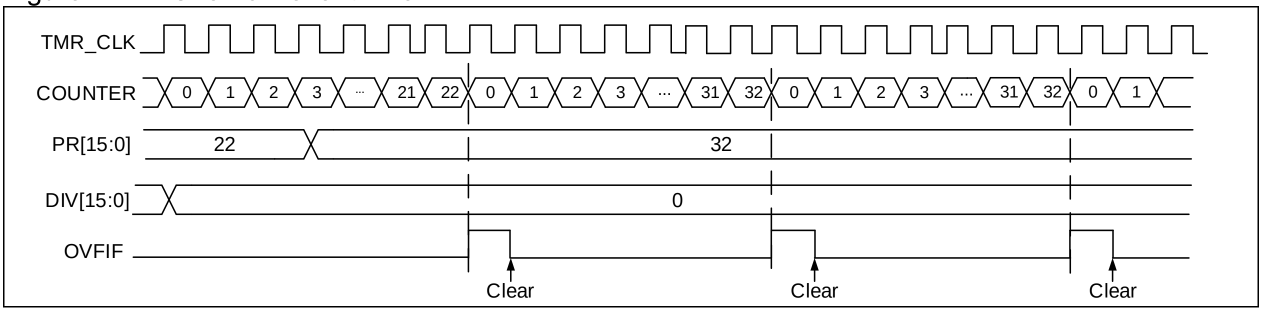


Figure 14-72 Overflow event when PRBEN=1

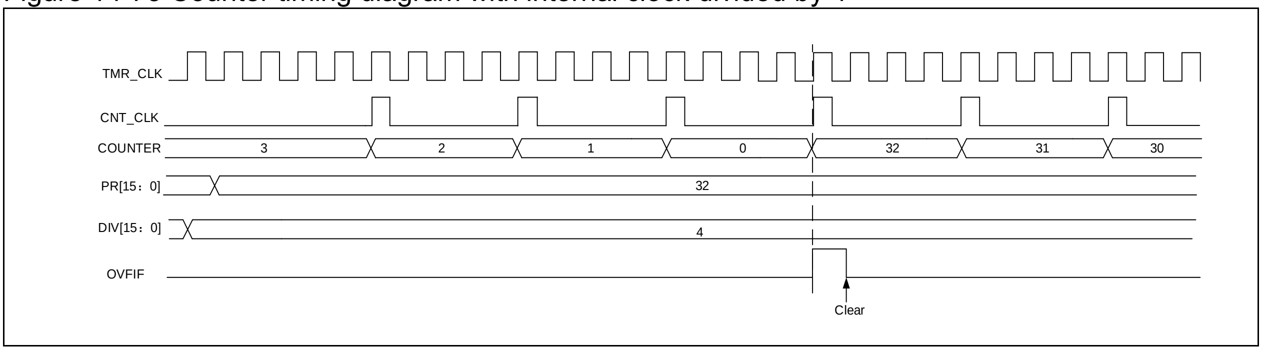


Downcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register.

In downcounting 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 14-73 Counter timing diagram with internal clock divided by 4



Up/down counting mode (center-aligned mode)

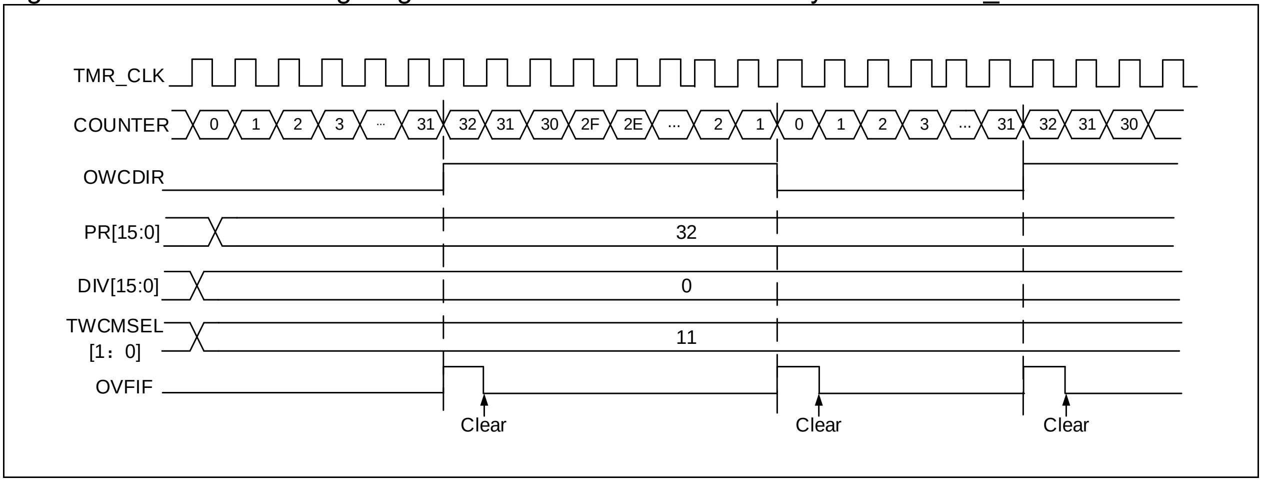
This mode is selected by setting CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register.

In up/down counting 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 used to select the condition under which the CxIF flag is set in two-way counting mode. In other words, when TWCMSEL[1:0]=2'b01 (counting mode 1) is selected, the CxIF flag is set only when the counter counts down; when TWCMSEL[1:0]=2'b10 (counting mode 2) is selected, the CxIF flag is set only when the counter counts up; when TWCMSEL[1:0]=2'b11 (counting mode 3) is selected, the CxIF flag is set when the counter counts up and down.

Note: The OWCDIR is ready-only in up/down counting mode.

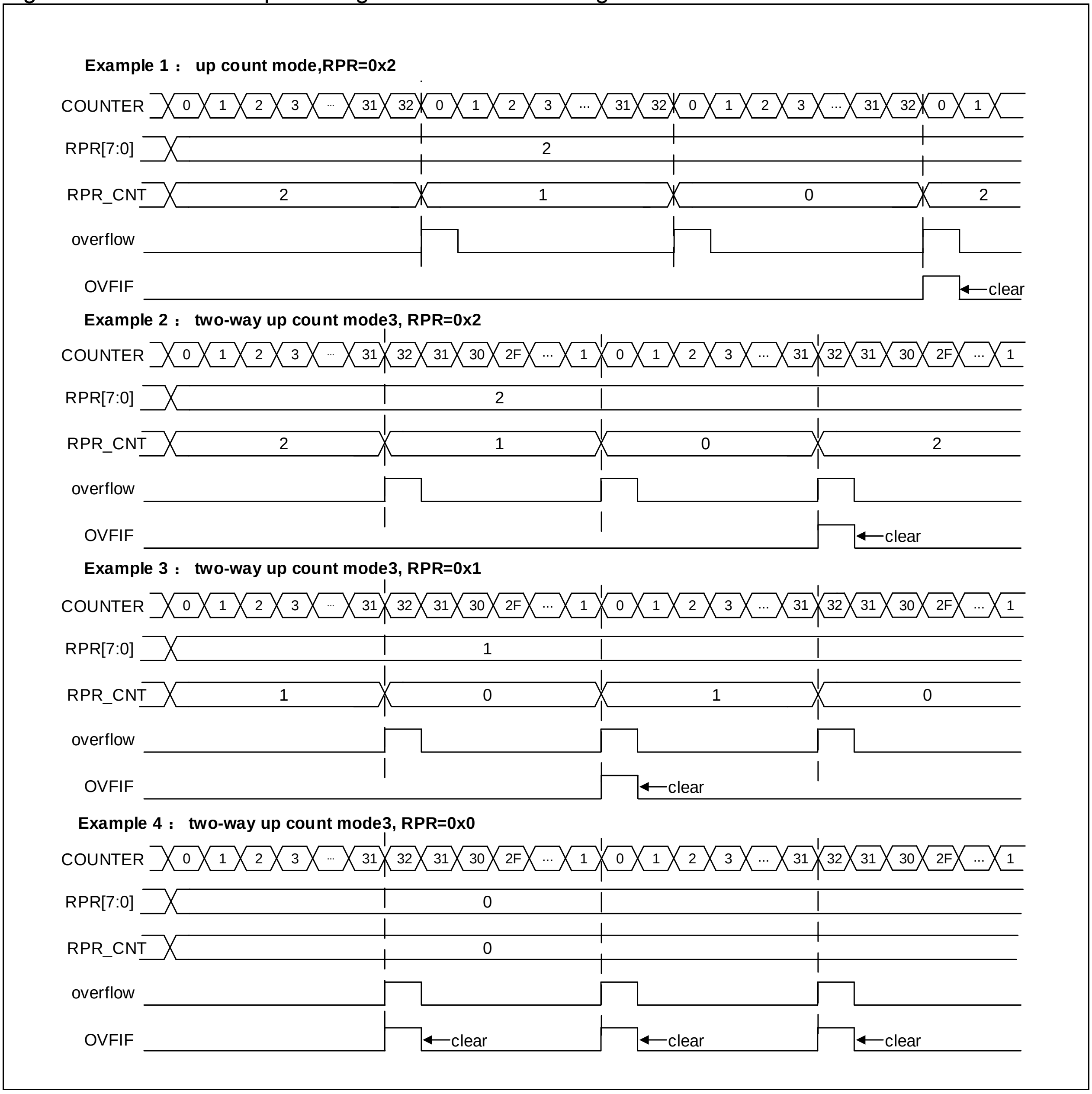
Figure 14-74 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Repetition counter mode:

The TMRx_RPR register is used to enable repetition counting mode. This mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event is generated when a counter overflow occurs (RPR[7:0]+1). The repetition counter is decremented at each counter overflow. An overflow event is generated when the repetition counter reaches 0. The frequency of the overflow event generation can be adjusted by setting the repetition counter value.

Figure 14-75 OVFIF in upcounting mode and central-aligned mode



14.4.3.3 TMR input function

Each of TMR10/11/13/14 has an independent channel that can be configured in input or output mode. As input, the channel input signal is processed as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSE bit is used to select TMRx_CHx as the source of C1IRAW
- CxIRAW goes through digital filter and generates the filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF goes through edge detector and generates 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 goes through capture signal selector and generates the CxIN signal after capture signal selection. The capture signal selection is defined by CxC bits. The CxIFPx can be used as a source of CxIN source.
- CxIN goes through input divider and generates the CxIPS signal. The divider factor can be defined as No division, /2, /4 or /8 by setting the CxIDIV bit.

Figure 14-76 Input/output channel 1 main circuit

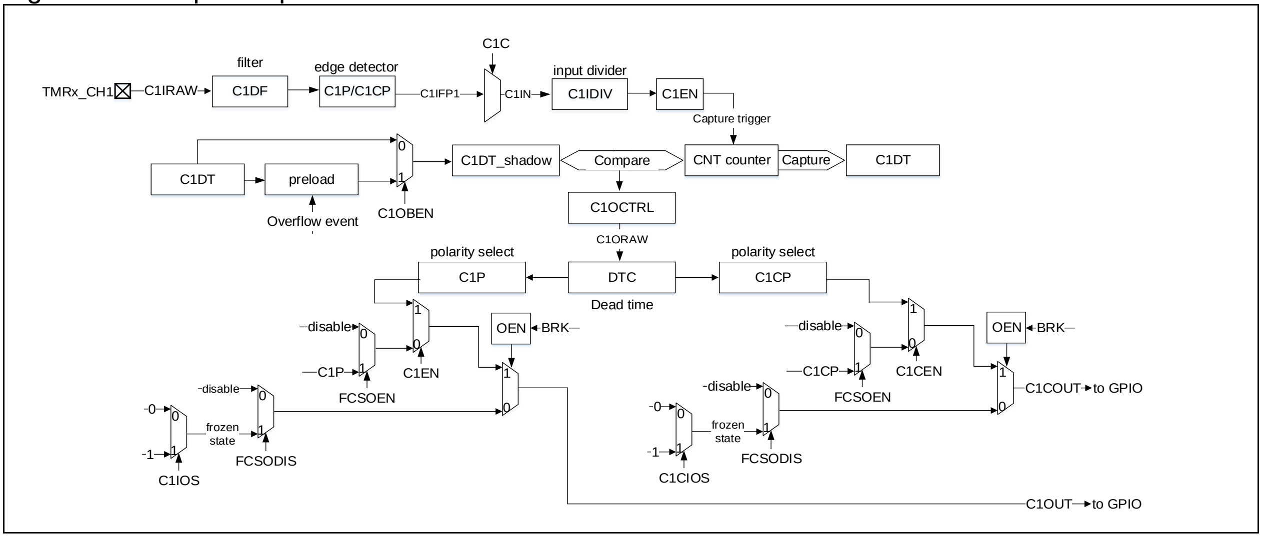
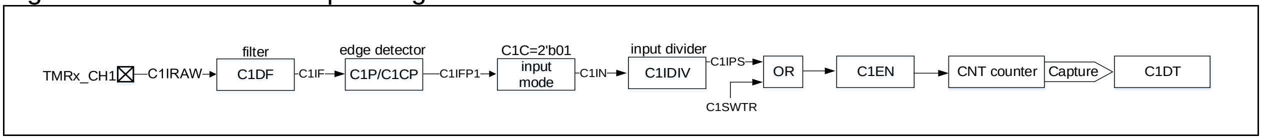


Figure 14-77 Channel 1 input stage



Input mode

In input mode, the TMRx_CxDT registers latches 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/DMA request will be generated if the CxIEN bit and CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event is generated, the previous counter value will be overwritten with the current counter value, and the CxRF is set to 1.

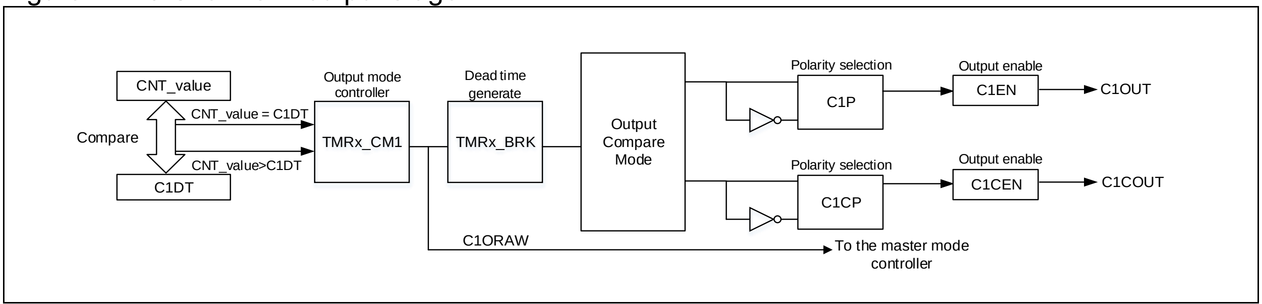
To capture the rising edge of C1IN input, following the procedure below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set C1IN signal filter bandwidth (CxDF[3: 0])
- Set the active edge of C1IN channel by writing C1P=0 (rising edge) in the TMRx_CCTRL register
- Program C1IN signal capture frequency divider (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

14.4.3.4 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 14-78 Channel 1 output stage



Output mode

Write CxC[1: 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 TMR15_PR register, while the duty cycle by the TMRx_CxDT register.

PWM mode A:

Enable PWM mode A by setting CxOCTRL=3'b110. In upcounting mode, C1ORAW outputs high when TMRx_C1DT>TMRx_CVAL, otherwise, it is low; in downcounting mode, C1ORAW outputs low when TMRx_C1DT<TMRx_CVAL, otherwise, it is high.

To use PWM mode A, the following procedures are recommended:

- Set PWM periods through TMRx_PR register
- Set PWM duty cycles through TMRx_CxD
- Select PWM mode A by setting CxOCTRL=3'b110 in the TMRx_CM1/CM2 register
- Set counting frequency through TMRx_DIV register
- Select counting mode by setting the TWCMSEL[1:0] bit in the TMRx_CTRL1 register
- Select output polarity through the CxP and CxCP bits in the TMRx_CCTRL register
- Enable channel output through the CxEN and CxCEN bits in the TMRx_CCTRL register
- Enable TMRx output through the OEN bit in the TMRx_BRK register
- Configure GPIOs corresponding to TMR output channels as multiplexed mode
- Enable TMRx to start counting through the TMREN bit in the TMRx_CTRL1 register.

PWM mode B:

Enable PWM mode B by setting CxOCTRL=3'b111. In upcounting mode, C1ORAW outputs low when TMRx_C1DT>TMRx_CVAL, otherwise, it is high; in downcounting mode, C1ORAW outputs high when TMRx_C1DT<TMRx_CVAL, otherwise, it is low.

Forced output mode:

Enable forced output mode by setting CxOCTRL=3'b100/101. In this case, the CxORAW is forced to be the programmed level, regardless of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.

Output compare mode:

Enable output compare mode by setting CxOCTRL=3'b001/010/011. 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. Enable one-pulse by setting OCMEN=1. 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 in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.

Fast output mode:

Enable this mode by setting CxOIEN=1. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but change 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 14-79 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-80 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 14-81 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 14-79 C1ORAW toggles when counter value matches the C1DT value

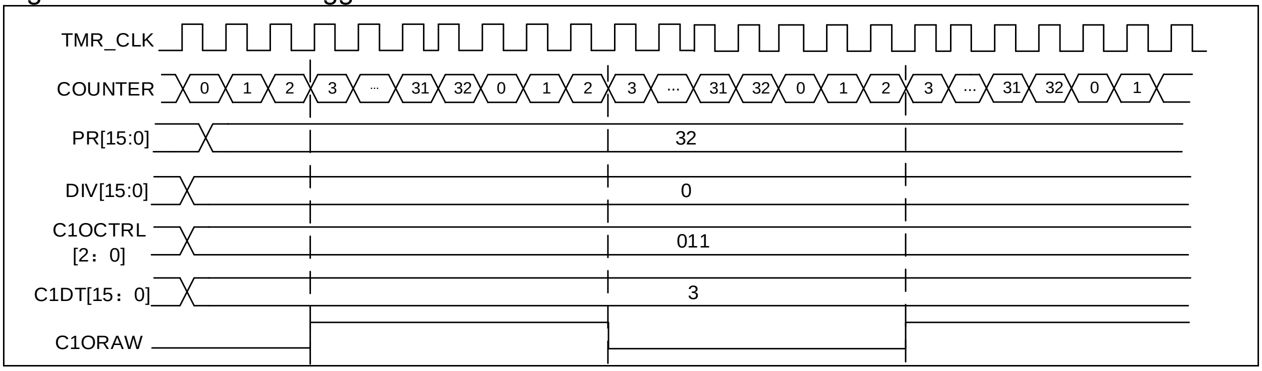


Figure 14-80 Upcounting mode and PWM mode A

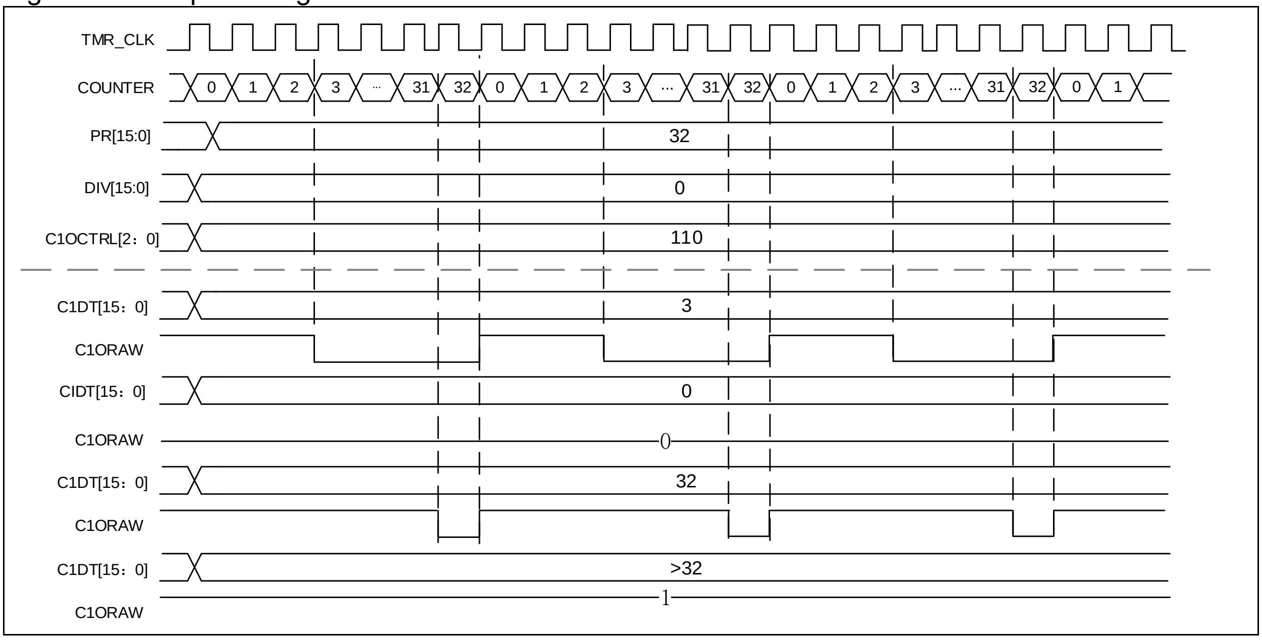
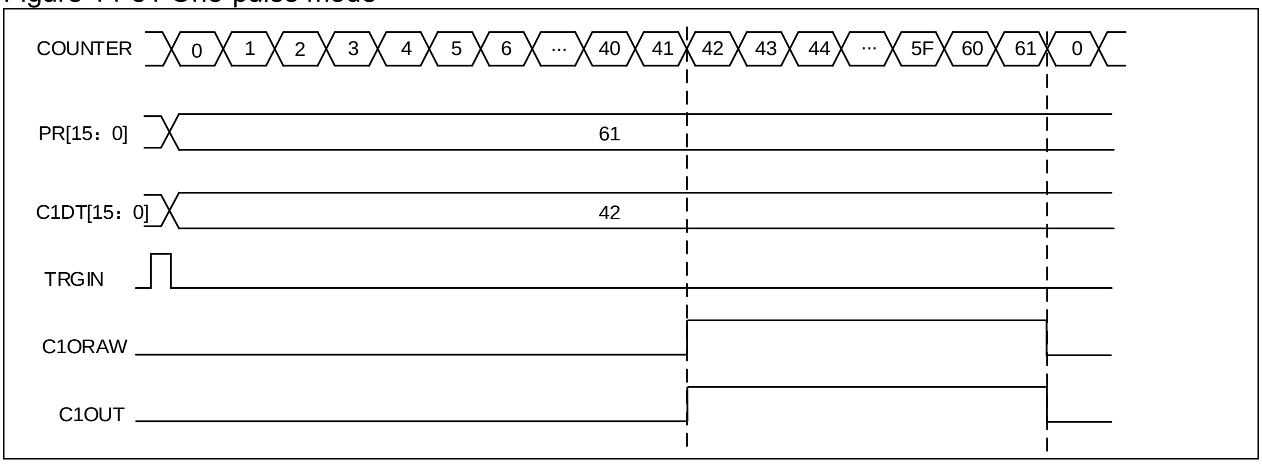


Figure 14-81 One-pulse mode



Dead-time insertion

The TMRx contains a set of reverse channel output. This function is enabled by the CxCEN bit and its polarity is selected by CxCP. Refer to Table 14-15 for more information about the output state of CxOUT and CxCOUT.

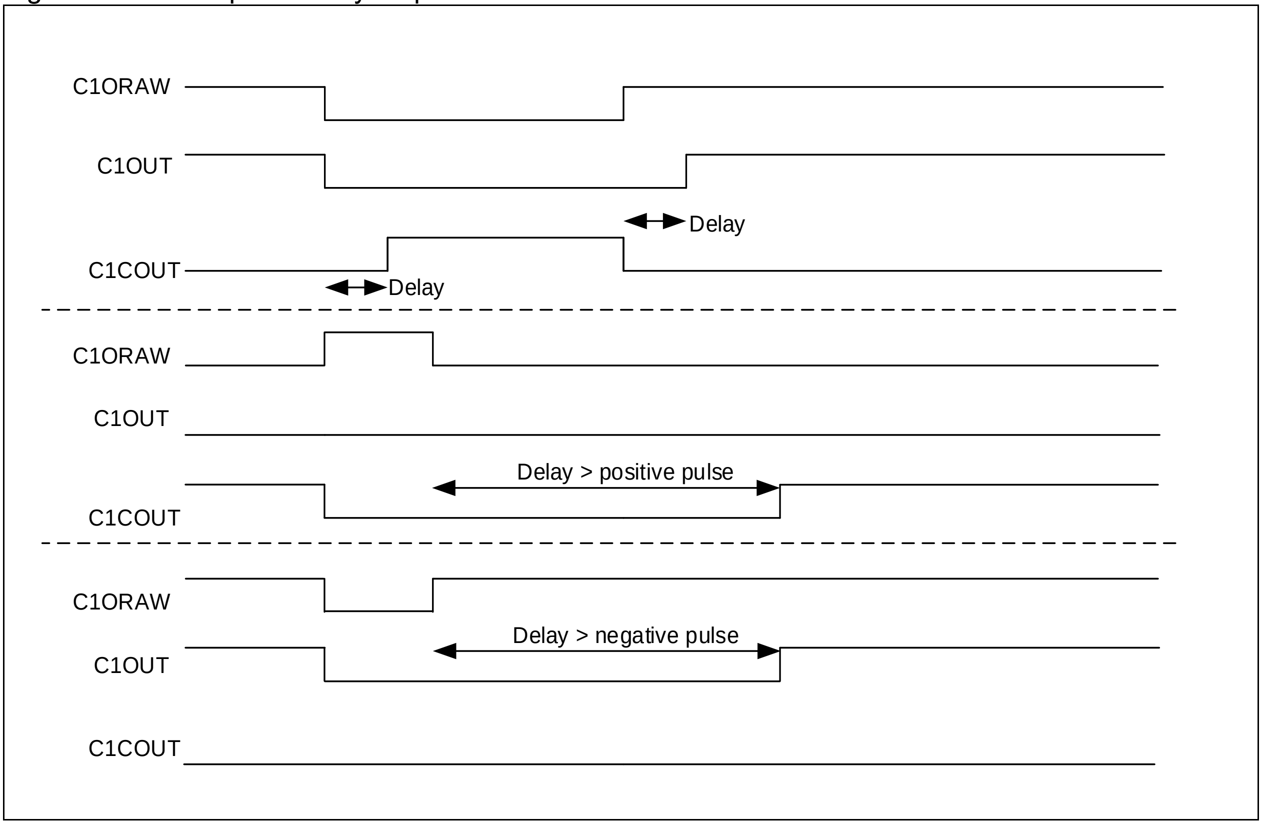
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

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 14-82 gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-82 Complementary output with dead-time insertion



14.4.3.5 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 14-15 for more details.

The break source can be a break input pin or a clock failure event. The polarity is controlled by 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.

  Note: Because of synchronization logic on OEN bit, 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 14-83 TMR output control

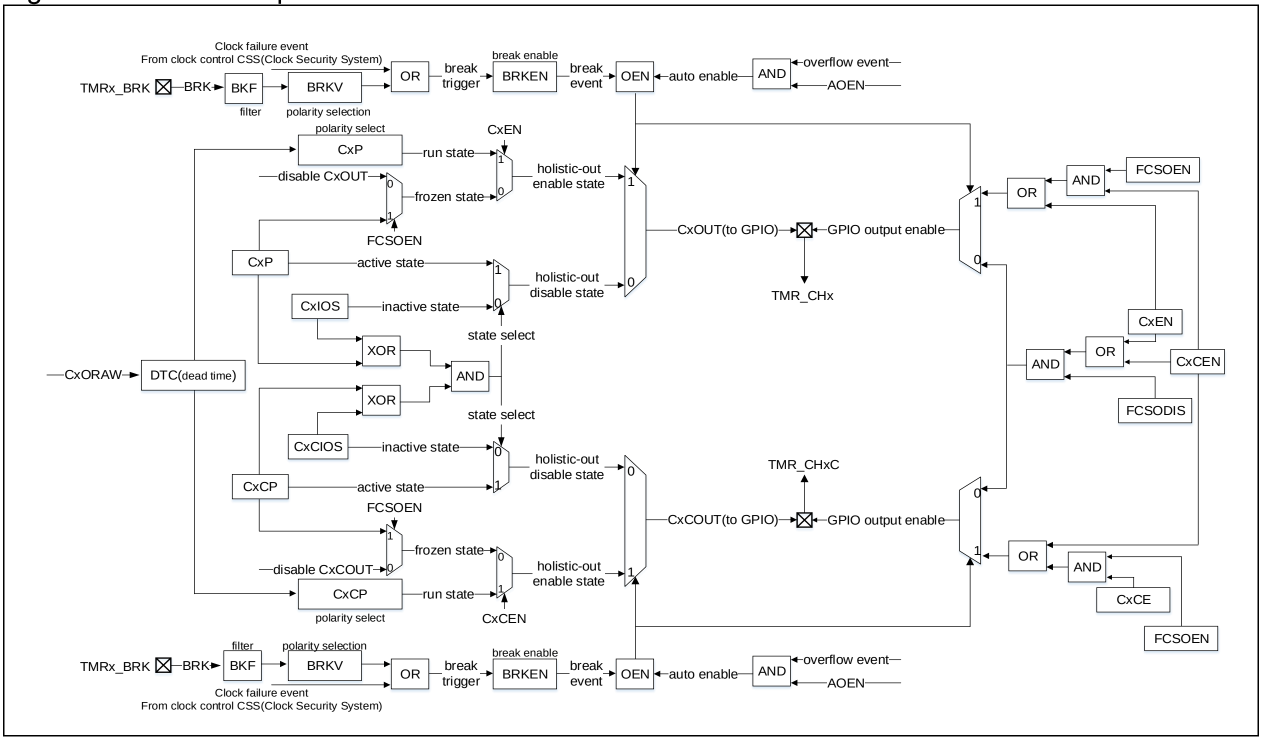
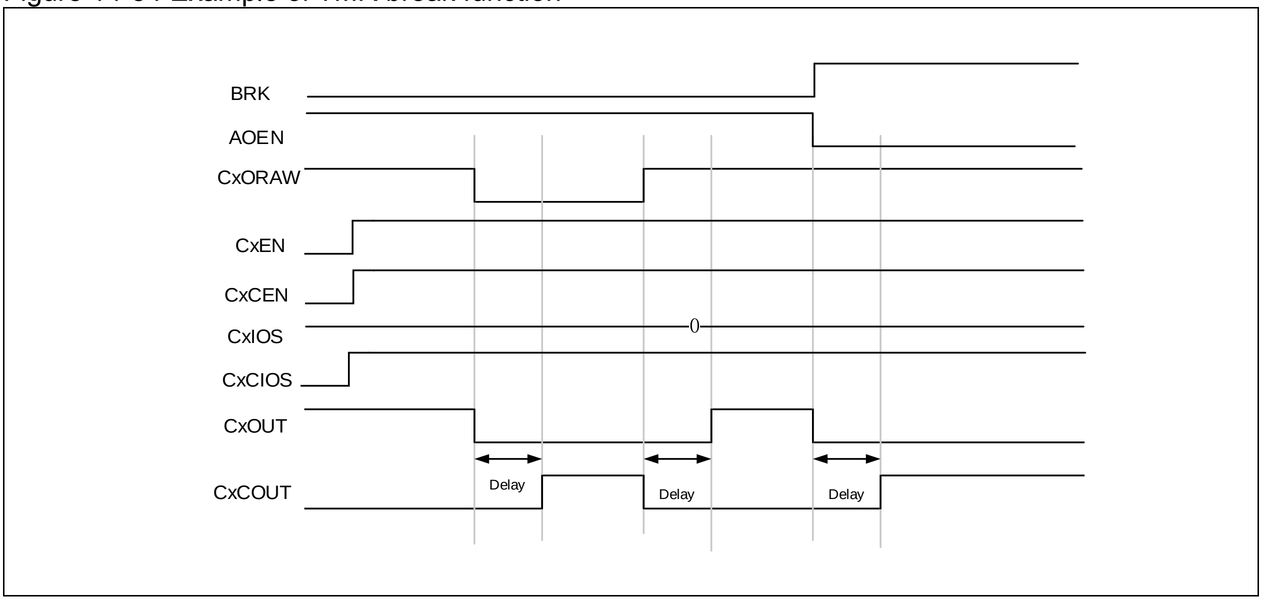


Figure 14-84 Example of TMR break function



14.4.3.6 Debug mode

When the microcontroller enters debug mode (Cortex®-M4F core halted), the TMRx counter stops counting by setting the TMRx_PAUSE in the DEBUG module.

14.4.4 TMRx registers

Table 14-10 TMR10/11/13/14 register map and reset value

Register	Offset	Reset value
TMRx_CTRL1	0x00	0x0000
TMRx_CTRL2	0x04	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_RPR	0x30	0x0000
TMRx_C1DT	0x34	0x0000
TMRx_BRK	0x44	0x0000
TMRx_DMACTRL	0x48	0x0000
TMRx_DMADT	0x4C	0x0000

14.4.4.1 TMRx control register1 (TMRx_CTRL1) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at 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}). 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 mode 1, count up and down alternately, the CxIF bit is set only when the counter counts down 10: Two-way counting mode 2, count up and down alternately, the CxIF bit is set only when the counter counts up 11: Two-way counting mode 3, 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: Enabled 1: Disabled

14.4.4.2 TMRx control register 2 (TMRx_CTRL2) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at default value.
Bit 9	C1CIOS	0x0	rw	Channel 1 complementary idle output state Output disabled (OEN = 0), after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state Output disabled (OEN = 0), after dead-time: 0: C1OUT=0 1: C1OUT=1
Bit 7:4	Reserved	0x0	resd	Kept at default value.
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 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.

14.4.4.3 TMRx DMA/interrupt enable register (TMRx_IDEN) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15:10	Reserved	0x00	resd	Kept at default value.
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	Reserved	0x0	resd	Kept at default value.
Bit 5	HALLIEN	0x0	rw	HALL interrupt enable 0: Disabled 1: Enabled
Bit 4: 2	Reserved	0x0	resd	Kept at 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

14.4.4.4 TMRx interrupt status register (TMRx_ISTS) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at 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	Reserved	0x0	resd	Kept at 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	Reserved	0x0	resd	Kept at default value.
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 occurred. 1: Hall event is detected. HALL even: CxEN, CxCEN and CxOCTRL are updated.
Bit 4: 2	Reserved	0x0	resd	Kept at 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 occurred 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 occurred 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 occurred 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.

14.4.4.5 TMRx software event register (TMRx_SWEVT) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at 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	Reserved	0x0	resd	Kept at default value.
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 with complementary output.
Bit 4: 2	Reserved	0x0	resd	Kept at 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.

14.4.4.6 TMRx channel mode register1 (TMRx_CM1) (x=10/11/13/14)

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 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_CVA, else low; —OWCDIR=1, C1ORAW is low once TMRx_ C1DT <TMRx_CVA, 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	Register	Reset value	Type	Description
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 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 C1IRAW 10: Input, C1IN is mapped on C2IRAW 11: Input, C1IN is mapped on STCI. This mode works only when the internal trigger input is selected by STIS.

14.4.4.7 TMRx Channel control register (TMRx_CCTRL) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 4	Reserved	0x000	resd	Kept at default value.
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 in output mode: 0: C1OUT is active high 1: C1OUT is active low

				When the channel 1 is configured in input mode: The active edge of the input signal is defined by C1CP/C1P. 00: C1IN active edge is on its rising edge. When used as external trigger, C1IN is not inverted. 01: C1IN active edge is on its falling edge. When used as external trigger, C1IN is inverted. 10: Reserved 11: C1IN active edge is on its falling edge and rising edge. When used as external trigger, C1IN is not inverted.
Bit 0	C1EN	0x0	rw	Channel 1 enable 0: Input or output is disabled 1: Input or output is enabled

Table 14-11 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 (corresponding IO is not driven by the timer, IO floating) 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	CxEN=CxCEN=0: output disabled (corresponding IO is not driven by the timer, IO floating) In other cases: off-state (corresponding channel output	
	1		0	1		

	1		1	0	invalid level) Asynchronously: CxOUT =CxP, Cx_EN=1, CxCOUT=CxCP, CxCEN=1;
	1		1	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.

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.

14.4.4.8 TMRx counter value (TMRx_CVAL) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.4.4.9 TMRx division value (TMRx_DIV) (x=10/11/13/14)

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.

14.4.4.10 TMRx period register (TMRx_PR) (x=10/11/13/14)

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.

14.4.4.11 TMRx repetition period register (TMRx_RPR) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15: 8	RPR	0x00	rw	Kept at 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.

14.4.4.12 TMRx channel 1 data register (TMRx_C1DT) (x=10/11/13/14)

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.

14.4.4.13 TMRx break register (TMRx_BRK) (x=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 19: 16	BKF	0x0	rw	Break input filter This field is used to set the filter for break input. The filter number N indicates that the input edge can pass through filter only after N sampling events. 0000: $f_{SAMPLING}=f_{DTS}$ (no filter) 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: $f_{SAMPLING}$= 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 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_STOP: DTC, STPEN, STPV and HOAEN 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_STOP: 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.

14.4.4.14 TMRX DMA control register (TMRX_DMACTRL) (X=10/11/13/14)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at 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 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 ......

14.4.4.15 TMRx DMA data register (TMRx_DMADT) (X=10/11/13/14)

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

14.4.4.16 TMR14 channel input remap register (TMRx_RMP)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at default value.
Bit 7: 6	TMR14_CH1_IRMP	0x0	rw	TMR14 channel 1 input remap 00: TMR14 channel 1 input is connected to GPIO 01: ERTC_CLK 10: Divided ERTC 32 is used as HEXT 11: CLK_OUT
Bit 5: 0	Reserved	0x00	resd	Kept at default value.

14.5 Advanced-control timers (TMR1)

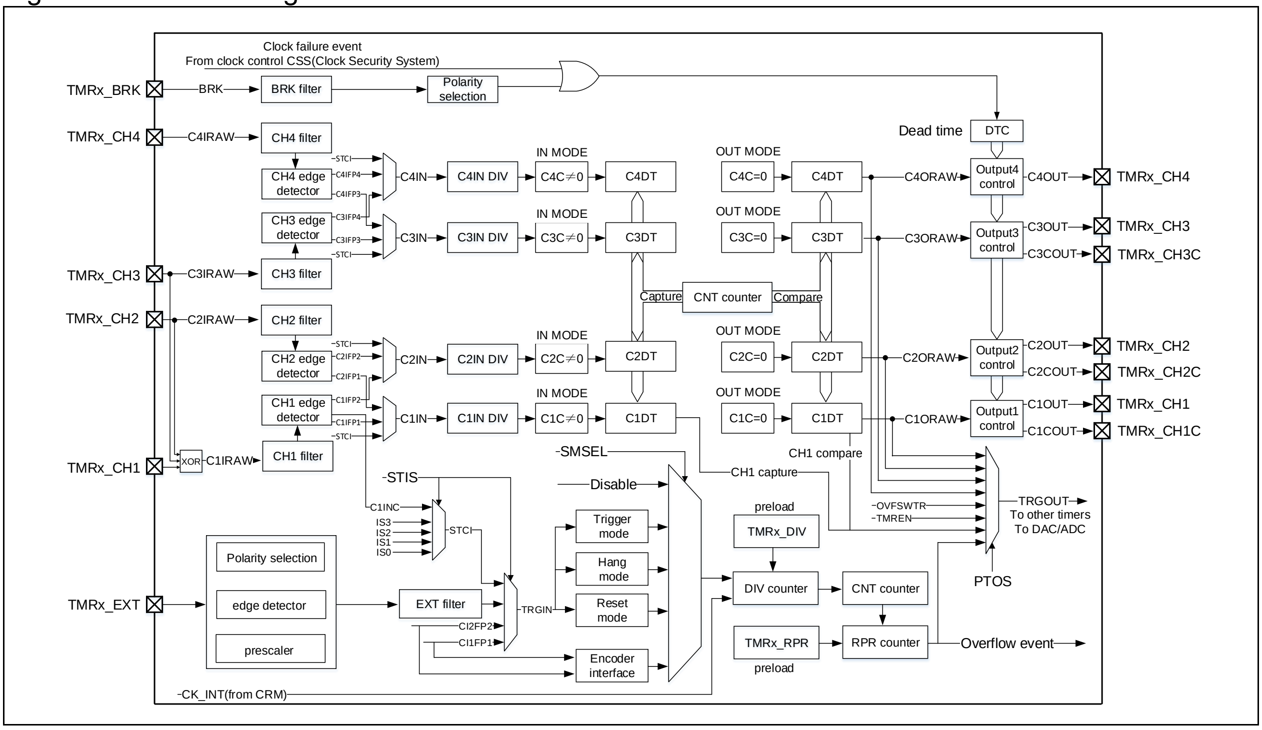
14.5.1 TMR1 introduction

The advanced-control timer TMR1 consists of a 16-bit counter supporting up and down counting modes, four channel registers, and four independent channels. It can be used for dead-time insertion, input capture and programmable PWM output.

14.5.2 TMR1 main features

- Clock 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 dead-time insertion
- Three independent channels for complementary output
- TMR break function
- Synchronization control between master and slave timers
- Interrupt/DMA generation at overflow event, trigger event, and channel events
- Support TMR burst DMA transfer

Figure 14-85 Block diagram of advanced-control timer

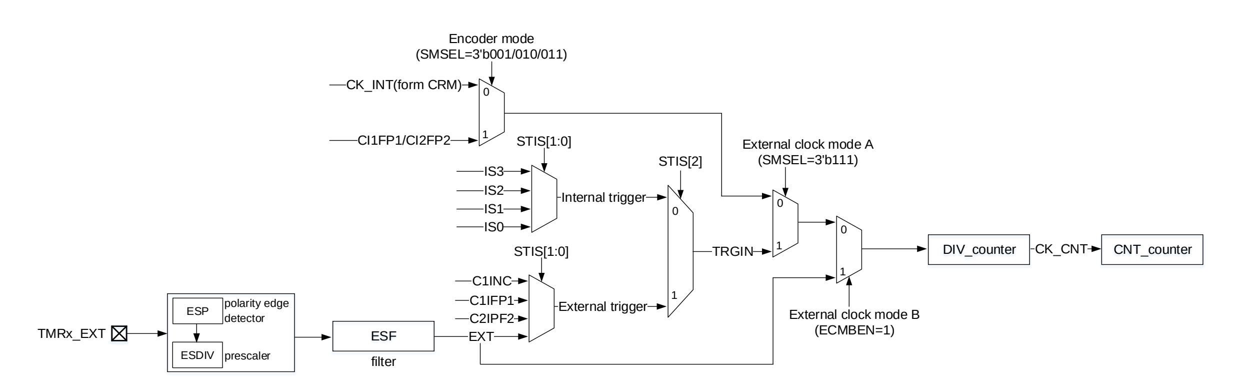


14.5.3 TMR1 functional overview

14.5.3.1 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 14-86 Counting clock



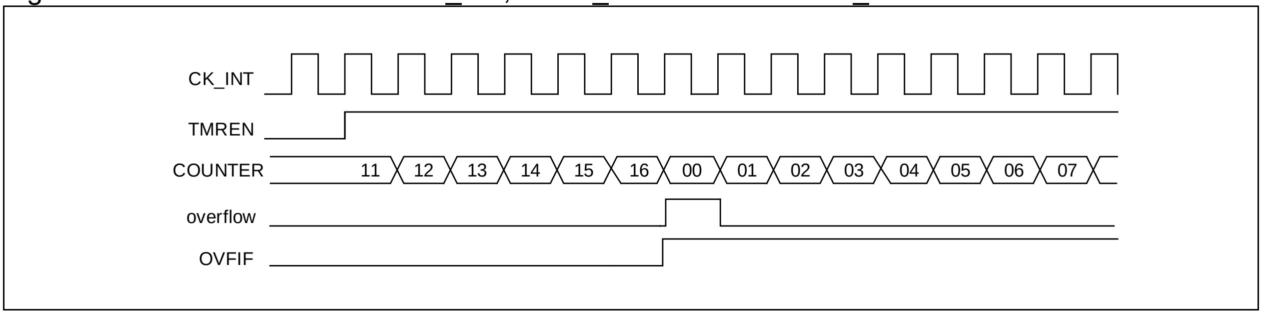
Internal clock (CK_INT)

By default, the CK_INT which is 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.

Follow the configuration steps below:

- Select a counting mode by setting the TWCMSEL[1:0] in TMRx_CTRL1 register. If a unidirectional aligned counting mode is selected, it is necessary to select a counting direction through the OWCDIR in TMRx_CTRL1 register.
- Set counting frequency through TMRx_DIV register
- Set counting cycles through TMRx_PR register
- Enable a counter by setting the TMREN bit in the TMRx_CTRL1 register

Figure 14-87 Control circuit with CK_INT, 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. By setting the STIS[2: 0] bit, select an external clock source TRGIN signal 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, a signal after channel 1 filter and polarity selection)

C2IFP2 (STIS=3'b110, a signal after channel 2 filter and polarity selection)

EXT (STIS=3'b111, external input signal after polarity selection, frequency division and filter).

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, and the EXT signal is used 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);

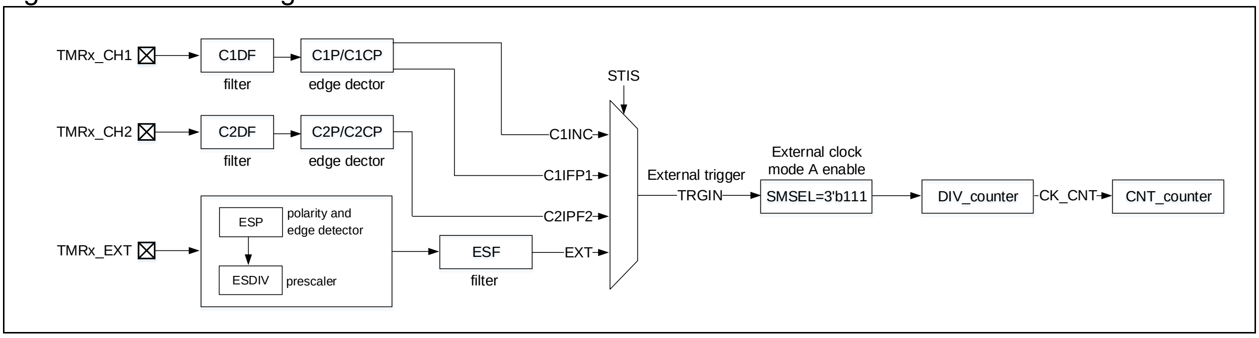
If the TMRx_EXT is used as a source of TRGIN, it is necessary to configure the external signal polarity (ESP in TMRx_STCTRL register), external signal frequency division (ESDIV[1:0] in TMRx_STCTRL) and external signal filter (ESF[3:0] in TMRx_STCTRL register).

