PADAUK 應廣科技

PEC930

RISC-V Motor-Specific ASSP 32-bit BLDC MCU

User Manual

Version 0.00

April. 01,2026

6F-6, No.1, Sec. 3, Gongdao 5th Rd., Hsinchu City 30069, Taiwan, R.O.C.

TEL: 886-3-572-8688 www.padauk.com.tw

IMPORTANT NOTICE

PADAUK Technology reserves the right to make changes to its products or to terminate production of its products at any time without notice. Customers are strongly recommended to contact PADAUK Technology for the latest information and verify whether the information is correct and complete before placing orders.

PADAUK Technology products are not warranted to be suitable for use in life-support applications or other critical applications. PADAUK Technology assumes no liability for such applications. Critical applications include, but are not limited to, those which may involve potential risks of death, personal injury, fire or severe property damage.

PADAUK Technology assumes no responsibility for any issue caused by a customer's product design. Customers should design and verify their products within the ranges guaranteed by PADAUK Technology. In order to minimize the risks in customers' products,customers should design a product with adequate operating safeguards.

Table of contents

List of figures

List of tables

Revision History

Version	Date	Discription
0.00	2026/04/01	Preliminary version.

Warning

User must read all application notes of the IC by detail before using it. Please download the related application notes from the following link:https://www.padauk.com.tw/en/product/search_list.aspx?kw=PEC

1. General Description

The ultra-low-power PEC930 adopts a high-performance 32-bit RISC-V microcontroller, operating up to 60 MHz. It integrates high-speed embedded memory (up to 4 KB SRAM and up to 32 KB Flash for program/data storage). With built-in 4 KB SRAM supporting 0-wait execution at up to 60 MHz, it provides efficient program operation.

Integrated peripherals include:

- 12-bit SAR ADC (<1 Msps, 16 channels)
- Dual PGAs, dual comparators, dual 8-bit DACs
- Basic timers (TIM0, TIM1), advanced timers(EPWM, TIM2), and low-power timer(LPTIM)
- Multiple communication interfaces: UART, SPI, I2C
- EPWM with up to 3 independent or 3 complementary outputs
- DSP hardware acceleration: 32-bit signed divider, 32-bit unsigned square root unit

With rich peripheral integration, high reliability, strong anti-interference, and compact design, the PEC930 MCU series is well-suited for applications such as three-phase BLDC motor drive control and single-phase BLDC motor drive control.

2. Feature

2.1. CPU Features

- 32-bit RISC-V CPU core, supporting the RV32E/M/C extension instruction sets.
- Sixteen 32-bit general-purpose registers
- Efficient 2-stage execution pipeline
- Built-in single-cycle 32×32-bit hardware multiplier array
- Built-in 32-bit hardware divider
- Built-in 64-bit system timer
- CJTAG/JTAG debugging interface

2.2. Enhanced Core Local Interrupt Controller (ECLIC)

- Supports 2 core-specified internal interrupts, such as software interrupts and timer interrupts.
- Supports 15 external interrupts
- Support software dynamically programmable division of interrupt levels and priorities values
- Support interrupts preemption based on interrupt levels

2.3. Power management

- Provides low-power modes: Sleep and Deep Sleep
- Sleep mode current: 3.3mA(Typ), SYSCLK=60MHz wake-up time < 2 μs
- Deep Sleep mode current: < 5μA(Typ)@Ta=25°C
- POR, PDR, LVR integrated
- Low-voltage detection, configurable as interrupt or reset
- Wake-up from Sleep: all interrupt sources available
- Wake-up from Deep Sleep: all external interrupt GPIO pins, WDG (running on SIRC), low-power wake-up timer

2.4. On-Chip Memory

- Instruction Memory32KB FLASH ,8KB NVR
- Data Memory: 4KB SRAM

2.5. Safety Mechanisms

- On-chip watchdog timer (WDG), configurable clock source for counting/timing
- Low-voltage monitor: generates interrupt or reset when supply voltage drops below safety threshold

2.6. UART

- 1 channel
- Flexible programmable baud rate settings, supporting industry standards: 9600 bps, 19200 bps, 28800 bps, 38400 bps, 57600 bps, 115.2 kbps, etc., or custom baud rates for special applications
- Programmable data transfer formats: 8-bit data, 7-bit data + parity, 8-bit data + parity, 9-bit data
- Configurable parity modes: odd/even
- Configurable stop bits: 0.5, 1, 1.5, or 2 bits
- Full-duplex transmission and reception
- Hardware error detection

2.7. SPI

- 1 channel
- Configurable transfer word size: supports 8/16/24/32-bit data size
- Supports both LSB-first and MSB-first transmission
- Maximum master transfer speed: 10 Mbps (system clock > 20 MHz)

2.8. I2C

- 1 channel
- Supports SMBus
- Supports multi-master I2C bus, configurable as master or slave mode
- Standard mode 100 kbps, Fast mode up to 400 kbps, High-speed mode up to 1 Mbps.

2.9. High Resolution ADC

- Power consumption < 130 μA/MHz in normal operation
- 12-bit SAR ADC, sampling rate < 1 Msps
- 16 channels: 12 external input pins, 1 internal temperature sensor, 2 PGA outputs, 1 internal reference voltage (1.5 V)
- External reference voltage: VDD
- Trigger sources: Software trigger, Hardware triggers: Timer (overflow), EPWM (overflow, underflow, rising/falling edge), COMP

2.10. Timer

- TIM0/1: 16-bit timers with prescaler
- LPTIM: 16-bit timer with prescaler, selectable clock source, can operate in Sleep/DeepSleep mode
- TIM2: 20-bit timer with prescaler, supports up-count, down-count, center-aligned (symmetric and asymmetric) modes, capable of generating 4 PWM outputs
- All standard peripherals have enable/disable control; when disabled, power consumption is negligible

2.11. Enhanced EPWM

- 20-bit timer with prescaler, supports up-count, down-count, center-aligned modes (symmetric/asymmetric), capable of generating 6 PWM outputs in 3 complementary pairs
- Standard peripherals are equipped with enable controls; when disabled, power consumption is negligible.
- Supports two aligned modes: edge-aligned and center-aligned.
- Center-aligned mode supports both symmetric and asymmetric counting
- Complementary PWM supports programmable dead-time generator
- PWM edge or period events can trigger ADC conversion
- Brake protection sources:
  - Ext External BKIN signal (high/low level)
  - Comparator 0/1 outputs (high/low level)
  - ADC value comparison (any two channels can be selected as brake source)
  - Low-voltage detection
  - WDG overflow event
  - CPU exception eventernal

2.12. OPAMP (PGA0/1)

- Selectable gain: 1X~16X
- PGA output/input:
  - Output to ADC channel
  - Output to comparator
  - Output to pin
  - Built-in clamp diodes at amplifier inputs (non-inverting), motor phase current can be directly fed through matching resistors into input, simplifying MOSFET current-sampling circuitry

2.13. DSP hardware accelerator module

- Built-in 32 -bit signed divider hardware
- Built-in 32 -bit unsigned square root hardware

2.14. Comparator (COMP0/1)

- Positive inputs: Multi-Channals Seltect
- Negative input: Port or DAC
- Supports hysteresis voltage
- Comparator outputs can trigger EPWM brake

2.15. DAC (8 bit) X2

- VREFH: VDD
- Multiple selectable output levels

2.16. Clock

- HIRC: 60 MHz, +5% ~ -2.5% accuracy(-40°C ~ 85°C), +3.5% ~ -2.5% accuracy(0°C ~ 85°C)
- SIRC: 32KHz，32 kHz, ±40% accuracy, ultra-low power (0.5 μA), available as WDG and LPTIM clock source
- Frequency divider: supports integer division of internal 60 MHz RC clock

2.17. GPIO

- Up to 22 pins support external interrupts
- GPIO support Digital/Analog alternate functions

2.18. Internal temperature sensor

- Build in thermal Sensor: -1.65mV/℃

2.19. Hardware CRC-16/32 module

- Supports Byte, Half-word, and Word operations
- Selectable CRC polynomials

2.20. 12bytes(96 bits)UID

- Provides 12-byte unique identifier for each chip

2.21. Debug mode

- Support JTAG and two-wire CJTAG debug interface

2.22. Operating Environment

- Operating Voltage: 2.5 ~ 5.5 V
- Operating temperature range: -40 ~ +85℃

2.23. Package Type

- PEC930-Y24ASSOP24(150MIL)
- PEC930-2J24AQFN24(4*4*0.75MM)

3. Functional Block Diagram

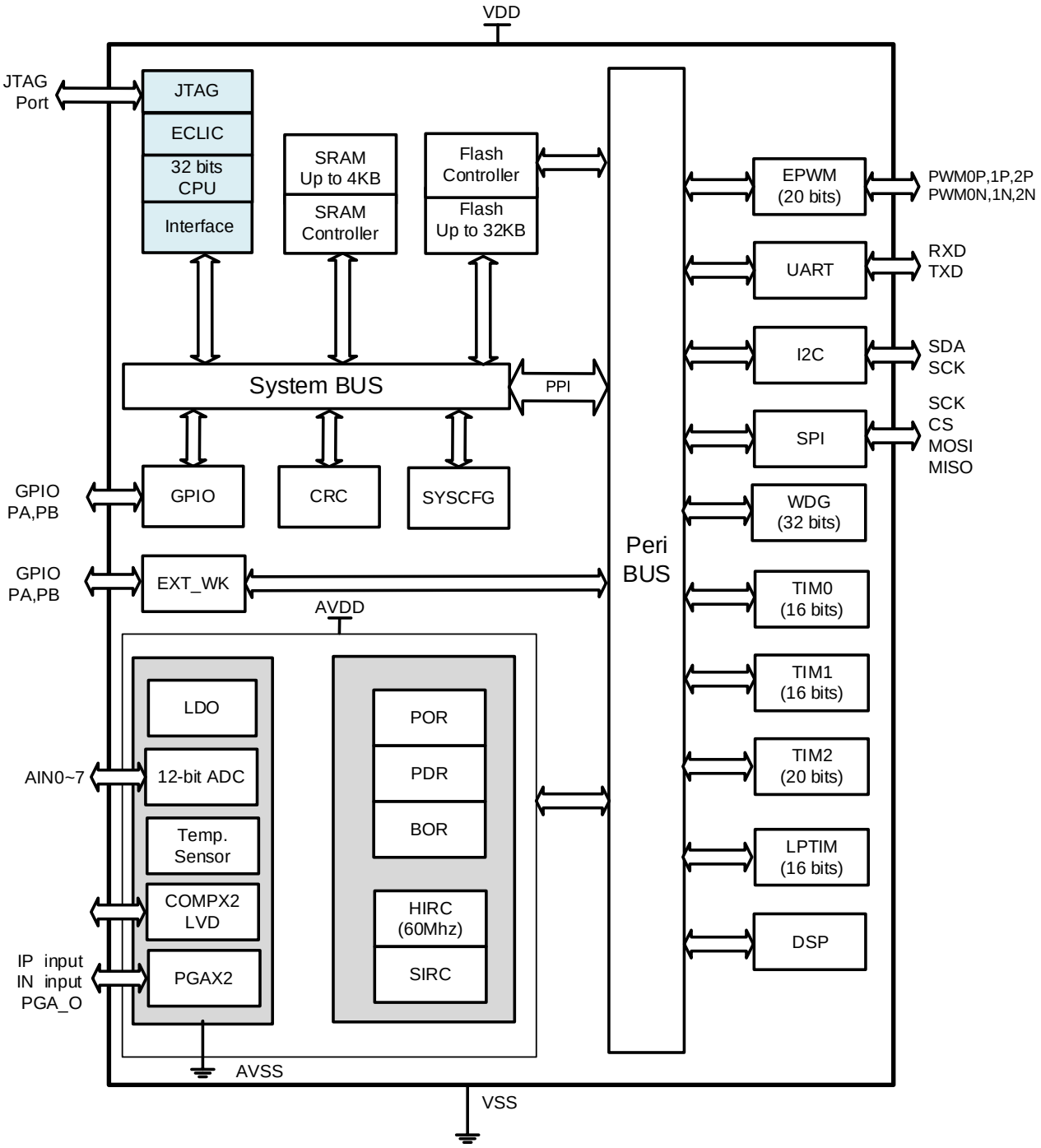


Fig. 3-1 PEC930 Block Diagram

3.1. CPU

The PEC930 integrates the latest generation of high-efficiency, low-power, and compact embedded RISC-V core solutions. It features a 32-bit processor core with an integrated hardware multiplier/divider. The main features of the core are as follows:

- 2-stage pipeline, single-issue processor architecture
- Supports both 2-wire CJTAG and standard JTAG debug interfaces
- Supports low-power Sleep/DeepSleep mode through standard WFI (Wait for Interrupt) and WFE (Wait for Event) mechanisms to achieve reduced dynamic and static power consumption

3.2. On-Chip Memory

The device has the following features:

- The built-in FLASH memory is divided into two arrays:
  - 32 KB Main area, used for storing programs and data.
  - 8 KB NVR area (7 KB for user data storage, 1 KB reserved for system use)
- Built-in 4KB SRAM, operable at a maximum of 60MHz.