- Set TRGIN signal source through 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 14-88 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 14-89 Counting in external clock mode A, PR=0x32 and DIV=0x0

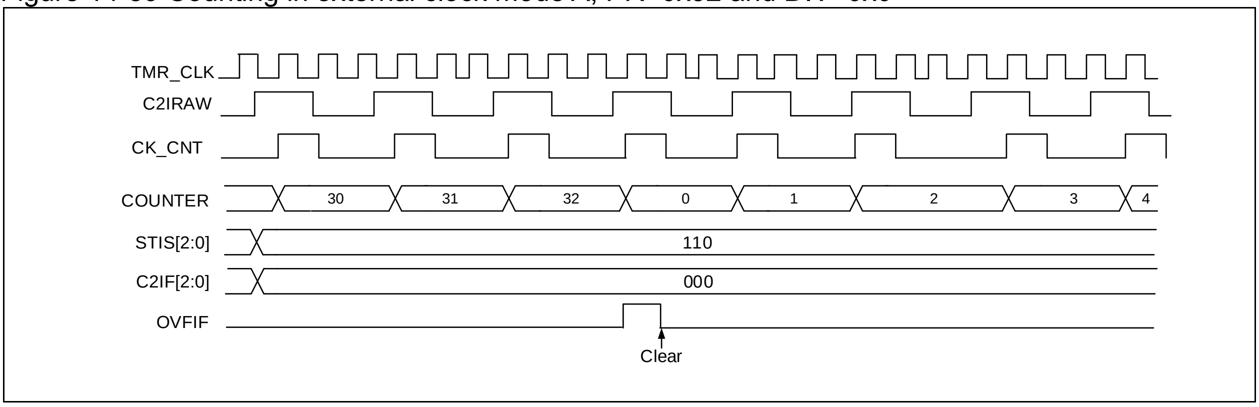
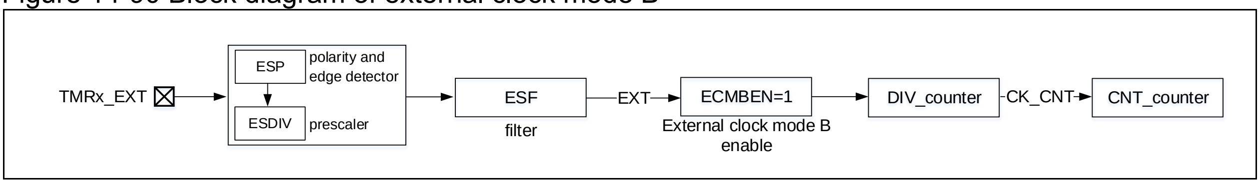
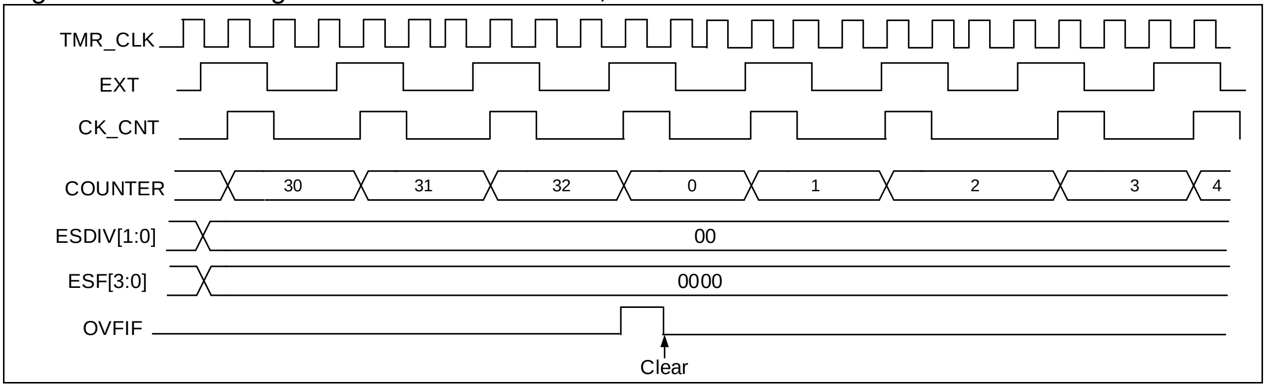


Figure 14-90 Block diagram of external clock mode B



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 14-91 Counting in external clock mode B, PR=0x32 and 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.

The advanced-control 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 TMR1_DIV register. The prescaler value can be modified at any time, but it takes effect only when the next overflow event occurs.

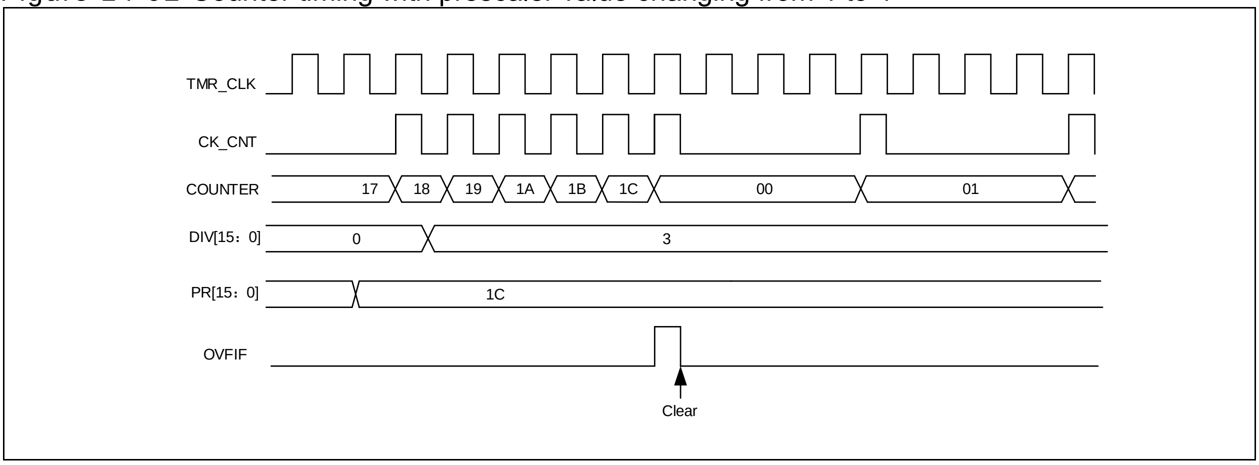
Below is the configuration procedure for internal trigger input:

- Set counting cycles through TMRx_PR register
- Set counting frequency through TMRx_DIV register
- Set counting modes through the TWCMSEL[1:0] in TMRx_CTRL1 register
- Select internal trigger by setting STIS[2:0]= 3'b000~3'b011 in TMRx_STCTRL register
- Select external clock mode A by setting SMSEL[2:0]=3'b111 in TMRx_STCTRL register
- Enable TMRx to start counting through the TMREN in TMRx_CTRL1 register

Table 14-12 TMRx internal trigger connection

Slave timer	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR1	TMR9	TMR2	TMR3	TMR4

Figure 14-92 Counter timing with prescaler value changing from 1 to 4



14.5.3.2 Counting mode

The advanced-control timer consists of a 16-bit counter supporting up, down, up/down counting modes.

The TMRx_PR register is used to define counting period of counter. 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.

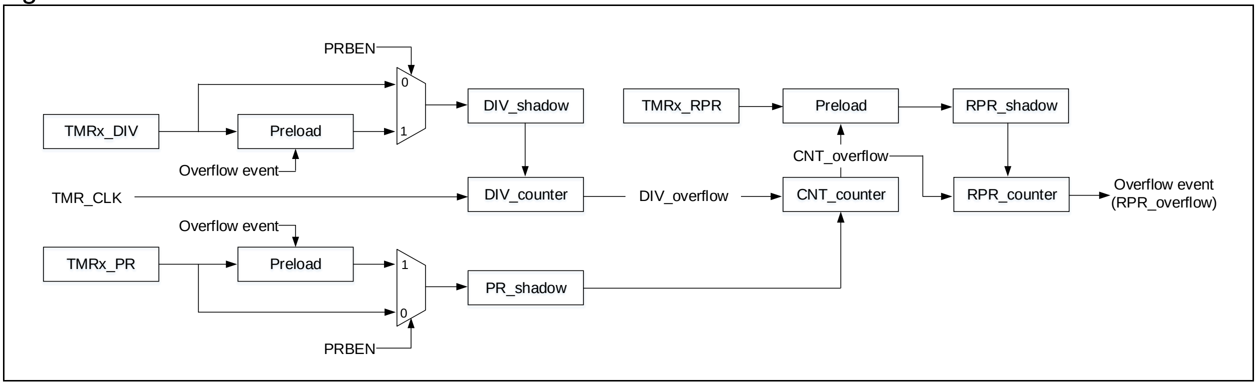
TMRx_DIV register is used to define the counter frequency of the counter. The counter counts once every DIV[15:0]+1 clock cycle. Similar to TMRx_PR register, after enabling periodic buffer, the value of the TMRx_DIV register are transferred into the shadow register at each overflow event.

Reading the TMRx_CNT register returns the current counter value. Writing the TMRx_CNT register will update the current counter value.

An overflow event is enabled by default. It can be disabled by setting OVFEN=1 in the TMRx_CTRL1 register. The OVFS bit in the TMRx_CTRL1 register is used to select the source of an overflow event, which is, by default, counter overflow or underflow, setting OVFSWTR, reset signal generated by slave mode timer controller in reset mode. Once the OVFS is set, an overflow event is generated only when overflow or underflow occurs.

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 14-93 Basic structure of a counter



Upcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b0 in the TMRx_CTRL1 register.

In upcounting mode, the counter counts from 0 to the value programmed in the TMR1_PR register, restarts from 0, and generates a counter overflow event, with setting OVFIF bit to 1. If the overflow event is disabled, the counter is no longer reloaded with the prescaler and re-loaded value on counter overflow, otherwise, the prescaler and re-loaded value will be updated on an overflow event.

Figure 14-94 Overflow event when PRBEN=0

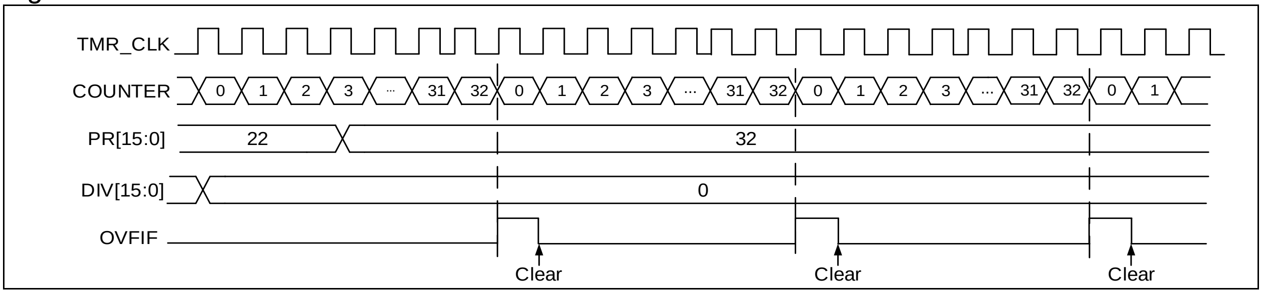
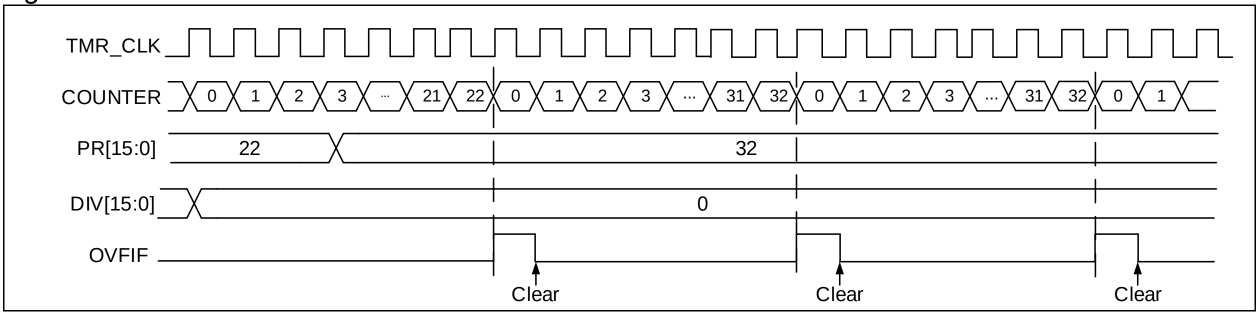


Figure 14-95 Overflow event when PRBEN=1

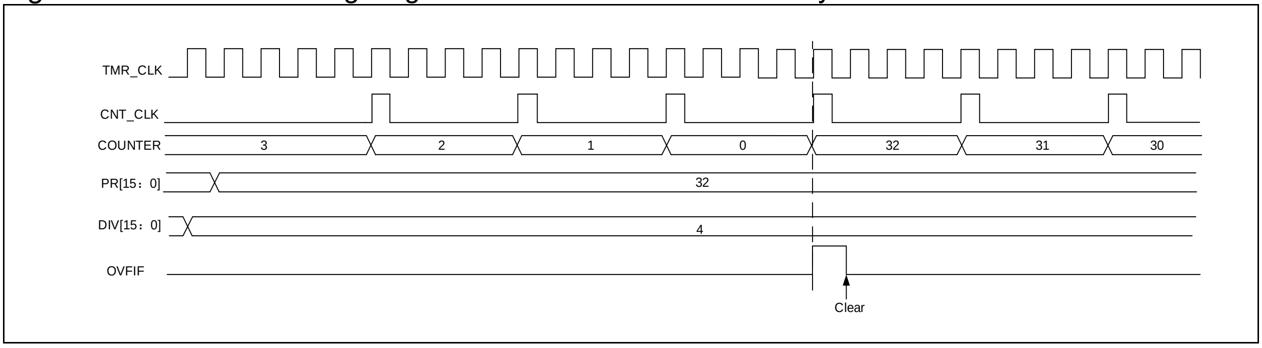


Downcounting mode

This mode is enabled by setting CMSEL[1:0]=2'b00 and OWCDIR=1'b1 in the TMRx_CTRL1 register.

In downcounting 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 14-96 Counter timing diagram with internal clock divided by 4



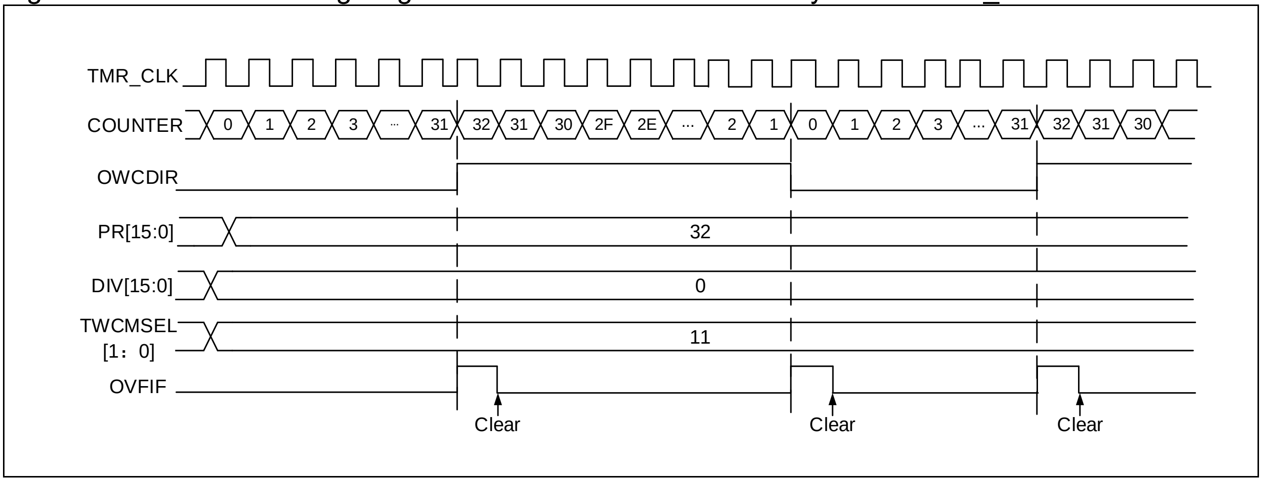
Up/down counting mode (center-aligned mode)

Up/down counting mode can be enabled by setting CMSEL[1:0]≠2'b00 in the TMRx_CTRL1 register. In up/down counting 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 used to select the condition under which the CxIF flag is set in two-way counting mode. In other words, when TWCMSEL[1:0]=2'b01 (counting mode 1) is selected, the CxIF flag is set only when the counter counts down; when TWCMSEL[1:0]=2'b10 (counting mode 2) is selected, the CxIF flag is set only when the counter counts up; when TWCMSEL[1:0]=2'b11 (counting mode 3) is selected, the CxIF flag is set when the counter counts up and down.

Note: The OWCDIR is ready-only in up/down counting mode.

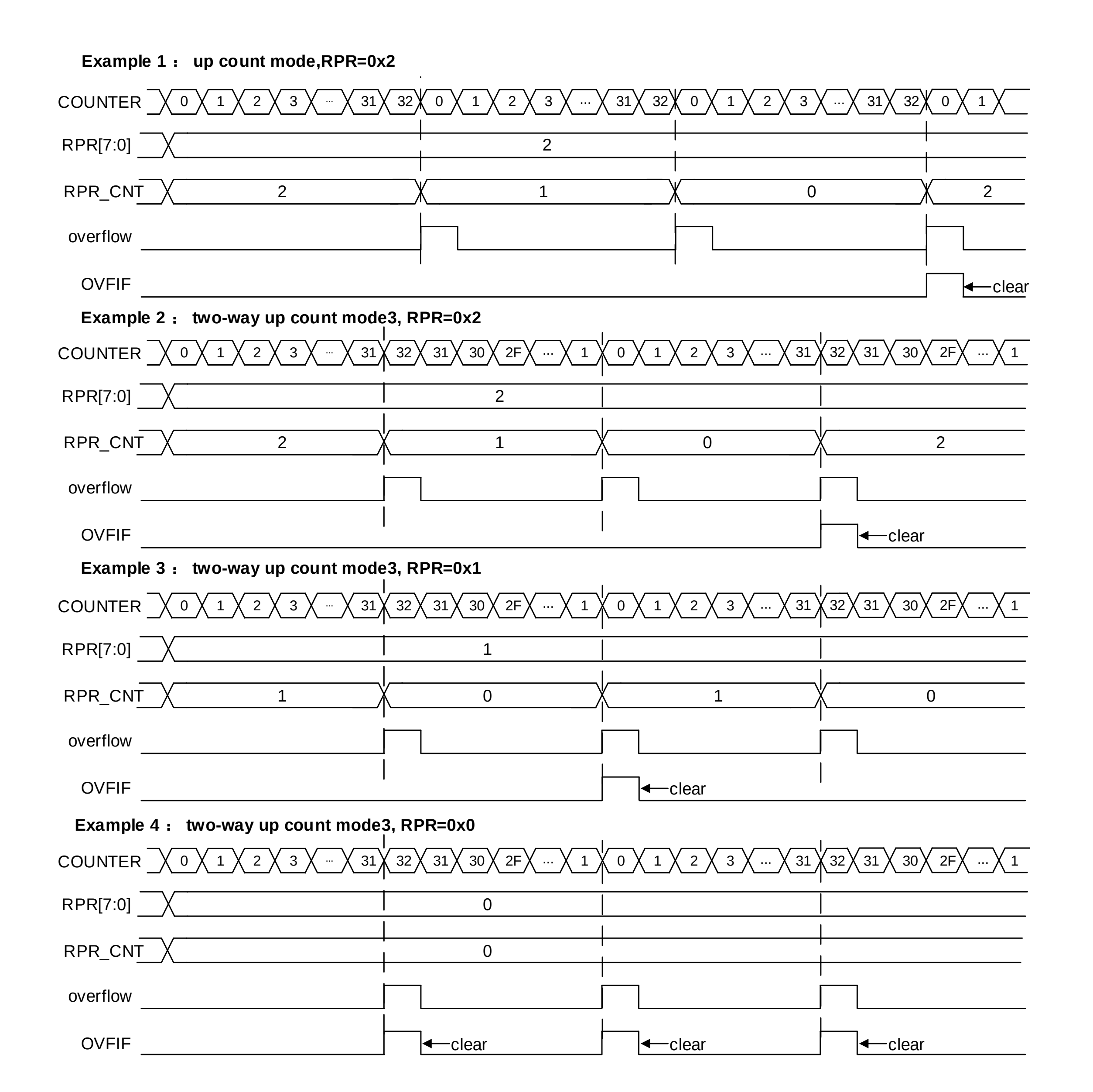
Figure 14-97 Counter timing diagram with internal clock divided by 1 and TMRx_PR=0x32



Repetition counter mode:

The TMRx_RPR register is used to set repetition counting mode. This mode is enabled when the repetition counter value is not equal to 0. In this mode, an overflow event is generated when a counter overflow occurs (RPR[7:0]+1). The repetition counter is decremented at each counter overflow. An overflow event is generated when the repetition counter reaches 0. The frequency of the overflow event can be adjusted by setting the repetition counter value.

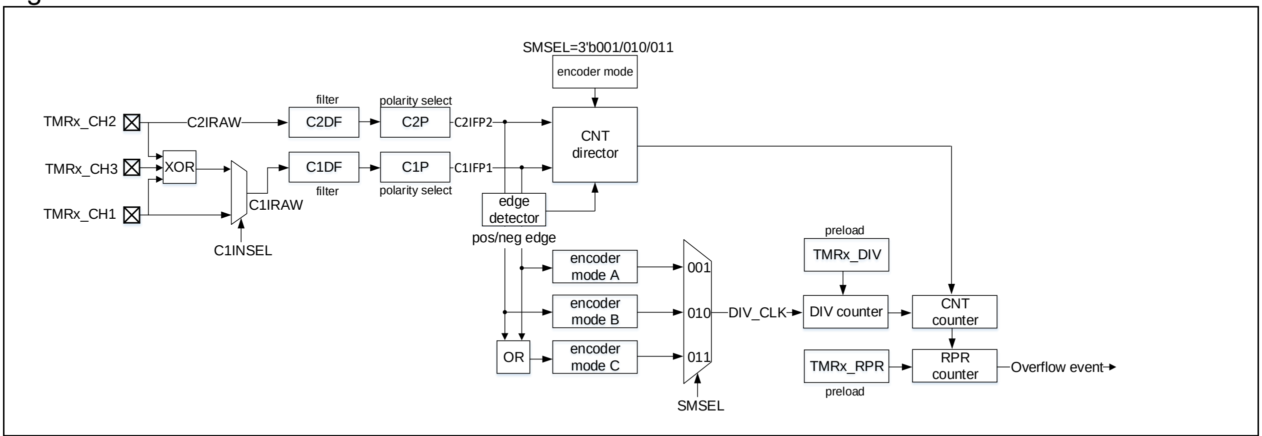
Figure 14-98 OVFIF behavior in upcounting mode and center-aligned mode



Encoder interface mode

In this mode, the two inputs (TMRx_CH1/ TMRx_CH2) are required. Depending on the level on one input, the counter counts up or down on the edge of the other input. The OWCDIR bit indicates the direction of the counter, as shown in the figure below:

Figure 14-99 Structure of encoder mode



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.

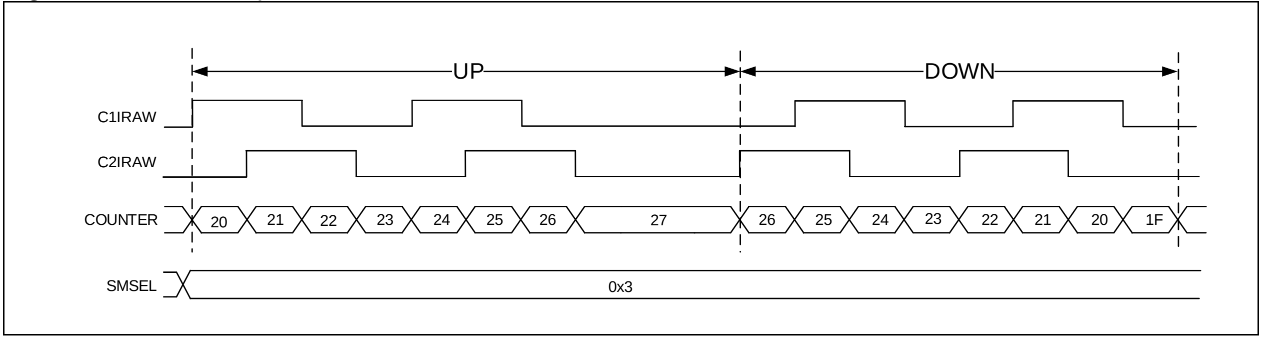
To use encoder mode, follow the procedures below:

- 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 14-13 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 14-100 Example of encoder interface mode C



14.5.3.3 TMR input function

The TMR1 has four independent channels. Each channel can be configured as input or output. As input, each channel input signal is processed as follows:

- TMRx_CHx outputs the pre-processed CxIRAW. The C1INSE bit is used to select TMRx_CHx, or the XOR-ed TMRx_CH1, TMRx_CH2 and TMRx_CH3 as the source of C1IRAW.
  The sources of C2IRAW, C3IRAW and C4IRAW are TMRx_CH2, TMRx_CH3, TMRx_CH4, respectively.
- CxIRAW goes through digital filter and generates the filtered CxIF signal. The digital filter uses the CxDF bit to program sampling frequency and sampling times.
- CxIF goes through edge detector, and generates 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 goes through capture signal selector, and generates the CxIN signal after capture signal 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, C1IFP2 is the C1IFP1 signal that is from channel 1 and processed by channel 2 edge detector). The STCI comes from slave timer controller, and its source is selected by STIS bit.
- CxIN goes through input divider and generates the CxIPS signal. The divider factor can be defined as No division, /2, /4 or /8, by the CxIDIV bit.

Figure 14-101 Input/output channel 1 main circuit

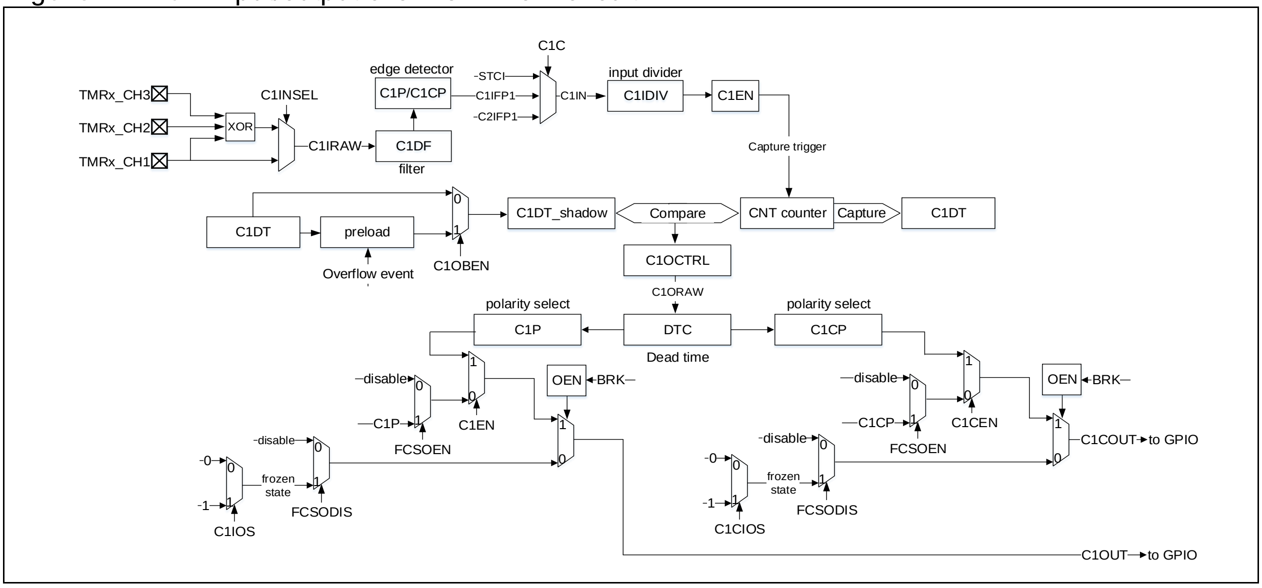
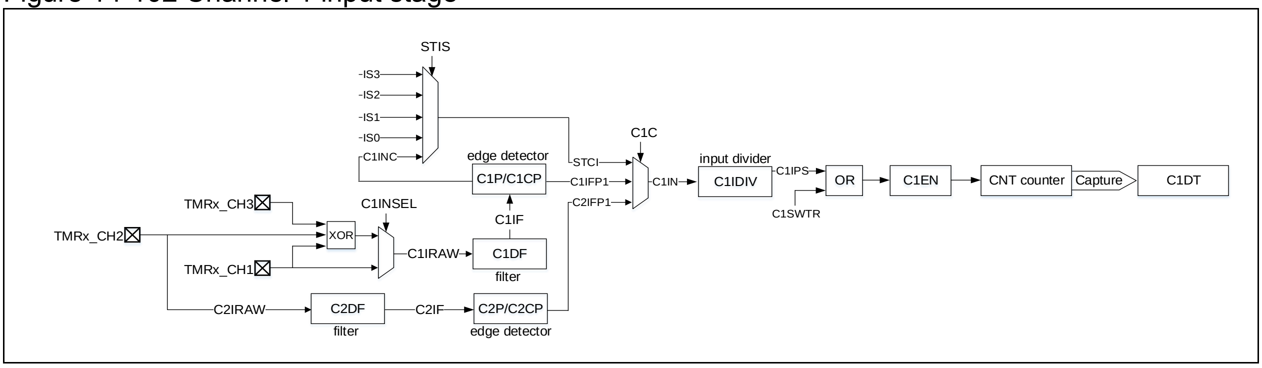


Figure 14-102 Channel 1 input stage



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/DMA request will be generated if the CxIEN bit and CxDEN bit are enabled. If the selected trigger signal is detected when the CxIF is set to 1, a capture overflow event is generated. The previous counter value will be overwritten with the current counter value, and the CxRF is set to 1.

To capture the rising edge of C1IN input, following the procedure below:

- Set C1C=01 in the TMRx_CM1 register to select the C1IN as channel 1 input
- Set C1IN signal filter bandwidth (CxDF[3: 0])
- Set the active edge on the C1IN channel by writing C1P=0 (rising edge) in the TMR1_CCTRL register
- Program C1IN signal capture frequency divider (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 (TMR1_CH1, TMR1_CH2 and TMR1_CH3) are connected to the channel 1 (selected by setting the C1INSE in the TMR1_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

PWM input mode is applied to channel 1 and 2. To use this mode, both C1IN and C2IN are mapped on the same TMRx_CHx, and the CxIFPx of either channel 1 or channel 2 must be configured as trigger input and slave mode controller is configured in reset mode.

The PWM input mode can be used to measure the period and duty cycle of the PWM input signal. For example, the user can measure the period and duty cycle of the PWM applied on channel 1 using the following procedures:

- Set C1C=2‘b01: select C1IN for C1IFP1
- Set C1P=1’b0, select C1IFP1 rising edge active
- Set C2C=2‘b10, select C2IN for C1IFP2
- Set C2P=1’b1, select C1IFP2 falling edge active
- Set STIS=3’b101, select the slave mode timer trigger signal as C1IFP1
- Set SMSEL=3‘b100: configure the slave mode controller in reset mode
- Set C1EN=1’b1 and C2EN=1’b1. Enable channel 1 and input capture

After above configuration, the rising edge of channel 1 input signal will trigger the capture and stores the capture value into C1DT register, and it will reset the counter at the same time. The falling edge of the channel 1 input signal triggers the capture and stores the capture value into C2DT register. The period of the channel 1 input signal is calculated through C1DT, and its duty cycle through C2DT.

Figure 14-103 PWM input mode configuration example

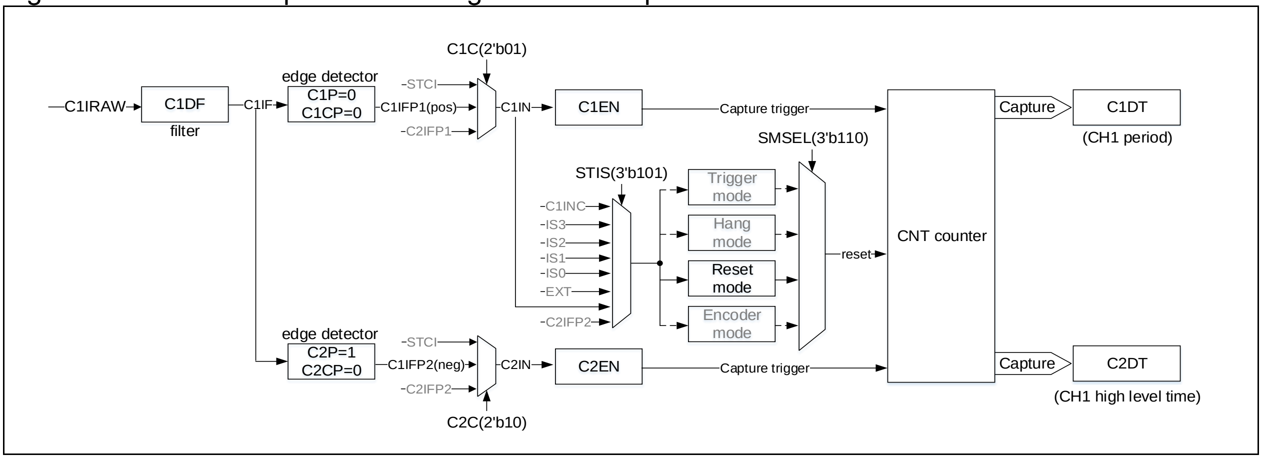
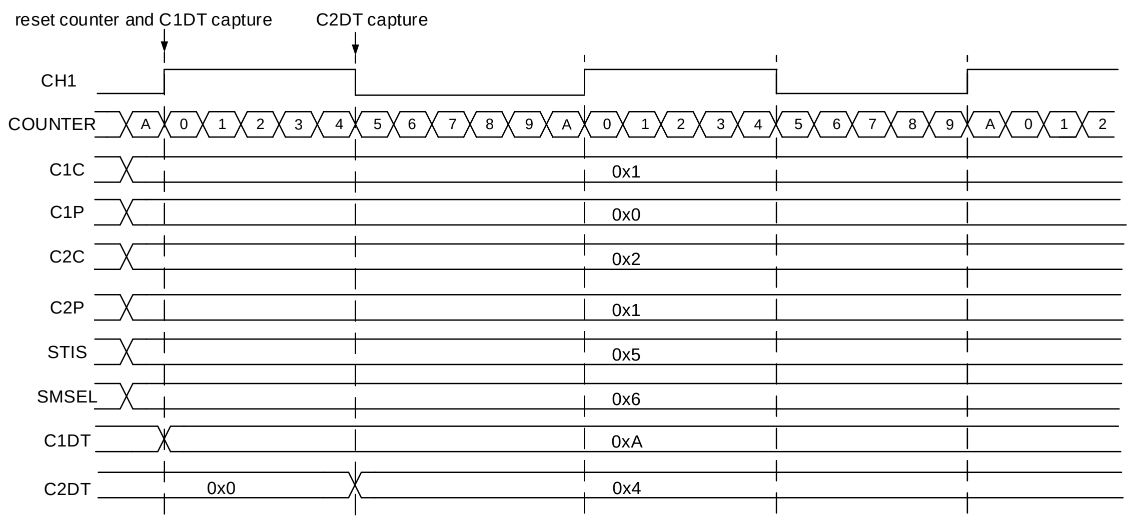


Figure 14-104 PWM input mode



14.5.3.4 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. The advanced-control timer output function varies from one channel to one channel.

Figure 14-105 Channel output stage (channel 1 to 3)

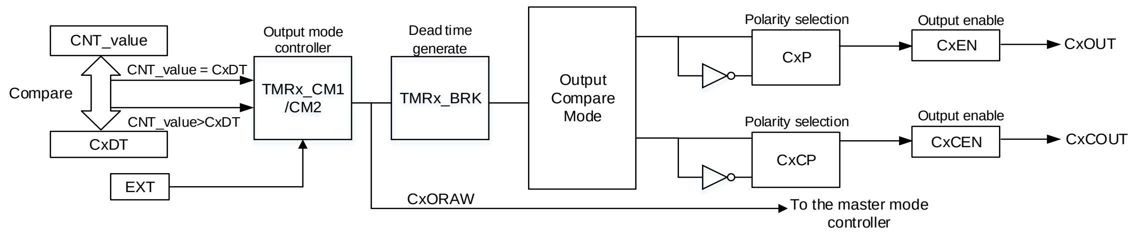
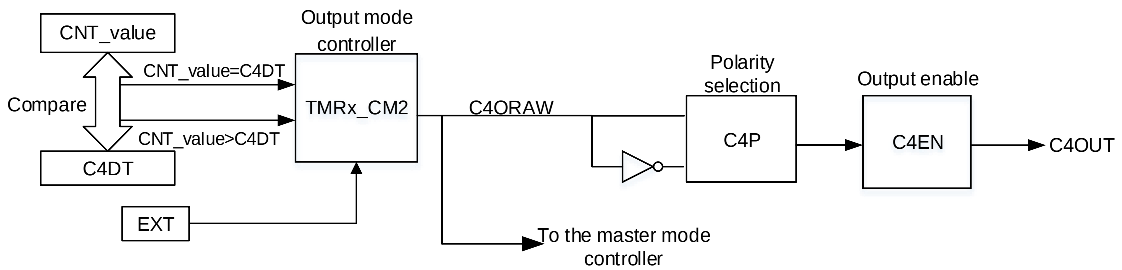


Figure 14-106 Channel 4 output stage



Output mode

Write CxC[1: 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:

Enable PWM mode A by setting CxOCTRL=3'b110. In upcounting mode, C1ORAW outputs high

when TMRx_C1DT>TMRx_CVAL, otherwise, it is low; In downcounting mode, C1ORAW outputs low when TMRx_C1DT<TMRx_CVAL, otherwise, it is high.
To use PWM mode A, the following procedures are recommended:

- Set PWM periods through TMRx_PR register
- Set PWM duty cycles through TMRx_CxD
- Select PWM mode A by setting CxOCTRL=3'b110 in the TMRx_CM1/CM2 register
- Set counting frequency through TMRx_DIV register
- Select counting mode by setting the TWCMSEL[1:0] bit in the TMRx_CTRL1 register
- Select output polarity through the CxP and CxCP bits in the TMRx_CCTRL register
- Enable channel output through the CxEN and CxCEN bits in the TMRx_CCTRL register
- Enable TMRx output through the OEN bit in the TMRx_BRK register
- Configure GPIOs corresponding to TMR output channels as multiplexed mode
- Enable TMRx to start counting through the TMREN bit in the TMRx_CTRL1 register.

PWM mode B:
Enable PWM mode B by setting CxOCTRL=3'b111. In upcounting mode, C1ORAW outputs low when TMRx_C1DT>TMRx_CVAL, otherwise, it is high; In downcounting mode, C1ORAW outputs high when TMRx_C1DT<TMRx_CVAL, otherwise, it is low.

Forced output mode:
Enable forced output mode by setting CxOCTRL=3'b100/101. In this case, the CxORAW is forced to be the programmed level, regardless of the counter value. Despite this, the channel flag bit and DMA request still depend on the compare result.

Output compare mode:
Enable output compare mode by setting CxOCTRL=3'b001/010/011. 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. Enable one-pulse by setting OCMEN=1. 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 in upcounting mode, the configuration must follow the rule: CVAL<CxDT≤PR; in downcounting mode, CVAL>CxDT is required.

Fast output mode:
Enable this mode by setting CxOIEN=1. If enabled, the CxORAW signal will not change when the counter value matches the CxDT, but change 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 14-107 gives an example of output compare mode (toggle) with C1DT=0x3. When the counter value is equal to 0x3, C1OUT toggles.

Figure 14-108 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 14-109 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 14-110 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 14-107 C1ORAW toggles when counter value matches the C1DT value

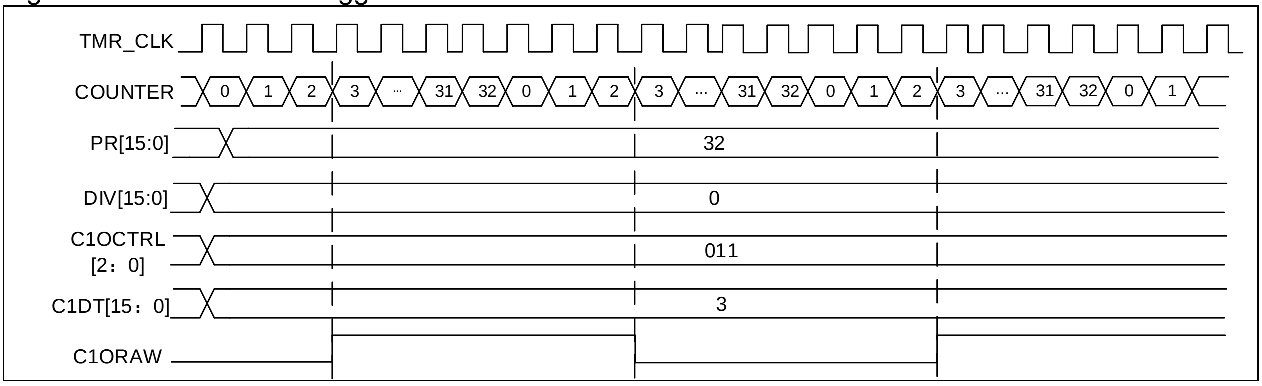


Figure 14-108 Upcounting mode and PWM mode A

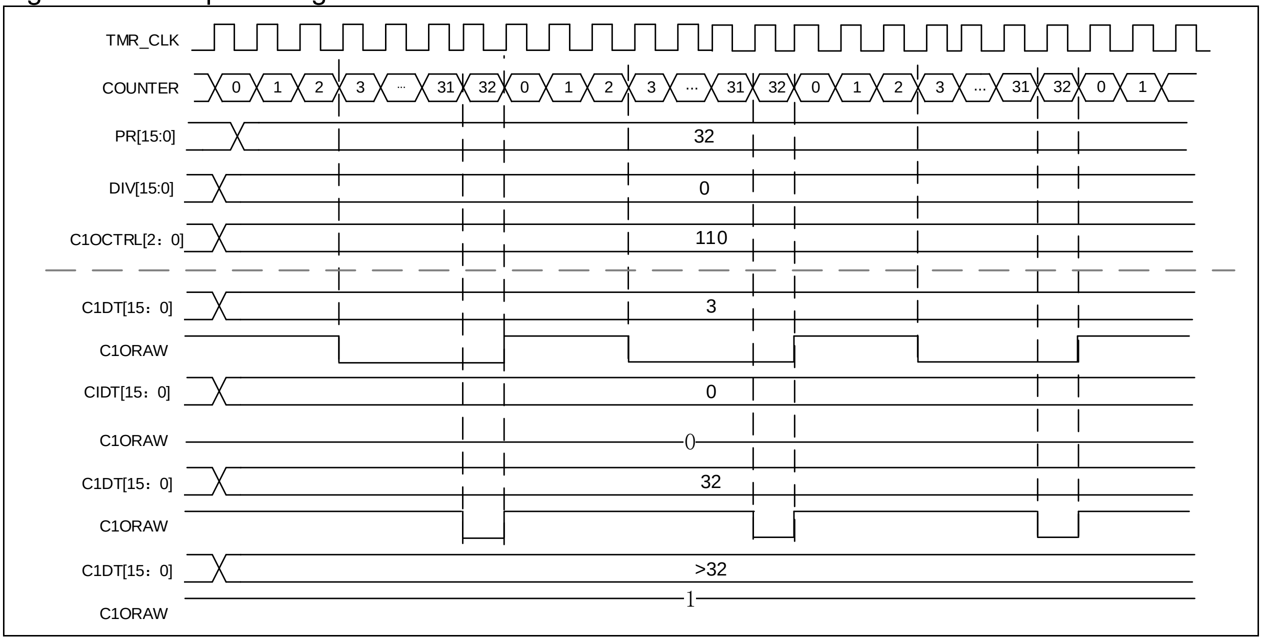


Figure 14-109 Up/down counting mode and PWM mode

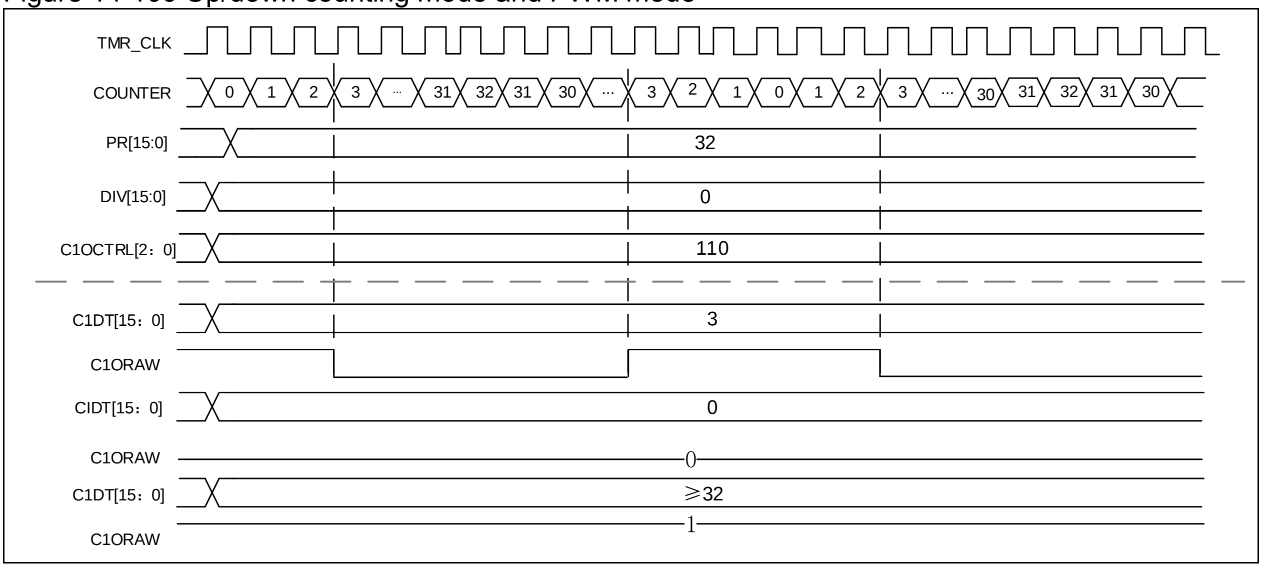
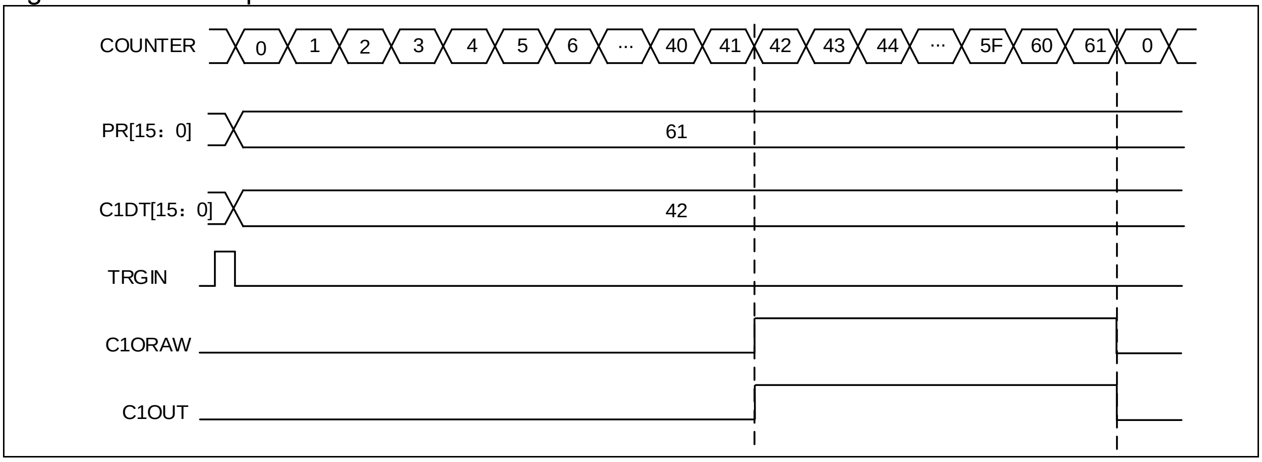


Figure 14-110 One-pulse mode



Master mode timer event output

When TMR is used as a master timer, one of the following source of signals can be selected as TRGOUT output to a slave mode timer. This is done by setting the PTOS bit in the TMRxCTRL2 register.