3.3. CRC calculation unit

The CRC (Cyclic Redundancy Check) computation unit uses a fixed polynomial generator to produce a CRC code. Among other applications, CRC-based techniques are used to verify data transmission or storage integrity. The CRC calculation unit can be used to compute the CRC signature of frames or data packets during data transmission and compare it with the signature appended by the sender when the packet was generated, in order to verify whether the data has been corrupted or has encountered errors during communication.

3.4. Low Power Mode

The PEC930 series supports two low-power operating modes that provide an optimal balance between low power consumption, fast wake-up time, and flexible wake-up sources:

- Sleep Mode:

  In Sleep Mode, only the CPU is halted while all peripheral modules remain operational. The CPU can be awakened by any interrupt or event trigger.

- DeepSleep Mode:

  DeepSleep Mode achieves the lowest power consumption while preserving the contents of SRAM and CPU registers. In this mode, the internal HIRC oscillator is powered down, and the voltage regulator (LDO) can be switched to low-power mode. If the LIRC oscillator is disabled, the microcontroller can be awakened from deep sleep mode by any signal configured as an IO interrupt source; If the LIRC oscillator is enabled, the WDG or LPTIM can also be used for wake-up.

3.5. Internal temperature sensor

The temperature sensor generates a voltage that varies linearly with temperature, operating within a supply range of 2.5V < VDDA < 5.5V. The sensor output is internally connected to an ADC input channel, which converts the analog voltage signal from the temperature sensor into a corresponding digital value.

3.6. ADC

The chip integrates a 12-bit Analog-to-Digital Converter (ADC) with 16 channels, allowing the ADC to measure voltages from 12 external input pins as well as several internal voltage channels. The sampling interval for each conversion must be greater than 1 us.

3.7. SPI

The SPI (Serial Peripheral Interface) communication standard was developed by Motorola. It is based on a three-wire connection scheme that enables data communication between a microcontroller and peripheral devices, or between multiple microcontrollers. By using a chip select (CS) signal to control communication with slave devices, an SPI bus can be configured in a master-slave architecture.

3.8. Timer

Provides the following timer resources:

- 2x16-bit base Timer (TIM0,TIM1)
- 2x20-bit Enhanced Timer(TIM2, EPWM)
- 1x16-bit Low-Power Wake up Timer (LPTIM)
- 1x16-bit Watchdog Timer (WDG)

3.9. I2C

I2C is a commonly used serial communication bus in embedded system design. It is based on a two-wire interface consisting of SCL (Serial Clock) and SDA (Serial Data), enabling bidirectional data communication among multiple interconnected devices in a master-slave configuration.

3.10. EPWM

This module provides three PWM (Pulse Width Modulation) channels for motor driving, with the corresponding functional pins EPWM0P, EPWM0N, EPWM1, EPWM1N, EPWM2P, and EPWM2N.

Complementary PWM configuration:

(1) EPWM0P=EPWM_CH1, EPWM0N= EPWM_CH1N=~(EPWM_CH1)

(2) EPWM1P=EPWM_CH2, EPWM1N= EPWM_CH2N=~(EPWM_CH2)

(2) EPWM2P=EPWM_CH3, EPWM2N= EPWM_CH3N=~(EPWM_CH3)

3.11. UART

The device embeds one universal synchronous serial receivers/transmitters that communicate at speeds of up to 115.2 kbps.

3.12. ECLIC interrupt controller

The ECLIC (Enhanced Core Local Interrupt Controller) is responsible for managing multiple interrupt sources. All types of interrupts within the core (except for debug interrupts) are centrally managed by the ECLIC. The core supports both external interrupts and internal interrupts.

External Interrupt	IRQ Number	Exception or Interrupt	Priority (Default)	Program Addr Vector	Type
	0	Reserved	(Lowest)	MTVT +4*0	Reserved
	1	Reserved		MTVT +4*1	Reserved
	2	Reserved		MTVT +4*2	Reserved
	3	Machine Software interrupt		MTVT +4*3	level
	4	Reserved		MTVT +4*4	Reserved
	5	Reserved		MTVT +4*5	Reserved
	6	Reserved		MTVT +4*6	Reserved
	7	Machine Timer interrupt		MTVT +4*7	level
	8	Reserved		MTVT +4*8	Reserved
	9	Reserved		MTVT +4*9	Reserved
	10	Reserved		MTVT +4*10	Reserved
	11	Reserved		MTVT +4*11	Reserved
	12	Reserved		MTVT +4*12	Reserved
	13~16	Reserved		MTVT +4*(13~16)	Reserved
	17	Reserved		MTVT +4*17	Reserved
	18	Reserved		MTVT +4*18	Reserved
1	19	TIM0		MTVT +4*19	level
2	20	TIM1		MTVT +4*20	level
3	21	TIM2		MTVT +4*21	level
4	22	LPTIM		MTVT +4*22	level
5	23	WDG		MTVT +4*23	level
6	24	SPI		MTVT +4*24	level
7	25	UART		MTVT +4*25	level
8	26	I2C-		MTVT +4*26	level
9	27	GPIOA		MTVT +4*27	level
10	28	GPIOB		MTVT +4*28	level
11	29	COMP0		MTVT +4*29	level
12	30	COMP1		MTVT +4*30	level
13	31	ADC		MTVT +4*31	level
14	32	EPWM		MTVT +4*32	level
15	33	LVD	(Highest)	MTVT +4*33	level

Table 3-1 ECLIC interrupt number

3.13. DSP

The DSP hardware accelerator block includes:

- 32-bit Signed Divider
- 32-bit Unsigned Square Root Unit

3.14. CRC

Used to verify the correctness and integrity of data transmission or storage; the protocol can select CRC-16/32.