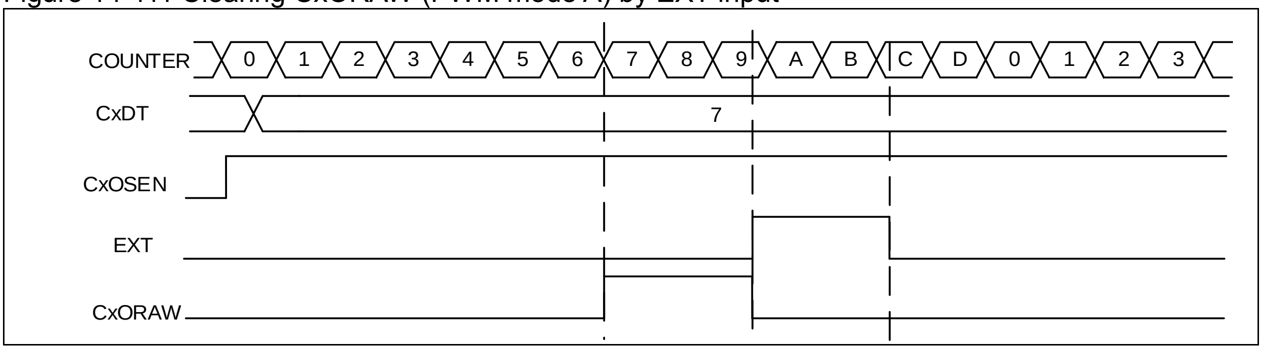
– PTOS=3'b000, TRGOUT output software overflow event (OVFSWTR bit in TMRx_SWEVT register)
– PTOS=3'b001, TRGOUT output counter enable
– PTOS=3'b010, TRGOUT output counter overflow event
– PTOS=3'b011, TRGOUT output capture and compare event
– PTOS=3'b100, TRGOUT output C1ORAW
– PTOS=3'b101, TRGOUT output C2ORAW
– PTOS=3'b110, TRGOUT output C3ORAW
– PTOS=3'b111, TRGOUT output C4ORAW

CxORAW clear

When the CxOSEN bit is set, 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 output mode. *Figure 14-111* shows the example of clearing CxORAW. 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 14-111 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 14-15 for more information about the output state of CxOUT and CxCOUT.

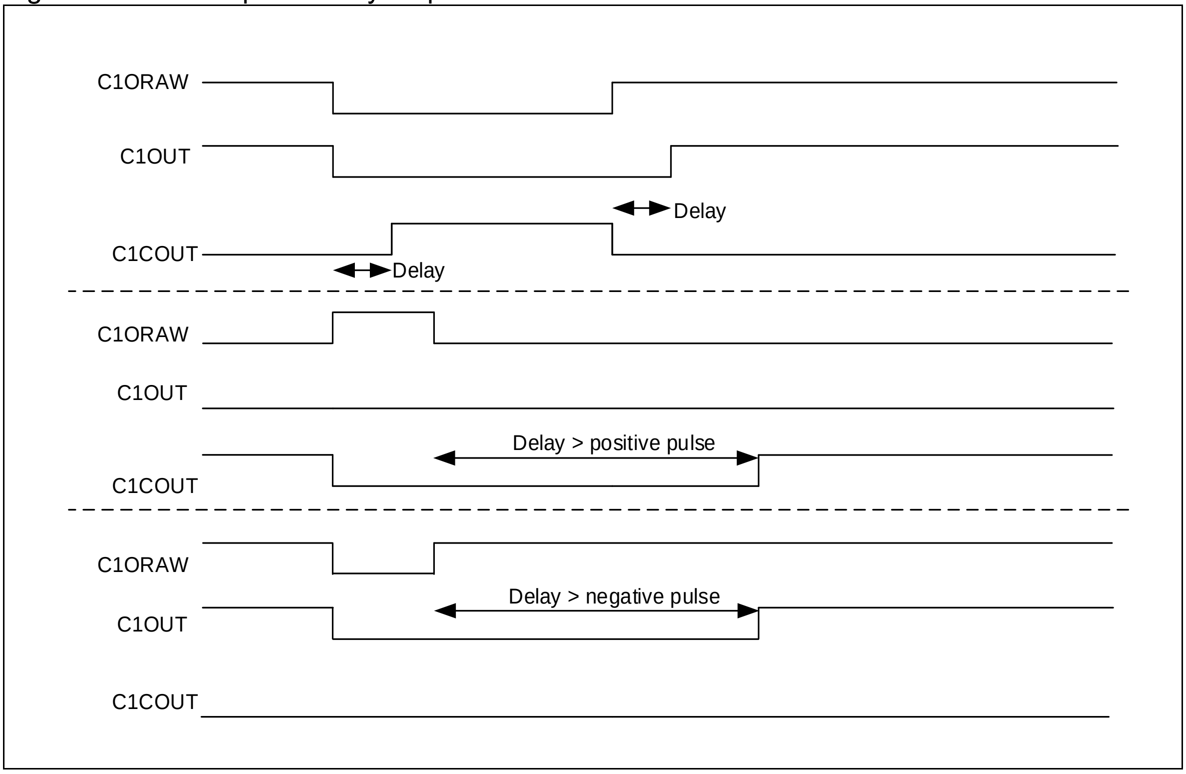
The dead-time is activated when switching to IDLEF state (OEN falling down to 0).

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, then theC1OUT and C1COUT will not generate corresponding pulses. Therefore the dead-time should be less than the width of the active output.

Figure 14-112 gives an example of dead-time insertion when CxP=0, CxCP=0, OEN=1, CxEN=1 and CxCEN=1.

Figure 14-112 Complementary output with dead-time insertion



14.5.3.5 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 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.

    Note: 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 14-113 TMR output control

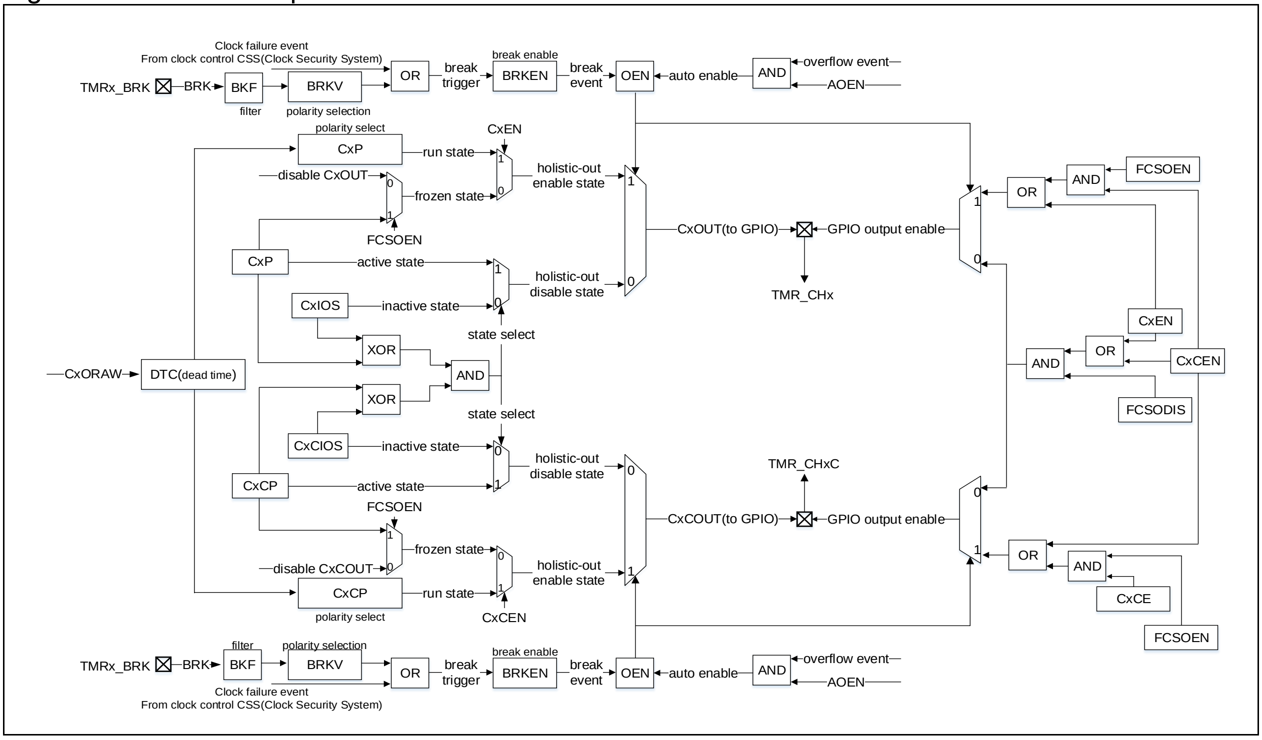
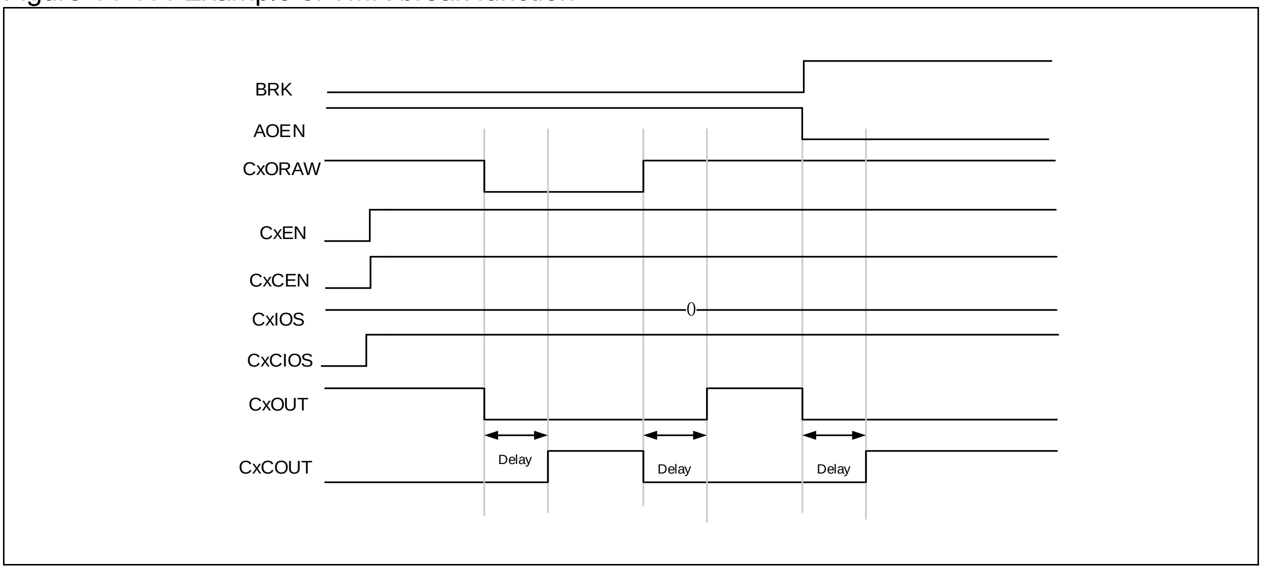


Figure 14-114 Example of TMR break function



14.5.3.6 TMR synchronization

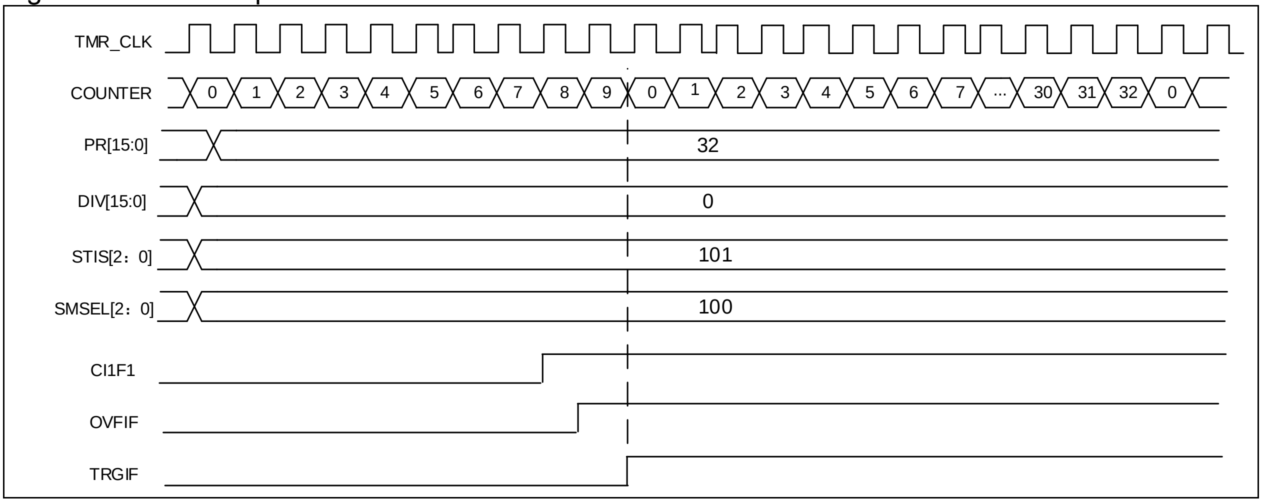
The timers are linked together internally for timer synchronization. Master timer is selected by setting the PTOS[2: 0] bit; Slave timer is selected by setting the SMSEL[2: 0] bit.

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.

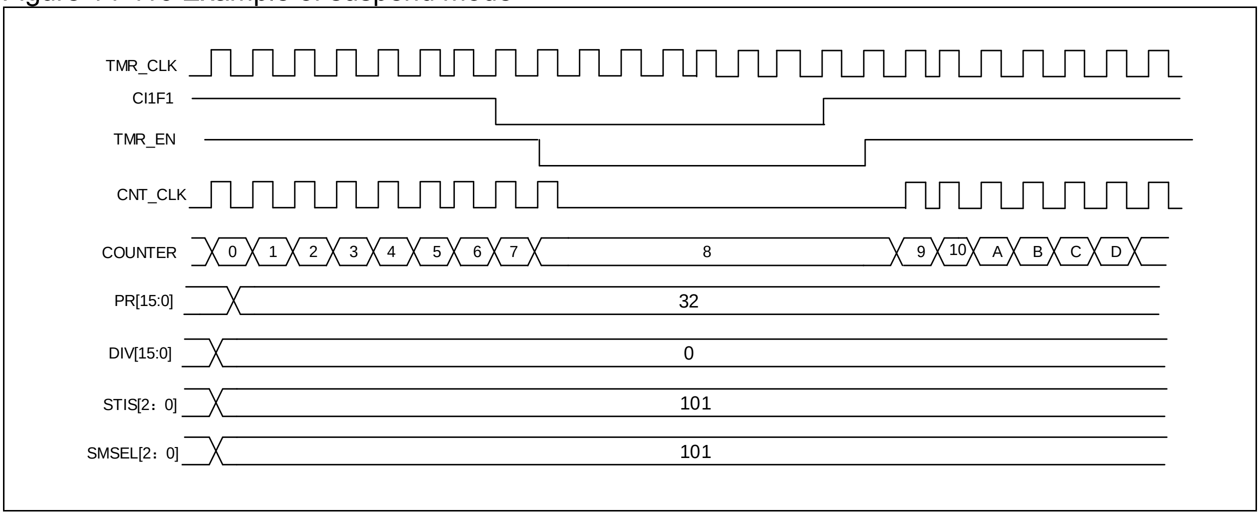
Figure 14-115 Example of reset mode



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.

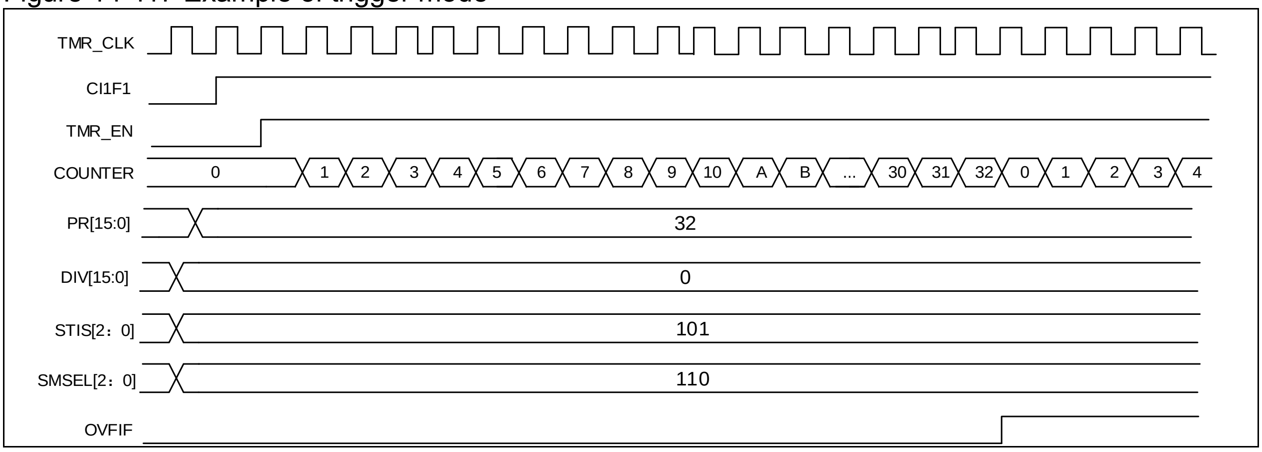
Figure 14-116 Example of suspend mode



Slave mode: Trigger mode

The counter can start counting on the rising edge of a selected trigger input (TMR_EN=1)

Figure 14-117 Example of trigger mode



For more examples on timer synchronization, please refer to section 14.2.3.5.

14.5.3.7 Debug mode

When the microcontroller enters debug mode (Cortex[TM]-M4F core halted), the TMR1 counter stops counting by setting the TMR1_PAUSE in the DEBUG module.

14.5.4 TMR1 registers

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

TMR1 registers are mapped into a 16-bit addressable space.

Table 14-14 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

14.5.4.1 TMR1 control register1 (TMR1_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 15: 10	Reserved	0x00	resd	Kept at 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 mode 1, count up and down alternately, the CxIF bit is set only when the counter counts down 10: Two-way counting mode 2, count up and down alternately, the CxIFbit is set only when the counter counts up 11: Two-way counting mode 3, 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

14.5.4.2 TMR1 control register2 (TMR1_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31	TRGOUT2EN	0x0	rw	TRGOUT2 enable 0: TRGOUT2 disabled 1: TRGOUT2 enabled
Bit 30: 15	Reserved	0x0000	resd	Kept at 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 Output disabled (OEN = 0), after dead-time: 0: C1OUTL=0 1: C1OUTL=1
Bit 8	C1IOS	0x0	rw	Channel 1 idle output state Output disabled (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 that have 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 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.

14.5.4.3 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) Please refer to Table 14-11 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.

14.5.4.4 TMR1 DMA/interrupt enable register (TMR1_IDEN)

Bit	Register	Reset value	Type	Description
Bit 15	Reserved	0x0	resd	Kept at 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

14.5.4.5 TMR1 interrupt status register (TMR1_ISTS)

Bit	Register	Reset value	Type	Description
Bit 15: 13	Reserved	0x0	resd	Kept at 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	Kept at 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.

14.5.4.6 TMR1 software event register (TMR1_SWEVT)

Bit	Register	Reset value	Type	Description
Bit 15: 8	Reserved	0x00	resd	Kept at 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.

14.5.4.7 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 TMR1_CVAL=TMR1_C1DT 010: C1ORAW is low when TMR1_CVAL=TMR1_C1DT 011: Switch C1ORAW level when TMR1_CVAL=TMR1_C1DT 100: C1ORAW is forced low 101: C1ORAW is forced high. 110: PWM mode A —OWCDIR=0, C1ORAW is high once TMR1_C1DT>TMR1_CVAL, else low; —OWCDIR=1, C1ORAW is low once TMR1_ C1DT <TMR1_CVAL, else high; 111: PWM mode B — OWCDIR=0, C1ORAW is low once TMR1_C1DT >TMR1_CVAL, else high; — OWCDIR=1, C1ORAW is high once TMR1_C1DT <TMR1_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.

14.5.4.8 TMR1 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.

14.5.4.9 TMR1 channel control register (TMR1_CCTRL)

Bit	Register	Reset value	Type	Description
Bit 15: 14	Reserved	0x0	resd	Kept at 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 14-15 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 (corresponding IO is not driven by the timer, IO floating)	
	0		0	1		

	0		1	0	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		1	1	
	1		0	0	CxEN=CxCEN=0: Output disabled (corresponding IO is not driven by the timer, IO floating) In other cases: 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.

14.5.4.10 TMR1 counter value (TMR1_CVAL)

Bit	Register	Reset value	Type	Description
Bit 15: 0	CVAL	0x0000	rw	Counter value

14.5.4.11 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.

14.5.4.12 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.

14.5.4.13 TMR1 repetition period register (TMR1_RPR)

Bit	Register	Reset value	Type	Description
Bit 15: 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.

14.5.4.14 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.

14.5.4.15 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.

14.5.4.16 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.

14.5.4.17 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.

14.5.4.18 TMR1 break register (TMR1_BRK)

Bit	Register	Reset value	Type	Description
Bit 19: 16	BKF	0x0	rw	Break input filter This field is used to set the filter for break input. The filter number N indicates that the input edge can pass through filter only after N sampling events. 0000: $f_{SAMPLING}=f_{DTS}$ (no filter) 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: $f_{SAMPLING}=f_{DTS}/4$, 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 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: TMR1_STOP: DTC, STPEN, STPV and HOAEN TMR1_CTRL2: CxIOS and CxCIOS 10: Write protection level 2. The following bits and all bits in level 3 are write protected: TMR1_CCTRL: CxP and CxCP TMR1_STOP: FCSODIS and FCSOEN 11: Write protection level 1. The following bits and all bits in level 2 are write protected: TMR1_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.

14.5.4.19 TMR1 DMA control register (TMR1_DMACTRL)

Bit	Register	Reset value	Type	Description
Bit 15:13	Reserved	0x0	resd	Kept at 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 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 ......

14.5.4.20 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: TMR1 peripheral address + ADDR*4 to TMR1 peripheral address + ADDR*4 + DTB*4

14.5.4.21 TMR1 channel mode register3 (TMR1_ CM3)

Bit	Register	Reset value	Type	Description
Bit 15: 6	Reserved	0x0	resd	Kept at default value.
Bit 7	C5OSEN	0x0	rw	Channel 5 output switch enable
Bit 6: 4	C5OCTRL	0x0	rw	Channel 5 output control
Bit 3	C5OBEN	0x0	rw	Channel 5 output buffer enable
Bit 2	C5OIEN	0x0	rw	Channel 5 output immediately enable
Bit 1: 0	Reserved	0x0	resd	Kept at default value.

14.5.4.22 TMR1 channel 5 data register (TMR1_C5DT)

Bit	Register	Reset value	Type	Description
Bit 15: 0	C5DT	0x0000	rw	Channel 5 data register C5DT holds the value that is to be compared with the CVAL. Whether the written data will takes effect immediately depends on the C5OBEN bit, and the corresponding output generates on the C5OUT bit.

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
- After 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.

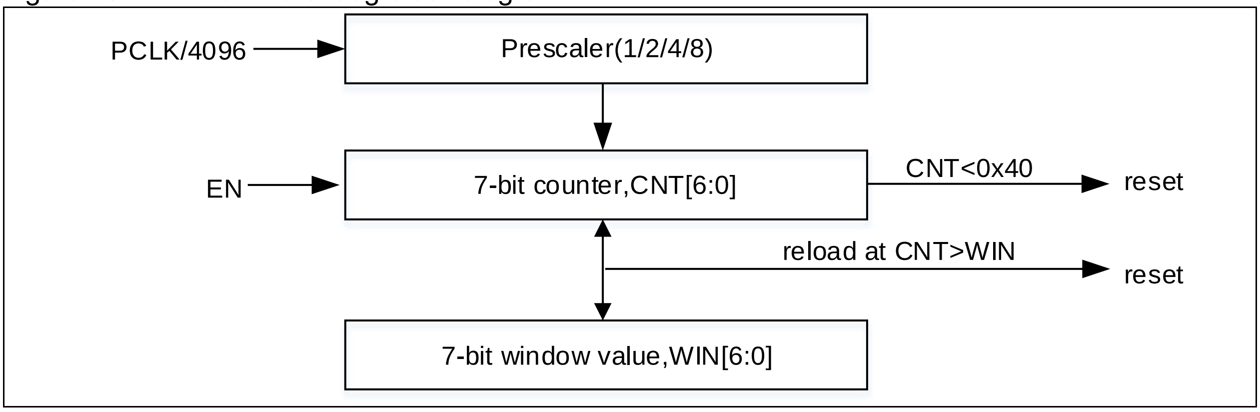
15.3 WWDT functional overview

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 7-bit downcounter is greater than the 7-bit window value programmed.

Figure 15-1 Window watchdog block diagram



To prevent system reset while reloading the counter value, 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. It can be used together with the WIN[6: 0] bit to adjust 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=1 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:

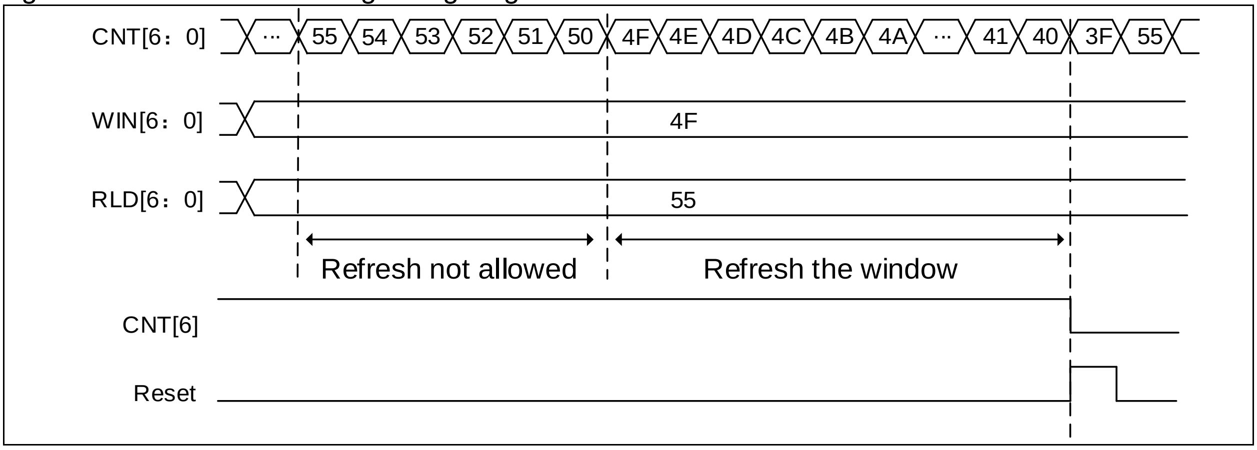
$$T_{WWDT} = T_{PCLK1} \times 4096 \times 2^{DIV[1:0]} \times (CNT[5:0] + 1); (ms)$$

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®-M4F core halted), the WWDT counter stops counting by setting the WWDT_PAUSE in the DEBUG module.

15.5 WWDT registers

These peripheral registers must be accessed by half words (16 bits) or words (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 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 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 will be 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 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 Deepsleep and Standby modes)
- The counter can be configured to stop counting either in Deepsleep or Standby mode
- A system reset is generated under the following circumstances:
  - When the counter value is decremented to 0
  - When the counter is reloaded outside the window

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 will 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. Besides, setting WIN[11:0]= 0xFFF (not default value) will enable window watchdog feature. In this case, when the counter value is greater than the window value, reloading counter value will trigger a system reset.

WDT write-protected:

The WDT_DIV, WDT_RLD and WDT_WIN registers are write-protected. Writing the value 0x5555 to the WDT_CMD register will unlock write protection. The update status of these three registers are indicated by the DIVF, RLDF and WINF bits in the WDT_STS register. If a different value is written to the WDT_CMD register, these three 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 LICK. The LICK is an internal RC clock in the range of 30kHz~60kHz. Therefore, 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 a relatively accurate WDT timeout. For more information about LICK, see *Section 4.1.1*.

WDT low power counting mode:

WDT can work in Sleep, Deepsleep and Standby modes. It is possible to stop counting in Deepsleep and Standby modes by setting the nWDT_DEPSLP and nWDT_STDBY bits in the User System Data area.

If the counter is disabled, it will stop decrementing as soon as the Deepsleep and Standby modes are entered. This means that the WDT would not generate a system reset in both low power modes. After waking up from these two modes, it continues downcounting from the value at the time of the entry of these modes.

Figure 16-1 WDT block diagram

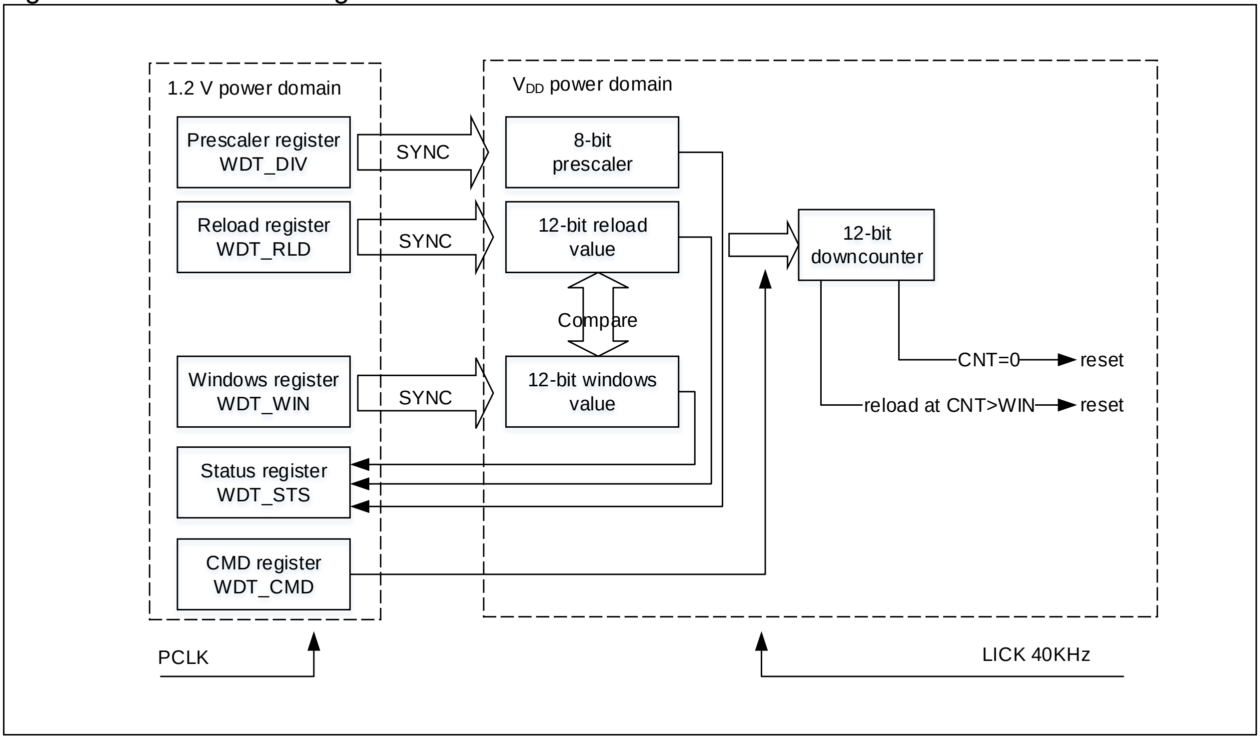


Table 16-1 WDT timeout period (LICK=40kHz)

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®-M4F core halted), the WDT counter stops counting by setting the WDT_PAUSE in the DEBUG module. Refer to Chapter 25.2 for more information.

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
WDT_WIN	0x10	0x0000 0FFF

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 default value.
Bit 15: 0	CMD	0x0000	wo	Command register 0xAAAA: Reload counter 0x5555: Unlock the write-protected WDT_DIV, WDT_RLD and WDT_WIN 0xCCCC: Enable WDT. If the hardware watchdog has been enabled, ignore this operation.

16.5.2 Divider register (WDT_DIV)

(Not reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 3	Reserved	0x0000 0000	resd	Kept at 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)

(Not reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at 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: 3	Reserved	0x0000 0000	resd	Kept at default value.
Bit 2	WINF	0x0	ro	Window value update complete flag 0: Window value update complete 1: Window value update is in process. The WDT_WIN register can be written only when RLDF=0
Bit 2	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

16.5.5 Window register (WDT_WIN)

(Not reset in Standby mode)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x000000	resd	Kept at default value.
Bit 11: 0	WIN	0xFFF	ro	Window value When the counter value is greater than the window value, reloading the counter will trigger a reset. The reload counter value falls between 0 and the window value.

17 Enhanced real-time clock (ERTC)

17.1 ERTC introduction

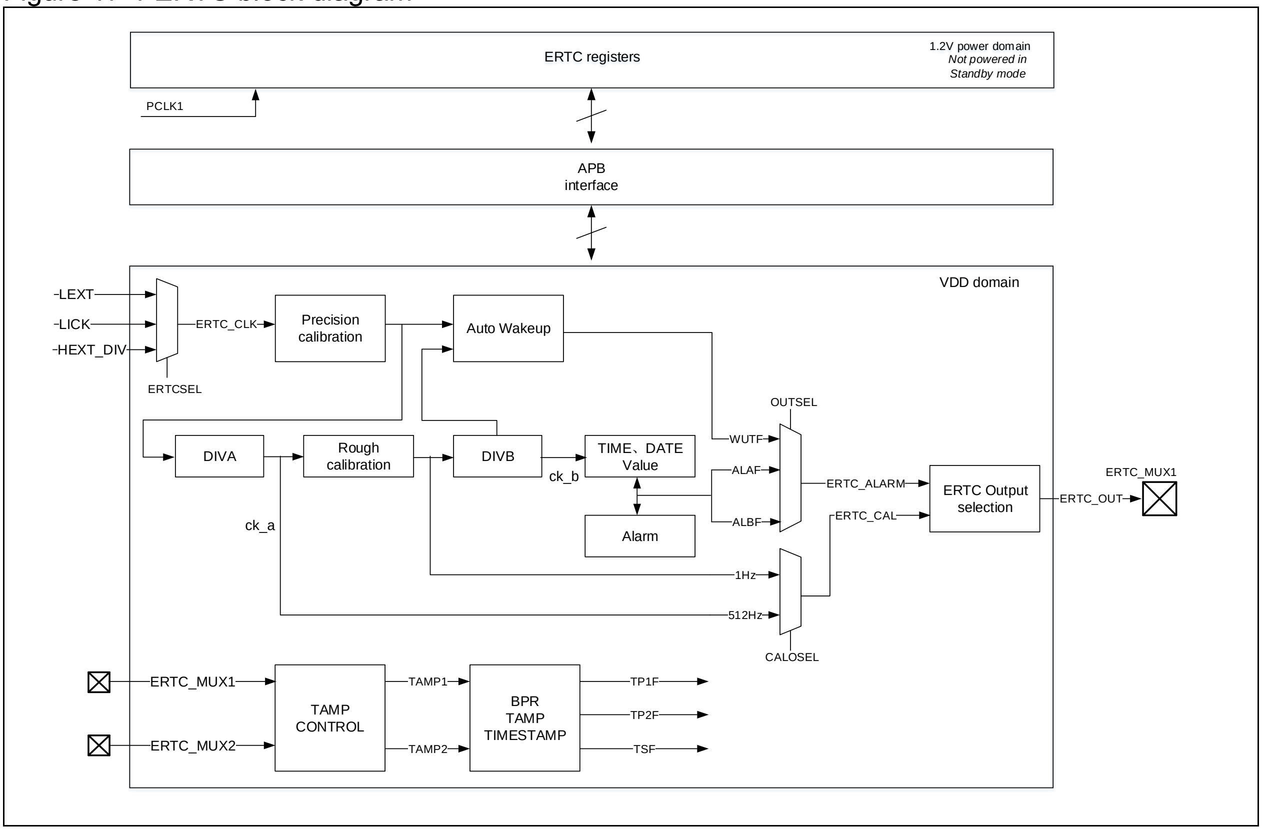
The real-time clock provides a calendar clock function. The time and date can be modified by modifying the ERTC_TIME and ERTC_DATE register.

The ERTC module is in the battery powered domain, which means that it keeps running and free from the influence of system reset as long as VBAT is powered (VBAT must be supplied through VDD domain).

17.2 ERTC main features

- Real-time calendar, one alarm. Compensations for 28-, 29- (leap year), 30-, and 31-day months are performed. When the year register is a multiple of 4, it represents a leap year
- Periodic auto-wakeup
- Reference clock detection
- 2x programmable tamper detection supporting time stamp feature
- Supports fine calibration
- 20 x battery powered registers
- 5 x interrupts: alarm A, alarm B, periodic auto-wakeup, tamper detection and time stamp
- Multiplexed function output, calibration clock output, alarm event or wakeup event
- Multiplexed function input, reference clock input, two-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 (by setting the ERTCSEL[1:0] in the CRM_BPDC register).

The HEXT frequency division value is configured through the ERTC_DIV[4:0] bit in the CRM_CFG register.

The ERTC embeds two dividers: A and B, selected by the DIVA[6: 0] and DIVB[14: 0] bits respectively. It is recommended that the DIVA is configured with a higher value in order to minimum power consumption. After being divided by dividers 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 auto wakeup. The clock frequencies of ck_a and ck_b are 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.

Note: If the divided HEXT is used to clock the ERTC_CLK, then the HEXT frequency division value must be configured first before switching the clock source to HEXT.

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 write protection.

To unlick 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 after the initialization mode is entered.

Table 17-1 RTC 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_ALBSBS	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. Enter initialization mode by setting IMEN =1.
2. Wait until the initialization flag INITF bit is set to 1.
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 to 1, 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=1 is activated, 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.

Reading the calendar

The ERTC offers two different ways to read the calendar, namely, synchronous read (DREN=0) and asynchronous read (DREN=1).

DREN=0: The clock and calendar values can be obtained by reading the 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.

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

Periodic automatic wakeup unit is used to wake up ERTC from low power consumption modes automatically. The period is programmed with the VAL[15: 0] bi (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)

Note: In debug mode, if the ERTC_CLK is selected as wakeup clock, the counter which is used for periodic wakeup works normally.

17.3.4 ERTC calibration

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 bit 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 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).

17.3.6 Time stamp function

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) will be set to 1 in the ERTC_STS register. 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 two tamper detection modes: TAMP1 and TAMP2. They can be configured as a level detection with filter or edge detection. TAMP1 uses the TSPIN bit to select either ERTC_MUX1 or ERTC_MUX2 as a tamper pin, while the TAMP2 can only select ERTC_MUX2 as a tamper pin.

The TP1F or TP2F will be set to 1 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 to 1, 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 or TP2EDG)
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. To use TAMP1 tamper detection mode, either ERTC_MUX1 or ERTC_MUX2 (TP1PIN bit) has to be selected as TAMPI mapping, and TAMP1 is enabled (TP1EN=1); to use TAMP2 tamper detection mode, the users only need enable TAMP2 by setting TP2EN=1.

How to configure level detection with filtering

1. Select level detection with filtering, and valid level sampling times (TPFLT≠00)
2. Select tamper detection valid level (TP1EDG or TP2EDG)
3. Select tamper detection sampling frequency (TPFREQ\)
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. To use TAMP1 tamper detection mode, either ERTC_MUX1 or ERTC_MUX2 (TP1PIN bit) has to be selected as TAMPI mapping, and TAMP1 is enabled (TP1EN=1); to use TAMP2 tamper detection mode, the users only need enable TAMP2 by setting TP2EN=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: Tamper detection is inactive when the battery powered domain is OFF.

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 event (OUTSEL=3)

When alarm clock or wakeup event is 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 clock, periodic auto wakeup, time stamp or tamper event. To enable an ERTC interrupt, follow the procedure below:

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 half words (16 bits) or 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_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 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 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 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 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 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 event
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	Reserved	0x0	resd	Kept at default value.
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_a 11x: ck_a 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 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	Reserved	0x0	resd	Kept at default value.
Bit 14	TP2F	0x0	rw0c	Tamper detection 2 flag 0: No tamper event 1: Tamper event occurred
Bit 13	TP1F	0x0	rw0c	Tamper detection 1 flag

				0: No tamper event 1: Tamper event occurred
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 occurred *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 occurred *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 ERTC_TIME, ERTC_DATE and ERTC_SBS are synchronized with the ERTC calendar value located in the battery powered domain. The synchronization is performed every two ERTC_CLK cycles.
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 default value.
Bit 22: 16	DIVA	0x7F	rw	Divider A
Bit 15	Reserved	0x0	resd	Kept at 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 default value.
Bit 15: 0	VAL	0xFFFF	rw	Wakeup timer reload value

17.4.7 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

17.4.8 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

17.4.9 ERTC write protection register (ERTC_WP)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at default value
Bit 7: 0	CMD	0x00	wo	Command register All ERTC register write protection is unlocked by writing 0xCA and then 0x53. Writing any other value will re-activate write protection.

17.4.10 ERTC subsecond register (ERTC_SBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at 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)

17.4.11 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 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))

17.4.12 ERTC time stamp time register (ERTC_TSTM)

Bit	Register	Reset value	Type	Description
Bit 31: 23	Reserved	0x000	resd	Kept at 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 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.

17.4.13 ERTC time stamp date register (ERTC_TSDT)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at 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 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.

17.4.14 ERTC time stamp subsecond register (ERTC_TSSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at 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.

17.4.15 ERTC smooth calibration register (ERTC_SCAL)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at default value
Bit 15	ADD	0x0	rw	Add ERTC clock 0: No ERTC clock added 1: One ERTC_CLK is inserted every 2^{11} 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 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 cycles. 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 cycles.

17.4.16 ERTC tamper configuration register (ERTC_TAMP)

Bit	Register	Reset value	Type	Description
Bit 31: 19	Reserved	0x0000	resd	Kept at default value
Bit 18	OUTTYPE	0x0	rw	Output type 0: Open-drain output 1: Push-pull output
Bit 17	TSPIN	0x0	rw	Time stamp detection pin selection 0: ERTC_MUX1 1: ERTC_MUX2
Bit 16	TP1PIN	0x0	rw	Tamper detection pin selection 0: ERTC_MUX1 1: ERTC_MUX2
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: 5	Reserved	0x0	resd	Kept at default value
Bit 4	TP2EDG	0x0		Tamper detection 2 valid edge No filtering (TPFLT=0): 0: Rising edge 1: Falling edge Filtering (TPFLT>0): 0: Low level 1: High level
Bit 3	TP2EN	0x0	rw	Tamper detection 2 enable 0: Tamper detection 2 disabled 1: Tamper detection 2 enabled
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

17.4.17 ERTC alarm clock A subsecond register (ERTC_ALASBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at 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 default value
Bit 14: 0	SBS	0x0000	rw	Sub-second value

17.4.18 ERTC alarm clock B subsecond register (ERTC_ALBSBS)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at 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 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/10-bit/8-bit/6-bit digital signal. Its sampling rate is as high as 5.33 MSPS. It has up to 26 channels for sampling and conversion.

18.2 ADC main features

In terms of analog:

- 12-bit/10-bit/8-bit/6-bit configurable resolution
- Self-calibration time: 205 ADC clock cycles
- ADC conversion time
  - Fast channel: ADC conversion time is 0.1875 μs (5.33 MSPS) at max. 80 MHz in 12-bit resolution
  - Slow channel: ADC conversion time is 0.2375 μs (4.21 MSPS) at max. 80 MHz in 12-bit resolution
  - Fast channel: ADC conversion time is 0.1126 μs (8.88 MSPS) at max. 80 MHz in 6-bit resolution
  - Slow channel: ADC conversion time is 0.1626 μs (6.15 MSPS) at max. 80 MHz in 6-bit resolution
- ADC supply requirement: 2.4 V to 3.6 V
- ADC input range: $V_{REF-} \leq V_{IN} \leq V_{REF+}$

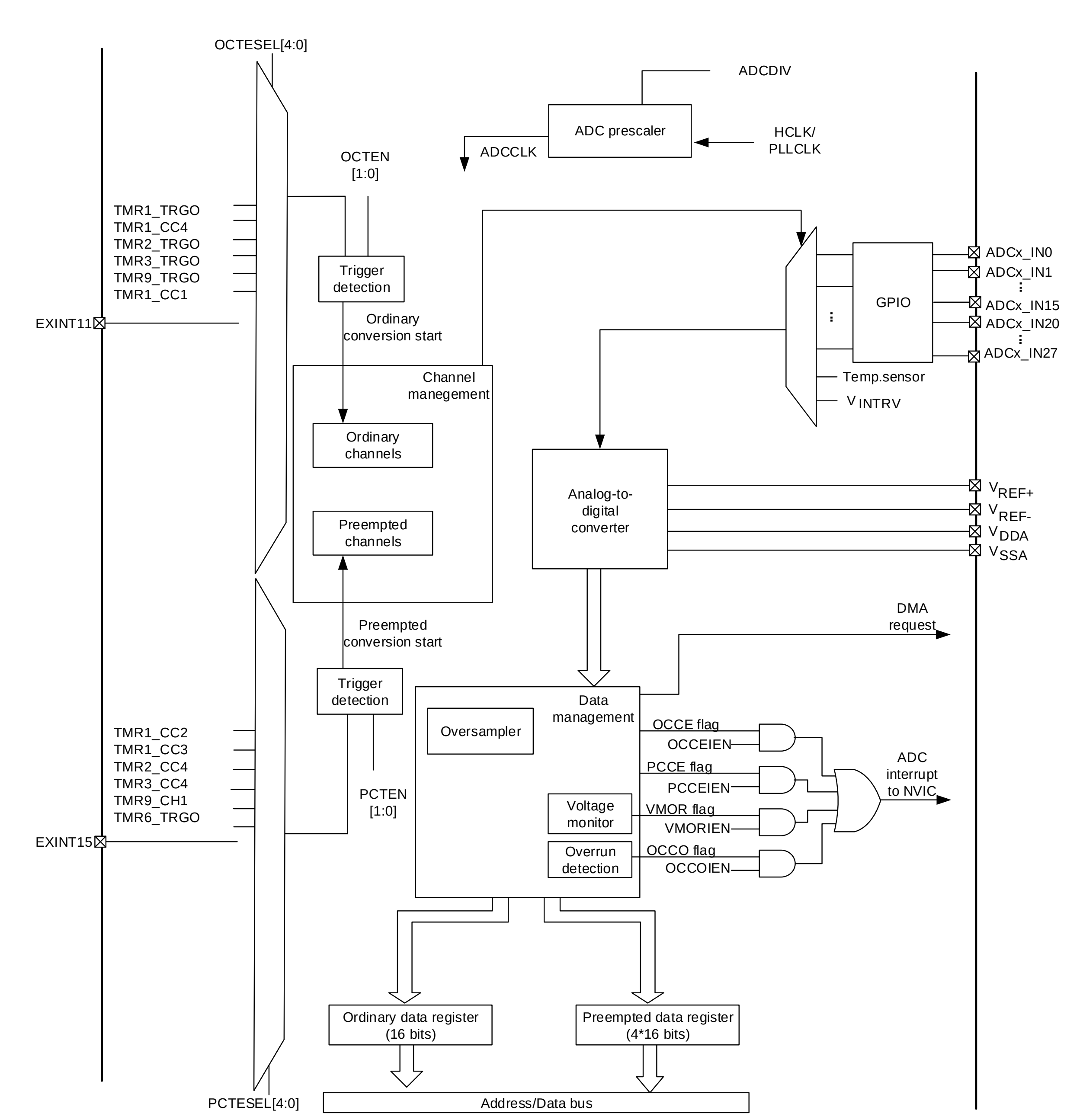
In terms of digital control:

- 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
- Oversampling: hardware oversampling up to 16-bit resolution
- Optional data alignment mode
- Programmable voltage monitoring threshold
- Regular channels with DMA transfers
- Interrupt generation at one of the following events:
  - Conversion overflow at regular channels
  - 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, can be connected to V_{DD}, or 2.4V ≤ V_{DDA} ≤ V_{DD} (2.6V)
- V_{SSA}: Analog supply ground, ADC analog supply ground, has to be connected to V_{SS}
- V_{REF+}: Analog reference positive, high/positive reference voltage for the ADC, 2.0V ≤ V_{REF+} ≤ V_{DDA}
- V_{REF-}: Analog reference negative, low/negative reference voltage for the ADC, has to be connected to V_{SS}
- ADCx_IN: Analog input signal channel

18.4 ADC functional overview

18.4.1 Channel management

Analog signal channel input:

There are 26 analog signal channel inputs for each of the ADCs, expressed by ADC_INx (x=0 to 17, 20 to 27).