3.15. GPIO

Up to 22 General-Purpose I/O (GPIO) pins available.

Four port modes:

- High-impedance / Input buffer disabled mode
- Pull high/low input mode
- Open-Drain output mode
- Push-Pull output mode

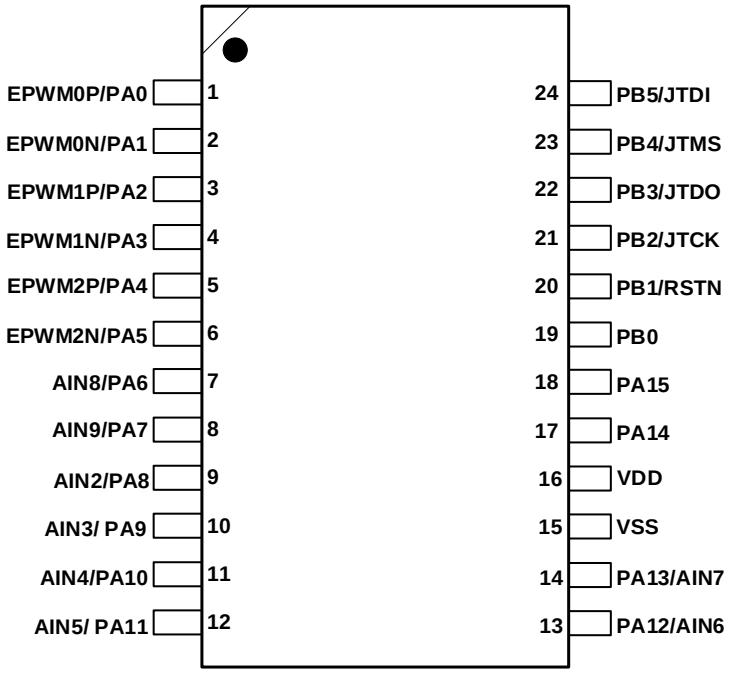
4. Device overview

Peripheral		PEC930
PIN		24
GPIO		22
CPU	Core	RISC-V 32-bit core
	Clock freq.	Max. 60 MHz
	Multiplier	32-bit (1T)
	Divider	32-bit (17T)
	SysTimer	64-bit
FLASH		32 KB
SRAM		4 KB
DSP		32-bit signed divider, 32-bit unsigned square root
Timer	Basic(16-bit)	TIM0,TIM1
	Enhanced(20-bit)	TIM2, EPWM
	Low Power(16-bit)	LPTIM
	Watchdog(32-bit)	WDG
Operating Voltage		2.5V ~ 5.5V
Operating Temperatures		-40℃ ~ 85℃
Debug Mode		CJTAG(2-wire)/JTAG(4-wire)
Unique Identifier(UID)		12 bytes
Communication	UART	1
	SPI	1
	I2C	1
CRC Verification		1 (CRC16/32)
Internal Temperature Sensor		1
Clock	HIRC	60 MHz
	SIRC	32 KHz
12-bit ADC		1 (16 CH)
Comparator(COMP)		2
OPAMP(PGA)		2
Package		SSOP24/QFN24

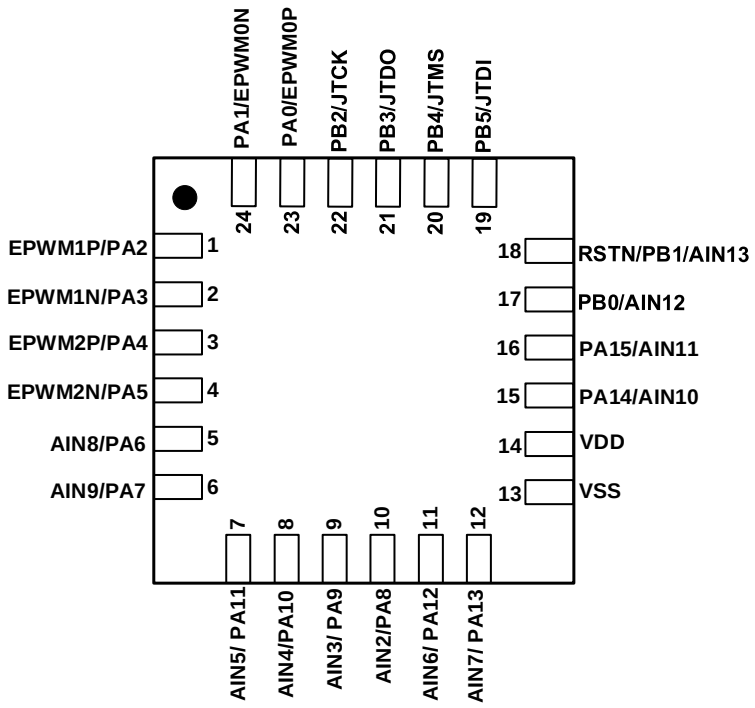
Table 4-1 PEC930 device features and peripheral counts

5. Pinouts and pin description

5.1. PIN Define



PEC930-Y24A: SSOP24(150MIL)



PEC930-2J24A: QFN24(4*4*0.75MM)

5.2. PIN Function Configuration

SSOP24 PIN Num.	PIN Name	PIN Type	Config	Default
1	PA0	I/O		
2	PA1	I/O		
3	PA2	I/O		
4	PA3	I/O		
5	PA4	I/O		
6	PA5	I/O		
7	PA6	I/O		
8	PA7	I/O		
9	PA8	I/O		
10	PA9	I/O		
11	PA10	I/O		
12	PA11	I/O		
13	PA12	I/O		
14	PA13	I/O		
15	VSS	GND		
16	VDD	Power		
17	PA14	I/O		
18	PA15	I/O		
19	PB0	I/O		
20	PB1	I/O	RSTN (POR latch)	RSTN
21	PB2	I/O	JTAG_TCK/CJ_TCK	JTAG_TCK/CJ_TCK
22	PB3	I/O	JTAG_TDO	JTAG_TDO
23	PB4	I/O	JTAG_TMS/CJ_TDIO	JTAG_TMS/CJ_TDIO
24	PB5	I/O	JTAG_TDI	JTAG_TDI

Table 5-1 PIN Function Configuration

5.3. PIN Alternate Function

PIN Num.		PIN Name	Analog Function			Digital Function						Special Function PIN
QFN 24	SSOP 24		ADC	PGA	COMP	UART	SPI	I2C	EPWM/ TIM2 in	BKIN	EPWM/ TIM2 out	
23	1	PA0				RXD/ TXD		SDA/ SCL			EPWM0P	
24	2	PA1				TXD/ RXD		SCL/ SDA			EPWM0N	
1	3	PA2					SCK				EPWM1P/ TIM2CH1	
2	4	PA3					MOSI/ MISO				EPWM1N/ TIM2CH2	
3	5	PA4					MISO/ MOSI				EPWM2P/ TIM2CH3	
4	6	PA5					CS				EPWM2N/ TIM2CH4	
5	7	PA6	AIN8		C0_N					BKIN		
6	8	PA7	AIN9	A0_O	C0_O					BKIN		
10	9	PA8	AIN2	A0_P	C1_P0/ C0_P0					BKIN		
9	10	PA9	AIN3	A0_N						BKIN		
8	11	PA10	AIN4	A1_N						BKIN		
7	12	PA11	AIN5	A1_P	C1_P1/ C0_P1					BKIN		
11	13	PA12	AIN6		C1_N					BKIN		
12	14	PA13	AIN7	A1_O	C1_O					BKIN		
13	15	VSS										Ground
14	16	VDD										Power
15	17	PA14	AIN10		C1_P2/ C0_P2		SCK		ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3			MCO[1*]
16	18	PA15	AIN11		C1_P3/ C0_P3	TXD/ RXD	MOSI/ MISO	SCL	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3			
17	19	PB0	AIN12			RXD/ TXD	MISO/ MOSI	SDA	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3			
18	20	PB1	AIN13				CS	SCL/ SDA	EPETR/ T2ETR	BKIN	TIM2CH1	RSTN/ MCO[1*]
22	21	PB2					SCK	SCL/ SDA	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3		EPWM2N/ TIM2CH1	JTAG_TCK/ CJ_TCK/ OSCIN
21	22	PB3				TXD	MOSI/ MISO	SCL	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	BKIN	EPWM2P	JTAG_TDO
20	23	PB4				TXD/ RXD	MISO/ MOSI		ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3		EPWM1P	JTAG_TMS/ CJ_TDIO
19	24	PB5				RXD/ TXD	CS	SDA	EPETR/ T2ETR	BKIN	EPWM1N/ TIM2CH1	JTAG_TDI

Table 5-2 PIN alternate function

Note:

1. BKIN is the optional external brake input pin for the EPWM module.
2. All I/O pins support timer counting, capture, external interrupt, and wake up functions.
3. All I2C pins have internal pull-up enable control; each I2C pin integrates a 10 kΩ pull-up resistor, which can be enabled or disabled by software.
4. RSTN is the reset pin. The pin includes a built-in 10 kΩ pull-up resistor, which is always enabled.

   When the RSTN function is reconfigured as a GPIO, the pull-up resistor can be disabled by software.
5. [1*]: MCO: System Clock output
6. [2*]: Complementary PWM

   (1) EPWM0P=PWMA, EPWM0N=PWMD=~(PWMA)

   (2) EPWM1P=PWMB, EPWM1N=PWME=~(PWMB)

   (3) EPWM2P=PWMC, EPWM2N=PWMF=~(PWMC)

5.4. PIN Alternate Selection

PIN Num.		PAx_AFR[2:0]/PBx_AFR[2:0] (PA0_AFR[2:0]~PA5_AFR[2:0],PA14_AFR[2:0], PA15_AFR[2:0], PB0_AFR[2:0]~PB5_AFR[2:0])							Special Function	External input	AMISC/ ADC Register
QFN 24	SSOP 24	0	1	2	3	4	5	6	CONFIG	IP Function	AMISC/ ADC Function
23	1	PA0	RXD	TXD	SDA[1*]	SCL[1*]		EPWM0P			
24	2	PA1	TXD	RXD	SCL[1*]	SDA[1*]		EPWM0N			
1	3	PA2	SCK				TIM2CH1	EPWM1P			
2	4	PA3	MOSI	MISO			TIM2CH2	EPWM1N			
3	5	PA4	MISO	MOSI			TIM2CH3	EPWM2P			
4	6	PA5	CS				TIM2CH4	EPWM2N			
5	7	PA6								BKIN	AIN8/ C0_N
6	8	PA7								BKIN	AIN9/ C0_O/ A0_O
10	9	PA8								BKIN	AIN2/ C1_P0/ C0_P0/ A0_P
9	10	PA9								BKIN	AIN3/ A0_N
8	11	PA10								BKIN	AIN4/ A1_N
7	12	PA11								BKIN	AIN5/ C1_P1/ C0_P1/ A1_P
11	13	PA12								BKIN	AIN6/ C1_N
12	14	PA13								BKIN	AIN7/ C1_O/ A1_O
13	15	VSS									
14	16	VDD									
15	17	PA14			SCK			MCO		ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	C1_P2/ C0_P2/ AIN10
16	18	PA15	TXD;	RXD	MOSI	MISO		SCL[1*]		ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	C1_P3/ C0_P3/ AIN11
17	19	PB0	RXD	TXD	MISO	MOSI		SDA[1*]		ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	AIN12
18	20	PB1	SCL [1*]	SDA [1*]	CS	TIM2CH1		MCO	RSTN	EPETR/ T2ETR/ BKIN	AIN13
22	21	PB2	SDA [1*]	SCL [1*]	SCK	TIM2CH1		EPWM2N	JTAG_TCK /CJ_TCK	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	OSCIN

21	**22**	PB3	TXD	SCL [1*]	MOSI	MISO		EPWM2P	JTAG_TDO	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3/ BKIN	
20	**23**	PB4	TXD	RXD	MISO	MOSI		EPWM1P	JTAG_TMS/ CJ_TDIO	ECAP1,2,3/ EPETR/ T2ETR/ T2CAP1,2,3	
19	**24**	PB5	RXD	TXD	CS	TIM2CH1	SDA[1*]	EPWM1N	JTAG_TDI	EPETR/ T2ETR/ BKIN	