- ADC_IN0 to ADC_IN15 represent external analog input, ADC_IN16 represents internal temperature sensor, ADC_IN17 represents internal reference voltage, and ADC_IN20 to ADC_IN27 represent external analog input.

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 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, the ITSRVEN bit in the ADC_CTRL register must be enabled and wait after power-on time.

Obtain the temperature using the following formula:

Temperature (in °C) = {(V 25 - V_{SENSE}) / Avg_Slope} + 25.

Where,

V_{25} = V_{SENSE} value for 25° C and

Avg_Slope = Average Slope for curve between Temperature vs. V_{SENSE} (given in mV/° C).

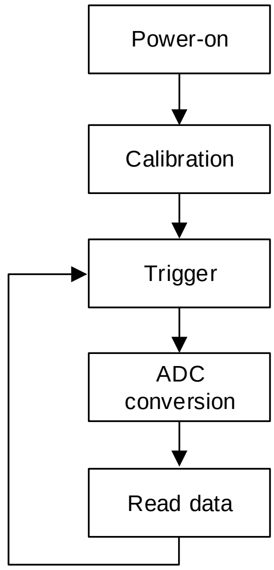
18.4.1.2 Internal reference voltage

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 ADCxEN 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 ADC_CCTRL register.

The PLLCLK_TO_ADC bit in the CRM_MISC1 register is used to select the clock source of ADC. When the PLL is selected, there is a need to ensure that the system clock SCLKSEL must select PLL/2.

Note: ADCCLK must be less than 80 MHz, while the ADCCLK frequency must be lower than PCLK2.

Then set the ADCEN bit in the ADC_CTRL2 register to supply the ADC, and wait until the RDF flag is set before starting ADC conversion. Clear the ADCEN bit will stop ADC conversion and result in a reset. In the meantime, the ADC is switched off to save power.

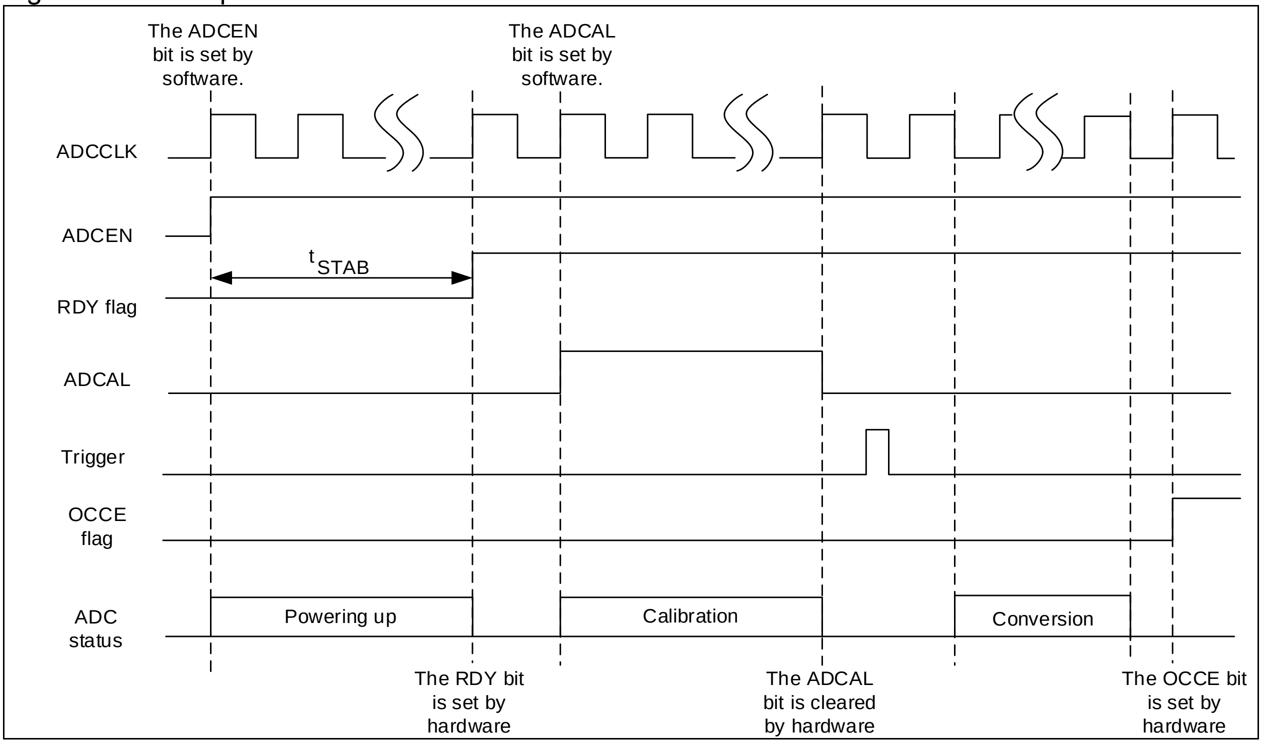
Calibration

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 the lower 7 bits of the ADC_CALVAL register, and then it 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 an interrupt or DMA request.

Note: The calibration can only be done when the 12-bit resolution is used. The switch of the resolution can be allowed only after the completion of the calibration. Then wait until the RDY flag is set before triggering conversion.

Figure 18-3 ADC power-on and calibration



18.4.2.2 Trigger

The ADC trigger contains 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. The valid polarity for external trigger sources is selected by the OCETE and PCETE bits 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, depending on the OCTESEL and PCTESEL bits in the ADC_CTRL2 register, as shown in Table 18-1.

Table 18-1 Trigger sources for ordinary and preempted channels

OCTESEL	Trigger source	PCTESEL	Trigger source
00000	TMR1_TRGOUT event	00000	TMR1_CH2 event
00001	TMR1_CH4 event	00001	TMR1_CH3 event
00010	TMR2_TRGOUT event	00010	TMR2_CH4 event
00011	TMR3_TRGOUT event	00011	TMR3_CH4 event
00100	TMR9_TRGOUT event	00100	TMR9_CH1 event
00101	TMR1_CH1 event	00101	TMR6_TRGOUT event
00110	EXINT line11 External pin	00110	EXINT line15 External pin
00111	OCSWTRG	00111	PCSWTRG
01000~1111	Reserved	01000~1111	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. Channel 0/1/10/11/12/13 represents fast channels, with a minimum 2.5 sampling cycles; others represent slow channels, with a minim 6.5 sampling cycles. The resolution can be programmed by setting the CRSEL bit in the ADC_CTRL1 register. A lower resolution has shorter conversion time. The duration required for a single conversion is calculated based on the following formula:

$$\text{A single conversion time (ADCCLK period)} = \text{sampling time} + \text{resolution bits} + 0.5$$

Example:

If the CSPTx selects 1.5 cycles, CRSEL is 12-bit, then a single conversion needs 2.5+12.5=15 ADCCLK cycles; If the CSPTx selects 7.5 cycles, then a conversion needs 6.5+10.5=17 ADCCLK cycles

18.4.3 Conversion sequence management

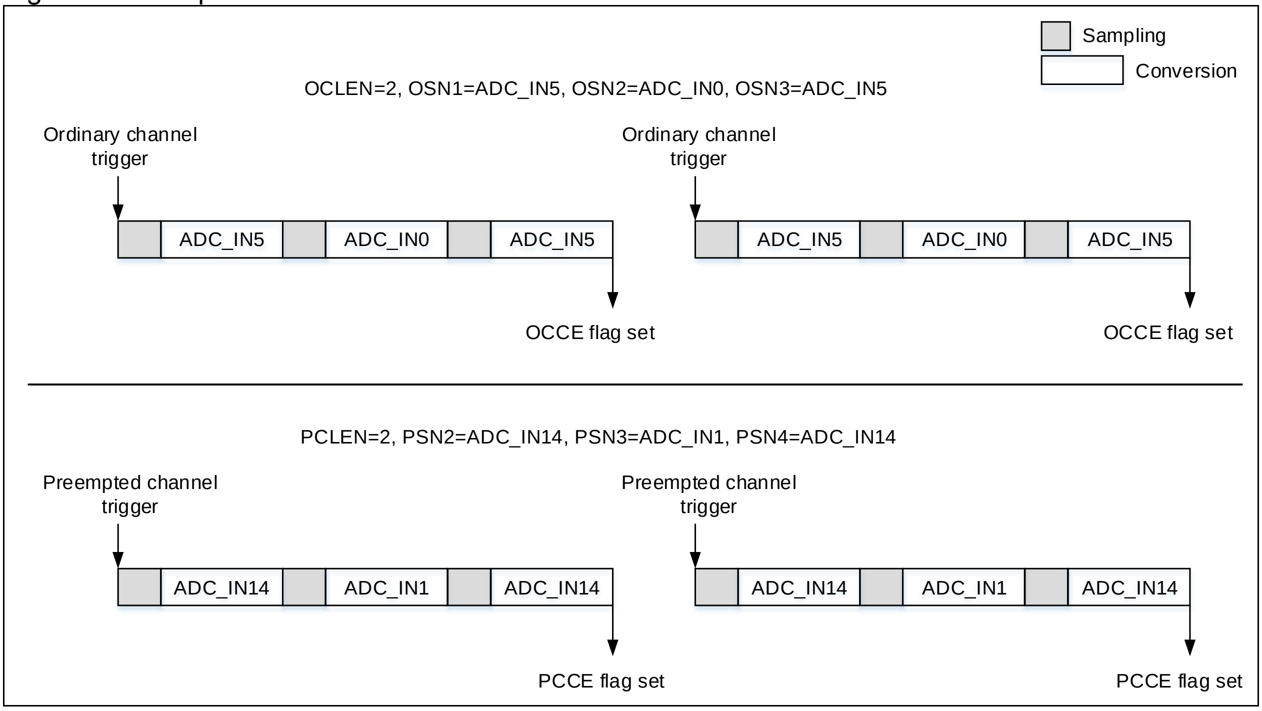
Only one channel is converted at each trigger by default, that is, OSN1-defined channel or PSN4-defined channel.

The following section describes various conversion sequence modes in detail. This mode enables multiple channels to be converted in a specific sequence.

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 the number of ordinary channels, while the ADC_PSQ register is used to define the sequence and the number of preempted channels. After 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.

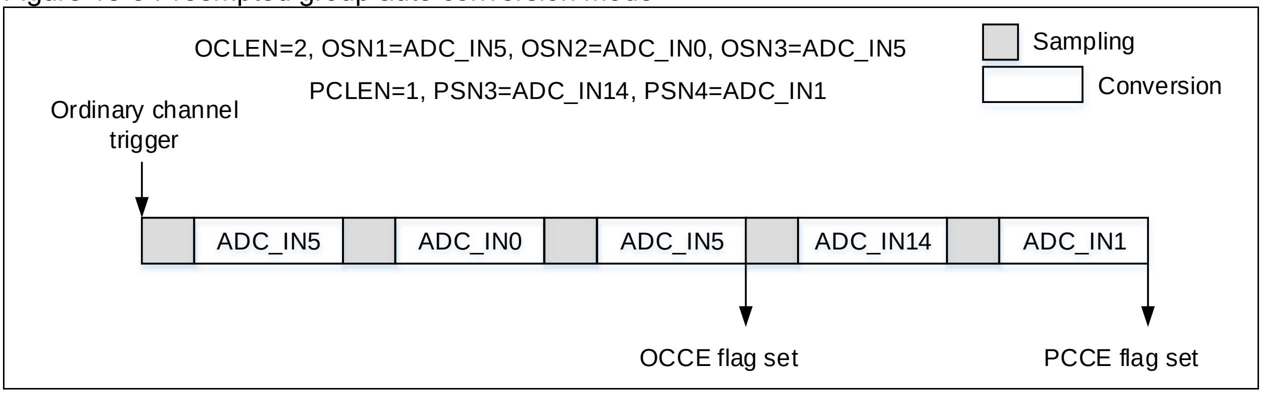
Figure 18-4 Sequence mode



18.4.3.2 Preempted group automatic conversion mode

The automatic conversion mode for preempted group is enabled by setting the PCAUTOEN bit in the ADC_CTRL1 register. In this mode, once the ordinary channel conversion is over, the preempted group will automatically continue its conversion. This mode can work in conjunction 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.

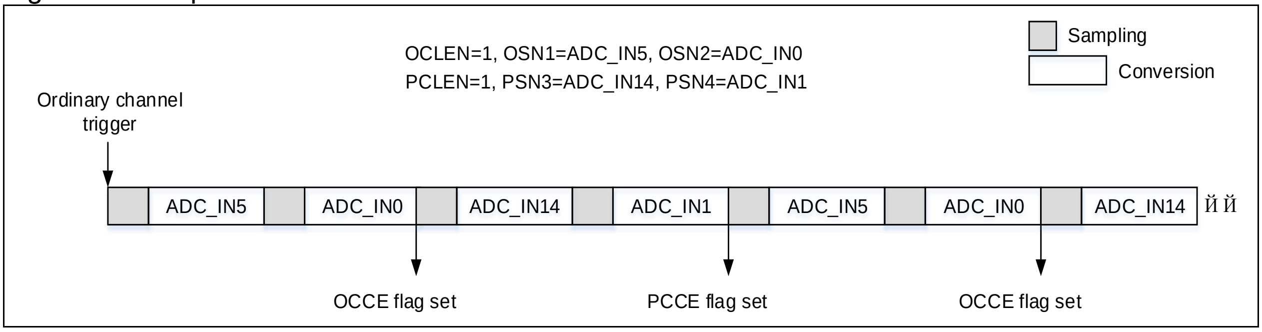
Figure 18-5 Preempted group auto conversion mode



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 in conjunction with the ordinary channel conversion in 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 19-6 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



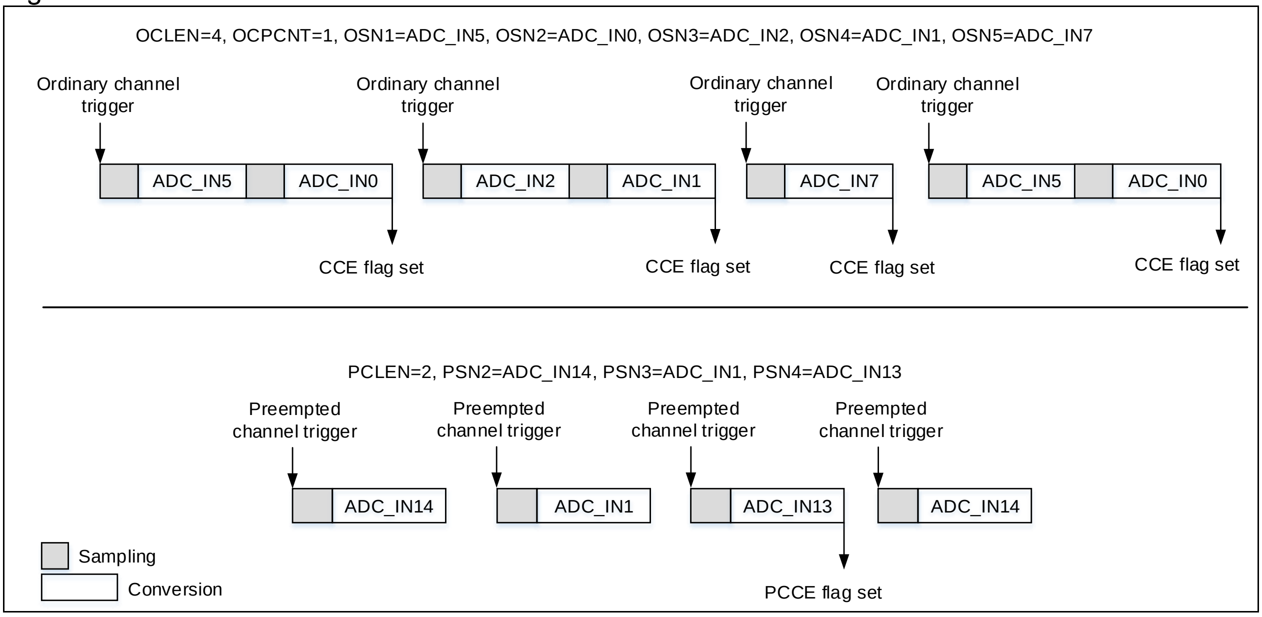
18.4.3.4 Partition mode

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 subgroups. The number of channels in a subgroup is specified through the OCPCNT bit in the ADC_CTRL1 register. When a trigger occurs, all channels in the subgroup will be converted. Each trigger selects a different subgroup in order.

Setting the PCPEN bit in the ADC_CTRL1 register enables the partition mode of preempted group. In this mode, the preempted group conversion sequence length (OCLEN bit in the ADC_OSQ1 register) is divided into subgroups, each of which has only one channel. When a trigger occurs, the channel in the subgroup will be converted. Each trigger event selects a different sub-group in order.

The partition mode and repetition mode cannot be used at the same time. *Figure 18-7* shows an example of the behavior in partition mode for ordinary group and preempted group.

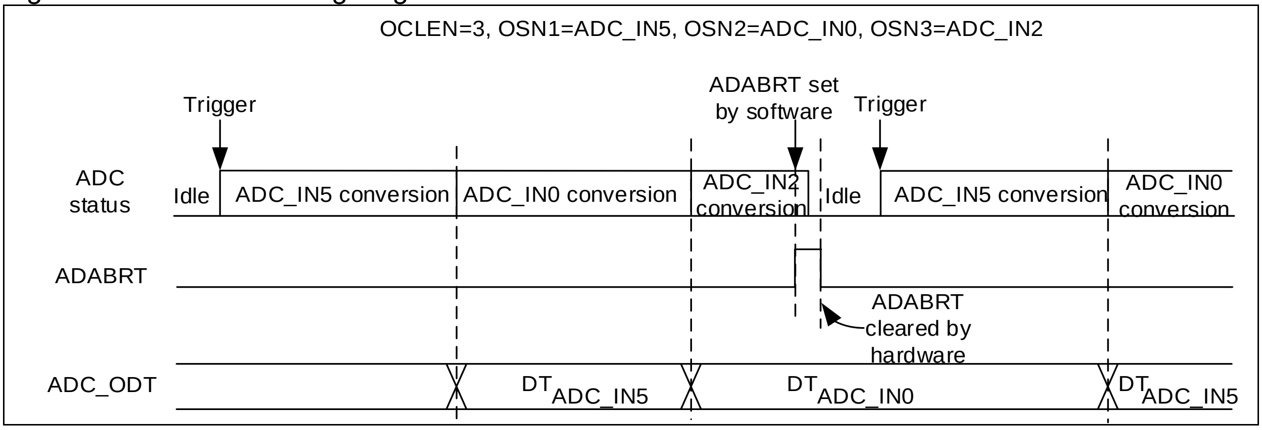
Figure 18-7 Partition mode



18.4.4 End of conversion

The ADABRT bit in the ADC_CTRL2 register is used to stop ADC conversions. At the end of the conversion, the conversion sequence returns to the first channel. This allows the user to configure a new sequence of channels, and the ADC starts conversions from the beginning based on the new order when a trigger occurs. Figure 18-8 shows the timing diagram when the ADABRT bit is set.

Figure 18-8 ADABRT timing diagram



18.4.5 Oversampling

The converted data of a single oversampling are obtained by enabling multiple conversions of the same channel and then averaging the cumulative converted data.

- Oversampling ratio is selected through the OSRSEL bit in the ADC_OVSP register. This bit is used to specify the oversampling multiple, which is done by converting the same channel many times
- Oversampling shift is selected through the OSSSEL bit in the ADC_OVSP register. This bit defines the average coefficient, which is done by right shift.

If the averaged data is greater than 16 bits, then only pick up the right-aligned 16-bit data and put them into a 16-bit data register, shown in Table 18-3.

Example:

If 4x oversampling is selected through the OSRSEL bit, then the same channel is converted by four times in a single oversampling conversion, and the converted data derived from 4 conversions is put together. If 6-bit resolution is selected through the OSSSE bit, then the cumulative data is divided by 2^6 and rounded up.

Table 18-2 Correlation between maximum cumulative data, oversampling multiple and shift digits

Oversampling multiple	**2x**	**4x**	**8x**	**16x**	**32x**	**64x**	**128x**	**256x**
Max cumulative data	**0x1FFE**	**0x3FFC**	**0x7FF8**	**0xFFF0**	**0x1FFE0**	**0x3FFC0**	**0x7FF80**	**0xFFF00**
No shift	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0	0xFF80	0xFF00
Shift 1 bit	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0	0xFF00
Shift 2 bit	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0	0xFFC0
Shift 3 bit	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0	0xFFE0
Shift 4 bit	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFF0
Shift 5 bit	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8
Shift 6 bit	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC
Shift 7 bit	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE
Shift 8 bit	0x020	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF

When using oversampling conversion mode, the DTALIGN and PCDTOx are ignored, and data must be right aligned. The CRSEL bit is used to select the desired oversampling resolution only, without changing the data operation mode.

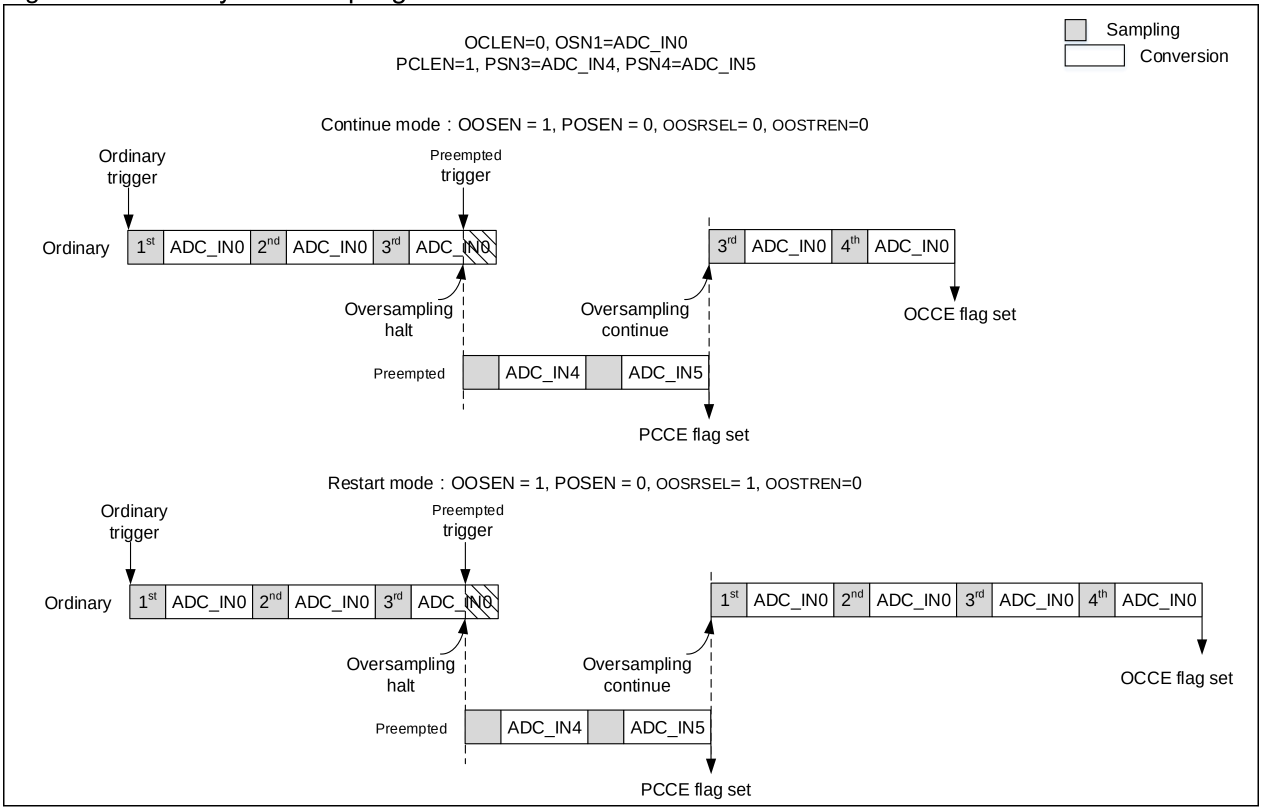
18.4.5.1 Oversampling of ordinary group of channels

The OOSRSEL bit in the ADC_OVSP register can be used to resume ordinary oversampling mode.

- OOSRSEL=0: continuous conversion mode. Ordinary group of channels, after being interrupted by preempted group of channels during oversampling, will retain the converted data and resume from the last interrupted ordinary conversion.
- OOSRSEL=1: restart mode. Ordinary group of channels, after being interrupted by preempted group of channels during oversampling, will be reset and restart the ordinary conversion from the beginning.

Figure 18-9 shows the differences between ordinary continuous mode and restart mode in 4x oversampling rate and sequence mode.

Figure 18-9 rdinary oversampling restart mode selection

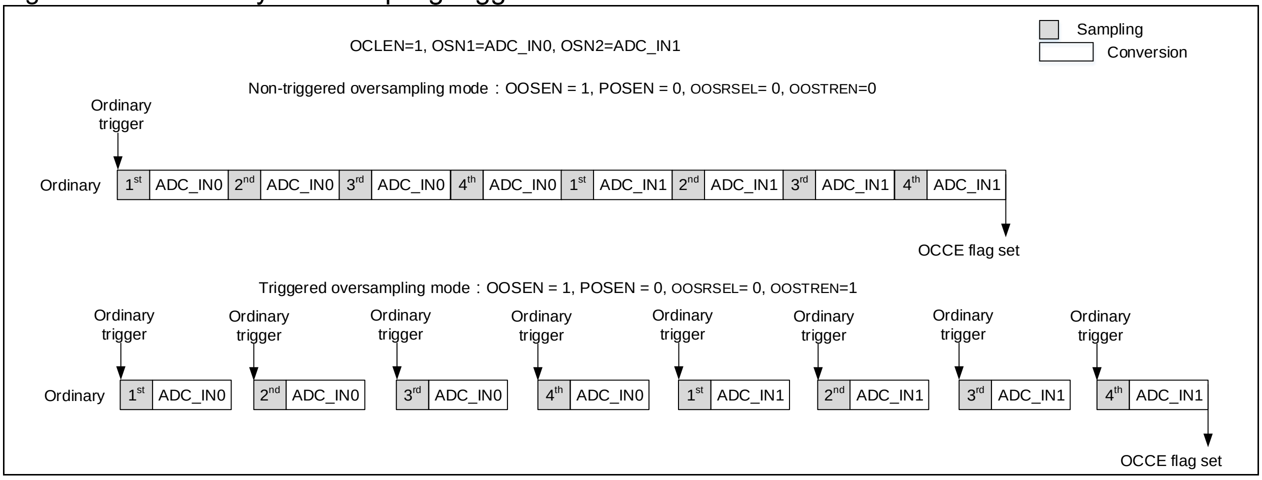


Trigger mode can be enabled by setting the OOSTREN bit in the ADC_OVSP register. The user must trigger each of the ordinary conversions. In this mode, once the ordinary conversion is interrupted by preempted group of channels, it is necessary to re-trigger ordinary group of channels before resuming the oversampling of ordinary channels.

When the trigger mode works in conjunction with conversion sequence management mode, trigger mode is applied, and the conversion complete flag follows the conversion sequence management mode. *Figure 18-10* shows the behavior when the ordinary trigger mode works together with resume mode in 4x oversampling rate and sequential mode.

Note: The trigger mode and repetition mode cannot be used concurrently.

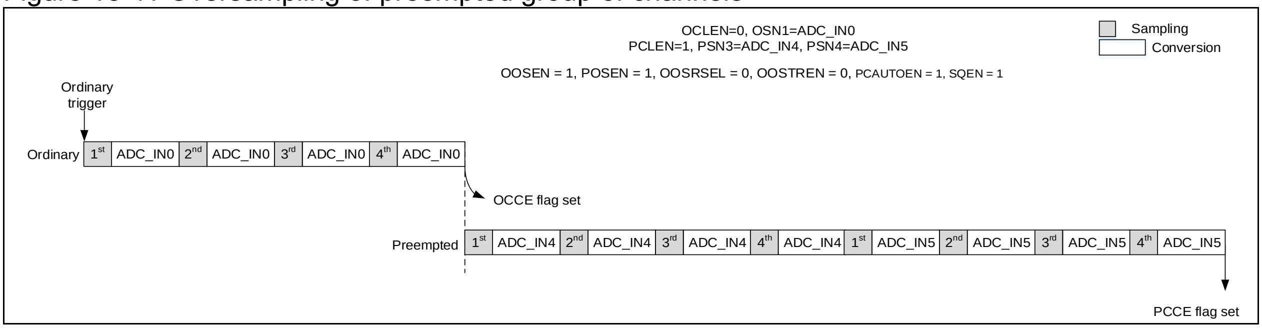
Figure 18-10 Ordinary oversampling trigger mode



18.4.5.2 Oversampling of preempted group of channels

It is possible to use both the preempted oversampling and ordinary oversampling simultaneously or individually. The oversampling of the preempted group of channels does not affect ordinary oversampling modes. *Figure 18-11* shows the behavior when the preempted oversampling and ordinary oversampling trigger mode are used simultaneously in 4x oversampling rate and auto-conversion of preempted group.

Figure 18-11 Oversampling of preempted group of channels



18.4.6 Data management

At the end of the conversion of the ordinary group, the converted data 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.

18.4.6.1 Data alignment

DTALIGN bit in the ADC_CTRL2 register selects the alignment of data (right-aligned or left-aligned). Besides, 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.

The data are aligned on a half-word basis expect when the resolution is set to 6-bit. In 6-bit resolution mode, the data are aligned on a byte basis, as shown in *Figure 18-12*.

Figure 18-12 Data alignment

Ordinary channel data 12 bits

Right-alignment

0	0	0	0	DT[11]	DT[10]	DT[9]	DT[8]	DT[7]	DT[6]	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]

Left-alignment

DT[11]	DT[10]	DT[9]	DT[8]	DT[7]	DT[6]	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]	0	0	0	0

Ordinary channel data 6 bits

Right-alignment

0	0	0	0	0	0	0	0	0	0	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]

Left-alignment

0	0	0	0	0	0	0	0	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]	0	0

Preempted channel data 12 bits

Right-alignment

SIGN	SIGN	SIGN	SIGN	DT[11]	DT[10]	DT[9]	DT[8]	DT[7]	DT[6]	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]

Left-alignment

SIGN	DT[11]	DT[10]	DT[9]	DT[8]	DT[7]	DT[6]	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]	0	0	0

Preempted channel data 6 bits

Right-alignment

SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]

Left-alignment

SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	DT[5]	DT[4]	DT[3]	DT[2]	DT[1]	DT[0]	0

18.4.6.2 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.

The EOCSFEN bit in the ADC_CTRL2 register can be used to select when to set the ordinary group conversion complete flag. In other words, the ordinary group conversion complete flag is set either at the end of sequence mode or each time the ordinary data register is updated, depending on the EOCSFEN bit.

When the OCDMAEN bit is set in the ADC_CTRL2 register, the ADC will issue DMA request each time the ADC_OTD register is updated.

When the EOCSFEN or OCDMAEN is set, the ADC will automatically start overflow detection. If an overflow event occurs, the OCCO flag will be set, the ADC stops conversion, and the last valid data is stored in the data register. If the DMA is used, the DMA request remains set so that the DMA can read the last valid data. The OCCO flag is cleared by software, and the ADC is triggered again so that it starts conversion from the next channel of the valid data. In this case, even if an overflow event occurs halfway, all the data read are valid and orderly.

The OCDRCEN bit in the ADC_CTRL2 register can be used to select whether to continually send DMA request after the DMA transfer register is reset.

18.4.7 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 monitoring on either a single channel or all the channels. The VMCSEL bit is used to select a specific channel for voltage monitoring.

Voltage monitoring is based on the comparison result between the original converted data and the 12-bit voltage monitor boundary register, regardless of the CRSEL, PCDTOx and DTALIGN bits.

When using an oversampler, voltage monitoring is based on the comparison result between the 16-bit registers (ADC_VMHB[15:0] and ADC_VMLB[15:0]) and the oversampled data.

18.4.7.1 Status flag and interrupts

ADC has its dedicated ADC_STS registers, namely, ready flag (RDY), overflow event flag (OCCO), ordinary channel conversion start flag (OCCS), preempted channel conversion start flag (PCCS), preempted channel conversion end flag (PCCE), ordinary channel conversion end flag (OCCE) and voltage monitor out of range (VMOR).

OCCO, PCCe, OCCE 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 18-3 lists ADC register map and their reset values.

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

Table 18-3 ADC register map and reset values

Register name	Offset	Reset value
ADC1_STS	0x000	0x0000 0000
ADC1_CTRL1	0x004	0x0000 0000
ADC1_CTRL2	0x008	0x0000 0000
ADC1_SPT1	0x00C	0x0000 0000
ADC1_SPT2	0x010	0x0000 0000
ADC1_PCDTO1	0x014	0x0000 0000
ADC1_PCDTO2	0x018	0x0000 0000
ADC1_PCDTO3	0x01C	0x0000 0000
ADC1_PCDTO4	0x020	0x0000 0000
ADC1_VMHB	0x024	0x0000 FFFF
ADC1_VMLB	0x028	0x0000 0000
ADC1_OSQ1	0x02C	0x0000 0000
ADC1_OSQ2	0x030	0x0000 0000
ADC1_OSQ3	0x034	0x0000 0000
ADC1_PSQ	0x038	0x0000 0000
ADC1_PDT1	0x03C	0x0000 0000
ADC1_PDT2	0x040	0x0000 0000
ADC1_PDT3	0x044	0x0000 0000
ADC1_PDT4	0x048	0x0000 0000
ADC1_ODT	0x04C	0x0000 0000
ADC1_SPT3	0x050	0x0000 0000
ADC1_OSQ4	0x054	0x0000 0000
ADC1_OSQ5	0x058	0x0000 0000
ADC1_OSQ6	0x05C	0x0000 0000
ADC1_OVSP	0x080	0x0000 0000
ADC1_CALVAL	0x0B4	0x0000 0000
ADC_CCTRL	0x304	0x0000 0000

18.5.1 ADC status register (ADC_STS)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x0000000	resd	Kept at default value.
Bit 6	RDY	0x0	rw0c	ADC conversion ready flag This bit is read only. It is set by hardware when the ADC is powered. 0: Not ready for ADC conversion 1: Ready for ADC conversion
Bit 5	OCCO	0x0	rw0c	Ordinary channel conversion overflow flag This bit is set by hardware and cleared by software (writing 0). 0: No overflow occurred 1: Overflow occurred Overflow detection is applicable to the case of DMA transfer enable or EOCSFEN =1
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
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: 27	Reserved	0x00	resd	Kept at default value.
Bit 26	OCCOIE	0x0	rw	Ordinary channel conversion overflow interrupt enable 0: Ordinary channel conversion overflow interrupt disabled 1: Ordinary channel conversion overflow interrupt enabled
Bit 25: 24	CRSEL	0x0	rw	Conversion resolution select

				00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit
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 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 …… 10001: ADC_IN17 channel 10100: ADC_IN20 channel 10101: ADC_IN21 channel …… 11011: ADC_IN27 channel 11011~11111: Unused, do not configure.

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 default value
Bit 30	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 29: 28	OCETE	0x0	rw	Ordinary channel external trigger edge select 00: Edge trigger forbidden 01: Rising edge 01: Falling edge 11: Any edge
Bit 31 Bit 27: 24	OCTESEL	0x0	rw	Ordinary channel conversion trigger event select Note: Refer to section *18.4.1.1* for details on bits.
Bit 22	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 21: 20	PCETE	0x0	rw	Preempted channel external trigger edge select 00: Edge trigger forbidden 01: Rising edge 01: Falling edge 11: Any edge
Bit 23 Bit 19: 16	PCTESEL	0x0	rw	Preempted channel conversion trigger event select Note: Refer to section *18.4.1.1* for details on bits.
Bit 15: 12	Reserved	0x0	resd	Kept at default value.
Bit 11	DTALIGN	0x0	rw	Data alignment 0: Right alignment 1: Left alignment
Bit 10	EOCSFEN	0x0	rw	Each ordinary channel conversion OCCE flag enable 0: Disabled 1: Enabled Note: Overflow detection is enabled automatically when this bit is set.
Bit 9	OCDRCEN	0x0	rw	Ordinary channel DMA request continue enable for independent mode 0: Disabled (After the completion of the programmed number of DMA transfers, no DMA request generated at the end of ordinary conversion) 1: Enabled (Don't care about the programmed number of DMA transfers, Each ordinary channel sends DMA request at the end of ordinary conversion) Note: This bit is applicable to non-master/salve mode only and when OCDMAEN = 1.
Bit 8	OCDMAEN	0x0	rw	DMA transfer enable of ordinary channels 0: Disabled 1: Enabled
Bit 7: 5	Reserved	0x0	resd	Kept at default value.
Bit 4	ADABRT	0x0	rw	ADC conversion abort

				0: No ADC conversion abort command 1: Abort current ADC conversion Note: This bit is cleared by hardware when the ADC conversion stops. After this bit is cleared, it is possible to retrigger conversion.
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 initialization 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 channel is converted each time when a trigger event arrives; when SQEN=1, a group of channels are converted 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. The application should pay attention to the fact that there is a delay of t_{STAB} between power up 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 default value.
Bit 23: 21	CSPT17	0x0	rw	Sample time selection of channel ADC_IN17 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 20: 18	CSPT16	0x0	rw	Sample time selection of channel ADC_IN16 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles

				111: 640.5 cycles
Bit 17: 15	CSPT15	0x0	rw	Sample time selection of channel ADC_IN15 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 14: 12	CSPT14	0x0	rw	Sample time selection of channel ADC_IN14 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 11: 9	CSPT13	0x0	rw	Sample time selection of channel ADC_IN13 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 8: 6	CSPT12	0x0	rw	Sample time selection of channel ADC_IN12 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 5: 3	CSPT11	0x0	rw	Sample time selection of channel ADC_IN11 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 2: 0	CSPT10	0x0	rw	Sample time selection of channel ADC_IN10 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles

				110: 247.5 cycles 111: 640.5 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 default value
Bit 29: 27	CSPT9	0x0	rw	Sample time selection of channel ADC_IN9 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 26: 24	CSPT8	0x0	rw	Sample time selection of channel ADC_IN8 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 23: 21	CSPT7	0x0	rw	Sample time selection of channel ADC_IN7 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 20: 18	CSPT6	0x0	rw	Sample time selection of channel ADC_IN6 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 17: 15	CSPT5	0x0	rw	Sample time selection of channel ADC_IN5 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

Bit 14: 12	CSPT4	0x0	rw	Sample time selection of channel ADC_IN4 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 11: 9	CSPT3	0x0	rw	Sample time selection of channel ADC_IN3 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 8: 6	CSPT2	0x0	rw	Sample time selection of channel ADC_IN2 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 5: 3	CSPT1	0x0	rw	Sample time selection of channel ADC_IN1 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 2: 0	CSPT0	0x0	rw	Sample time selection of channel ADC_IN0 000: Reserved 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 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 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 monitoring high threshold register (ADC_VWHB)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x00000	resd	Kept at default value
Bit 15: 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 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 default value
Bit 23: 20	OCLEN	0x0	rw	Ordinary conversion sequence length 0000: 1 conversion 0001: 2 conversions …… 11111: 32 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 0~17,20~27. For example, if the number is set to 3, it means that the 13^{th} 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 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	Register	Reset value	Type	Description
Bit 4: 0	OSN7	0x00	rw	Number of 7th conversion in ordinary sequence Note: The number can be 0~17, 20~27. 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 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 0~17, 20~27. For example, if the number is set to 17, 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 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 0~17, 20~27. 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 default value
Bit 15: 0	PDTx	0x0000	rw	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	Reserved	0x0000	resd	Kept at default value
Bit 15: 0	ODT	0x0000	ro	Conversion data of ordinary channel

18.5.15 ADC sampling time register 3 (ADC_SPT3)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0	resd	Kept at default value.
Bit 23:21	CSPT27	0x0	rw	Sample time selection of channel ADC_IN27 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 20:18	CSPT26	0x0	rw	Sample time selection of channel ADC_IN26 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 17:15	CSPT25	0x0	rw	Sample time selection of channel ADC_IN25 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 14:12	CSPT24	0x0	rw	Sample time selection of channel ADC_IN24 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 11:9	CSPT23	0x0	rw	Sample time selection of channel ADC_IN23 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 8:6	CSPT22	0x0	rw	Sample time selection of channel ADC_IN22 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles

				011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 5:3	CSPT21	0x0	rw	Sample time selection of channel ADC_IN21 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles
Bit 2:0	CSPT20	0x0	rw	Sample time selection of channel ADC_IN20 000: 2.5 cycles 001: 6.5 cycles 010: 12.5 cycles 011: 24.5 cycles 100: 47.5 cycles 101: 92.5 cycles 110: 247.5 cycles 111: 640.5 cycles

18.5.16 ADC ordinary sequence register 4 (ADC_OSQ4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31:30	Reserved	0x0	resd	Kept at default value.
Bit 29:25	OSN22	0x00	rw	Number of 22nd conversion in ordinary sequence
Bit 24:20	OSN21	0x00	rw	Number of 21st conversion in ordinary sequence
Bit 19:15	OSN20	0x00	rw	Number of 20th conversion in ordinary sequence
Bit 14:10	OSN19	0x00	rw	Number of 19th conversion in ordinary sequence
Bit 9:5	OSN18	0x00	rw	Number of 18th conversion in ordinary sequence
Bit 4:0	OSN17	0x00	rw	Number of 17th conversion in ordinary sequence Note: The number can be set from 0 to 17. 20 to 27. For example, 17 represents that the first conversion channel is ADC_IN17.

18.5.17 ADC ordinary sequence register 5 (ADC_OSQ5)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31:30	Reserved	0x0	resd	Kept at default value.
Bit 29:25	OSN28	0x00	rw	Number of 28th conversion in ordinary sequence
Bit 24:20	OSN27	0x00	rw	Number of 27th conversion in ordinary sequence
Bit 19:15	OSN26	0x00	rw	Number of 26th conversion in ordinary sequence
Bit 14:10	OSN25	0x00	rw	Number of 25th conversion in ordinary sequence
Bit 9:5	OSN24	0x00	rw	Number of 24th conversion in ordinary sequence
Bit 4:0	OSN23	0x00	rw	Number of 23nd conversion in ordinary sequence Note: The number can be set 0~17, 20~27. For example, 17 represents that the first conversion channel is ADC_IN17.

18.5.18 ADC ordinary sequence register 6 (ADC_OSQ6)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31:20	Reserved	0x0	resd	Kept at default value.
Bit 19:15	OSN32	0x00	rw	Number of 32nd conversion in ordinary sequence
Bit 14:10	OSN31	0x00	rw	Number of 31th conversion in ordinary sequence
Bit 9:5	OSN30	0x00	rw	Number of 30th conversion in ordinary sequence
Bit 4:0	OSN29	0x00	rw	Number of 29th conversion in ordinary sequence Note: The number can be set 0~17, 20~27. For example, 17 represents that the first conversion channel is ADC_IN17.

18.5.19 ADC oversampling register (ADC_OVSP)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 11	Reserved	0x0000	resd	Kept at default value.
Bit 10	OOSRSEL	0x0	rw	Ordinary oversampling restart mode select When the ordinary oversampling is interrupted by preempted conversions, this bit can be used to select where to resume ordinary conversions. 0: Continuous mode (ordinary oversampling buffer will be reserved) 1: Restart mode (ordinary oversampling buffer will be cleared, that is, the previously oversampled times are reset)
Bit 9	OOSTREN	0x0	rw	Ordinary oversampling trigger mode enable 0: Disabled (only one trigger is needed for all oversampling conversions) 1: Enabled (Each oversampling conversion needs a trigger)
Bit 8: 5	OSSSEL	0x0	rw	Oversampling shift select This field is used to define the number of right shift used in the oversampling results. 0000: No shift 0001: Shift 1 bit 0010: Shift 2 bits 0011: Shift 3 bits 0100: Shift 4 bits 0101: Shift 5 bits 0110: Shift 6 bits 0111: Shift 7 bits 1000: Shift 8 bits 1001~1111: Unused. Do not configure.
Bit 4: 2	OSRSEL	0x0	rw	Oversampling ratio select 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x 111: 256x
Bit 1	POSEN	0x0	rw	Preempted oversampling enable 0: Preempted oversampling disabled 1: Preempted oversampling enabled
Bit 0	OOSEN	0x0	rw	Ordinary oversampling enable 0: Ordinary oversampling disabled 1: Ordinary oversampling enabled

18.5.20 ADC calibration value register (ADC_CALVAL)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 7	Reserved	0x0000	resd	Kept at default value
Bit 6: 0	CALVAL	0x0	rw	A/D Calibration value

18.5.21 ADC common control register (ADC_CCTRL)

Accessed by words.