Table 5-3 PIN alternate selection

Note:

(1) All I2C build in Pull high 10kΩ , can software set enable/disable

(2) When ADC, COMP, and PGA channels are in use, the digital input buffers must be disabled

(3) [1*]: When select as SDA or SCL, set I2CEN = 1 & FN2_AFR.I2CPULL = 1 to enable internal pull high 10kΩ

PIN Num.		GPIO Port	WK [5:0]	BKIN [3:0]	ETETR [2:0]	ETCAP0 [2:0]	ETCAP1 [2:0]	ETCAP2 [2:0]	T2ETR [2:0]	T2CAP0 [2:0]	T2CAP1 [2:0]	T2CAP2 [2:0]
QFN	SSOP		WK	EPWM (BKIN)	EPWM	EPWM	EPWM	EPWM	TIM2	TIM2	TIM2	TIM2
24	24											
23	1	PA0	WK(1)									
24	2	PA1	WK(2)									
1	3	PA2	WK(3)									
2	4	PA3	WK(4)									
3	5	PA4	WK(5)									
4	6	PA5	WK(6)									
5	7	PA6	WK(7)	BKIN(1)								
6	8	PA7	WK(8)	BKIN(2)								
10	9	PA8	WK(9)	BKIN(3)								
9	10	PA9	WK(10)	BKIN(4)								
8	11	PA10	WK(11)	BKIN(5)								
7	12	PA11	WK(12)	BKIN(6)								
11	13	PA12	WK(13)	BKIN(7)								
12	14	PA13	WK(14)	BKIN(8)								
13	15	VSS										
14	16	VDD										
15	17	PA14	WK(15)		EPETR(1)	ECAP1(1)	ECAP2(1)	ECAP3(1)	T2ETR (1)	T2CAP1 (1)	T2CAP2 (1)	T2CAP3 (1)
16	18	PA15	WK(16)		EPETR(2)	ECAP1(2)	ECAP2(2)	ECAP3(2)	T2ETR (2)	T2CAP1 (2)	T2CAP2 (2)	T2CAP3 (2)
17	19	PB0	WK(17)		EPETR(3)	ECAP1(3)	ECAP2(3)	ECAP3(3)	T2ETR (3)	T2CAP1 (3)	T2CAP2 (3)	T2CAP3 (3)
18	20	RSTN (PB1)	WK(18)	BKIN(9)	EPETR(4)				T2ETR (4)			
22	21	PB2	WK(19)		EPETR(5)	ECAP1(4)	ECAP2(4)	ECAP3(4)	T2ETR (5)	T2CAP1 (4)	T2CAP2 (4)	T2CAP3 (4)
21	22	PB3	WK(20)	BKIN(10)	EPETR(6)	ECAP1(5)	ECAP2(5)	ECAP3(5)	T2ETR (6)	T2CAP1 (5)	T2CAP2 (5)	T2CAP3 (5)
20	23	PB4	WK(21)		EPETR(7)	ECAP1(6)	ECAP2(6)	ECAP3(6)	T2ETR (7)	T2CAP1 (6)	T2CAP2 (6)	T2CAP3 (6)
19	24	PB5	WK(22)	BKIN(11)	EPETR(8)				T2ETR (8)			

Table 5-4 PIN input alternate selection

5.5. PIN Functional Discription

SSOP24	PIN Name	Type	Description
1	PA0	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	RXD	IO	UART Data input
	TXD	IO	UART Data output
	SDA	IO	I2C Data input/ouput
	WK	IO	Wake Up
	EPWM0P	IO	EPWM PWM Postive ouput channel 0
2	PA1	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	TXD	IO	UART Data output
	RXD	IO	UART Data input
	SCL	IO	I2C Clock input/output
	WK	IO	Wake Up
	EPWM0N	IO	EPWM PWM Negtive output channel 0
3	PA2	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	SCK	IO	SPI Clock input/output
	WK	IO	Wake Up
	EPWM1P	IO	EPWM PWM Postive output channel 1
	TIM2CH1	IO	TIM2 PWM output channel 1
4	PA3	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	MOSI	IO	SPI Master ouput/Slave input Data
	WK	IO	Wake Up
	MISO	IO	SPI Master input/Slave output Data
	EPWM1N	IO	EPWM PWM Negtive output channel 1
	TIM2CH2	IO	TIM2 PWM output channel 2
5	PA4	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	MISO	IO	SPI Master input/Slave output Data
	WK	IO	Wake Up
	MOSI	IO	SPI Master ouput/Slave input Data
	EPWM2P	IO	EPWM PWM Postive output channel 2
	TIM2CH3	IO	TIM2 PWM output channel 3
6	PA5	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	CS	IO	SPI CS enable
	WK	IO	Wake Up
	EPWM2N	IO	EPWM PWM Negtive output channel 2
	TIM2CH4	IO	TIM2 PWM output channel 4

SSOP24	PIN Name	Type	Description
7	PA6	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN8	A	ADC Channel 8 input
	C0_N	A	COMP0 Negtive input channel
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
8	PA7	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN9	A	ADC Channel 9 input
	A0_O	A	PGA0 output
	C0_O	IO	COMP0 output
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
9	PA8	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN2	A	ADC Channel 2 input
	A0_P	A	PGA0 Postive Input
	C1_P0/C0_P0	A	COMP1 Postive input channel 0/COMP0 Postive input channel 0
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
10	PA9	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN3	A	ADC Channel 3 input
	A0_N	A	PGA0 Negtive Input
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
11	PA10	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN4	A	ADC Channel 4 input
	A1_N	A	PGA1 Negtive Input
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
12	PA11	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN5	A	ADC Channel 5 input
	A1_P	A	PGA1 Postive Input
	C1_P1/C0_P1	A	COMP1 Postive input channel 1/COMP0 Postive input channel 1
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
13	PA12	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN6	A	ADC Channel 6 input
	C1_N	A	COMP1 Negtive Input channel
	WK	IO	Wake Up