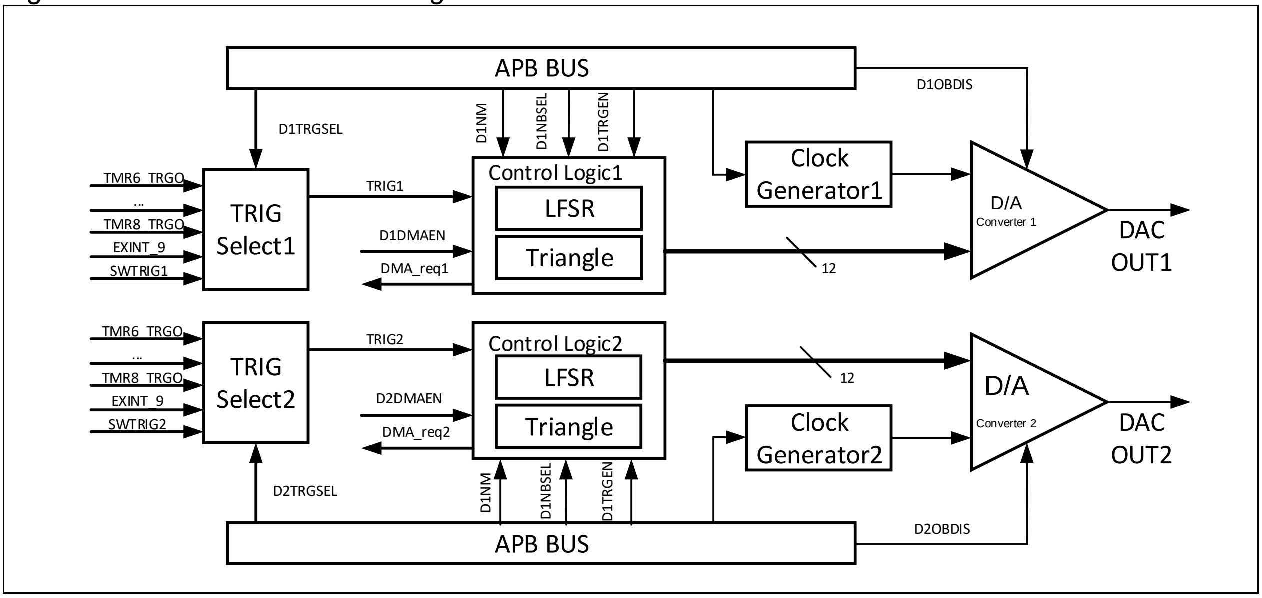
Bit	Register	Reset value	Type	Description
Bit 31: 24	Reserved	0x0000	resd	Kept at default value.
Bit 23	ITSRVEN	0x0	rw	Internal temperature sensor and V_{INTRV} enable 0: Disabled 1: Enabled
Bit 22: 20	Reserved	0x0	resd	Kept at default value.
Bit 19: 16	ADCDIV	0x0	rw	ADC division 0000: ADC input clock/2 0001: ADC input clock/3 … 1111: ADC input clock/17 Note: The clock divided by this field are used by all ADCs. The maximum value of the ADCCLK is 80 MHz. After this division, the ADCCLK cannot be higher than PCLK2.
Bit 15:0	Reserved	0x0	resd	Kept at default value.

19 Digital-to-analog converter (DAC)

19.1 DAC introduction

The DAC uses a 12-bit digital input to generate an analog output between 0 and reference voltage. The digital part of the DAC can be configured in 8-bit or 12-bit mode and can be used in conjunction with the DMA. It supports left or right alignment in a single /dual DAC modes. DAC1 and DAC2 have their own converter each. DAC1/DAC2 can be converted independently. Dual DAC can be triggered for conversion simultaneously. The input reference voltage V_{REF+} makes conversion more accuracy.

Figure 19-1 DAC1/DAC2 block diagram



19.2 DAC main features

- A single/dual DAC 8-bit or 12-bit digital input
- Left or right data alignment
- Noise-wave/Triangular-wave generation
- Dual DAC or single DAC1/DAC2 independent conversion
- DMA mode for DAC1/DAC2
- Software or external triggers for conversion
- Input reference voltage V_{REF+}

19.3 Design hints and tips

The following information can be used as DAC design reference:

- **Analog module configuration**
  The analog part of the DAC1/DAC2 can be enabled by setting the ENx bit in the DAC_CTRL register, but its digital part is not subject to this bit. Besides, the DAC integrates two output gains that can be used to reduce the output impedance, and to drive external loads directly without the need of an external operational amplifier. The DAC1/DAC2 output gain can be enabled or disabled through the DxOBDIS bit in the DAC_CTRL register.
- **DMA capability**
  The DAC1/DAC2 both have a DMA capability that can be enabled by setting the DxDMAEN bit in the DAC_CTRL register. One DMA request is generated when a trigger signal is active while the DxTRGEN bit is set. The DAC DMA request is not added up, meaning the new DAM request will be ignored and no error is reported.

  In dual DAC mode, the application can handle two channels (DAC1/DAC2) by using only one DMA request and a DMA channel.

- **DMA underflow**

  When the DAC DMA feature is enabled, an overflow occurs if a second external trigger arrives before the acknowledgement for the first external trigger is received. In this case, no new external trigger is handled, and no new DMA request is issued, and the DxDMAUDRF bit in the DAC_SR register is set, reporting the error. An interrupt is generated if the DxDMAUDRIEN bit is set in the DAC_CTRL register.

  The software clears the DxDMAUDRF bit by writing 1. The DxDMAEN bit should be cleared in the DAC_CTRL register before re-starting a DMA transfer and re-initializing DMA and DAC.

- **Input/output configuration**

  The digital input is linearly converted to analog voltage output by the DAC, and it is between 0 and V_{REF+}. The analog DAC module is supplied by VDDA. The positive analog reference voltage input falls between 2.0 V and VDDA. To avoid parasitic interruption and excessive consumption, the PA4 or PA5 should be configured as analog input.

  DAC output = V_{REF+} x (DxODT[11: 0]/4095)

19.4 Functional overview

19.4.1 Trigger events

If the DxTRGEN bit is set to 1 in the DAC_CTRL register, the DAC conversion can be triggered by an external event (timer counter, external interrupt line) or by software. The DxTRGSEL[2: 0] is used to select trigger sources.

Table 19-1 Trigger source selection

Source	DxTRGSEL [2:0]	Description
TMR6_TRGOUT	000	On-chip signals
TMR3_TRGOUT	001	
TMR7_TRGOUT	010	
TMR9_TRGOUT	011	
TMR2_TRGOUT	100	
TMR4_TRGOUT	101	
EXINT_9	110	External signal
DxSWTRG	111	Software trigger

When the DxTRGEN bit is set to 1, the data stored into the HDRx register is transferred into the DAC_DxODT register each time a DAC detects an active trigger event. If the software trigger is selected, the DxSWTRG flag is cleared by hardware after software writing 1. As long as the data is loaded into the DAC_DxODT register, the DAC output becomes active after a period of time.

When the DxTRGEN bit is cleared, each data written to the data register is immediately transferred into the DAC_DxODT register without the need of a trigger event.

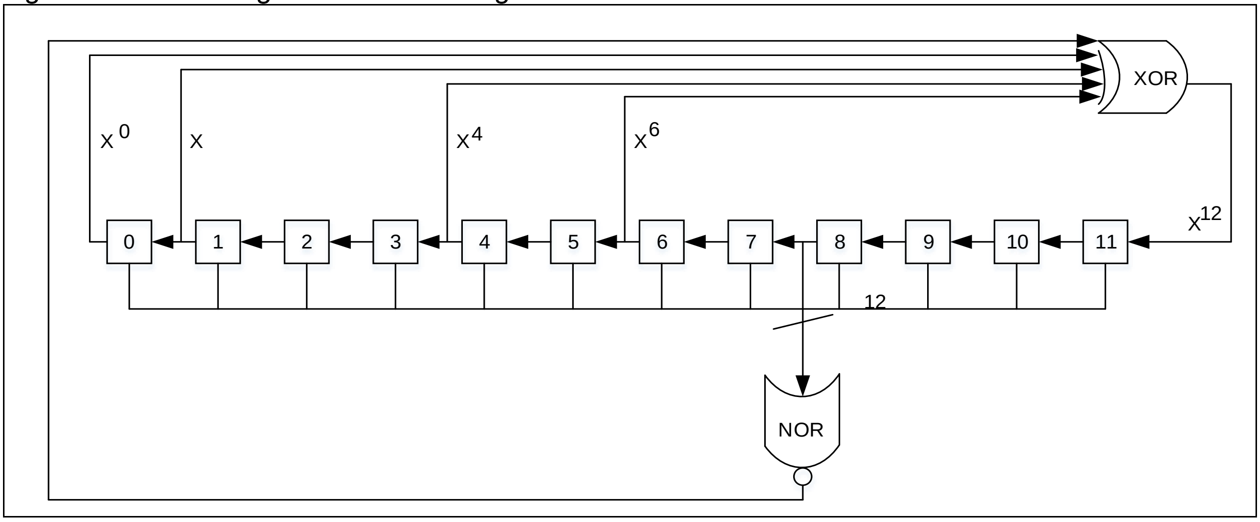
19.4.2 Noise/Triangular-wave generation

The DAC can output a variable-amplitude pseudonoise and a triangular wave, which is done by the LENinear Feedback Shift Register and triangle wave generator respectively. The DAC variable-amplitude pseudo noise generation is selected by setting DxNM[1:0]=01 in the LFSR, while the DAC triangular-wave generation is selected by setting the DxNM[1:0]=1x.

LFSR logic

The preloaded value in the LFSR is 0xAAA. This register is updated after each trigger event according to a specific calculation algorithm.

Figure 19-2 LFSR register calculation algorithm



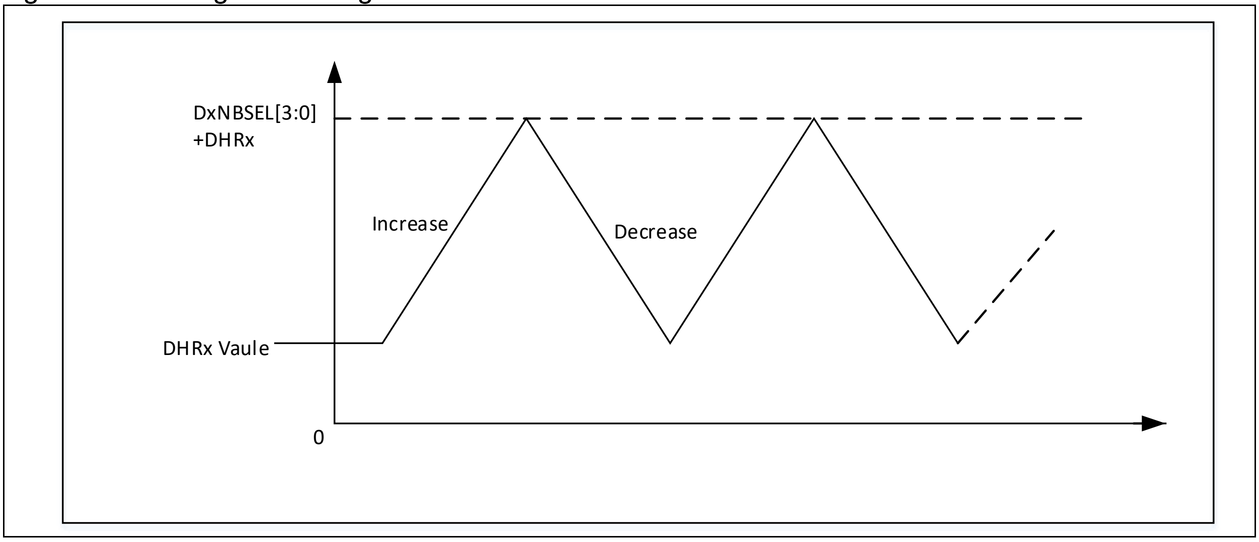
The DxNBSEL [3: 0] bit in the DAC_CTRL register is set to mark partially or totally LFSR data. The resulting LSFR value is then added up to the DHRx value without overflow and this value is loaded into the DAC_DxODT register. It is possible to disable LFSR function and reset LFSR wave generation algorithm by setting DxNM[1: 0]=00.

Triangular wave logic

The DAC triangular-wave generation is selected by setting DxNM[1: 0]=1x. The amplitude is configured through the DxNBSEL [3: 0] bit in the DAC_CTRL register. An internal triangular-wave counter is incremented at each trigger event. Once the maximum amplitude programmed in the DxNBSEL [3: 0] is reached, the value of this counter is decremented down to 0, then incremented again, and so on.

Meanwhile, the value of this counter is then added up to the DHRx register without overflow and the resulting value is loaded into the DAC_DxODT register. It is possible to disable/reset the triangular-wave generation by setting DxNM[1: 0]=00.

Figure 19-3 Triangular-wave generation



19.4.3 DAC data alignment

The DAC supports a single DAC and dual DAC mode. The data format is dependent on the selected configuration mode.

Single DAC data format:

8-bit right alignment: load data into the DAC_DxDTH8R [7:0]

12-bit left alignment: load data into the DAC_DxDTH12L [15: 4]

12-bit right alignment: load data in the DAC_DxDTH12R [11: 0]

Dual DAC data format:

8-bit right alignment: load data into the DAC_DDTH8R [7: 0] and DAC_DDTH8R [15: 8]

12-bit left alignment: load data into the DAC_DDTH12L [15: 4] and DAC_DDTH12L [31: 20]

12-bit right alignment: load data into the DAC_DDTH12R [11: 0] and DAC_DDTH12R [27:16]

The loaded 8-bit data corresponds to the DHRx[11:4] and the loaded 12-bit data corresponds to the DHRx[11: 0]

19.5 DAC registers

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

Table 19-2 DAC register map and reset values

Register name	Offset	Reset value
DAC_CTRL	000h	0x0000 0000
DAC_SWTRG	004h	0x0000 0000
DAC_D1DTH12R	008h	0x0000 0000
DAC_D1DTH12L	00Ch	0x0000 0000
DAC_D1DTH8R	010h	0x0000 0000
DAC_D2DTH12R	014h	0x0000 0000
DAC_D2DTH12L	018h	0x0000 0000
DAC_D2DTH8R	01Ch	0x0000 0000
DAC_DDTH12R	020h	0x0000 0000
DAC_DDTH12L	024h	0x0000 0000
DAC_DDTH8R	028h	0x0000 0000
DAC_D1ODT	02Ch	0x0000 0000
DAC_D2ODT	030h	0x0000 0000
DAC_STS	034h	0x0000 0000

19.5.1 DAC control register (DAC_CTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at default value
Bit 29	D2DMAUDRIEN	0x0	rw	DAC2 DMA transfer underrun interrupt enable This bit is set and cleared by software. 0: DAC2 DMA transfer underrun interrupt disabled 1: DAC2 DMA transfer underrun interrupt enabled
Bit 28	D2DMAEN	0x0	rw	DAC2 DMA transfer enable This bit is set and cleared by software. 0: DAC2 DMA mode disabled 1: DAC2 DMA mode enabled
Bit 27: 24	D2NBSEL	0x0	rw	DAC2 noise bit select These bits are used to select the mark bit in noise generation mode or select an amplitude in triangular-wave generation mode.

				0000: Unmask LSFR bit0 /Triangle amplitude is equal to 1 0001: Unmask LSFR bit[1: 0] /Triangle amplitude is equal to 3 0010: Unmask LSFR bit[2: 0] /Triangle amplitude is equal to 7 0011: Unmask LSFR bit[3: 0] /Triangle amplitude is equal to 15 0100: Unmask LSFR bit[4: 0] /Triangle amplitude is equal to 31 0101: Unmask LSFR bit[5: 0] /Triangle amplitude is equal to 63 0110: Unmask LSFR bit[6: 0] /Triangle amplitude is equal to 127 0111: Unmask LSFR bit[7: 0] /Triangle amplitude is equal to 255 1000: Unmask LSFR bit[8: 0] /Triangle amplitude is equal to 511 1001: Unmask LSFR bit[9: 0] /Triangle amplitude is equal to 1023 1010: Unmask LSFR bit[10:0] /Triangle amplitude is equal to 2047 ≥1011: Unmask LSFR bit[11: 0] /Triangle amplitude is equal to 4095
Bit 23: 22	D2NM	0x0	rw	DAC2 noise mode 00: Wave generation disabled 01: Noise wave generation enabled 1x: Triangular wave generation enabled
Bit 21: 19	D2TRGSEL	0x0	rw	DAC2 trigger select 000: TMR6 TRGOUT event 001: TMR3 TRGOUT event 010: TMR7 TRGOUT event 011: TMR9 TRGOUT event 100: TMR2 TRGOUT event 101: TMR4 TRGOUT event 110: External interrupt line 9 111: Software trigger Note: These bits can be valid only when D2TRGEN = 1.
Bit 18	D2TRGEN	0x0	rw	DAC2 trigger enable 0: DAC2 trigger disabled 1: DAC2 trigger enabled Note: When the DAC2 trigger is disabled, the data written into the DAC_D2DTHx register is transferred into the DAC_D2ODT register after one APB1 clock cycle. When the DAC2 trigger is enabled, the data written into the DAC_D2DTHx register is transferred into the DAC_ D2ODT register after three APB1 clock cycles. If the software trigger is selected, it takes one APB1 clock cycle to have the data written into the DAC_D2DTHx register transferred into the DAC_D2ODT register.
Bit 17	D2OBDIS	0x0	rw	DAC2 output buffer disable 0: DAC2 output buffer enabled 1: DAC2 output buffer disabled
Bit 16	D2EN	0x0	rw	DAC2 enable 0: DAC2 disabled 1: DAC2 enabled
Bit 15: 14	Reserved	0x0	resd	Kept at default value
Bit 13	D1DMAUDRIEN	0x0	rw	DAC1 DMA transfer underrun interrupt enable This bit is set and cleared by software. 0: DAC1 DMA transfer underrun interrupt disabled 1: DAC1 DMA transfer underrun interrupt enabled
Bit 12	D1DMAEN	0x0	rw	DAC1 DMA transfer enable 0: DAC1 DMA transfer disabled

				1: DAC1 DMA transfer enabled
Bit 11: 8	D1NBSEL	0x0	rw	DAC1 noise bit select These bits are used to select the mark bit in noise generation mode or amplitude in triangular-wave generation mode. 0000: Unmask LSFR bit0/Triangle amplitude is equal to 1 0001: Unmask LSFR bit[1:0]/Triangle amplitude is equal to 3 0010: Unmask LSFR bit[2: 0]/Triangle amplitude is equal to 7 0011: Unmask LSFR bit[3: 0]/Triangle amplitude is equal to 15 0100: Unmask LSFR bit[4: 0]/Triangle amplitude is equal to 31 0101: Unmask LSFR bit[5: 0]/Triangle amplitude is equal to 63 0110: Unmask LSFR bit[6: 0]/Triangle amplitude is equal to 127 0111: Unmask LSFR bit[7: 0]/Triangle amplitude is equal to 255 1000: Unmask LSFR bit[8: 0]/Triangle amplitude is equal to 511 1001: Unmask LSFR bit[9: 0]/Triangle amplitude is equal to 1023 1010: Unmask LSFR bit[10: 0]/Triangle amplitude is equal to 2047 ≥1011: Unmask LSFR bit[11:0]/Triangle amplitude is equal to 4095
Bit 7: 6	D1NM	0x0	rw	DAC1 noise mode 00: Wave generation disabled 01: Noise wave generation enabled 1x: Triangular wave generation enabled
Bit 5: 3	D1TRGSEL	0x0	rw	DAC1 trigger select 000: TMR6 TRGOUT event 001: TMR3 TRGOUT event 010: TMR7 TRGOUT event 011: TMR9 TRGOUT event 100: TMR2 TRGOUT event 101: TMR4 TRGOUT event 110: External interrupt line 9 111: Software trigger Note: These bits can be valid only when D1TRGEN = 1.
Bit 2	D1TRGEN	0x0	rw	DAC1 trigger enable 0: DAC1 trigger disabled 1: DAC1 trigger enabled Note: When the DAC1 trigger is disabled, the data written into the DAC_D1DTHx register is transferred into the DAC_D1ODT register after one APB1 clock cycle. When the DAC1 trigger is enabled, the data written into the DAC_D1DTHx register is transferred into the DAC_ D1ODT register after three APB1 clock cycles If the software trigger is selected, it takes one APB1 clock cycle to have the data written into the DAC_D1DTHx register transferred into the DAC_ D1ODT register.
Bit 1	D1OBDIS	0x0	rw	DAC1 output buffer disable 0: DAC1 output buffer enabled 1: DAC1 output buffer disabled
Bit 0	D1EN	0x0	rw	DAC1 enable 0: DAC1 disabled 1: DAC1 enabled

19.5.2 DAC software trigger register (DAC_SWTRG)

Bit	Register	Reset value	Type	Description
Bit 31: 2	Reserved	0x0000 0000	resd	Kept at default value
Bit 1	D2SWTRG	0x0	rw	DAC2 software trigger 0: DAC2 software trigger disabled 1: DAC2 software trigger enabled Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC_D2DTH data is loaded into the DAC_D2ODT register.
Bit 0	D1SWTRG	0x0	rw	DAC1 software trigger 0: DAC1 software trigger disabled 1: DAC1 software trigger enabled Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC_D1DTH data is loaded into the DAC_D1ODT register.

19.5.3 DAC1 12-bit right-aligned data holding register (DAC_D1DTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at default value
Bit 11: 0	D1DT12R	0x000	rw	DAC1 12-bit right-aligned data

19.5.4 DAC1 12-bit left-aligned data holding register (DAC_D1DTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at default value
Bit 15: 4	D1DT12L	0x000	rw	DAC1 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at default value

19.5.5 DAC1 8-bit right-aligned data holding register (DAC_D1DTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at default value
Bit 7: 0	D1DT8R	0x00	rw	DAC1 8-bit right-aligned data

19.5.6 DAC2 12-bit right-aligned data holding register (DAC_D2DTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at default value
Bit 11: 0	D2DT12R	0x000	rw	DAC2 12-bit right-aligned data

19.5.7 DAC2 12-bit left-aligned data holding register (DAC_D2DTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at default value
Bit 15: 4	D2DT12L	0x000	rw	DAC2 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at default value

19.5.8 DAC2 8-bit right-aligned data holding register (DAC_D2DTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 8	Reserved	0x000000	resd	Kept at default value
Bit 7: 0	D2DT8R	0x00	rw	DAC2 8-bit right-aligned data

19.5.9 Dual DAC 12-bit right-aligned data holding register (DAC_DDTH12R)

Bit	Register	Reset value	Type	Description
Bit 31: 28	Reserved	0x0	resd	Kept at default value
Bit 27: 16	DD2DT12R	0x000	rw	DAC2 12-bit right-aligned data
Bit 15: 12	Reserved	0x0	resd	Kept at default value
Bit 11: 0	DD1DT12R	0x000	rw	DAC1 12-bit right-aligned data

19.5.10 Dual DAC 12-bit left-aligned data holding register (DAC_DDTH12L)

Bit	Register	Reset value	Type	Description
Bit 31: 20	DD2DT12L	0x000	rw	DAC2 12-bit left-aligned data
Bit 19: 16	Reserved	0x0	resd	Kept at default value
Bit 15: 4	DD1DT12L	0x000	rw	DAC1 12-bit left-aligned data
Bit 3: 0	Reserved	0x0	resd	Kept at default value

19.5.11 Dual DAC 8-bit right-aligned data holding register (DAC_DDTH8R)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at default value
Bit 15: 8	DD2DT8R	0x00	rw	DAC2 8-bit right-aligned data
Bit 7: 0	DD1DT8R	0x00	rw	DAC1 8-bit right-aligned data

19.5.12 DAC1 data output register (DAC_D1ODT)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at default value
Bit 11: 0	D1ODT	0x000	rw	DAC1 output data

19.5.13 DAC2 data output register (DAC_D2ODT)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at default value
Bit 11: 0	D2ODT	0x000	rw	DAC2 output data

19.5.14 DAC status register (DAC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at default value
Bit 29	D2DMAUDRF	0x0	w1c	DAC2 DMA transfer underrun flag 0: No DAC2 DMA transfer underrun 1: DAC2 DMA transfer underrun occurs Note: This bit is cleared by writing 1.
Bit 28: 14	Reserved	0x0000	resd	Kept at default value
Bit 13	D1DMAUDRF	0x0	w1c	DAC1 DMA transfer underrun flag 0: No DAC1 DMA transfer underrun 1: DAC1 DMA transfer underrun occurs Note: This bit is cleared by writing 1.
Bit 12: 0	Reserved	0x0000	resd	Kept at default value

20 Controller area network (CAN)

20.1 CAN introduction

CAN (Controller Area Network) is a serial communication protocol for real-time and reliable data communication among nodes. It supports the CAN protocol version 2.0A and 2.0B.

20.2 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

20.3 Baud rate

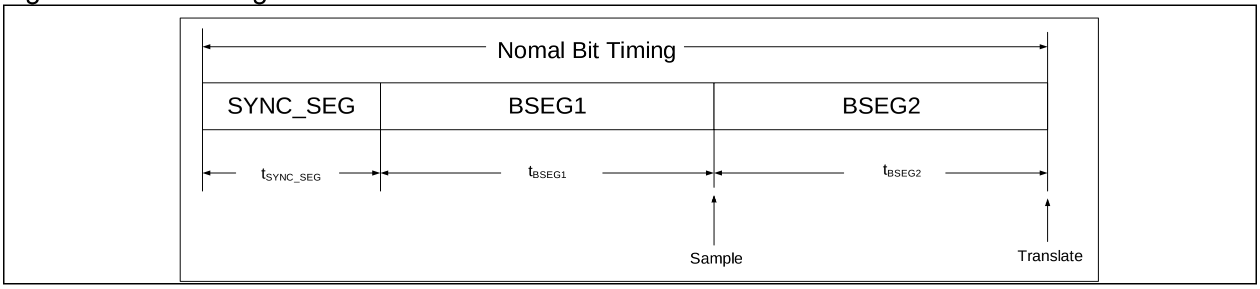
The nominal bit time of the CAN bus consists of three parts as follows:

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 configurable between 1 and 16 time units, depending on 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 configurable between 1 and 8 time units, depending on the BTS2[2: 0] bit.

Figure 20-1 Bit timing



Baud rate formula:

$$BaudRate = \frac{1}{\mathrm{Nomal\ \ Bit\ \ Timimg}}$$

$$Nomal\ \ Bit\ \ Timing = \mathrm{t}_{SYNC\_SEG} + \ \mathrm{t}_{BSEG1} + \ \mathrm{t}_{BSEG2}$$

with

$$\mathrm{t}_{SYNC\_SEG} = 1\ \ \mathrm{x}\ \ \mathrm{t}_q$$

$$\mathrm{t}_{BSEG1} = (1 + \mathrm{BTS1}[3:\ \ 0])\ \ \mathrm{x}\ \ \mathrm{t}_q$$

$$\mathrm{t}_{BSEG2} = (1 + \mathrm{BTS2}[2:\ \ 0])\ \ \mathrm{x}\ \ \mathrm{t}_q$$

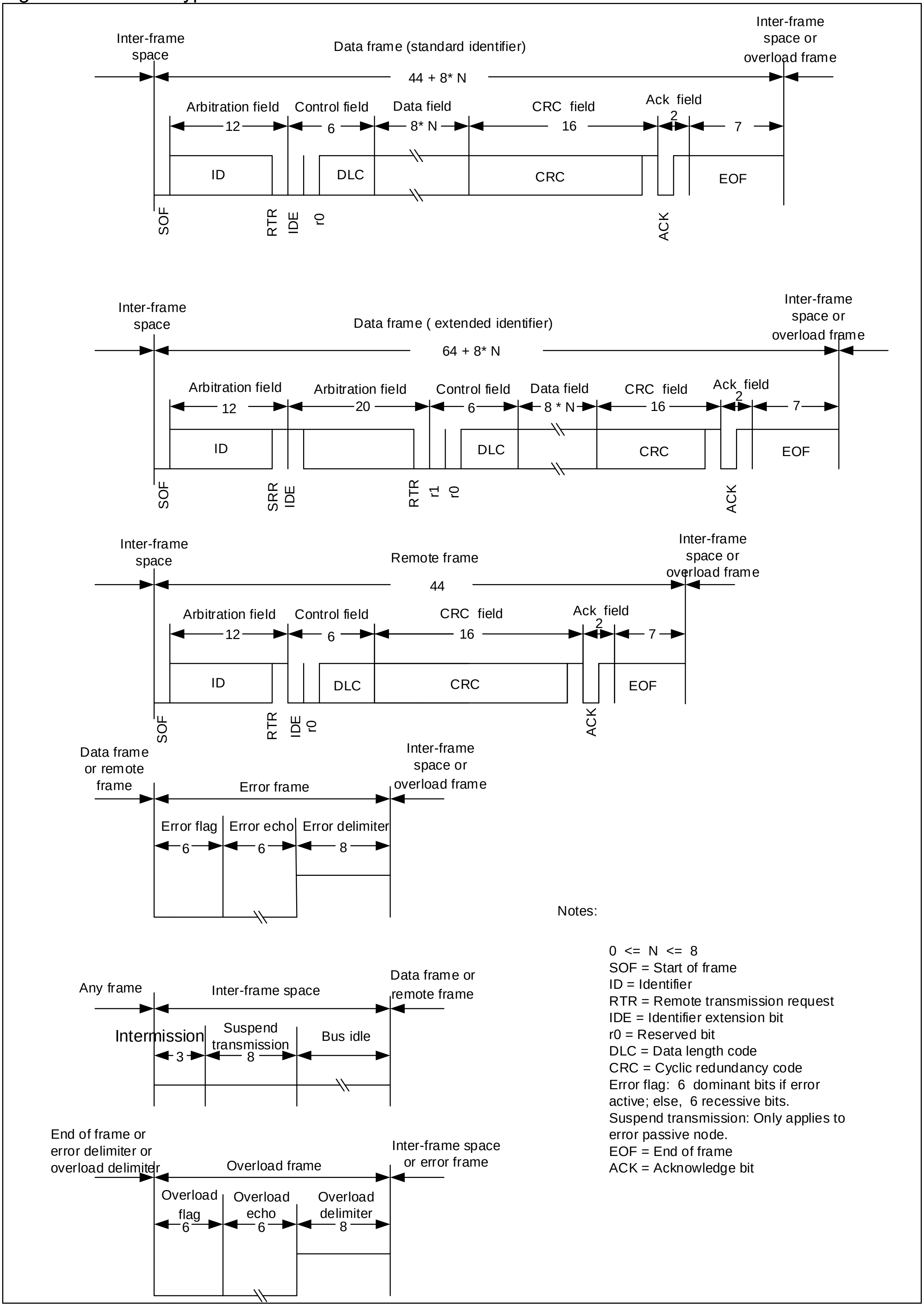
$$\mathrm{t}_q = (1 + \mathrm{BRDIV}[11:\ \ 0])\ \ \mathrm{x}\ \ \mathrm{t}_{pclk}$$

Hard synchronization and resynchronization

The start location of each bit in CAN nodes is always in synchronization segment by default. The sampling is performed at the edge location of bit segment 1 and big segment 2 simultaneously.

During the actual transmission, there are certain phase drifts among the bits of CAN nodes due to the oscillator drifts of nodes, transmission latency and noise interference. Therefore it is necessary to avoid the impact on the communication by synchronizing on the start-bit edge and resynchronizing the following falling edges. The length of synchronization to compensate for phase drifts must not be greater than the resynchronization adjustment width (it is programmable between 1 and 4 time units through the RSAW[1:0] bit).

Figure 20-2 Frame type



20.4 Interrupt management

The CAN controller has four interrupt vectors that can be used to enable or disable interrupts by setting the CAN_INTEN register.

Figure 20-3 Transmit interrupt generation

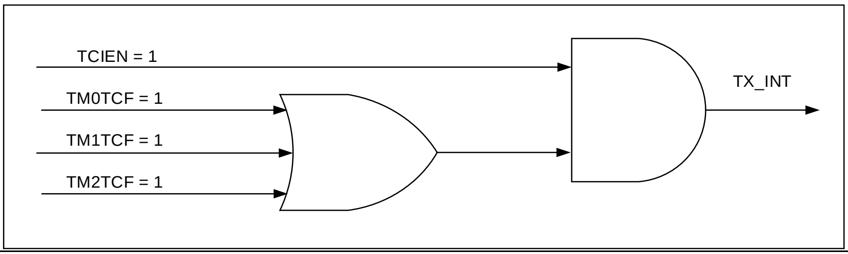


Figure 20-4 Receive interrupt 0 generation

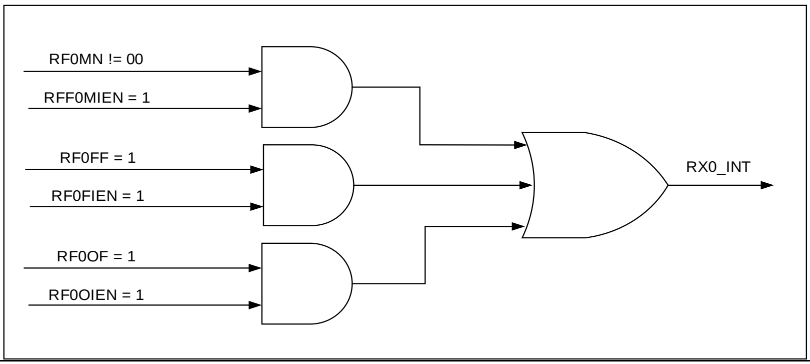


Figure 20-5 Receive interrupt 1 generation

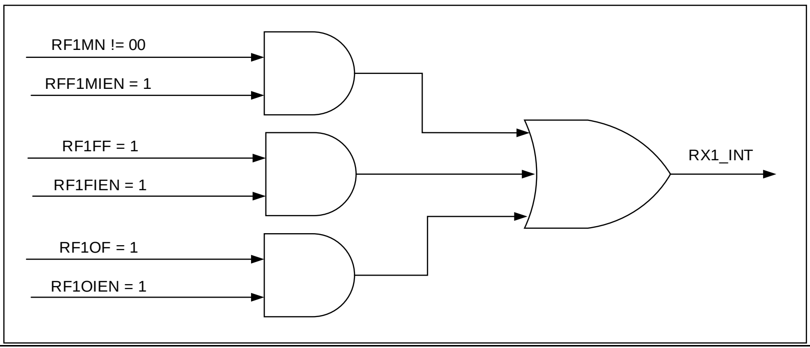
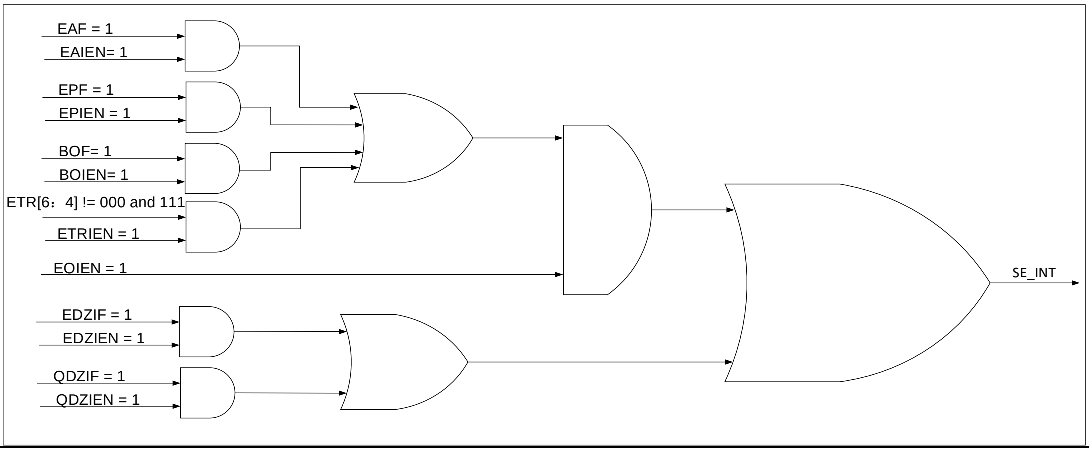


Figure 20-6 Status error interrupt generation



20.5 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 at each CAN bit time, and it 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 (filter mode configuration register), CAN_FBWCFG (filter bit width configuration register) and CAN_FRF (filter bit register) registers can be modified only when FCS=1. The CAN_FiFBx register can be modified only when FCS=1 (in filter configuration mode) or FAENx=0 (filter is disabled).

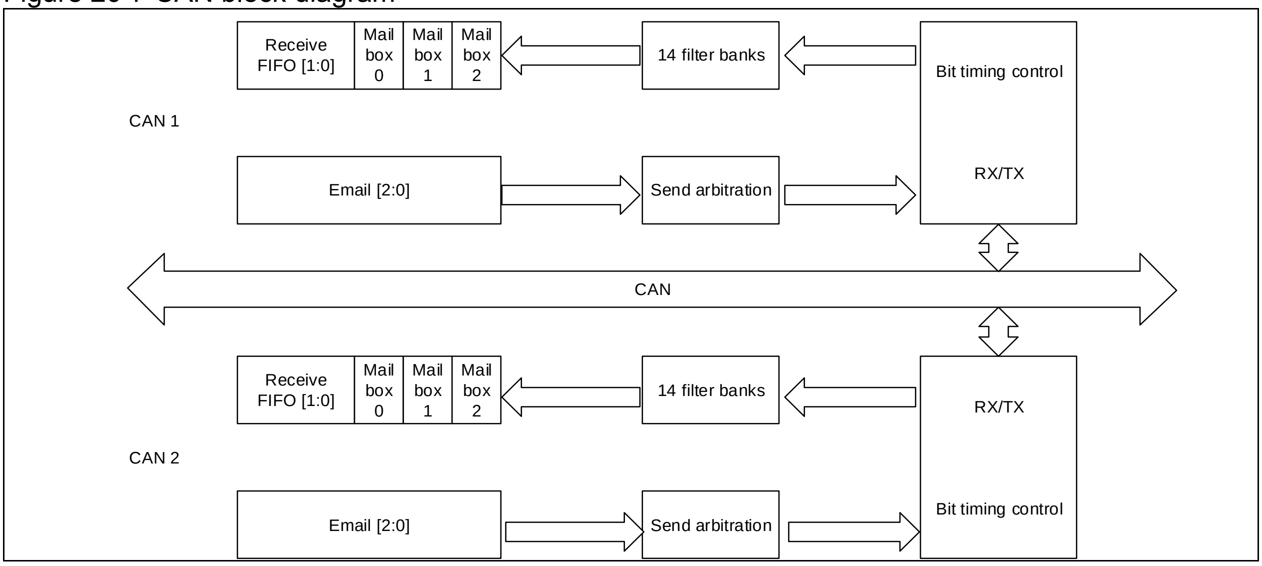
20.6 Functional overview

20.6.1 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 conditions above, the CAN controller provides 14 scalable/configurable identifier filter banks, 2 receive FIFOs capable of 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 20-7 CAN block diagram



20.6.2 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.

  Sleep mode 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.

  Sleep mode 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 (CAN bit timing register) and CAN_MCTRL (CAN master control register) 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.

  Frozen mode 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.

  Frozen mode 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.

  Communication mode 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.

  Communication mode 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.

20.6.3 Test modes

The CAN controller defines three test modes, including Listen-only mode, Loop back mode and Listen-only combined with 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 it sends only recessive bits on the CANTX pin. In the meantime, the dominant bits on the CANTX can be detected 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 of its own node. Meanwhile, the CAN can send data to the external bus. The Loop back mode is mainly used for self-test functions.
- **Loop back mode combined with Listen-only mode:** It is possible to combine the Listen-only and Loop back mode by setting [31:30] bits=11 in the CAN_BTMG register. In this mode, 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.

20.6.4 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 20-8 32-bit identifier mask mode

ID	CAN_FiFB1[31:21]	CAN_FiFB1[20:3]	CAN_FiFB1 [2:0]		
Mask	CAN_FiFB2[31:21]	CAN_FiFB2[20:3]	CAN_FiFB2 [2:0]		
Mapping	SID[10:0]	EID[17:0]	IDT	RTR	0

Figure 20-9 32-bit identifier list mode

ID	CAN_FiFB1[31:21]	CAN_FiFB1[20:3]	CAN_FiFB1 [2:0]		
ID	CAN_FiFB2[31:21]	CAN_FiFB2[20:3]	CAN_FiFB2 [2:0]		
Mapping	SID[10:0]	EID[17:0]	IDT	RTR	0

Figure 20-10 16-bit identifier mask mode

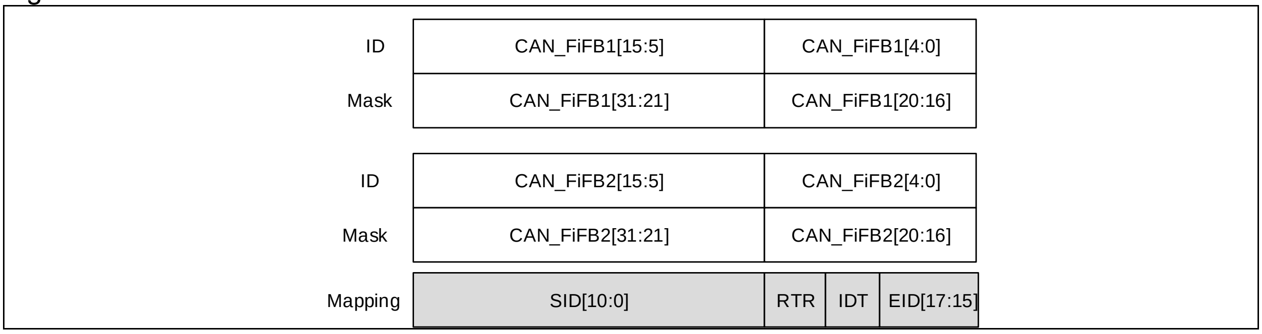
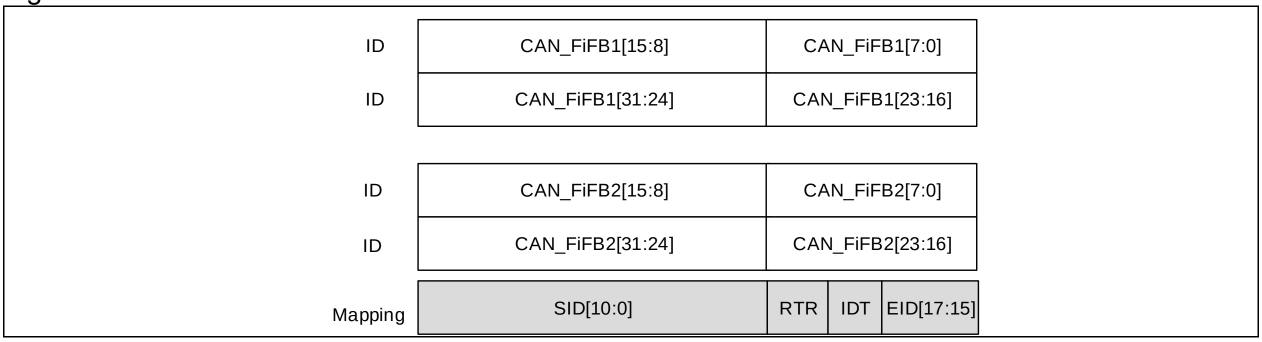


Figure 20-11 16-bit identifier list mode



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 filter number N, 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

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_F10FB2[31:16]-Mask				CAN_F11FB2[31:0]-ID		13
12	CAN_F12FB1[15:0]-ID	No	13	13	CAN_F13FB1[15:0]-ID	Yes	14
	CAN_F12FB1[31:16]-ID		14		CAN_F13FB1[31:16]-ID		15
	CAN_F12FB2[15:0]-ID		15		CAN_F13FB2[15:0]-ID		16
	CAN_F12FB2[31:16]-ID		16		CAN_F13FB2[31:16]-ID		17

Priority rules

It may occur that CAN controller receives a frame of message that pass through several filters successfully. 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 equal bit width, the identifier list mode has priority over the identifier mask mode
- For filter with equal bit width and identifier mode, the lower number has priority over the higher number.

Filter configuration

- Configure the CAN filters by setting the FCS bit in the CAN_FCTRL register.
- Select Identifier mask mode or identifier list mode by setting the FMSELx bit in the CAN_FMCFG register.
- Configure the filter bit width as two 16 bits or one 32 bits by setting the FBWSELx bit in the CAN_FBWCFG register.
- Associate the filter x with FIFO0 or FIFO1 by setting the FRFSELx bit in the CAN_FRF register.
- Activate the filter banks x by setting FAENx=1 in the CAN_FACFG register.
- Configure 0~13 filter banks by writing to the CAN_FiFBx register (i=0…27; x=1,2).
- Complete the CAN filter configuration by setting FCS=0 in the CAN_FCTRL register.

20.6.5 Message transmission

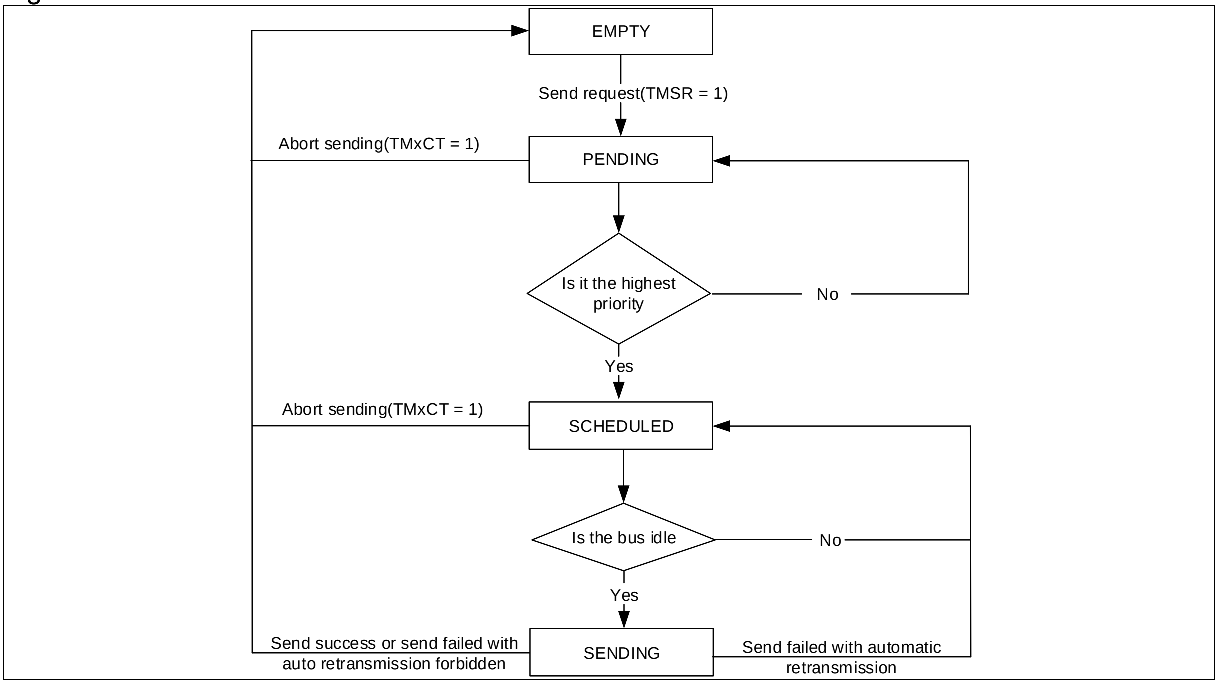
Register configuration

To transmit a message, the application must select one transmit mailbox and configure 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 into pending state immediately after the mailbox is configured and the CAN controller receives a 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 into 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 20-12 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.

20.6.6 Message reception

Register configuration

The CAN_RFIx (receive FIFO mailbox identifier register), CAN_RFCx (receive FIFO mailbox data length and time stamp register), CAN_RFDTLx (receive FIFO mailbox data register low) and CAN_RFDTHx (receive FIFO mailbox data register high) registers can be used by user applications to obtain valid messages.

Message reception

The CAN controller has 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 considered 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

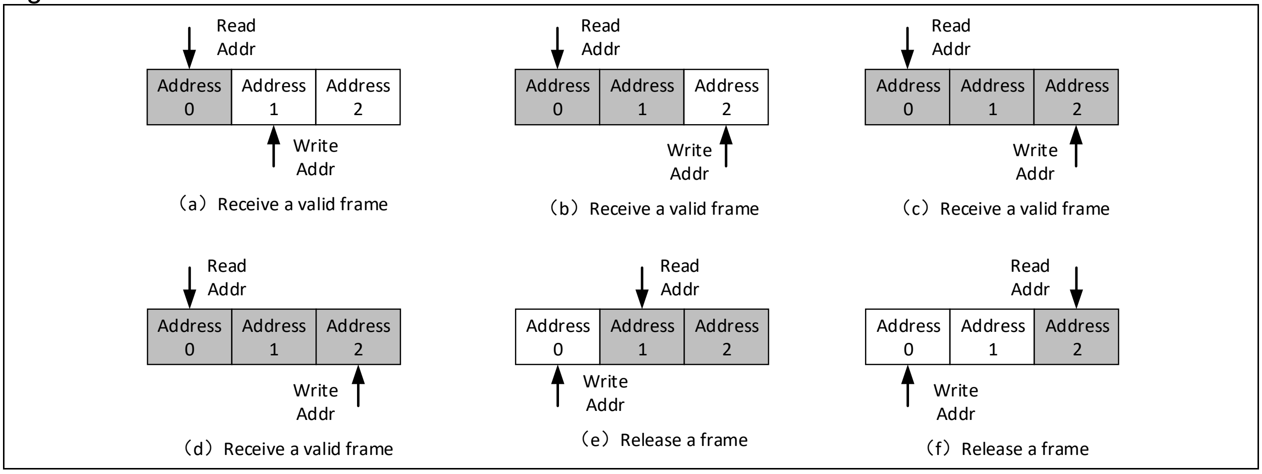
RFxMN[1: 0], RFxFF and RFxOF bits in the RFx register are used to indicate the status of receive FIFO.

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 20-13*.

RFxOF: indicates that a new valid message has been received while the FIFOx is full, as shown in (d) of *Figure 20-13*.

Figure 20-13 Receive FIFO status



20.6.7 Error management

The status of the current CAN node is indicated by the receive error counter (TEC) and transmit error counter (REC) bits in the CAN_ESTS register. The ETR[6: 4] 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 bus-off state, the node is not able to transmit and receive messages. The CAN recovers from bus-off state in two ways:

Option 1: When AEBOEN=0 in the CAN_MCTRL register, in communication mode, the CAN will recover from bus-off state when the 128 occurrence of 11 consecutive recessive bits are detected on the CAN RX pin, and the software requests to enter Frozen mode and exit Frozen mode.

Option 2:

When AEBOEN=1 in the CAN_MCTRL register, in communication mode, 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.

20.7 CAN registers

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

Table 20-1 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

20.7.1 CAN control and status registers

20.7.1.1 CAN master control register (CAN_MCTRL)

Bit	Register	Reset value	Type	Description
Bit 31: 17	Reserved	0x0000	resd	Kept at 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 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.

20.7.1.2 CAN master status register (CAN_MSTS)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at 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 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 set, 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.

20.7.1.3 CAN 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	ro	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 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 TSMF2, ALMF2 and TEMF2 bits of mailbox 2 by hardware
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 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 TSMF1, ALMF1 and TEMF1 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 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 TSMF0, ALMF0 and TEMF0 bits of mailbox 0.

20.7.1.4 CAN receive FIFO 0 register (CAN_RF0)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at 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 there are three messages pending in the FIFO 0. It is cleared by software by writing 1.
Bit 2	Reserved	0x0	resd	Kept at 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.

20.7.1.5 CAN receive FIFO 1 register (CAN_RF1)

Bit	Register	Reset value	Type	Description
Bit 31: 6	Reserved	0x0000000	resd	Kept at 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 receive 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 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.

20.7.1.6 CAN interrupt enable register (CAN_INTEN)

Bit	Register	Reset value	Type	Description
Bit 31: 18	Reserved	0x0000	resd	Kept at 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 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 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

20.7.1.7 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 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 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)

20.7.1.8 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 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 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 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).

20.7.2 CAN mailbox registers

This section describes the registers of the transmit and receive mailboxes. Refer to *Section 20.6.5* for more information on register map.

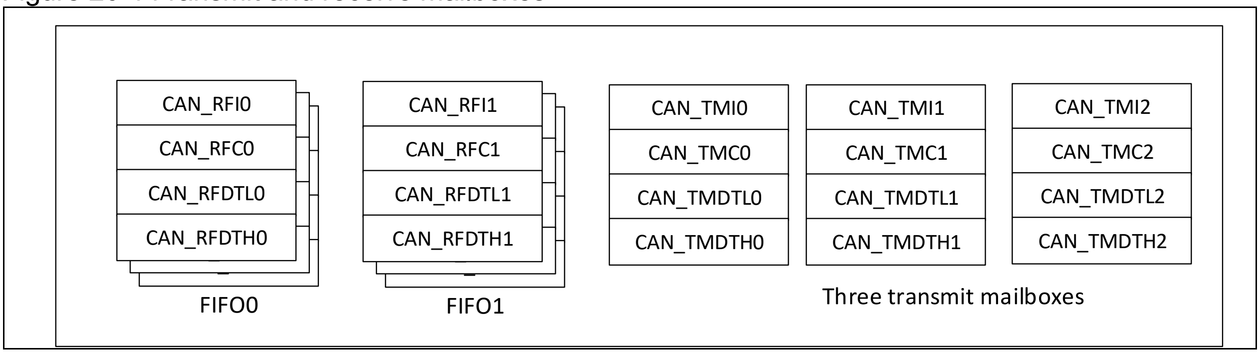
Transmit and receive mailboxes are the same except:

- RFFMN field in the CAN_RFCx register
- A receive mailbox is read only
- A transmit mailbox can be written only when empty. TM2S=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 20-14 ransmit and receive mailboxes



20.7.2.1 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)

20.7.2.2 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 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 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.

20.7.2.3 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

20.7.2.4 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

20.7.2.5 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 default value

20.7.2.6 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 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.

20.7.2.7 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

20.7.2.8 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

20.7.3 CAN filter registers

20.7.3.1 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.

20.7.3.2 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 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

20.7.3.3 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 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

20.7.3.4 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 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

20.7.3.5 CAN filter activation control register (CAN_ FACFG)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Kept at default value
Bit 13: 0	FAENx	0x0000	rw	Filter active enable Each bit corresponds to a filter bank. 0: Disabled 1: Enabled

20.7.3.6 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	0x0000 0000	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 being 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.

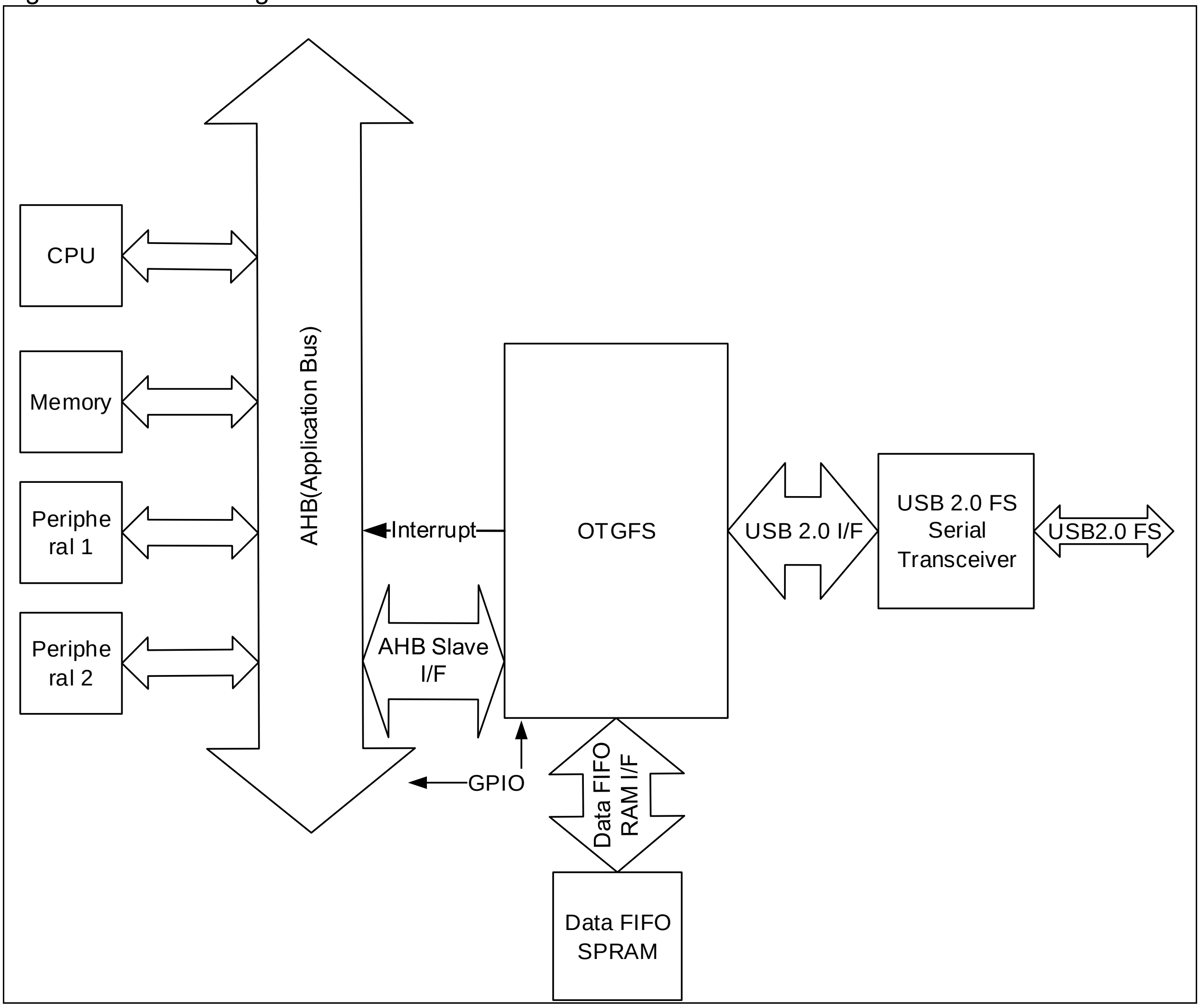
21 Universal serial bus full-seed device interface (OTGFS)

As a full-speed dual-role device, the OTGFS is fully compliant with the Universal Serial Bus Specification Revision2.0.

21.1 OTGFS structure

Figure 21-1 hows the block diagram of the OTGFS structure. The OTGFS module is connected to the AHB and has a dedicated SRAM of 1280 bytes.

Figure 21-1 Block diagram of OTGFS structure



21.2 OTGFS functional description

The OTGFS module consists of an OTGFS controller, built-inPHY and a dedicated 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. The status of the ID line determines whether the OTGFS works as a host or device. When the ID line is floating, the OTGFS works as a device. It is used as a host while the ID line is grounded. 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, 7 IN endpoints, and 7 OUT endpoints; in hose mode, the OTGFS supports 16 host channels.

The OTGFS supports SOF pulse and OE pulse functions: a SOF pulse generates at a SOF packet, the pulse is output on PIN and the timer 2; an OE pulse generates when the OTGFS outputs data, the pulse is output on PIN.

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. It enters this mode if a bus signal is not received within three minutes after the STOPPCLK bit is set in the OTGFS_PCGCCTL register; besides, the PHY reception can be disabled by setting the LP_MODE bit in the OTGS_GCCFG register in order to reduce power consumption.

21.3 OTGFS clock and pin configuration

21.3.1 OTGFS clock configuration

The OTGFS interface has two clocks: USB control clock and AHB bus clock. The USB full-speed device bus speed standard is 12Mb/s±0.25%. Therefore, the 48MHz±0.25% has to be provided to the OTGFS to perform USB bus sampling.

USBFS 48M clock has two sources:

- **HICK 48M**

  When the HICK 48M clock is used as an USBI clock, it is recommended to enable ACC feature.
- **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 30MHz when OTGFS is enabled.

21.3.2 OTGFS pin configuration

The OTGFS input/output pins are multiplexed with GPIOs. The GPIOs are used as OTGFS in one of the following conditions:

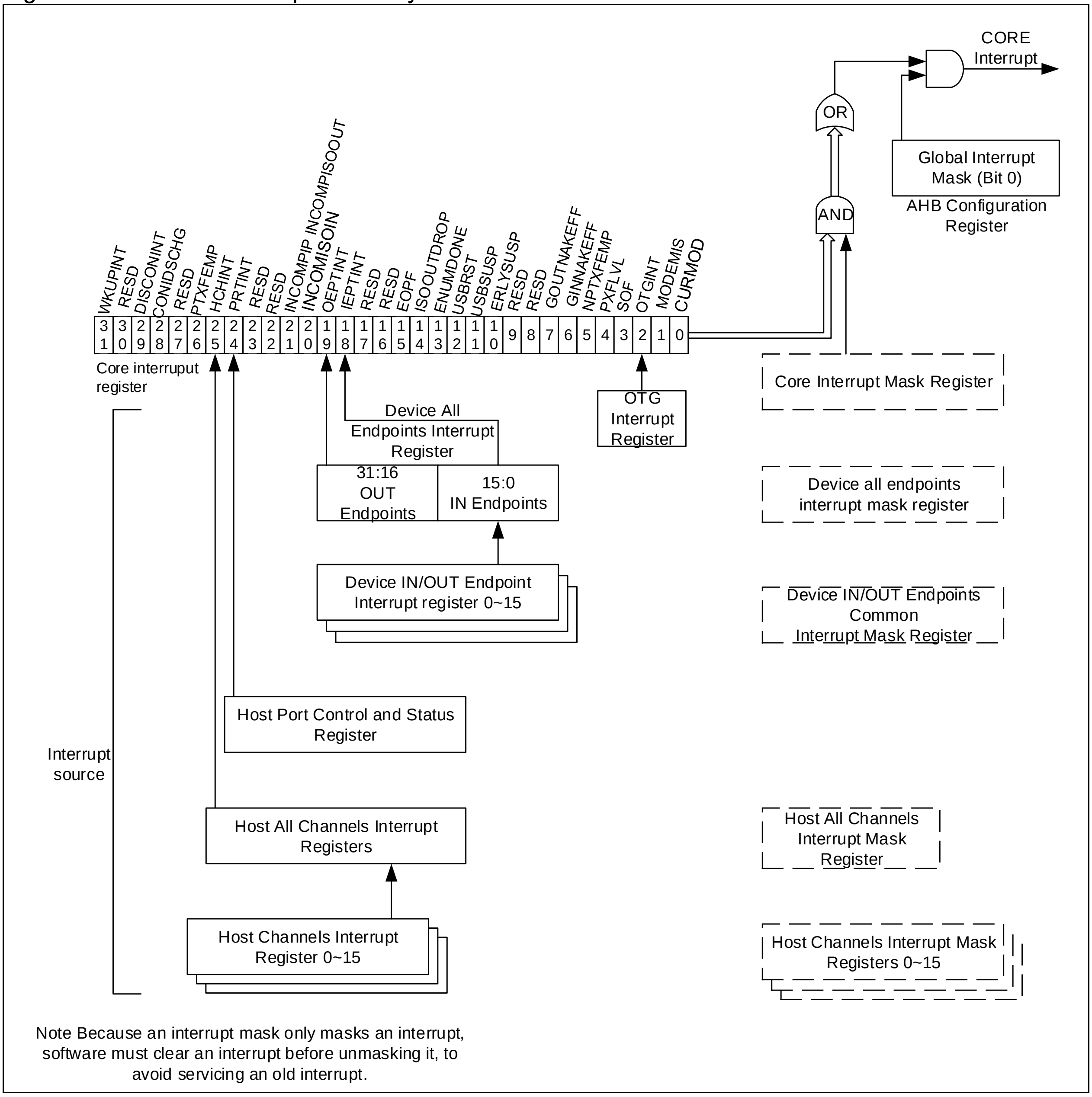
Table 21-1 OTGFS input/output pins

Pin	GPIO	Description
OTGFS_SOF	PA8	Enable OTG in CRM, and configure PA8 multiplexed function register as 0x3
OTGFS_VBUS	PA9	Configure PA9 as multiplexed function mode and PA9 multiplexed function register as 0xA
OTGFS_ID	PA10	Enable OTG in CRM, configure PA10 as multiplexed function mode and PA10 multiplexed function register as 0xA
OTGFS_D-	PA11	Enable OTG in CRM, and PWRDOWN=1
OTGFS_D+	PA12	Enable OTG in CRM, and PWRDOWN=1
OTGFS_OE	PA13	Enable OTG in CRM, and configure PA13 multiplexed function register as 0xA
	PC9	Enable OTG in CRM, and configure PC9 multiplexed function register as 0xA

21.4 OTGFS interrupts

Figure 21-2 shows the OTGFS interrupt hierarchy. Refer to the OTGFS interrupt register (OTGFS_GINTSTS) and OTGFS interrupt mask register (OTGFS_GINTMSK).

Figure 21-2 OTGFS interrupt hierarchy



21.5 OTGFS functional description

21.5.1 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 operating mode. When A-type plug is connected, the OTGFS controller works in host mode; when B-type plug is connection, the OTGFS controller works in device mode.

This section explains the initialization of the OTGFS controller after power-on. The application must follow the initialization sequence, whatever mode (in host or device mode). All controller global registers are initialized according to the controller configuration.

1. Configure the following fields in the AHB global configuration register:

- Global interrupt mask bit = 0x1

- Non-periodic transmit FIFO empty level
- Periodic transmit FIFO empty level

2. Configure the following fields in the AHB global configuration register:

- OTGFS_GINTMSK.RXFLVLMSK = 0x0

3. Configure 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 in host or device mode.

21.5.2 OTGFS FIFO configuration

21.5.2.1 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 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 multiple synchronous endpoints are enabled, at least two (largest packet size/4) + 1 must be allocated to receive back-to-back 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 21-2 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.

21.5.2.2 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 run. 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 DWORDs 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 number of 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 21-3 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.

21.5.2.3 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.

21.5.3 OTGFS host mode

21.5.3.1 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 to select full-speed or high-speed host mode
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.

21.5.3.2 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)

21.5.3.3 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-periodic 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

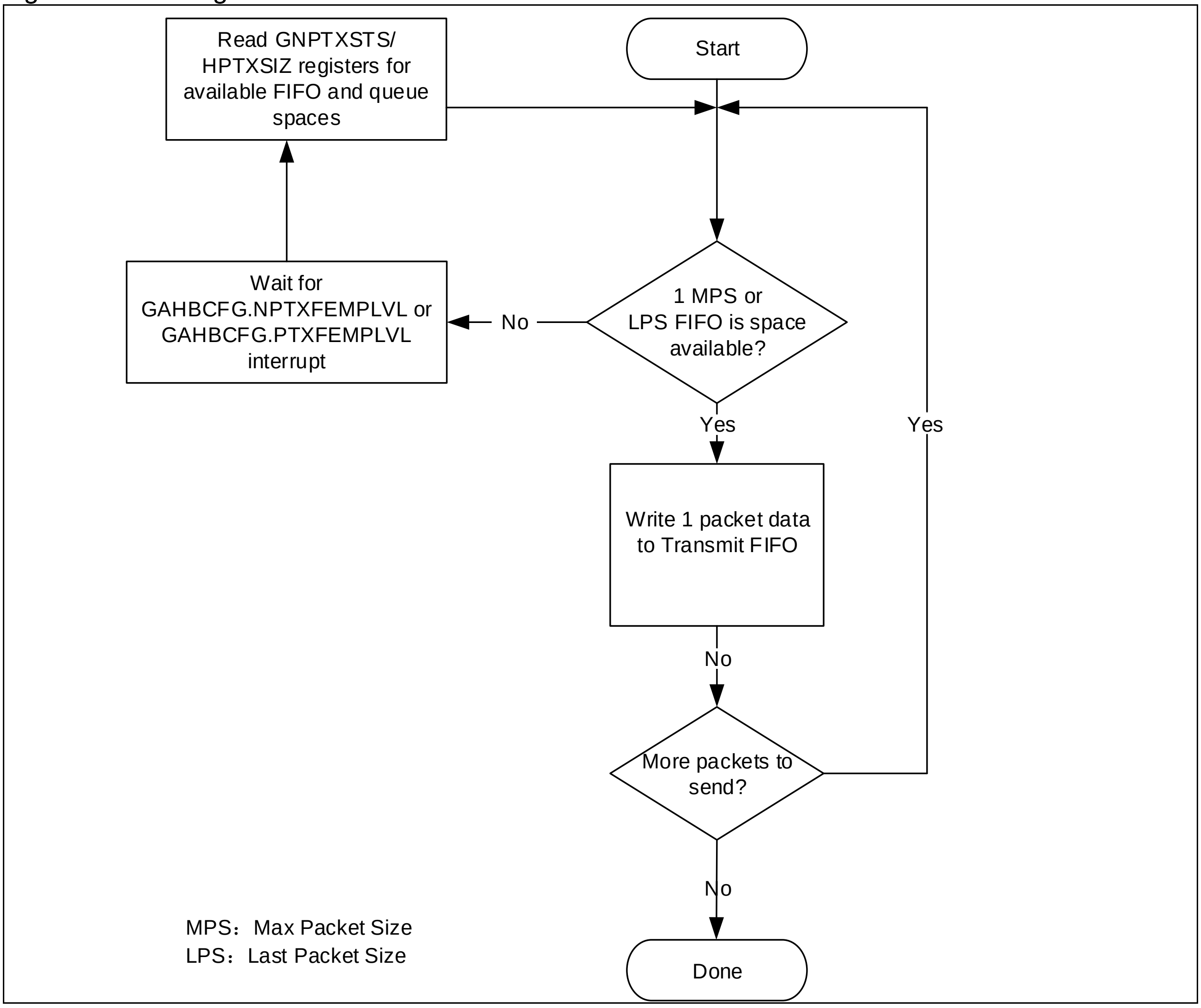
21.5.3.4 Queue depth

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 21-3 shows the flow chart of writing the transmit FIFO. The OTGFS host automatically writes an entry (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.

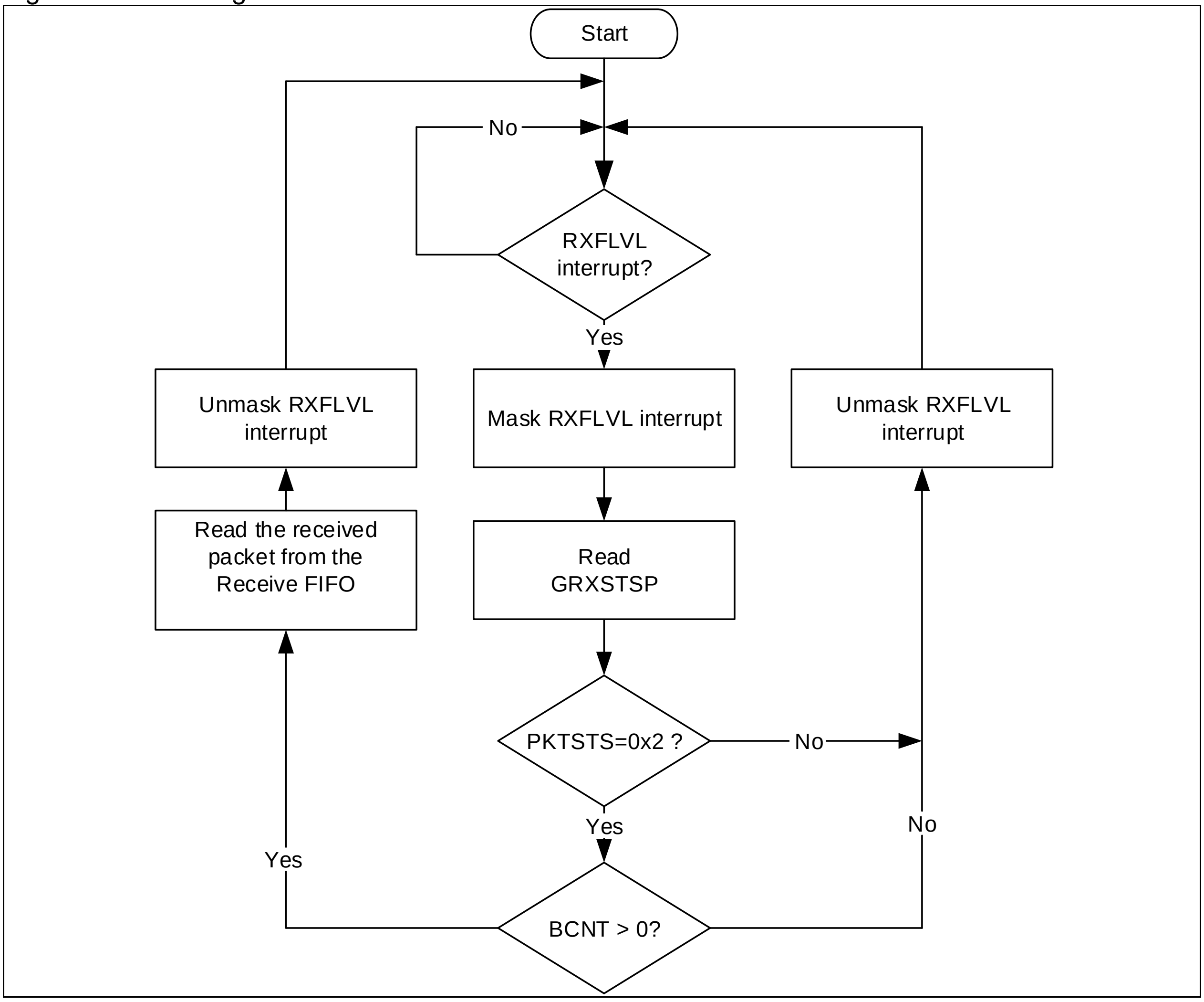
Figure 21-3 Writing the transmit FIFO



- Reading the receive FIFO

Figure 21-4 shows the flow chart of reading the receive FIFO. The application must ignore all packet statuses other than IN data packet (0x0010)

Figure 21-4 Reading the receive FIFO



21.5.3.5 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

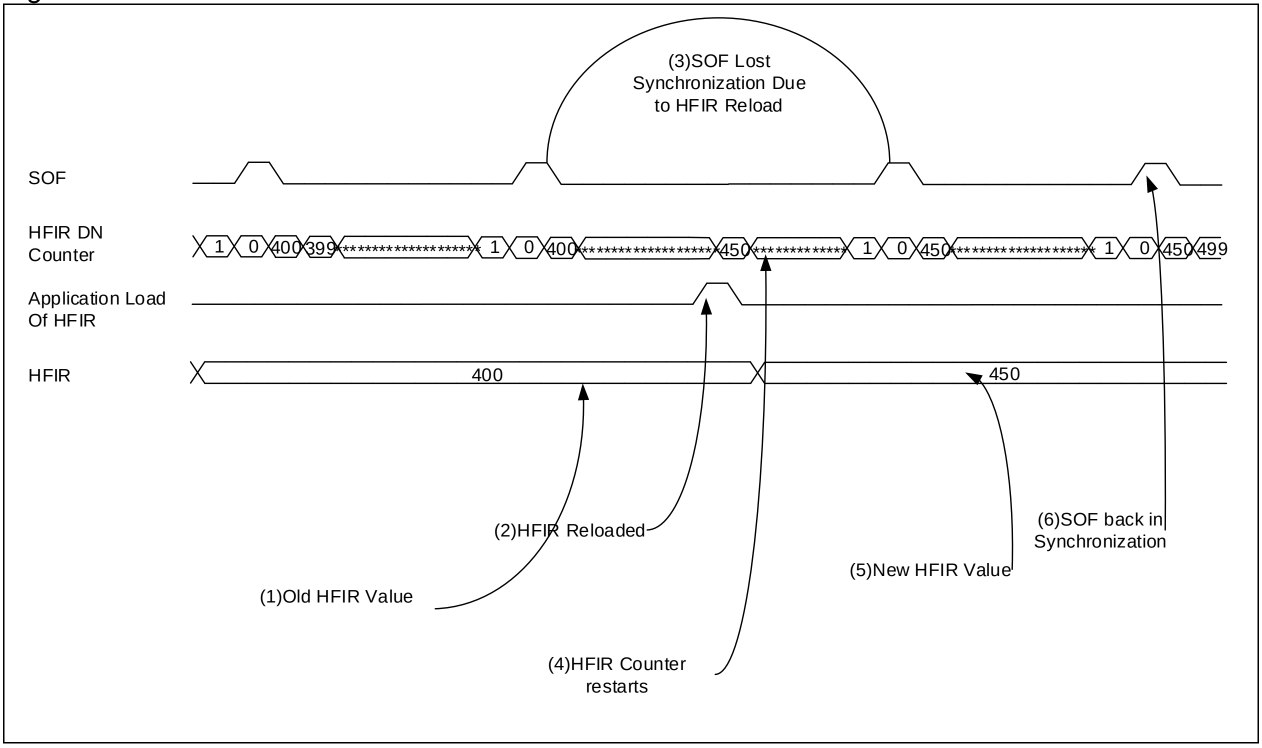
If the device is suddenly disconnected, an interrupt is generated on a disconnect event (DISCONINT bit in the OTGFS_GINTSTS register). Upon receiving this interrupt, the application must start a software reset through the CSFTRST in the OTGFS_GRSTCTL register.

21.5.3.6 Host HFIR feature

The host frame interval register (HFIR) defines the 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 21-5 shows the HFIR behavior when the HFIRRLDCTRL is set to 0x0 in the OTGFS_HFIR register.

Figure 21-5 HFIR behavior when HFIRRLDCTRL=0x0

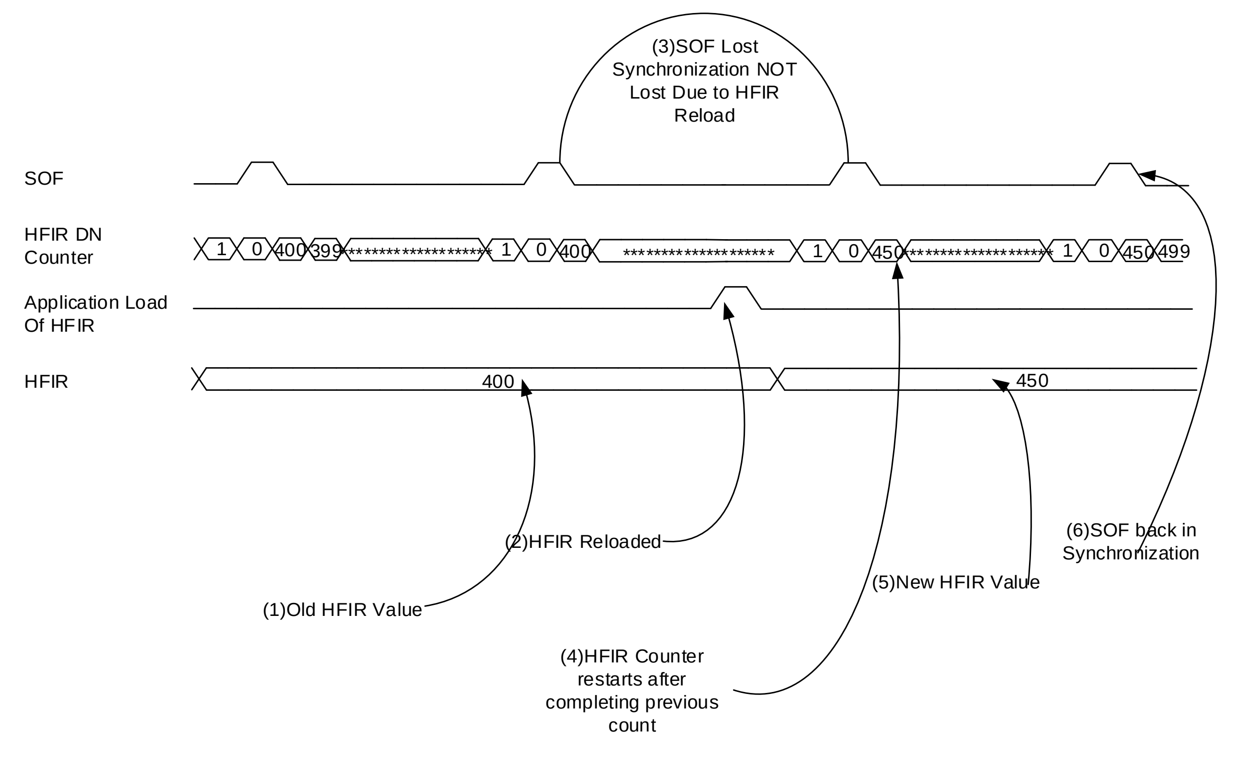


The sequence of operation is as follows:

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 21-6 shows the HFIR behavior when HFIRRLDCTRL=0x1 in the OTGFS_HFIR register.

Figure 21-6 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.

21.5.3.7 Initialize bulk and control IN transfers

Figure 21-7 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 21-7 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 the XFERC interrupt as soon as the receive packet is read
9. To handle the 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 the 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
}
```

21.5.3.8 Initialize bulk and control OUT/SETUP transfers

Figure 21-7 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:

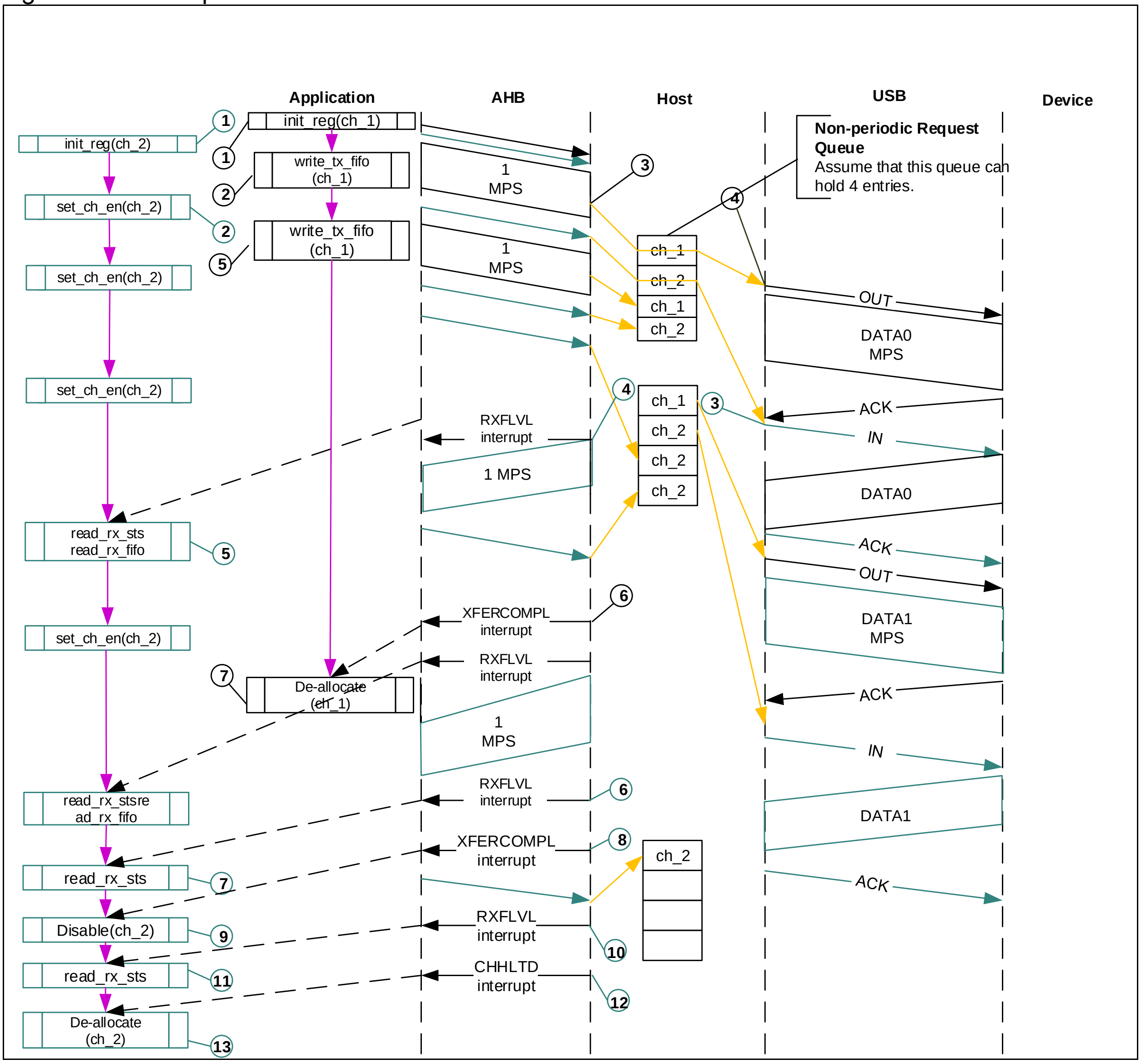
- The application is attempting to send two largest-packet-size packets (transfer size is 1024 bytes)
- The non-periodic transmit FIFO can store two packets (128 bytes for full-speed transfer)
- The non-periodic request queue depth is 4

(1) OUT/SETUP operation process for common bulk and control transfer

The sequence of operations shown in Figure 21-7 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 21-7 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 space. 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

21.5.3.9 Initialize interrupt IN transfers

Figure 21-8 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 1024 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 21-8 (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) Interrupt handling

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).

21.5.3.10 Initialize interrupt OUT transfers

Figure 21-8 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 packet (transfer size is 1024 bytes) to every frame
- The periodic transmit FIFO can store one packet (1KB bytes for full-speed transfer)
- The periodic request queue depth is 4

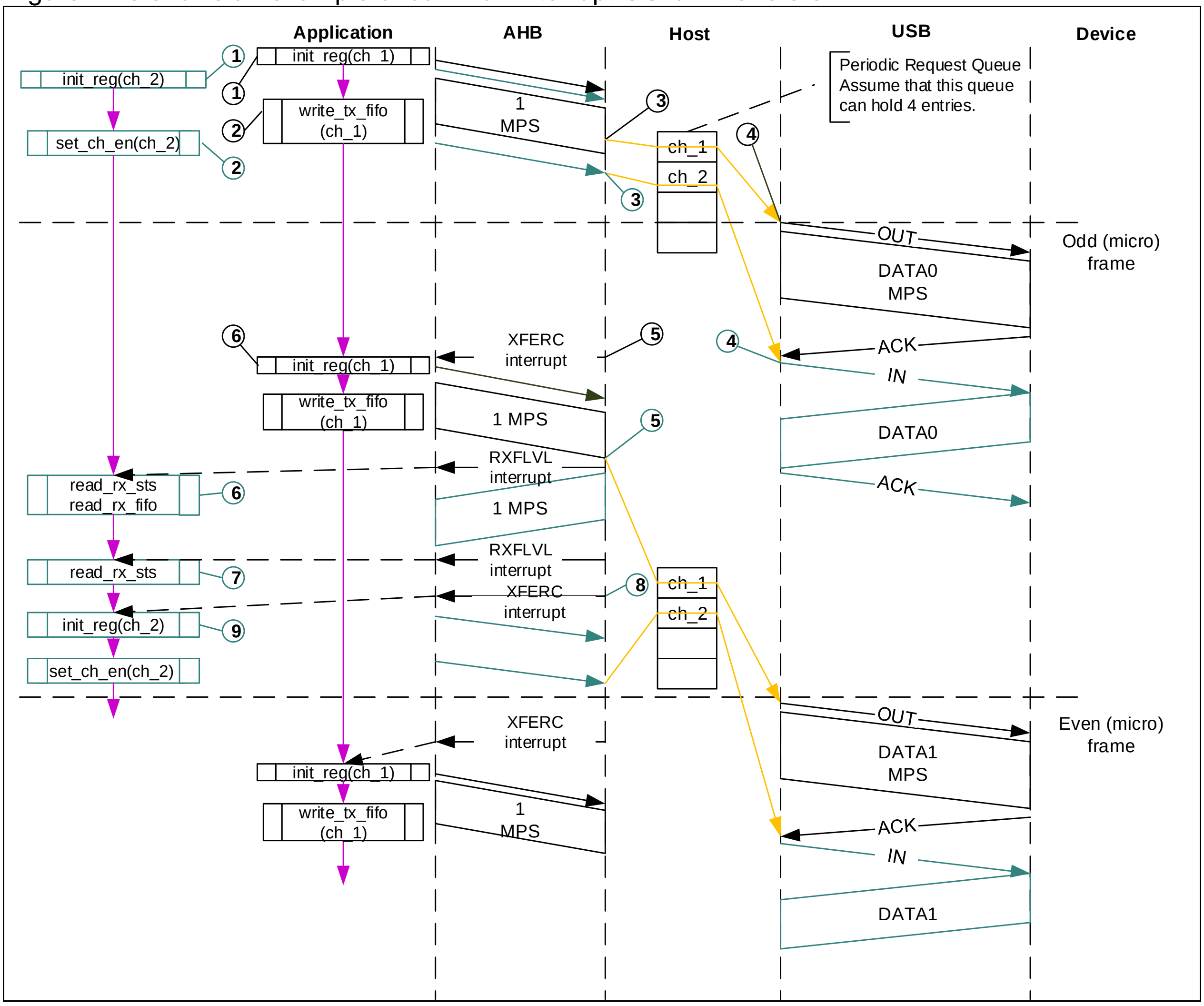
(1) Common interrupt IN operation process

The sequence of operations shown in Figure 21-8 (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) Interrupt handling

Figure 21-8 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.

21.5.3.11 Initialize synchronous IN transfers

Figure 21-9 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 1024 bytes) 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 21-9 (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) Interrupt handling

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
    }
}
```

21.5.3.12 Initialize synchronous OUT transfers

Figure 21-9 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 1024 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

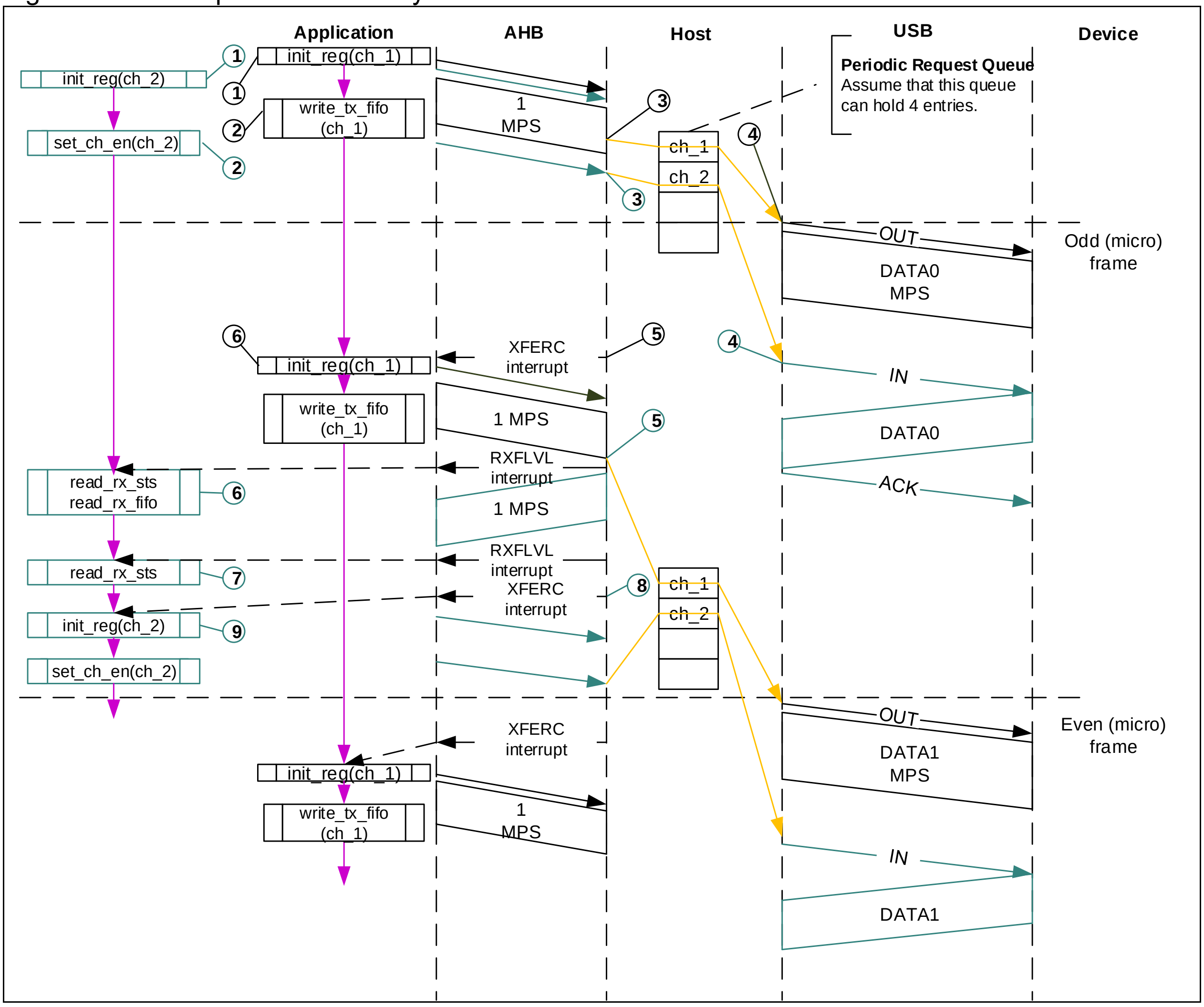
The sequence of operations shown in Figure 21-9 (channel 2) is as follows:

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) Interrupt handling

Figure 21-9 shows an example of common synchronous OUT transfers

Figure 21-9 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
}
```

21.5.4 OTGFS device mode

21.5.4.1 Device initialization

The application must perform the following steps to initialize the controller during power-on or after switching from host mode to device mode:

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.

21.5.4.2 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 NAB 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.

21.5.4.3 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.

21.5.4.4 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

21.5.4.5 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.

21.5.4.6 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 = 1
- 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

21.5.4.7 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.

21.5.4.8 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 according to 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.

21.5.4.9 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.

21.5.4.10 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 re-write 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 execute the received SETUP commands

3. For status IN stage, the application must program the registers according to “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.

21.5.4.11 Read FIFO packets

This section describes how to read FIFO packets (OUT data and SETUP packets)

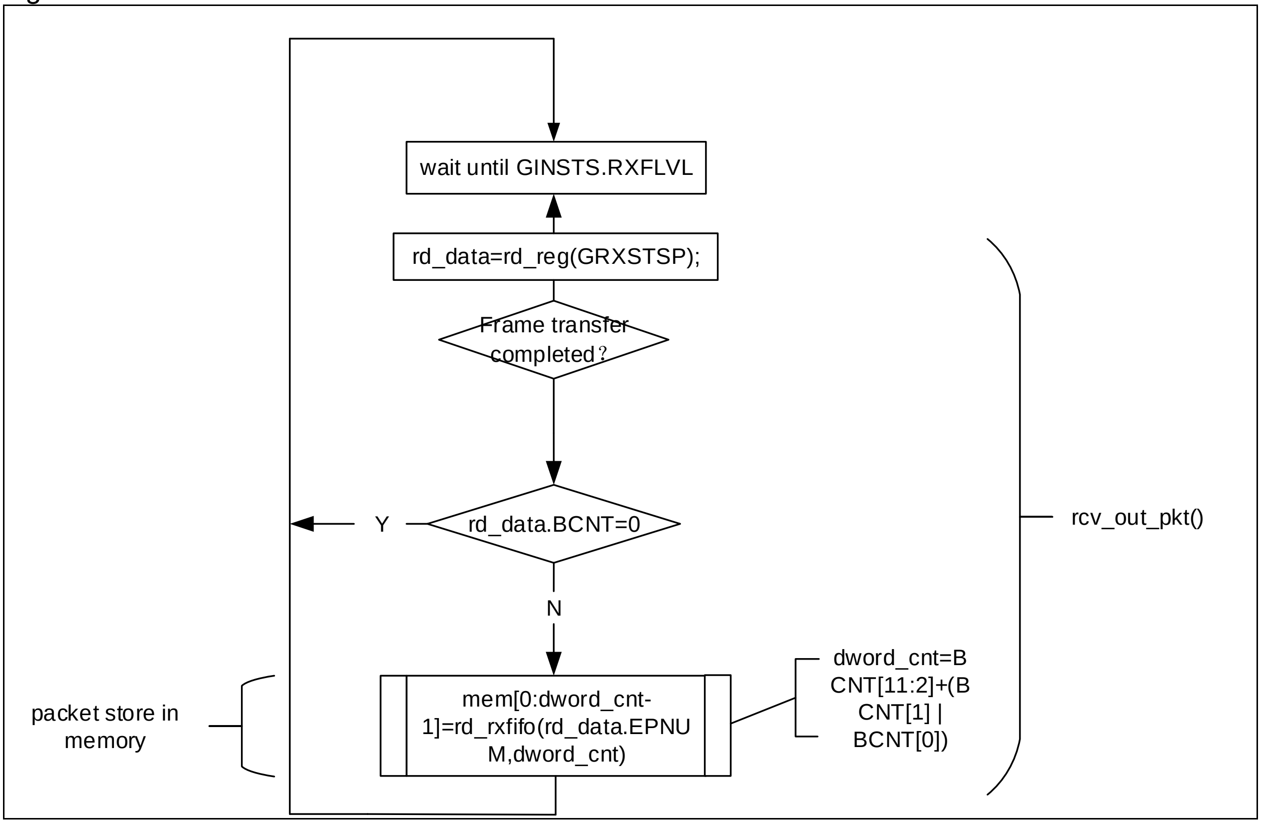
1. The application must read the OTGFS_GRXSTSP register as soon as the RXFLVL interrupt bit is detected in the OTGFS_GINTSTS register
2. The application can mask the RXFLVL interrupt bit in the OTGFS_GINTSTS register by setting RXFLVL = 0x0 in the OTGFS_GINTMSK register, until it has read the data packets 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 = Dont Care (0x0) and DPID = Dont 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 21-10* shows a flowchart.

Figure 21-10 Read receive FIFO



21.5.4.12 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 for one SETUP packet, one WORD for Setup stage and 8 WORDs for 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.

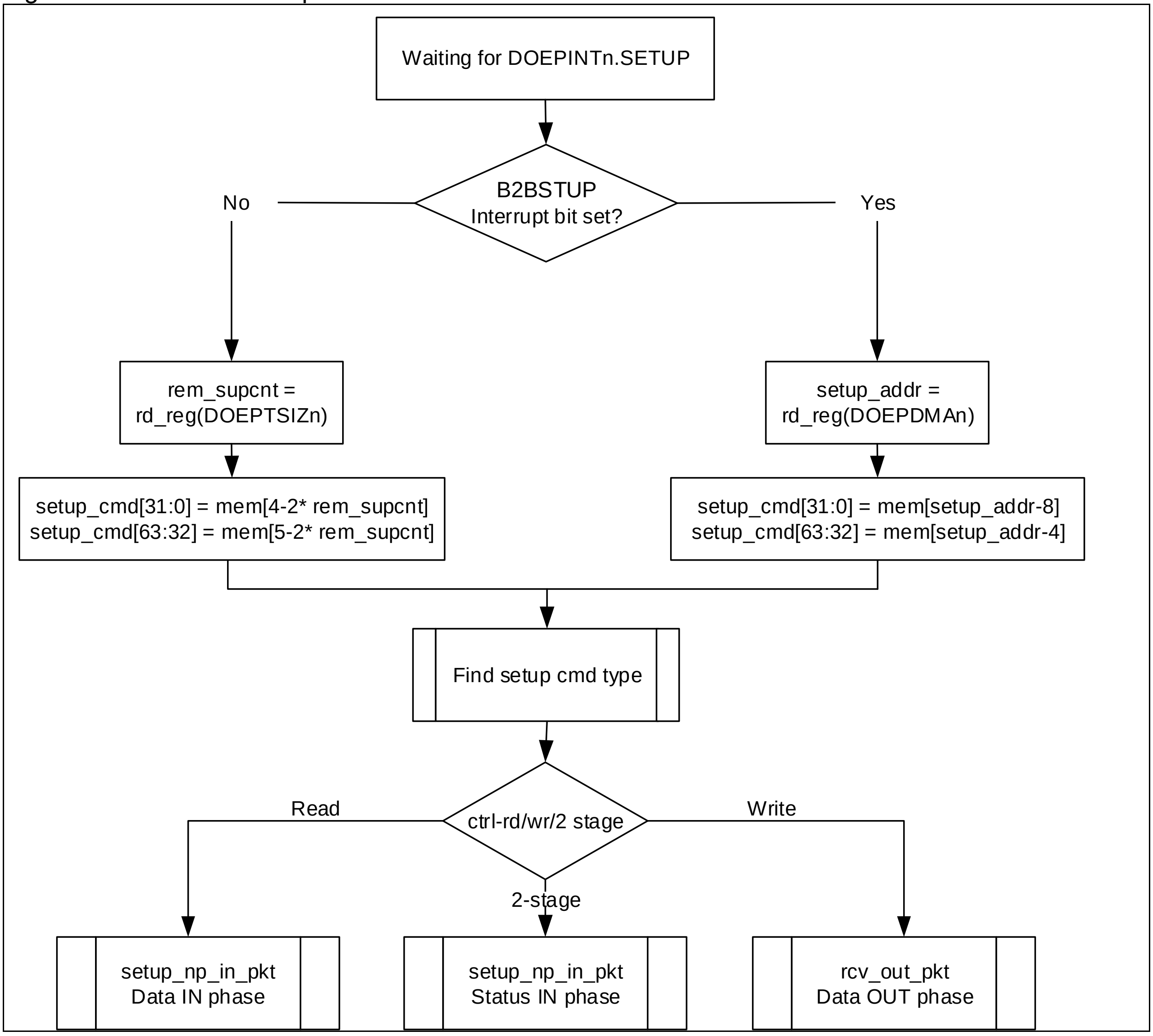
【Application programming process】

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 21-11 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).

21.5.4.13 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.

21.5.4.14 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 Initializtion. Before communicating with a host, the controller must follow the steps defined in Device Initializtion 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.

21.5.4.15 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 in 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.

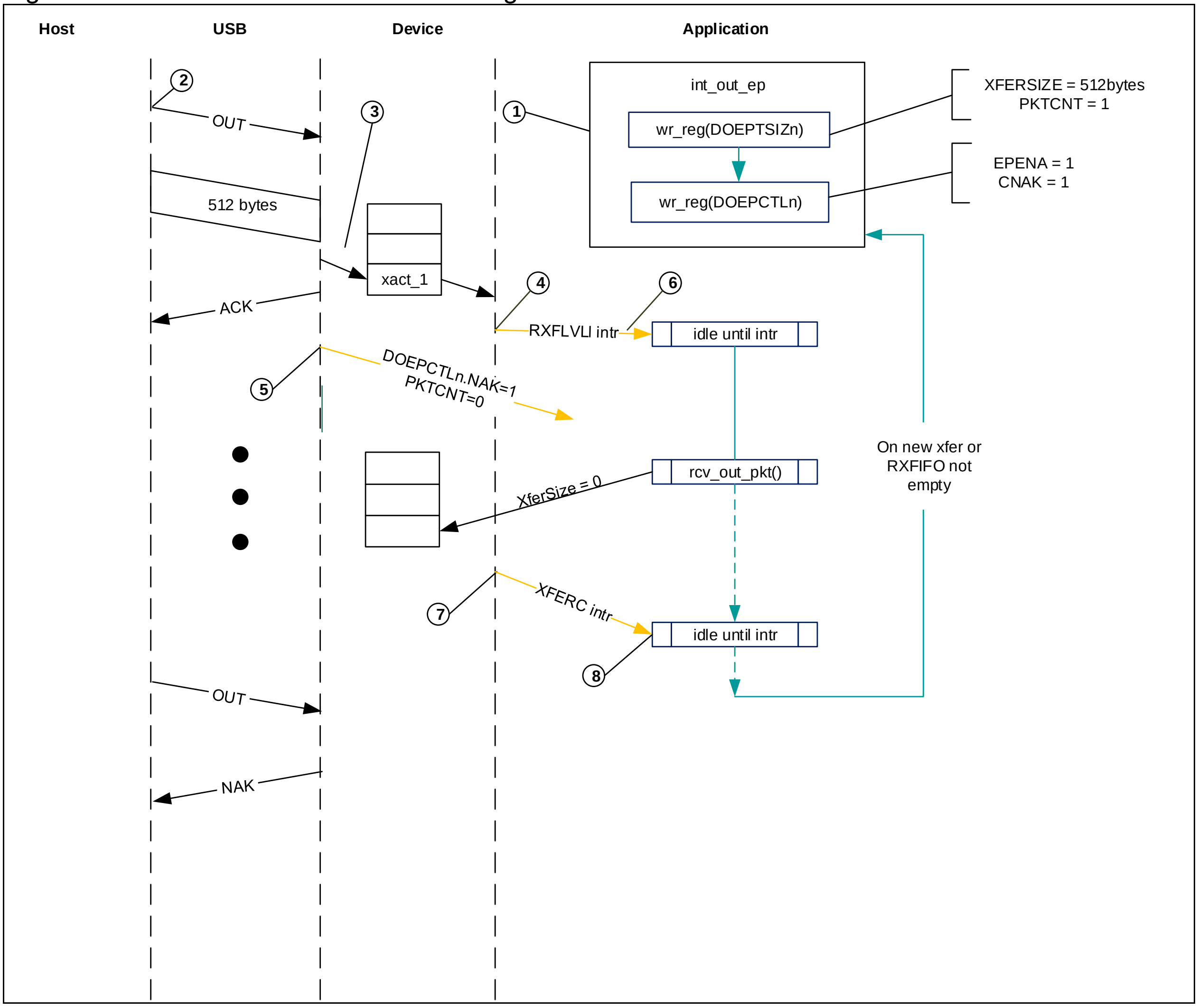
   This step can be repeated, depending on the transfer size

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 21-12 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 21-12 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.

21.5.4.16 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.

21.5.4.17 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 incomplSOOUT 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.

21.5.4.18 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.

21.5.4.19 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.

21.5.4.20 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.

21.6 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.

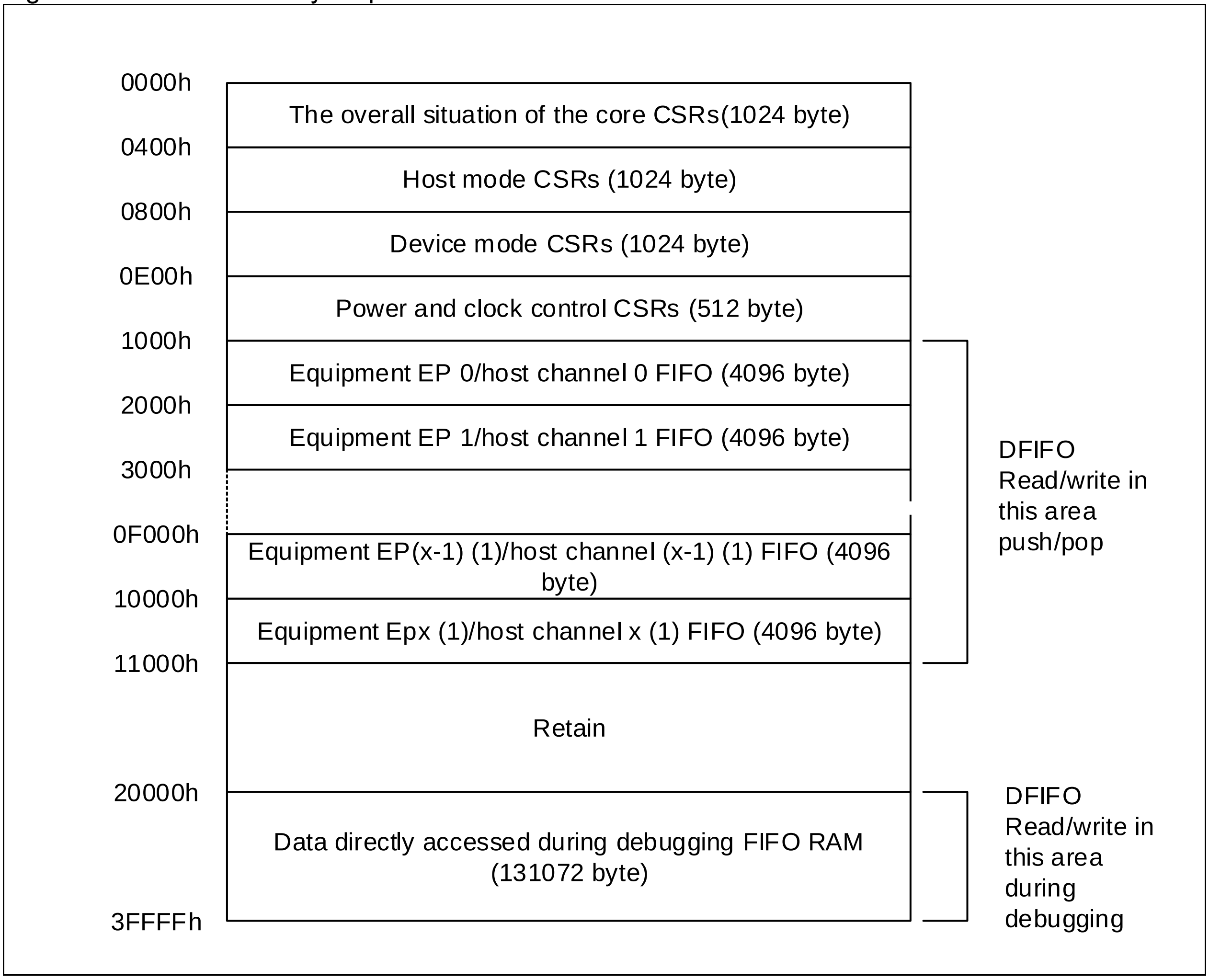
Only the controller global, power and clock control, data FIFO access and host port control and status registers are active in both host and device modes. When the OTGFS controller operates in either host or device mode, the application must not access the register group from the other mode. If an illegal 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)

21.6.1 CSR register map

The host and device mode registers occupy different addresses. All registers are located in the AHB clock domain

Figure 21-13 CSR memory map



x = 7 in device mode, x =15 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.

21.6.2 OTGFS register address map

Table 21-4 shows the USB OTG register map and their reset values.

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

Table 21-4 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	0x028	0x0000 0200
OTGFS_GNPTXSTS	0x02C	0x0008 0200
OTGFS_GCCFG	0x038	0x0000 0000
OTGFS_GUID	0x03C	0x0000 1000
OTGFS_HPTXFSIZ	0x100	0x0200 0600
OTGFS_DIEPTXF1	0x104	0x0200 0400
OTGFS_DIEPTXF2	0x108	0x0200 0400
OTGFS_DIEPTXF3	0x10C	0x0200 0400
OTGFS_DIEPTXF4	0x110	0x0200 0400
OTGFS_DIEPTXF5	0x114	0x0200 0400
OTGFS_DIEPTXF6	0x118	0x0200 0400
OTGFS_DIEPTXF7	0x11C	0x0200 0400
OTGFS_DIEPTXF8	0x120	0x0200 0400
OTGFS_DIEPTXF9	0x124	0x0200 0400
OTGFS_DIEPTXF10	0x128	0x0200 0400
OTGFS_DIEPTXF11	0x12C	0x0200 0400
OTGFS_DIEPTXF12	0x130	0x0200 0400
OTGFS_DIEPTXF13	0x134	0x0200 0400
OTGFS_DIEPTXF14	0x138	0x0200 0400
OTGFS_DIEPTXF15	0x13C	0x0200 0400
OTGFS_DIEPTXF16	0x140	0x0200 0400
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_HCCHAR9	0x620	0x0000 0000
OTGFS_HCINT9	0x628	0x0000 0000
OTGFS_HCINTMSK9	0x62C	0x0000 0000
OTGFS_HCTSIZ9	0x630	0x0000 0000
OTGFS_HCCHAR10	0x640	0x0000 0000
OTGFS_HCINT10	0x648	0x0000 0000
OTGFS_HCINTMSK10	0x64C	0x0000 0000
OTGFS_HCTSIZ10	0x650	0x0000 0000
OTGFS_HCCHAR11	0x660	0x0000 0000
OTGFS_HCINT11	0x668	0x0000 0000

OTGFS_HCINTMSK11	0x66C	0x0000 0000
OTGFS_HCTSIZ11	0x670	0x0000 0000
OTGFS_HCCHAR12	0x680	0x0000 0000
OTGFS_HCINT12	0x688	0x0000 0000
OTGFS_HCINTMSK12	0x68C	0x0000 0000
OTGFS_HCTSIZ12	0x690	0x0000 0000
OTGFS_HCCHAR13	0x6A0	0x0000 0000
OTGFS_HCINT13	0x6A8	0x0000 0000
OTGFS_HCINTMSK13	0x6AC	0x0000 0000
OTGFS_HCTSIZ13	0x6B0	0x0000 0000
OTGFS_HCCHAR14	0x6C0	0x0000 0000
OTGFS_HCINT14	0x6C8	0x0000 0000
OTGFS_HCINTMSK14	0x6CC	0x0000 0000
OTGFS_HCTSIZ14	0x6D0	0x0000 0000
OTGFS_HCCHAR15	0x6E0	0x0000 0000
OTGFS_HCINT15	0x6E8	0x0000 0000
OTGFS_HCINTMSK15	0x6EC	0x0000 0000
OTGFS_HCTSIZ15	0x6F0	0x0000 0000
OTGFS_DCFG	0x800	0x0220 0000
OTGFS_DCTL	0x804	0x0000 0002
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_DIEPCTL4	0x980	0x0000 0000
OTGFS_DIEPINT4	0x988	0x0000 0080
OTGFS_DIEPTSIZ4	0x990	0x0000 0000
OTGFS_DTXFSTS4	0x998	0x0000 0200
OTGFS_DIEPCTL5	0x9A0	0x0000 0000
OTGFS_DIEPINT5	0x9A8	0x0000 0080
OTGFS_DIEPTSIZ5	0x9B0	0x0000 0000
OTGFS_DTXFSTS5	0x9B8	0x0000 0200
OTGFS_DIEPCTL6	0x9C0	0x0000 0000
OTGFS_DIEPINT6	0x9C8	0x0000 0080
OTGFS_DIEPTSIZ6	0x9D0	0x0000 0000
OTGFS_DTXFSTS6	0x9D8	0x0000 0200
OTGFS_DIEPCTL7	0x9E0	0x0000 0000
OTGFS_DIEPINT7	0x9E8	0x0000 0080
OTGFS_DIEPTSIZ7	0x9F0	0x0000 0000
OTGFS_DTXFSTS7	0x9F8	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_DOEPCTL4	0xB80	0x0000 0000
OTGFS_DOEPINT4	0xB88	0x0000 0080
OTGFS_DOEPTSIZ4	0xB90	0x0000 0000
OTGFS_DOEPCTL5	0xBA0	0x0000 0000
OTGFS_DOEPINT5	0xBA8	0x0000 0080
OTGFS_DOEPTSIZ5	0xBB0	0x0000 0000
OTGFS_DOEPCTL6	0xBC0	0x0000 0000
OTGFS_DOEPINT6	0xBC8	0x0000 0080
OTGFS_DOEPTSIZ6	0xBD0	0x0000 0000
OTGFS_DOEPCTL7	0xBE0	0x0000 0000
OTGFS_DOEPINT7	0xBE8	0x0000 0080
OTGFS_DOEPTSIZ7	0xBF0	0x0000 0000

OTGFS_PCGCCTL	0xE00	0x0000 0000

21.6.3 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.

21.6.3.1 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 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 default value.
Bit 16	CONIDSTS	0x1	ro	Accessible in both host and device modes Connector ID status This bit indicates the connecter 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 default value.

21.6.3.2 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 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 default value.

21.6.3.3 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 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 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

21.6.3.4 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 default value.
Bit 14	Reserved	0x0	resd	Kept at 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 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.

21.6.3.5 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 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 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.

21.6.3.6 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 21-2*.

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 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 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 periodic 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 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 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 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

21.6.3.7 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 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 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 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 default value.
Bit 16	Reserved	0x0	resd	Kept at 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 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 default value.

21.6.3.8 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 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 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	Register	Reset value	Type	Description
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.

21.6.3.9 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 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.

21.6.3.10 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.

21.6.3.11 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

21.6.3.12 OTGFS general controller configuration register (OTGFS_GCCFG)

Bit	Register	Reset value	Type	Description
Bit 31: 22	Reserved	0x000	resd	Kept at 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: 18	Reserved	0x0	resd	Kept at default value.
Bit 17	LP_MODE	0x0	rw	Low-power mode This bit is used to control the OTG PHY consumption. When this bit is set to 1 by software, the OTG PHY enters low-power mode; when this bit is cleared by software, the OTG PHY operates in normal mode. 0: Non-low-power mode 1: Low-power mode
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 default value.

21.6.3.13 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.

21.6.3.14 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.

21.6.3.15 OTGFS device IN endpoint Tx FIFO size register (OTGFS_DIEPTXFn) (x=1…7, 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

21.6.4 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:

21.6.4.1 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

21.6.4.2 OTGFS host frame interval register (OTGFS_HFIR)

This register is used to program the current

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

Bit	Register	Reset value	Type	Description
				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)

21.6.4.3 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.

21.6.4.4 OTGFS host periodic Tx FIFO/request queue register (OTGFS_HPTXSTS)

This is a ready-only register containing the free space information of the periodic 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

21.6.4.5 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 22-2.* 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.

21.6.4.6 OTGFS host all channels interrupt mask register (OTGFS_HAINTMSK)

The host all channels interrupt mask register works with the host all channels interrupt register 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 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.

21.6.4.7 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 22-2*. 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 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 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

21.6.4.8 OTGFS host channelx characteristics register (OTGFS_HCCHARx) (x = 0...15, 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 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.

21.6.4.9 OTGFS host channelx interrupt register (OTGFS_HCINTx) (x = 0...15, where x= channel number)

This register contains the status of a channel related to USB and AHB events, as shown in *Figure 22-2*. 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 for 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 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 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 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.

21.6.4.10 OTGFS host channelx interrupt mask register (OTGFS_HCINTMSKx) (x = 0...15, 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 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 default value.
Bit 1	CHHLTDMSK	0x0	rw	Channel halted mask
Bit 0	XFERCMSK	0x0	rw	Transfer completed mask

21.6.4.11 OTGFS host channelx transfer size register (OTGFS_HCTSIZx) (x = 0...15, where x= channel number)

Bit	Register	Reset value	Type	Description
Bit 31	Reserved	0x0	resd	Kept at 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.

21.6.5 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.

21.6.5.1 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 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 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)

21.6.5.2 OTGFS device control register (OTGFS_DCTL)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Kept at 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

Bit	Register	Reset value	Type	Description
				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	0x1	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 21-5 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 21-5 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

21.6.5.3 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 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.

21.6.5.4 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 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 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 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

21.6.5.5 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 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 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 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 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

21.6.5.6 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 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 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.

21.6.5.7 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 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 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

21.6.5.8 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 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

21.6.5.9 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 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 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 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 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

21.6.5.10 OTGFS device IN endpoint-x control register (OTGFS_DIEPCTLx) (x=x=1…7, 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 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 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.

21.6.5.11 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 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 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 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 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

21.6.5.12 OTGFS device control OUT endpoint-x control register (OTGFS_DOEPCTLx) (x= x=1…7, 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 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 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.

21.6.5.13 OTGFS device IN endpoint-x interrupt register (OTGFS_DIEPINTx) (x=0…7, where x if endpoint number)

This register indicates the status of an endpoint when USB and AHB-related events occurs, as shown in *Figure 21-2.* 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 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 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 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.

21.6.5.14 OTGFS device OUT endpoint-x interrupt register (OTGFS_DOEPINTx) (x=0…7, 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 21-2*. 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 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 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.

21.6.5.15 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 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 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.

21.6.5.16 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 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 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 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.

21.6.5.17 OTGFS device IN endpoint-x transfer size register (OTGFS_DIEPTSIZx) (x=1…7, 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 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.

21.6.5.18 OTGFS device IN endpoint transmit FIFO status register (OTGFS_DTXFSTSx) (x=1…7, 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 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

21.6.5.19 OTGFS device OUT endpoint-x transfer size register (OTGFS_DOEPTSIZx) (x=1…7, 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 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.

21.6.6 Power and clock control registers

21.6.6.1 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 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.

22 HICK auto clock calibration (ACC)

22.1 ACC introduction

HICK auto clock calibration (HICK ACC), which uses the SOF signal (1 ms of period) generated as a reference signal, implements the sampling and calibration for the HICK clocks.

The main purpose of this module is to provide a clock of 48MHz±0.25% for the USB device.

It is able to make the calibrated frequency as close to the target frequency as possible by means of "cross and return" algorithm.

22.2 Main features

- Programmable center frequency
- Programmable boundary frequency that triggers calibration function
- Center frequency precision ±0.25%
- Status detection flags
  - Calibration ready flag
  - Error detection flag
  - Reference signal lost error flag
- Two interrupt sources with flags
  - Calibration ready flag
  - Reference signal lost error flag
- Two calibration modes: coarse calibration and smooth calibration

22.3 Interrupt requests

Table 22-1 ACC interrupt requests

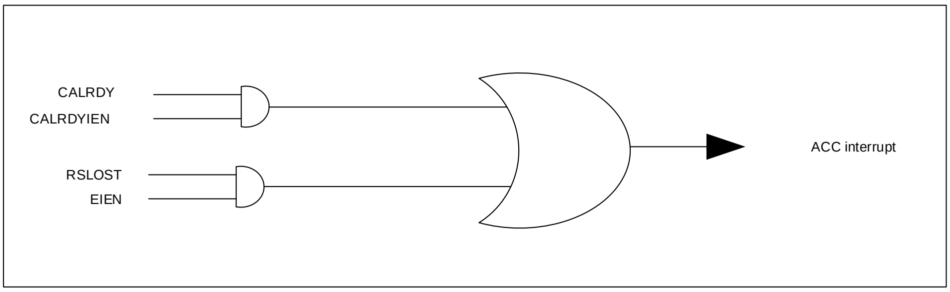
Interrupt event	Event flag	Enable bit
Calibration ready	CALRDY	CALRDYIEN
Reference signal lost	RSLOST	EIEN

ACC interrupt events are linked to the same interrupt vector (see *Figure 22-1*). Interrupt events include:

- During calibration process: When the calibration gets ready or reference signal lost occurs,

the corresponding interrupt will be generated if the corresponding enable bit is enabled.

Figure 22-1 ACC interrupt mapping diagram



22.4 Functional description

Auto clock calibration (HICK ACC), which uses the SOF signal (1 ms of period) generated as a reference signal, implements the sampling and calibration for the HICK clocks. In particular, the HICK clock frequency can be calibrated to a precision of ±0.25% so as to meet the needs of the high-precision clock applications such as USB.

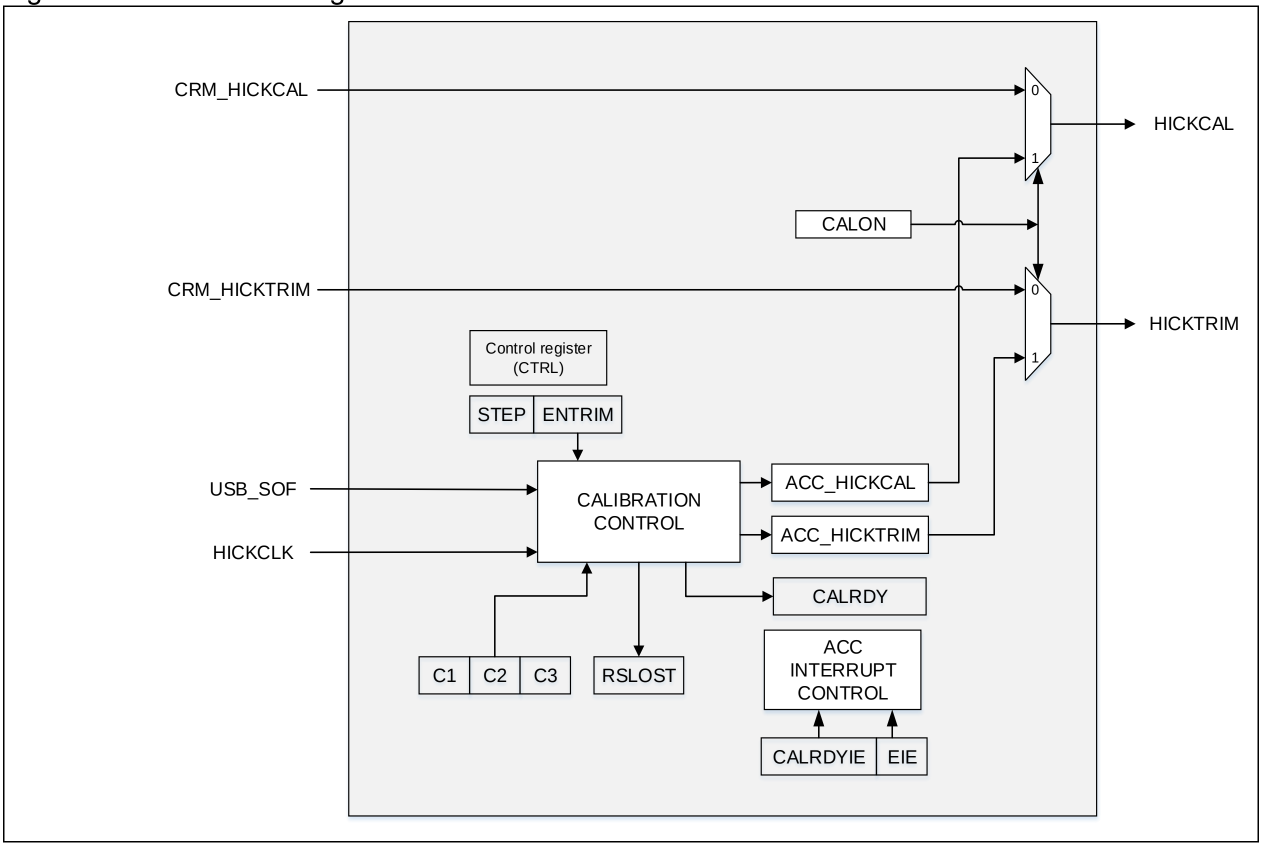
The signals of the module are connected to the CRM and HICK inside the microcontroller instead of being connected to the pins externally.

- CRM_HICKTWK: the HICKTWK bit in the CRM module. This signal is used to calibrate the HICK in bypass mode. The value is defined by the HICKTWK[5: 0] in the CRM_CTRL register.
- CRM_HICKTRIM: the HICKTRIM bit in the CRM module. This signal is used to calibrate the HICK in bypass mode. The value is defined by the HICKTRIM[5: 0] in the CRM_CTRL register.

  The default value of the HICK is 32, which can be calibrated to 8MHz±0.25%. The HICK frequency can be adjusted by 20kHz (design value) each time when the CRM_HICKTWK value changes. In other words, the HICK output frequency will increase by 20kHz each time the CRM_HICKTWK value is decremented by one; the HICK output frequency will reduce by 20kHz each time the CRM_HICKTWK value is decremented by one.
- USB_SOF: USB Start-of-Frame signal given by the USB device. Its high-level width is 64 system clock cycles, a pulse signal of 1 ms.
- HICKCLK: HICK clock. The original HICK output frequency is 48MHz, but the sampling clock used by the HICK calibration module is frequency divider (1/6) clock, about 8MHz.
- HICKCAL: HICK module calibration signal. For the HICK clock after frequency division (1/6), the HICK clock frequency will change by 40KHz (design value) each time the HICKCAL changes, which is positively correlated. In other words, the HICK clock frequency will increase by 40KHz (design value) each time the HICKCAL is incremented by one; the HICK clock frequency will reduce by 40KHz each time the HICKCAL is decremented by one.
- HICKTWK: HICK module calibration signal. For the HICK clock after frequency division (1/6), the HICK clock frequency will change by 20KHz (design value) each time the HICKTWK changes, which is positively correlated.

Refer to *Section 22.6* for more information about the bit definition in the registers.

Figure 22-2 ACC block diagram

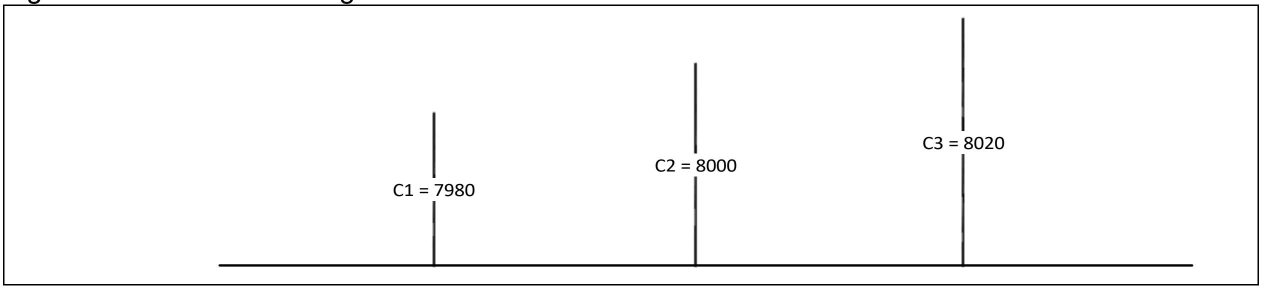


22.5 Principle

USB_SOF period signal: 1ms of period must be accurate, which is a prerequisite of the normal operation for an auto calibration module.

cross-return algorithm: This is used to calculate a calibration value closest to the theoretic value. In theory, the actual frequency after calibration can be adjusted to be within an accuracy range of about 0.5 steps from the target frequency (8MHz).

Figure 22-3 Cross-return algorithm



From the above figure, auto calibration function will adjust the HICKCAL or HICKTWK according to the specified step as soon as the condition for triggering auto calibration is reached.

Cross:

If the auto calibration condition is met, the actual sampling data in the first 1ms period will be either less than C2, or greater than C2.

When this value is less than C2, the auto calibration module will start increasing either the HICKCAL or HICKTWK according to the step definition until the actual sampling value is greater than C2. In this way, the actual value will cross over C2 from small to large.

When this value is greater than C2, the auto calibration module will decrease either the HICKCAL or HICKTWK according to the step definition until the actual sampling value become less than C1. In this way, the actual value will cross over C2 from large to small.

Return:

After cross operation is completed, the actual value closest to C2 can be obtained by comparing the difference (calculated as absolute value) between the actual sampling value and C2 before and after crossing C2 so as to get the best calibration value HICKCAL or HICKTWK.

If the difference after crossing is less than the one before crossing C2, the calibration value after crossing prevails, and stops the calibration process until the next condition for auto calibration appears.

If the difference after crossing is greater than the one before crossing C2, the calibration value before crossing prevails, and it will return by one step to the one before crossing, and stops the calibration process until the next condition for auto calibration appears.

According to the cross-return strategy, in theory, it is possible to get the frequency accuracy that is 0.5 steps away from the center frequency.

Four conditions for enabling auto calibration function are as follows:

1. The rising edge of the CANLON (from 0 to 1)
2. When CALON=1, reference signal is lost and restored
3. When the sample counter is less than C1
4. When the sample counter is greater than C3

Even though the sampling counter is between C1 and C3, at the rising edge the CANLON, the auto calibration module can also be activated so that the HICK frequency can be adjusted to be within a range of 0.5 steps of the center frequency as soon as the CANLON is enabled.

Under one of the above-mentioned circumstances, the HICK frequency can be calibrated to be within 0.5 steps of the center frequency. To achieve the best calibration accuracy, it is recommended to remain step as 1 (default value). If the step is set to 0, either HICKCAL or HICKTWK cannot be calibrated.

22.6 Register description

Refer to the list of abbreviations used in register descriptions.

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

22.6.1 ACC register map

Table 22-2 ACC register map and reset values

Register name	Offset	Reset value
ACC_STS	0x00	0x0000 000
ACC_CTRL1	0x04	0x0000 0100
ACC_CTRL2	0x08	0x0000 2080
ACC_C1	0x0C	0x0000 1F2C
ACC_C2	0x10	0x0000 1F40
ACC_C3	0x14	0x00000 1F54

22.6.2 Status register (ACC_STS)

Bit	Register	Reset value	Type	Description
Bit 31: 9	Reserved	0x0000000	resd	Kept at default value.
Bit 1	RSLOST	0x0	ro	Reference Signal Lost 0: Reference Signal is not lost 1: Reference Signal is lost Note: During the calibration, when the sample counter of the calibration module is twice that of C2, if a SOF reference signal is not detected, it means that the reference signal is lost. The internal statue machine will move to the idle state unless another SOF signal is detected, otherwise, the internal clock sample counter remains 0. The RSLOST bit is immediately cleared after the CALON bit is cleared or when the RSLOST is written with 0. Reference signal detection occurs only when CALON=1.
Bit 0	CALRDY	0x0	ro	Internal high-speed clock calibration ready 0: Internal 8MHz oscillator calibration is not ready 1: Internal 8MHz oscillator calibration is ready Note: This bit is set by hardware to indicate that internal 8MHz oscillator has been calibrated to the frequency closest to 8MHz. The CALRDY is immediately cleared after the CALON bit is cleared or when the CALRDY is written with 0.

22.6.3 Control register 1 (ACC_CTRL1)

Bit	Register	Reset value	Type	Description
Bit 31: 12	Reserved	0x00000	resd	Forced to 0 by hardware.
Bit 11: 8	STEP	0x1	rw	Calibrated step This field defines the value after each calibration. Note: It is recommended to set the step bit in order to get a more accurate calibration result. While ENTRIM=0, only the HICKCAL is calibrated. If the step is incremented or decremented by one, the HICKCAL will be incremented or decremented by one accordingly, and the HICK frequency will increase or decrease by 40KHz (design value). This is a positive relationship. While ENTRIM=1, only the HICKTRIM is calibrated. If the step is incremented or decremented by one, the HICKTRIM will be incremented or decremented by one accordingly, and the HICK frequency will increase or decrease by 20KHz (design value). This is a positive

				relationship.
Bit 7: 6	Reserved	0x0	rw	Forced by hardware to 0
Bit 5	CALRDYIEN	0x0	rw	CALRDY interrupt enable This bit is set or cleared by software. 0: Interrupt generation disabled 1: ACC interrupt is generated when CALRDY=1 in the ACC_STS register
Bit 4	EIEN	0x0	rw	RSLOST error interrupt enable This bit is set or cleared by software. 0: Interrupt generation disabled 1: ACC interrupt is generated when RSLOST=1 in the ACC_STS register
Bit 3: 2	Reserved	0x0	rw	Forced by hardware to 0
Bit 1	ENTRIM	0x0	rw	Enable trim This bit is set or cleared by software. 0: HICKCAL is calibrated. 1: HICKTRIM is calibrated. Note: It is recommended to set ENTRIM=1 in order to get higher calibration accuracy.
Bit 0	CALON	0x0	rw	Calibration on This bit is set or cleared by software. 0: Calibration disabled 1: Calibration enabled, and starts searching for a pulse on the USB_SOF. Note: This module cannot be used without the USB_SOF reference signal. If there are no requirements on the accuracy of the HICK clock, it is unnecessary to enable this module.

22.6.4 Control register 2 (ACC_CTRL2)

Bit	Register	Reset value	Type	Description
Bit 31: 14	Reserved	0x00000	resd	Forced to 0 by hardware
Bit 13: 8	HICKTWK	0x20	ro	Internal high-speed auto clock trimming This field is read only, but not written. Internal high-speed clock is adjusted by ACC module, which is added to the ACC_HICKCAL[7: 0] bit. These bits allow the users to input a trimming value to adjust the frequency of the HICKRC oscillator according to the variations in voltage and temperature. The default value is 32, which can trim the HICK to 8MHz±0.25. The trimming value is 20kHz (design value) between two consecutive ACC_HICKTWK steps.
Bit 7: 0	HICKCAL	0x80	ro	Internal high-speed auto clock calibration This field is read only, but not written. Internal high-speed clock is adjusted by ACC module. These bits allow the users to input a trimming value to adjust the frequency of the HICKPC oscillator according to the variations in voltage and temperature. The default value is 128, which can trim the HICK to 8MHz±0.25. The trimming value is 40kHz (design value) between two consecutive ACC_HICKCAL steps.

22.6.5 Compare value 1 (ACC_C1)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C1	0x1F2C	rw	Compare 1 This value is the lower boundary for triggering calibration, and its default value is 7980. When the number of clocks sampled by ACC in 1ms period is less than or equal to C1, auto calibration is triggered automatically. When the actual sampling value (number of clocks in 1ms) is greater than C1 but less than C3, auto calibration

				is not enabled.

22.6.6 Compare value 2 (ACC_C2)

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C2	0x1F40	rw	Compare 2 This value defines the number of clocks sampled for 8MHz (ideal frequency) clock in 1ms period , and its default value is 8000 (theoretical value) As a center point of cross-return strategy, this value is used to calculate the calibration value closest to the theoretical value. In theory, the actual frequency after calibration can be trimmed to be within an accuracy of 0.5 steps from the target frequency (8MHz)

22.6.7 Compare value 3 (ACC_C3)

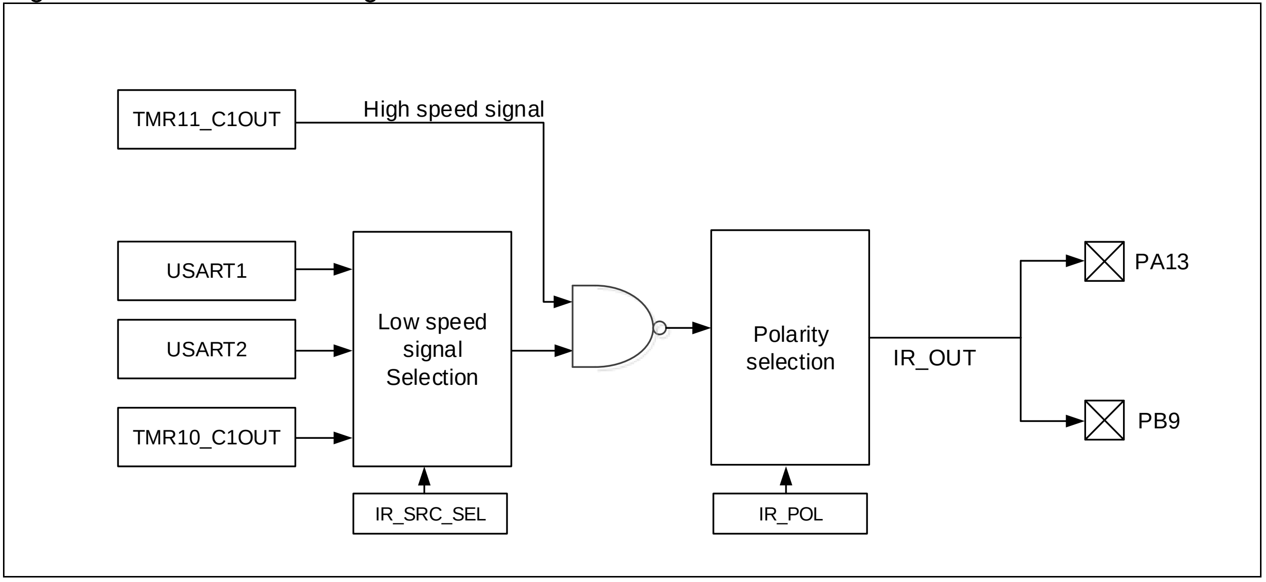
Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Forced to 0 by hardware
Bit 15: 0	C3	0x1F54	rw	Compare 3 This value is the upper boundary for triggering calibration. When the number of clock sampled by ACC in 1ms period is greater than or equal to C3, auto calibration is triggered automatically. When the actual sampling value (number of clocks in 1ms period) is greater than C1 but less than C3, auto calibration is not enabled.

23 Infrared timer (IRTMR)

The IRTMR (Infrared Timer) is used to generate the IR_OUT signal that drives the infrared LED to achieve infrared control.

The IR_OUT signals consists of a low-frequency modulation envelope and high-frequency carrier signals. The low-frequency modulation envelope signal selects from TMR10_C1OUT, USART1 and USART2 through the IR_SRC_SEL[1: 0] bit in the SCFG_CFG1 register, while the high-frequency carrier signal is provided by the TMR11_C1OUT register. The IR_POL bit in the SCFG_CFG1 register controls whether the IR_OUT output is reversed or not. The IR_OUT signal is output through multiplexed function via PB9 or PA13 (multiplexed mode needs to be configured in advance).

Figure 23-1 IRTMR block diagram



24 External memory controller (XMC)

24.1 XMC introduction

XMC block is able to translate the AHB transactions into external memory signals and vice versa.It features two chip-select signals at different pins, for interfacing up to two external memories at a time. The supported external memories include SRAM, NOR Flash and PSRAM.

24.2 XMC main features

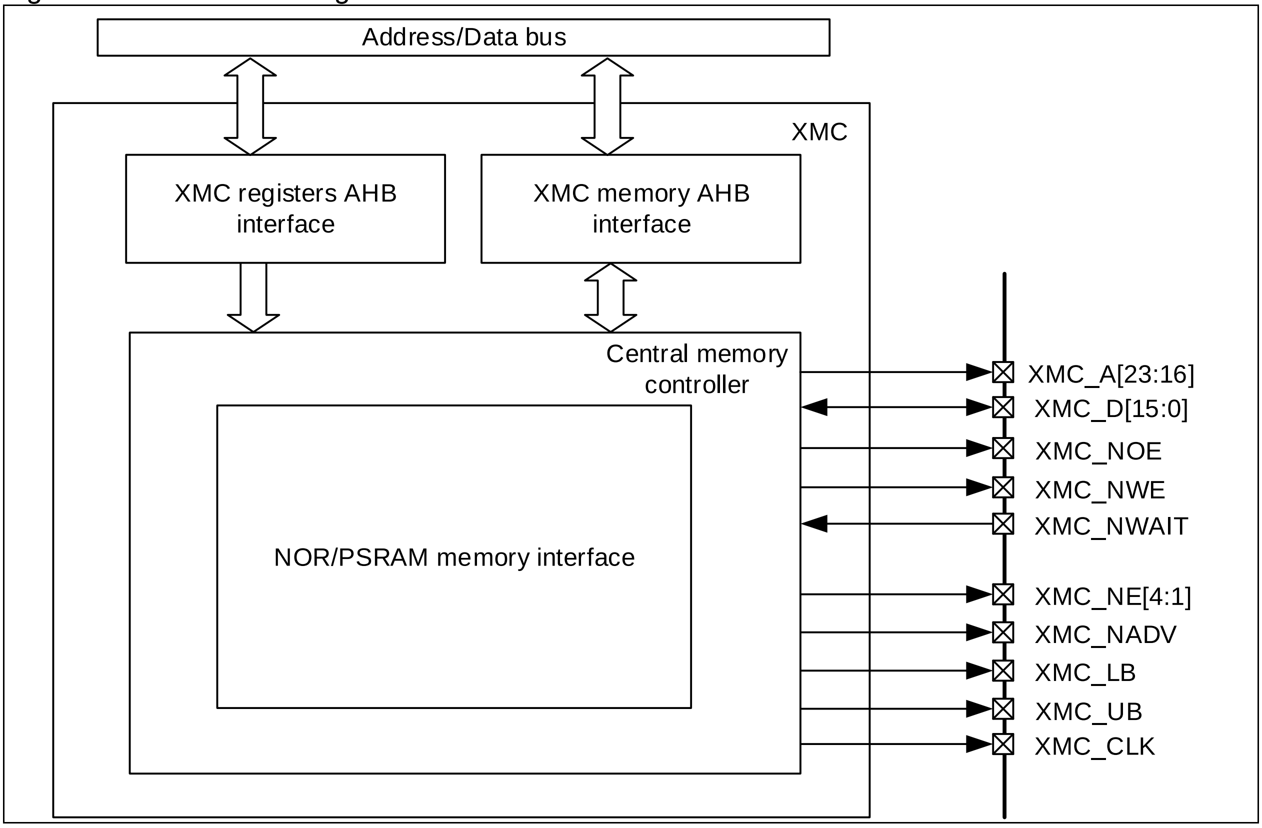
NOR/PSRAM has the following features:

- Chip select signals supporting 3 external memories, each of which has their own control register
- Support access to static memory devices, including
  - Static random access memory (SRAM)
  - NOR Flash
  - PSRAM
- 8-bit or 16-bit wide memory
- Various timing mode selection
  - Two modes with the same timings for read and write
  - Four modes with different timings for read and write
  - Multiplexed address/data mode
  - Synchronous mode
- Programmable timing control registers
- Translate the AHB data size into the appropriate external memory data size

24.3 XMC architecture

24.3.1 Block diagram

Figure 24-1 XMC block diagram



While interfacing to the external memory, NOR/PSRAM use different pins as shown in .
Table 24-1.

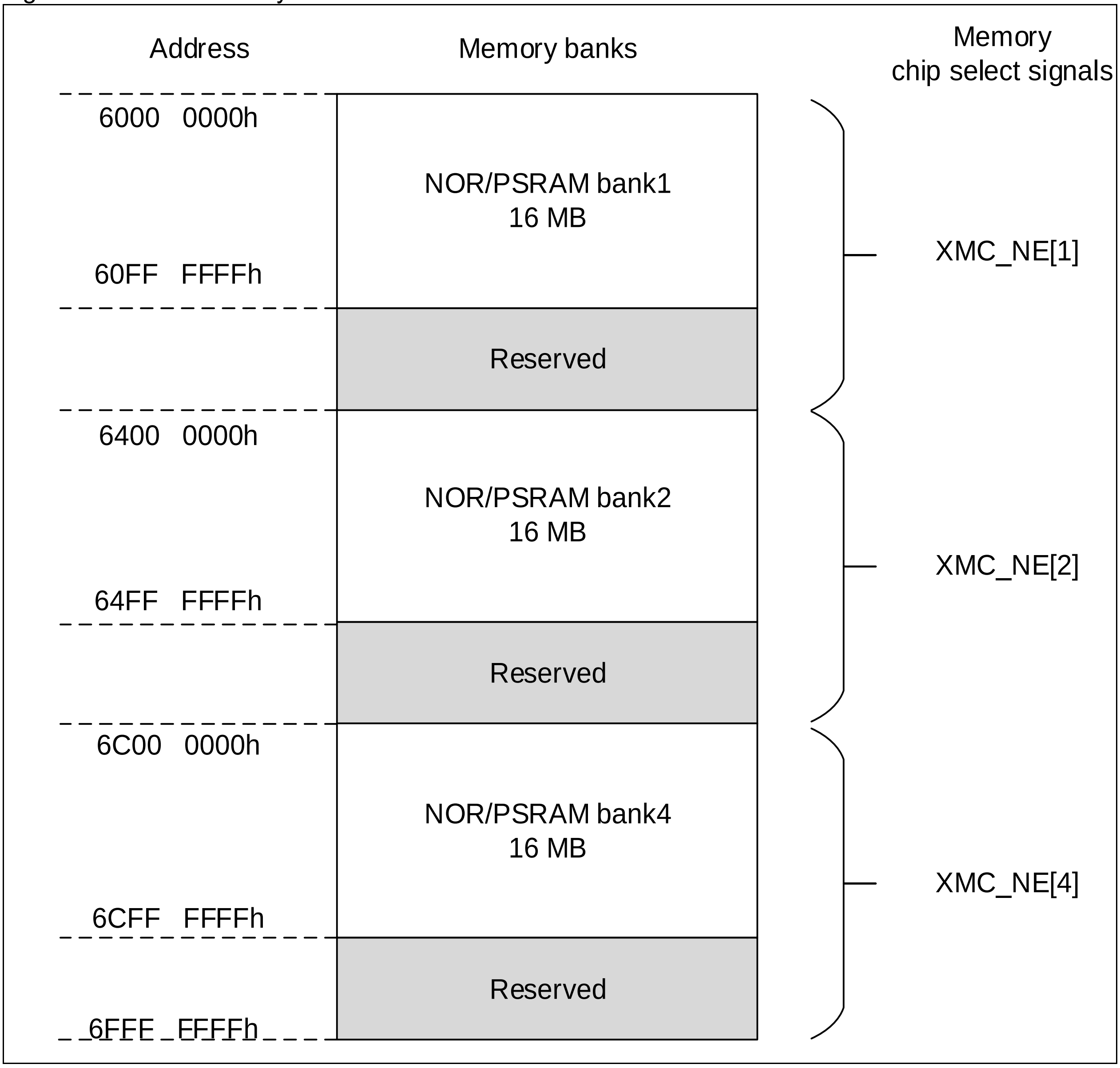
Table 24-1 NOR/PSRAM pins

Pin name	I/O	Description
XMC_CLK	Output	Clock
XMC_NE[x], x=1,4	Output	Chip select
XMC_NADV	Output	Address latch or address valid (NL) signal
XMC_A[x]	Output	Address bus
XMC_NOE	Output	Output enable signal
XMC_NWE	Output	Write enable signal
XMC_LB and XMC_UB	Output	Byte select signal
XMC_D[15: 0]	Read input/write output	Data bus/multiplexed address data
XMC_NWAIT	Input	Wait signal

24.3.2 Address mapping

XMC addresses are divided into multiple memory banks, as shown below.

Figure 24-2 XMC memory banks



HADDR[31:28] bits are used to select one of the three memory banks as shown in Table 24-2.

Table 24-2 Memory bank selection

HADDR[31: 28]	HADDR[27: 26]
0110: NOR/PSRAM	00: bank1
	01: bank2
	11: bank4

24.4 NOR/PSRAM

NOR/PSRAM offers multiple access modes with different timings to drive multiple memories including NOR Flash, SRAM, PSRAM and Cellular RAM.

There are three banks : bank 1, bank2 and bank 4, each of which has its own control register. They are are accessed by means of different timings and different chip-select signals.

24.4.1 Operating mode

Pin function:

Pin signals vary from one external memory to another. Table 24-3 lists typical pin signals.

Table 24-3 Pin signals for NOR and PSRAM

XMC pin name	NOR Flash	PSRAM
XMC_CLK	Clock (synchronous mode)	Clock (synchronous mode)
XMC_NE[x]	Chip-select	Chip-select
XMC_NADV	Address latch or address valid	Address latch or address valid
XMC_A[23:16]	Address bus	Address bus
XMC_NOE	Output enable	Output enable
XMC_NWE	Write enable	Write enable
XMC_LB, XMC_UB	Without using XMC_LBL[1:0]	XMC_LB: lower byte XMC_UB: upper byte
XMC_D[15: 0]	Data bus multiplexed address data bus (multiplex and synchronous mode)	Data bus multiplexed address data bus (multiplex and synchronous mode)
XMC_NWAIT	NOR Flash wait request	PSRAM wait request

Note: If the memory data size is 8-bit, the typical data bus is XMC_D[7: 0].

Access address

The upper bytes of the HADDR bit is used to select a memory bank while the lower bytes is used to select data memory address. HADDR is a byte address whereas the XMC supports the memory addressed in words or half words. Address translation between them is shown in *Table 24-4*. As long as read/write access to a specific address occurs, the XMC uses HADDR bit to enable chip-select signals and then write/read operation to the external memories

Table 24-4 Address translation between HADDR and external memory

External memory data width	Address connection	Accessible maximum memory space (bits)
8-bit	HADDR[23: 0] is linked to XMC_A[23: 0]. In multiplexed and synchronous mode, HADDR[15: 0] is connected to XMC_D[15: 0] during address latch period.	64 Mbyte x8 =512 Mbit
16-bit	HADDR[23: 1] is connected to XMC_A[22: 0]. In multiplexed and synchronous mode, HADDR[16: 1] is connected to XMC_D[15: 0] during address latch period	(64 Mbyte x 16)/2=512 Mbit

Data access

In case that the AHB data width is not equal to that of the memory, the XMC will make appropriate arrangement according to the typical signals of the external memories. *Table 24-5* lists the operation modes supported by XMC.

Table 24-5 Data access width vs. external memory data width

Memory	Mode	AHB data width	Memory width	Description
SRAM	Asynchronous read/write	8/16/32	8	One-time access, or split into 2 or 4 accesses
	Asynchronous read/write	8/16/32	16	XMC_LB and XMC_UB, One-time, or split into two access
NOR Flash	Asynchronous read	8	16	
	Asynchronous read/write	16	16	
	Asynchronous read/write	32	16	Split into 2 XMC accesses
	Synchronous read	16	16	
	Synchronous read	32	16	Split into 2 XMC accesses
PSRAM	Asynchronous read	8	16	
	Asynchronous write	8	16	Use XMC_LB and XMC_UB

	Asynchronous read/write	16	16	
	Asynchronous read/write	32	16	Split into 2 XMC accesses
	Synchronous write	8	16	Use XMC_LB and XMC_UB
	Synchronous read/write	16	16	
	Synchronous read/write	32	16	Split into 2 XMC accesses

24.4.2 Access mode

The XMC offers various access modes. Each access mode is operated based on timing parameters as shown in *Table 24-6*. Users can perform programming operations according to the specifications of the external memory and application needs.

Access modes available in the XMC:

- Multiplexed address data lines
- Clock-based synchronous mode

Table 24-6 NOR/PSRAM parameter registers

Parameter register	Function	Access mode	Unit
ADDRST	Address set-up time	1, 2, A, B, C, D and multiplexed	HCLK cycle
ADDRHT	Address-hold time	D and multiplexed	HCLK cycle
DTST	Data set-up time	1, 2, A, B, C, D and multiplexed	HCLK cycle
DTLAT	Data latency time	Synchronous	XMC_CLK cycle
CLKPSC	Clock prescaler	Synchronous	HCLK cycle

In addition to timing parameter registers for timing control, if the wait enable bit (NWASEN or NWSEN) is enabled, the XMC starts checking whether the XMC_NWAIT signal is in wait request state during data set time. If so, the XMC then waits until the XMC_NWAIT returns to the ready state before data transfer.

24.4.2.1 Multiplexed mode

As configured in *Table 24-7* and *Table 24-8*, the XMC uses multiplexed mode to access the external memory. The timing of read operation is shown in *Figure 24-3*. The timing of write operation is shown in *Figure 24-4*.

Table 24-7 Multiplexed mode — SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x0
Bit 18: 16	CRPGS: CRAM page size	0x0
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	Configure according to memory specifications.
Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	0x0
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	0x0
Bit 10	WRAPEN: Wrapped enable	0x0
Bit 9	NWPOL: NWAIT polarity	Configure according to memory specifications.
Bit 8	SYNCBEN: Synchronous burst enable	0x0
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	Configure according to memory specifications.
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Configure according to memory specifications. It is valid except 0x0 (SRAM)
Bit 1	ADMUXEN: Address/data multiplexing enable	0x1
Bit 0	EN: Memory bank enable	0x1

Table 24-8 Multiplexed mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG) configuration

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	0x0
Bit 23: 20	CLKPSC: Clock frequency division factor	0x0
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	Refer to *Figure 24-3* and *Figure 24-4*. Configure according to needs and memory specifications.
Bit 7: 4	ADDRHT: Address-hold time	Refer to *Figure 24-3* and *Figure 24-4*. Configure according to needs and memory specifications.
Bit 3: 0	ADDRST: Address setup time	Refer to *Figure 24-3* and *Figure 24-4*. Configure according to needs and memory specifications.

Figure 24-3 NOR/PSRAM multiplexed mode read access

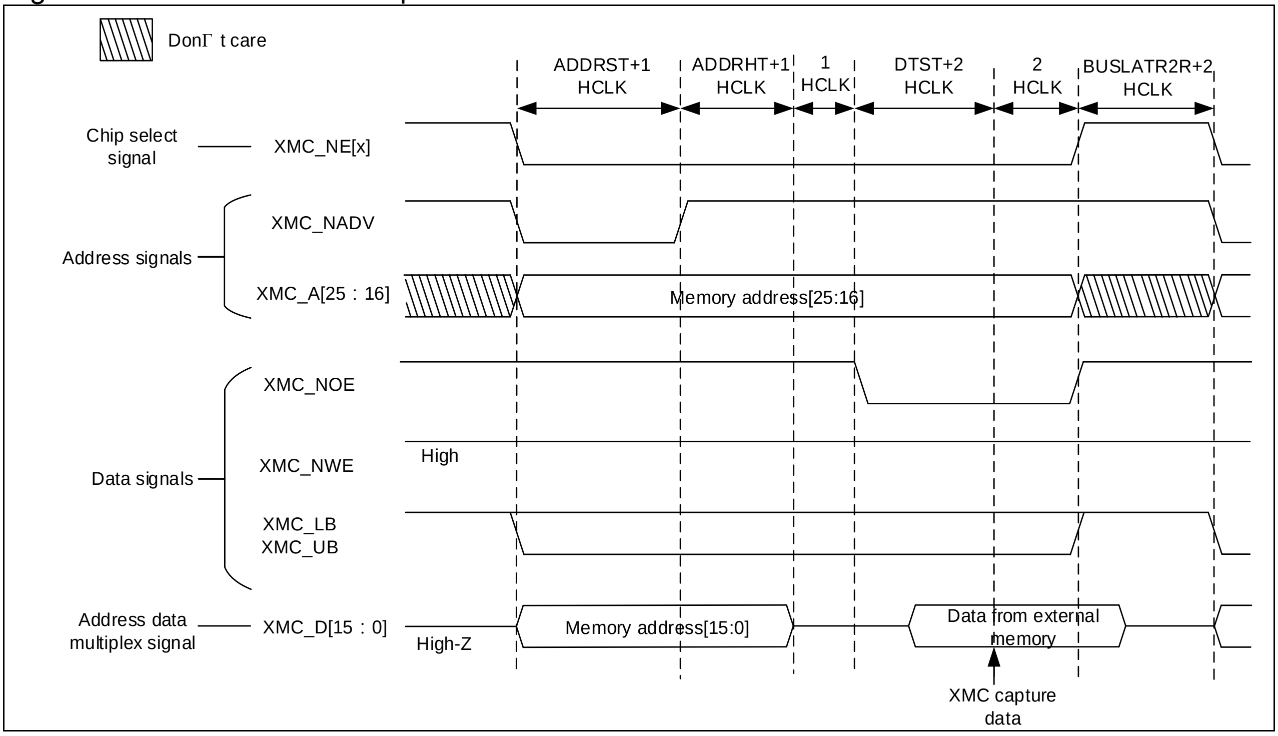
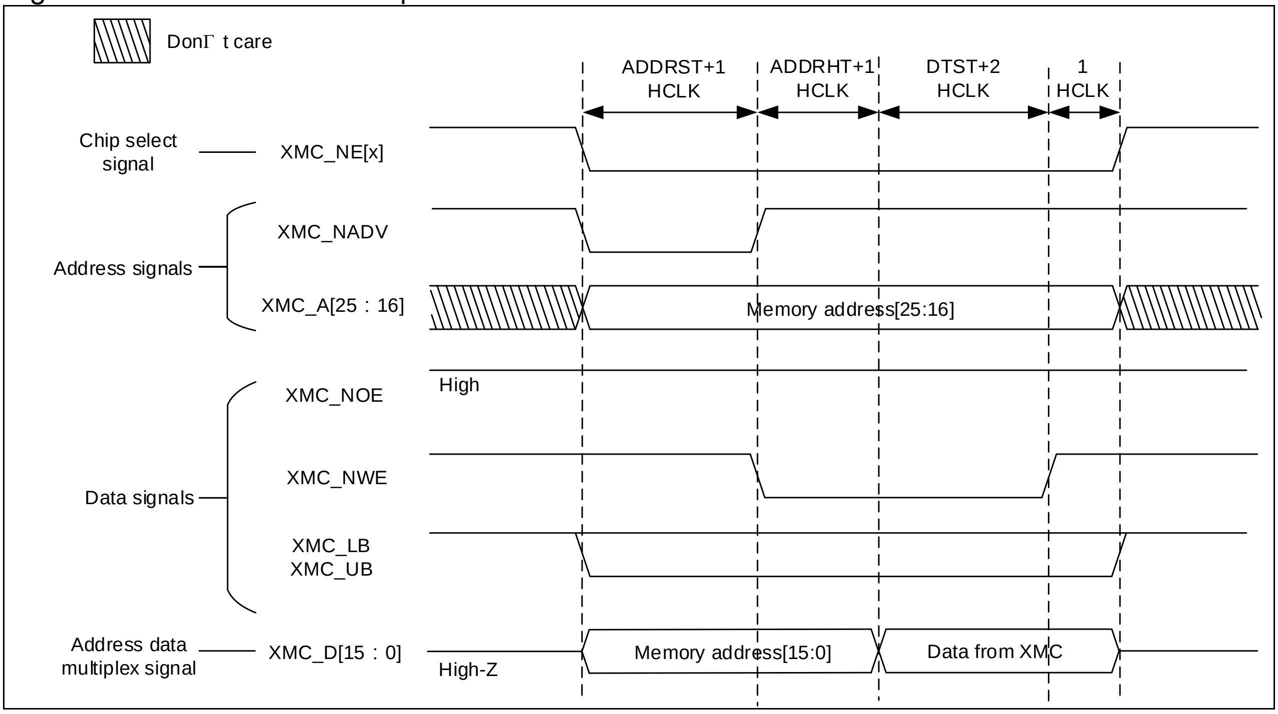


Figure 24-4 NOR/PSRAM multiplexed mode write access



24.4.2.2 Synchronous mode

As configured in *Table 24-9* and

Table 24-10, the XMC uses synchronous mode to access the external memories.

If the memory inserted XMC_NWAIT signal between the address latch and data transfer, the XMC would not only need wait (DTLAT+1) XMC_CLK clock cycles but also have to take into account the XMC_NWAIT signal. In the process of data transmission, the XMC needs wait at the current cycle when wait signal is active, or wait at the next cycle, depending on the NWTCFG bit in the XMC_BK1CTRL1 register.

Figure 24-5 shows an example of read access timing, while *Figure 24-6* shows write access timing. *Figure 24-5* and *Figure 24-6* examples are based on NWTCFG=0 (XMC waits at the next cycle after XMC_NWAIT signal is active)

Table 24-9 Synchronous mode — SRAM/NOR Flash chip select control register

Bit	Description	Configuration
Bit 31: 20	Reserved	0x0
Bit 19	MWMC: Memory write mode control	0x1
Bit 18: 16	CRPGS: CRAM page size	Configure according to memory specifications.
Bit 15	NWASEN: NWAIT in asynchronous transfer enable	0x0
Bit 14	RWTD: Read-write timing different	0x0
Bit 13	NWSEN: NWAIT in synchronous transfer enable	Configure according to memory specifications.
Bit 12	WEN: Write enable	Configure according to needs.
Bit 11	NWTCFG: NWAIT timing configuration	Configure according to memory specifications.
Bit 10	WRAPEN: Wrapped enable	Configure according to needs.
Bit 9	NWPOL: NWAIT polarity	c
Bit 8	SYNCBEN: Synchronous burst enable	0x1
Bit 7	Reserved	0x1
Bit 6	NOREN: NOR Flash access enable	Write synchronization: 0x0 Read synchronization: Configure according to memory specifications.
Bit 5: 4	EXTMDBW: External memory data bus width	Configure according to memory specifications.
Bit 3: 2	DEV: Memory device type	Write synchronization: 0x1 Read synchronization: Configure according to memory specifications. It is valid except 0x0 (SRAM)

Bit 1	ADMUXEN: Address/data multiplexing enable	Configure according to needs.
Bit 0	EN: Memory bank enable	0x1

Table 24-10 Synchronous mode—SRAM/NOR Flash chip select timing register (XMC_BK1TMG)

Bit	Description	Configuration
Bit 31: 30	Reserved	0x0
Bit 29: 28	ASYNCM: Asynchronous mode	0x0
Bit 27: 24	DTLAT: Data latency	Refer to *Figure 24-5* and *Figure 24-6*
Bit 23: 20	CLKPSC: Clock frequency division factor	XMC_CLK cycle is HCLK cycle*(CLKPSC+1). Refer to *Figure 24-5* and *Figure 24-6*
Bit 19: 16	BUSLAT: Bus latency	Indicates the time the XMC_NE[x] from the rising edge to the falling edge. Configure according to needs and memory specifications
Bit 15: 8	DTST: Data setup time	0x0
Bit 7: 4	ADDRHT: Address-hold time	0x0
Bit 3: 0	ADDRST: Address setup time	0x0

Figure 24-5 NOR/PSRAM synchronous multiplexed mode read access

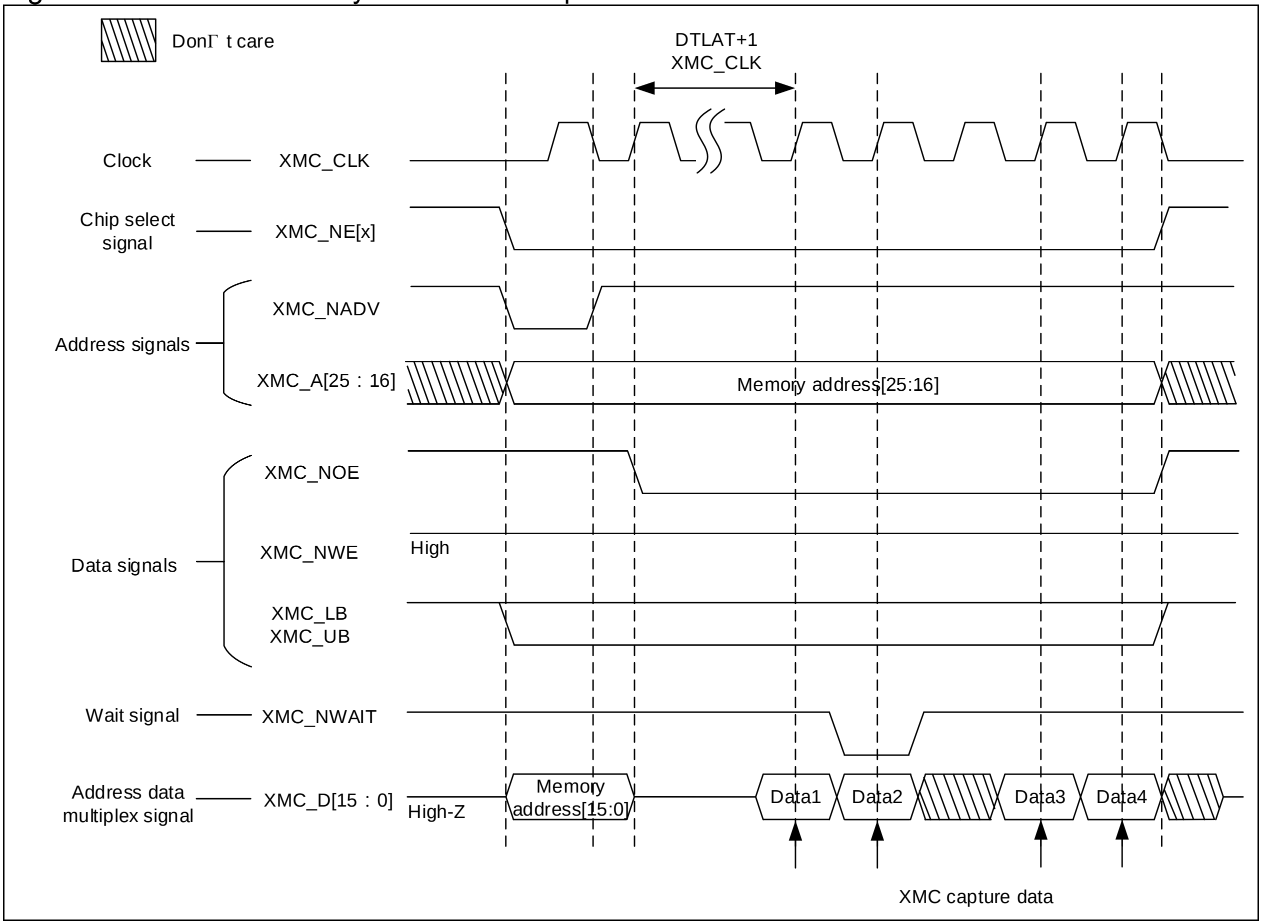
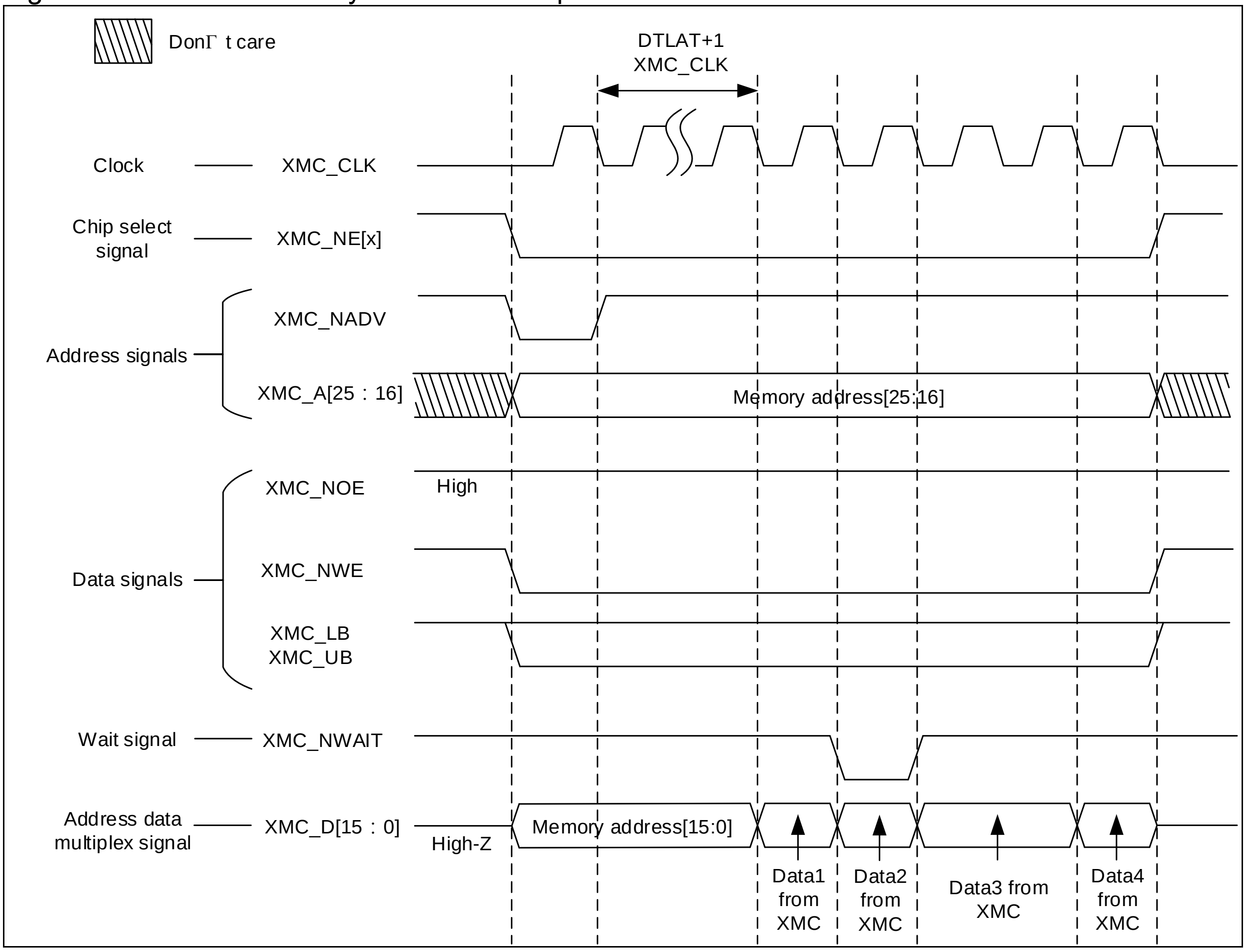


Figure 24-6 NOR/PSRAM synchronous multiplexed mode write access



24.5 XMC registers

These peripheral registers have to be accessed by word (32 bits).

Table 24-11 XMC register address mapping

Register name	Offset	Reset value
XMC_BK1CTRL1	0x000	0x0000 30DB
XMC_BK1TMG1	0x004	0x0FFF FFFF
XMC_BK1CTRL2	0x008	0x0000 30D2
XMC_BK1TMG2	0x00C	0x0FFF FFFF
XMC_BK1CTRL4	0x018	0x0000 30D2
XMC_BK1TMG4	0x01C	0x0FFF FFFF
XMC_BK1TMGWR1	0x104	0x0FFF FFFF
XMC_BK1TMGWR2	0x10C	0x0FFF FFFF
XMC_BK1TMGWR4	0x11C	0x0FFF FFFF
XMC_EXT1	0x220	0x0000 0808
XMC_EXT2	0x224	0x0000 0808
XMC_EXT4	0x22C	0x0000 0808

24.5.1 NOR Flash and PSRAM control registers

These peripherals registers have to be accessed by words (32 bits).

24.5.1.1 SRAM/NOR Flash chip select control register 1 (XMC_BK1CTRL1)

Accessed by words

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at default value.
Bit 19	MWMC	0x0	rw	Memory write mode control 0: Write operations are performed in asynchronous mode 1: Write operations are performed in synchronous mode
Bit 18: 16	CRPGS	0x0	rw	CRAM page size Cellular RAM 1.5 does not allow synchronous access to cross the address boundaries between pages. When these bits are configured in synchronous mode, the XMC will automatically split the access when the page size is reached. 000: No split access when crossing address boundary (default value) 001: 128 bytes 010: 256 bytes 011: 512 bytes 100: 1024 bytes Others: Reserved.
Bit 15	NWASEN	0x0	rw	NWAIT in asynchronous transfer enable 0: NWAIT signal is disabled 1: NWAIT signal is enable
Bit 14	RWTD	0x0	rw	Read-write timing different Different timings are used for read and write operations, that is, the XMC_BK1TMGWR register is released. 0: Same timings for read and write operations 1: Different timings for read and write operations
Bit 13	NWSEN	0x1	rw	NWAIT enable during synchronous transfer 0: NWAIT signal is disabled 1: NWAIT signal is enabled
Bit 12	WEN	0x1	rw	Write enable 0: Disabled 1: Enabled
Bit 11	NWTCFG	0x0	rw	NWAIT timing configuration It is valid only in synchronous mode. 0: NWAIT signal is active one data cycle before the wait state 1: NWAIT signal is active one data cycle during the wait state
Bit 10	WRAPEN	0x0	rw	Wrapped enable This bit defines whether the XMC will split a wrapped AHB access into two accesses. 0: Direct wrapped access is not allowed 1: Direct wrapped access is allowed
Bit 9	NWPOL	0x0	rw	NWAIT polarity This bit defines the polarity of the NWAIT signal in synchronous mode. 0: NWAIT active low 1: NWAIT active high
Bit 8	SYNCBEN	0x0	rw	Synchronous burst enable This bit allows synchronous access to Flash memories. 0: Synchronous burst disabled 1: Synchronous burst enabled
Bit 7	Reserved	0x1	resd	Kept at default value.
Bit 6	NOREN	0x1	rw	Nor flash access enable 0: Nor flash access is disabled 1: Nor flash access is enabled
Bit 5: 4	EXTMDBW	0x1	rw	External memory data bus width

				This field defines the external memory data bus width. 00: 8 bits 01: 16 bits 10: Reserved 11: Reserved
Bit 3: 2	DEV	0x2	rw	Memory device type 00: SRAM/ROM 01: PSRAM (Cellular RAM or CRAM) 10: NOR Flash 11: Reserved
Bit 1	ADMUXEN	0x1	rw	Address/data multiplexing enable This bit must be set to 1. 0: Address/data not multiplexed 1: Address/data multiplexed
Bit 0	EN	0x1	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled

24.5.1.2 SRAM/NOR Flash chip select control register x (x=2, 4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 20	Reserved	0x000	resd	Kept at default value.
Bit 19	MWMC	0x0	rw	Memory write mode control 0: Write operations are performed in asynchronous mode 1: Write operations are performed in synchronous mode
Bit 18: 16	CRPGS	0x0	rw	CRAM page size Cellular RAM 1.5 does not allow synchronous access to cross the address boundaries between pages. When these bits are configured in synchronous mode, the XMC will automatically split the access when the page size is reached. 000: No split access when crossing address boundary (default value) 001: 128 bytes 010: 256 bytes 011: 512 bytes 100: 1024 bytes Others: Reserved.
Bit 15	NWASEN	0x0	rw	NWAIT enable during asynchronous transfer 0: NWAIT signal is disabled 1: NWAIT signal is enable
Bit 14	RWTD	0x0	rw	Read-write timing different Different timings are used for read and write operations, that is, the XMC_BK1TMGWR register is released. 0: Same timings for read and write operations 1: Different timings for read and write operations
Bit 13	NWSEN	0x1	rw	NWAIT enable during synchronous transfer 0: NWAIT signal is disabled 1: NWAIT signal is enabled
Bit 12	WEN	0x1	rw	Write enable 0: Disabled 1: Enabled
Bit 11	NWTCFG	0x0	rw	NWAIT timing configuration It is valid only in synchronous mode. 0: NWAIT signal is active one data cycle before the wait state 1: NWAIT signal is active during the wait state
Bit 10	WRAPEN	0x0	rw	Wrapped enable This bit defines whether the XMC will split a wrapped AHB access into two accesses. 0: Direct wrapped access is not allowed 1: Direct wrapped access is allowed

Bit 9	NWPOL	0x0	rw	NWAIT polarity This bit defines the polarity of the NWAIT signal in synchronous mode. 0: NWAIT active low 1: NWAIT active high
Bit 8	SYNCBEN	0x0	rw	Synchronous burst enable This bit allows synchronous access to Flash memories. 0: Synchronous burst disabled 1: Synchronous burst enabled
Bit 7	Reserved	0x1	resd	Kept at default value.
Bit 6	NOREN	0x1	rw	Nor flash access enable 0: Nor flash access is disabled 1: Nor flash access is enabled
Bit 5: 4	EXTMDBW	0x1	rw	External memory data bus width This field defines the external memory data bus width. 00: 8 bits 01: 16 bits 10: Reserved 11: Reserved
Bit 3: 2	DEV	0x0	rw	Memory device type 00: SRAM/ROM 01: PSRAM (Cellular RAM or CRAM) 10: NOR Flash 11: Reserved
Bit 1	ADMUXEN	0x1	rw	Address/data multiplexing enable This bit must be set to 1. 0: Address/data not multiplexed 1: Address/data multiplexed
Bit 0	EN	0x0	rw	Memory bank enable 0: Memory bank disabled 1: Memory bank enabled

24.5.1.3 SRAM/NOR Flash chip select timing register x (x=1,2,4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at default value.
Bit 29: 28	ASYNCM	0x0	rw	Asynchronous mode This field is valid only when the RWTD bit is enabled. 00: Mode A 01: Mode B 10: Mode C 11: Mode D
Bit 27: 24	DTLAT	0xF	rw	Data latency This field is valid only in synchronous mode. 0000: 0 XMC_CLK cycle is inserted 0001: 1 additional XMC_CLK cycle is inserted …… 1111: 15 additional XMC_CLK cycles are inserted
Bit 23: 20	CLKPSC	0xF	rw	Clock prescaler This field is valid only in synchronous mode. It defines the frequency of the XMC_CLK clock. 0000: Reserved 0001: XMC_CLK cycle= 2 x HCLK clock cycles 0010: XMC_CLK cycle =3 x HCLK clock cycles …… 1111: XMC_CLK cycle = 16 x HCLK cycles
Bit 19: 16	BUSLAT	0xF	rw	Bus latency To avoid data bus conflict, a latency is inserted on the data bus after one read operation in multiplexed or synchronous mode. 0000: 1 HCLK cycle is inserted 0001: 2 HCLK cycles are inserted

				…… 1111: 16 HCLK cycles are inserted
Bit 15: 8	DTST	0xFF	rw	Data setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 7: 4	ADDRHT	0xF	rw	Address-hold time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 3: 0	ADDRST	0xF	rw	Address setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted

24.5.1.4 SRAM/NOR Flash write timing register x (x=1,2,4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 30	Reserved	0x0	resd	Kept at default value.
Bit 29: 28	ASYNCM	0x0	rw	Asynchronous mode This field is valid only when the RWTD bit is enabled. 00: Mode A 01: Mode B 10: Mode C 11: Mode D
Bit 27: 20	Reserved	0xFF	resd	Kept at default value.
Bit 19: 16	BUSLAT	0xF	rw	Bus latency To avoid data bus conflict, a latency is inserted on the data bus after one read operation in multiplexed or synchronous mode. 0000: 1 HCLK cycle is inserted 0001: 2 HCLK cycles are inserted …… 1111: 16 HCLK cycles are inserted
Bit 15: 8	DTST	0xFF	rw	Data setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 7: 4	ADDRHT	0xF	rw	Address-hold time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted
Bit 3: 0	ADDRST	0xF	rw	Address setup time 0000: 0 HCLK cycle is inserted 0001: 1 additional HCLK cycle is inserted …… 1111: 15 additional HCLK cycles are inserted

24.5.1.5 SRAM/NOR Flash extra timing register x(XMC_EXTx) (x=1,2, 4)

Accessed by words.

Bit	Register	Reset value	Type	Description
Bit 31: 16	Reserved	0x0000	resd	Kept at default value.
Bit 15: 8	BUSLATR2R	0x08	rw	Bus turnaround phase for consecutive read duration This field is used to define the bus turnaround phase duration for consecutive read operations. A delay is inserted between two consecutive read operations in order to avoid bus conflicts. 00000000: 1 HCLK cycle is inserted for consecutive read

				operations 00000001: 2 HCLK cycles are inserted for consecutive read operations …… 00001000: 9 HCLK cycles are inserted for consecutive read operations (default value) …… 11111111: 256 HCLK cycles are inserted for consecutive read operations
Bit 7: 0	BUSLATW2W	0x08	rw	Bus turnaround phase for consecutive write duration This field is used to define the bus turnaround phase duration for consecutive write operations. A delay is inserted between two consecutive write operations in order to avoid bus conflicts. 00000000: 1 HCLK cycle is inserted for consecutive write operations 00000001: 2 HCLK cycles are inserted for consecutive write operations …… 00001000: 9 HCLK cycles are inserted for consecutive write operations (default value) …… 11111111: 256 HCLK cycles are inserted for consecutive write operations

25 Debug (DEBUG)

25.1 Debug introduction

Cortex®-M4F 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.

ARM Cortex®-M4F reference documentation:

- Cortex®-M4 Technical Reference Manual (TRM)
- ARM Debug Interface V5
- ARM CoreSight Design Kit revision r1p0 Technical Reference Manual

25.2 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.

25.3 I/O pin control

The AT32F423 uses its two general-purpose I/O ports for SW-DP debugging. After reset, the SW-DP can be immediately used by the debugger by default.

It uses 5 general-purpose I/O ports. After reset, the SWJ-DP can be immediately used by the debugger as a default function. To ensure that JTAG input pins are not floating (particularly SWCLK/JTCK), NJTRST, JTDI and JTMS/SWDIO have enabled their internal pull-up feature each by default, while JTCK/SWCLK have enabled their internal pull-down feature each.

GPIO and IOMUX registers can be configured to allow users to switch between debug ports or disable debug feature.

25.4 DEGUB registers

Table 25-1 shows DEBUG register map and reset values.

These peripheral registers must be accessed by word (32 bits)

Table 25-1 DEBUG register address and reset value

Register name	Offset	Reset value
DEBUG_IDCODE	0xE004 2000	0xXXXX XXXX
DEBUG_CTRL	0xE004 2004	0x0000 0000
DEBUG_APB1_PAUSE	0xE004 2008	0x0000 0000
DEBUG_APB2_PAUSE	0xE004 200C	0x0000 0000

25.4.1 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
0x700A_3240	AT32F423VCT7	256KB	LQFP100
0x700A_21C1	AT32F423VBT7	128KB	LQFP100
0x7003_2102	AT32F423V8T7	64KB	LQFP100
0x700A_3243	AT32F423RCT7	256KB	LQFP64
0x700A_21C4	AT32F423RBT7	128KB	LQFP64
0x7003_2105	AT32F423R8T7	64KB	LQFP64
0x700A_3246	AT32F423RCT7-7	256KB	LQFP64
0x700A_21C7	AT32F423RBT7-7	128KB	LQFP64
0x7003_2108	AT32F423R8T7-7	64KB	LQFP64
0x700A_3249	AT32F423CCT7	256KB	LQFP48
0x700A_21CA	AT32F423CBT7	128KB	LQFP48
0x7003_210B	AT32F423C8T7	64KB	LQFP48
0x700A_324C	AT32F423CCU7	256KB	QFN48
0x700A_21CD	AT32F423CBU7	128KB	QFN48
0x7003_210E	AT32F423C8U7	64KB	QFN48
0x700A_3250	AT32F423TCU7	256KB	QFN36
0x700A_21D1	AT32F423TBU7	128KB	QFN36
0x7003_2112	AT32F423T8U7	64KB	QFN36
0x700A_3253	AT32F423KCU7-4	256KB	QFN32
0x700A_21D4	AT32F423KBU7-4	128KB	QFN32
0x7003_2115	AT32F423K8U7-4	64KB	QFN32

25.4.2 DEBUG control register (DEBUG_CTRL)

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 31:3	Reserved	0x0000 0000	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.

25.4.3 DEBUG APB1 pause register (DEBUG_ APB1_PAUSE)

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: 29	Reserved	0x0	resd	Kept at default value.
Bit 28	I2C3_SMBUS_TIMEOUT	0x0	rw	I2C3 pause control bit 0: I2C3 SMBUS timeout control works normally 1: I2C3 SMBUS timeout control stops running
Bit 27	I2C2_SMBUS_TIMEOUT	0x0	rw	I2C2 pause control bit 0: I2C2 SMBUS timeout control works normally 1: I2C2 SMBUS timeout control stops running
Bit 26	CAN2_PAUSE	0x0	rw	CAN2 pause control bit 0: CAN2 works normally 1: CAN2 receive register pauses (does not receive data)
Bit 25	CAN1_PAUSE	0x0	rw	CAN1 pause control bit 0: CAN1 works normally 1: CAN1 receive register pauses (does not receive data)
Bit 24	I2C1_SMBUS_TIMEOUT	0x0	rw	I2C1 pause control bit 0: I2C1 SMBUS timeout control works normally 1: I2C1 SMBUS timeout control stops running
Bit 23: 16	Reserved	0x00	resd	Kept at default value.
Bit 15	ERTC_512_PAUSE	0x0	rw	ERTC 512Hz output clock pause control bit 0: ERTC 512Hz output clock works normally 1: Froze 512Hz output clock
Bit 14: 13	Reserved	0x0	rw	Kept at default value.
Bit 12	WDT_PAUSE	0x0	rw	WDT pause control bit 0: WDT works normally 1: WDT stops running

Bit 11	WWDT_PAUSE	0x0	rw	WWDT pause control bit 0: WWDT works normally 1: WWDT stops running
Bit 10	ERTC_PAUSE	0x0	rw	ERTC pause control bit 0: ERTC works normally 1: ERTC stops running
Bit 9	Reserved	0x0	rw	Kept at default value.
Bit 8	TMR14_PAUSE	0x0	rw	TMR14 pause control bit 0: TMR14 works normally 1: TMR14 stops running
Bit 7	TMR13_PAUSE	0x0	rw	TMR13 pause control bit 0: TMR13 works normally 1: TMR13 stops running
Bit 6	TMR12_PAUSE	0x0	rw	TMR12 pause control bit 0: TMR12 works normally 1: TMR12 stops running
Bit 5	TMR7_PAUSE	0x0	rw	TMR7 pause control bit 0: TMR7 works normally 1: TMR7 stops running
Bit 4	TMR6_PAUSE	0x0	rw	TMR6 pause control bit 0: TMR6 works normally 1: TMR6 stops running
Bit 3	Reserved	0x0	rw	Kept at default value.
Bit 2	TMR4_PAUSE	0x0	rw	TMR4 pause control bit 0: TMR4 works normally 1: TMR4 stops running
Bit 1	TMR3_PAUSE	0x0	rw	TMR3 pause control bit 0: TMR3 works normally 1: TMR3 stops running
Bit 0	TMR2_PAUSE	0x0	rw	TMR2 pause control bit 0: TMR2 works normally 1: TMR2 stops running

25.4.4 DEBUG APB2 pause register (DEBUG_ APB2_PAUSE)

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: 19	Reserved	0x0000	resd	Kept at default value.
Bit 18	TMR11_PAUSE	0x0	rw	TMR11 pause control bit 0: TMR11 works normally 1: TMR11 stops running
Bit 17	TMR10_PAUSE	0x0	rw	TMR10 pause control bit 0: TMR10 works normally 1: TMR10 stops running
Bit 16	TMR9_PAUSE	0x0	rw	TMR9 pause control bit 0: TMR9 works normally 1: TMR9 stops running
Bit 15: 1	Reserved	0x000	resd	Kept at default value.
Bit 0	TMR1_PAUSE	0x0	rw	TMR1 pause control bit 0: TMR2 works normally 1: TMR2 stops running

26 Revision history

Document Revision History

Date	Version	Revision Note
2023.3.28	2.00	Initial release.
2023.4.25	2.01	Updated description of the first page.
2023.08.02	2.02	1. Updated descriptions of *section 12.1 USART introduction and 12.8.3 Start bit and noise detection* 2. Updated descriptions of some sections of chapter *14 Timer*