SSOP24	PIN Name	Type	Description
	BKIN	IO	EPWM Brake Signal Input
14	PA13	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN7	A	ADC Channel 7 input
	A1_O	A	PGA1 Output
	C1_O	IO	COMP1 Output
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
15	VSS	Gound	0V Ground
16	VDD	Power	5V Power
17	PA14	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN10	A	ADC Channel 10 input
	C1_P2/C0_P2	A	COMP1 Postive input channel 2/COMP0 Postive input channel 2
	SCK	IO	SPI Clock input/output
	WK	IO	Wake Up
	MCO	IO	System Clock output
	EPETR	IO	EPWM External trigger input
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
18	PA15	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN11	A	ADC Channel 11 input
	C1_P3/C0_P3	A	COMP1 Postive input channel 3/COMP0 Postive input channel 3
	TXD	IO	UART Data output
	MOSI	IO	SPI Master ouput/Slave input Data
	MISO	IO	SPI Master input/Slave output Data
	SCL	IO	I2C Clock input/output
	WK	IO	Wake Up
	EPETR	IO	EPWM External trigger input
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
19	PB0	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN12	A	ADC Channel 12 input
	RXD	IO	UART Data Input
	TXD	IO	UART Data output
	MISO	IO	SPI Master input/Slave output Data
	MOSI	IO	SPI Master ouput/Slave input Data
	SDA	IO	I2C Data input/Output

SSOP24	PIN Name	Type	Description
	WK	IO	Wake Up
	EPETR	IO	EPWM External trigger input
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
20	PB1	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	AIN13	A	ADC Channel 13 input
	CS	IO	SPI CS enable
	SCL	IO	I2C clock input/output
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
	TIM2CH1	IO	TIM2 PWM output channel 1
	RSTN	I	Default Chip Reset Active Low
	MCO	IO	System Clock Output
	EPETR	IO	EPWM External trigger input
	T2ETR	IO	TIM2 External trigger input
21	PB2	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	SCK	IO	SPI Clock input/output
	SCL	IO	I2C Clock input/output
	WK	IO	Wake Up
	EPWM2N	IO	EPWM PWM Negtive output channel 2
	TIM2CH1	IO	TIM2 PWM output channel 1
	JTAG_TCK/CJ_TCK	IO	JTAG Interface
	OSCIN	IO	External OSC input
	EPETR	IO	EPWM External trigger input
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
22	PB3	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	TXD	IO	UART Data output
	MOSI	IO	SPI Master ouput/Slave input Data
	MISO	IO	SPI Master input/Slave output Data
	SCL	IO	I2C Clock input/output
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
	EPWM2P	IO	EPWM PWM Postive output channel 2
	JTAG_TDO	IO	JTAG interface
	EPETR	IO	EPWM External trigger input

SSOP24	PIN Name	Type	Description
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
23	PB4	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	TXD	IO	UART Data output
	RXD	IO	UART Data input
	MISO	IO	SPI Master input/Slave output Data
	MOSI	IO	SPI Master ouput/Slave input Data
	WK	IO	Wake Up
	EPWM1P	IO	EPWM PWM Postive output channel 1
	JTAG_TMS/CJ_TDIO	IO	JTAG interface
	EPETR	IO	EPWM External trigger input
	ECAP1,2,3	IO	EPWM Capture input channel
	T2ETR	IO	TIM2 External trigger input
	T2CAP1,2,3	IO	TIM2 Capture input channel
24	PB5	IO	GPIO I/O configuration (direction, pull-up/pull-down functionality, etc.) is managed through control registers
	RXD	IO	UART Data input
	TXD	IO	UART Data output
	CS	IO	SPI CS enable
	SDA	IO	I2C Data input/output
	WK	IO	Wake Up
	BKIN	IO	EPWM Brake Signal Input
	EPWM1N	IO	EPWM PWM Negtive output channel 1
	TIM2CH1	IO	TIM2 PWM output channel 1
	JTAG_TDI	IO	JTAG interface
	EPETR	IO	EPWM External trigger input
	T2ETR	IO	TIM2 External trigger input

Table 5-5 PIN functional discription

[Note] IO: Logic input/output; A:Analog Ouput

6. Memory Mapping

The PEC930 utilizes a unified physical address space (or unified memory space) architecture. The specific memory address map for the chip is detailed below.

6.1. AHB Address Mapping

Internal AHB Address Mapping as follows:

AHB Memory Address	Size	Peripherals
0x0000 0000 – 0x0000 7FFF	32KB	**FLASH main code area**
0x0000 8000 – 0x001F FFFF	-	Reserved
0x0020 0000 – 0x0020 1FFF	8KB	**NVR information configuration area**
0x0020 2000 – 0x17FF FFFF	-	Reserved
0x1800 0000 – 0x1800 0FFF	4KB	**CORE control and status area**
0x1800 1000 – 0x1FFF FFFF	-	Reserved
0x2000 0000 – 0x2000 0FFF	4KB	**SRAM data area**
0x2000 1000 – 0x3FFF FFFF	-	Reserved
0x4000 0000 – 0x4000 FFFF	64KB	**APB peripheral module area**
0x4001 0000 – 0x4001 0FFF	-	Reserved
0x4001 1000 – 0x4001 1FFF	4KB	**GPIOA**
0x4001 2000 – 0x4001 2FFF	4KB	**GPIOB**
0x4001 3000 – 0x4001 DFFF	-	Reserved
0x4001 E000 – 0x4001 EFFF	4KB	**CRC**
0x4001 F000 – 0x4001 FFFF	4KB	**SYSCFG**
0x4002 0000 – 0x5FFF FFFF	-	Reserved
0x6000 0000 – 0x9FFF FFFF	-	Reserved
0xA000 0000 – 0xDFFF FFFF	-	Reserved
0xE000 0000 – 0xFFFF FFFF	-	Reserved

Table 6-1 AHB address mapping table

6.2. APB Address Mapping

Internal APB Address Mapping As follows:

No.	APB Memory Address	Size	Peripherals
0	0x4000 0000 – 0x4000 07FF	2KB	**TIM0**
1	0x4000 0800 – 0x4000 0FFF	2KB	**TIM1**
2	0x4000 1000 – 0x4000 17FF	2KB	**TIM2**
3	0x4000 1800 – 0x4000 1FFF	2KB	Reserved
4	0x4000 2000 – 0x4000 27FF	2KB	**UART**
5	0x4000 2800 – 0x4000 2FFF	2KB	Reserved
6	0x4000 3000 – 0x4000 37FF	2KB	**I2C**
7	0x4000 3800 – 0x4000 3FFF	2KB	**SPI**
8	0x4000 4000 – 0x4000 47FF	2KB	**WDG**
9	0x4000 4800 – 0x4000 4FFF	2KB	**ADC controller**
10	0x4000 5000 – 0x4000 57FF	2KB	Reserved
11	0x4000 5800 – 0x4000 5FFF	2KB	**AMISC(include PGA0/PGA1)**
12~15	0x4000 6000 – 0x4000 67FF	8KB	Reserved
16	0x4000 8000 – 0x4000 87FF	2KB	**DSP**
17	0x4000 8800 – 0x4000 8FFF	2KB	**COMP0**
18	0x4000 9000 – 0x4000 97FF	2KB	Reserved
19	0x4000 9800 –0x4000 9FFF	2KB	**COMP1**
20~23	0x4000 A000 –0x4000 BFFF	8KB	Reserved
24	0x4000 C000 –0x4000 C7FF	2KB	**EPWM**
25	0x4000 C800 – 0x4000 CFFF	2KB	**LPTIM**
26~30	0x4000 D000 – 0x4000 F7FF	10KB	Reserved
31	0x4000 F800 – 0x4000 FFFF	2KB	**FLASH controller**

Table 6-2 APB address mapping table

6.3. FLASH NVR Configuration

The NVR region of the FLASH memory consists of 16 sectors, with each sector containing 128 words. The analog calibration parameters are stored in the 14th and 15th sectors, starting from the AHB base address 0x20_1C00. These sectors hold critical calibration data required by analog modules, which are programmed during mass production and automatically loaded to configure the modules upon power-up. The remaining sectors 0 to 13 are available for user data storage, starting from the AHB base address 0x20_0000.

7. System Configuration (SYSCFG)

7.1. Characteristics

PEC930 system configuration area includes registers related to system usage.The base address of the system control register is 0x4001_F000. Users can configure system-related settings according to application requirements, such as:

- Clock configuration

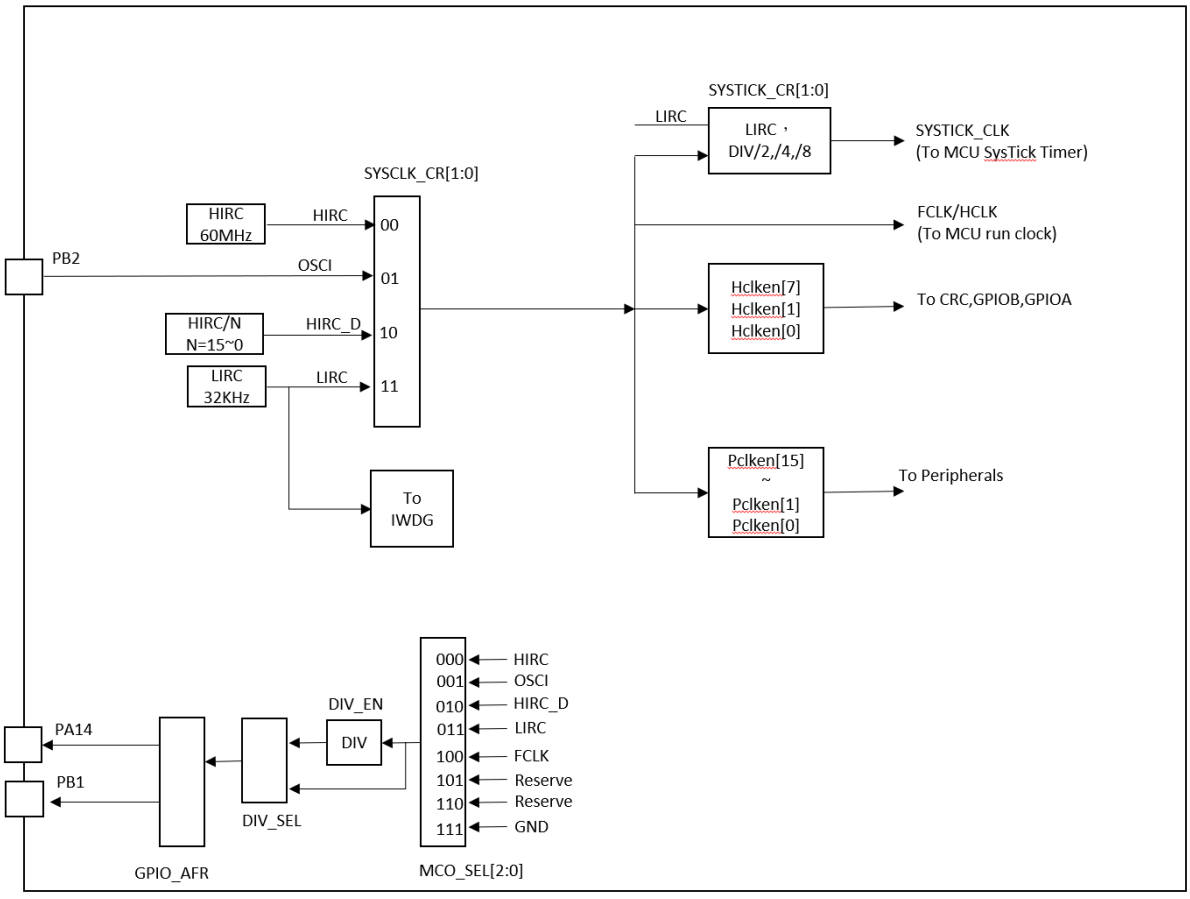


Fig. 7-1 Clock Block Diagram

Refer clock output control register(SYSCFG_MCOCR), System clock config register(SYSCFG_SYSCLKCR), Peripheral Clock Switch(SYSCFG_PCLKEN), Peripheral 1 Clock Switch(SYSCFG_HCLKEN)

- Reset System configuration
  - Refer Reset flag register(SYSCFG_SYSRSTSR),Reset config register(SYSCFG_SYSRSTCR),Pin reset enable register(SYSCFG_ICEIOCR)
- Wake up configuration (DeepSleep and Sleep)
  - Refer low power control register(SYSCFG_PMUCR)
- TIM2,EPWM input channel configuration
  - Refer SYSCFG_TIM2_CON_SEL regitser ,SYSCFG_EPWM_CON_SEL register

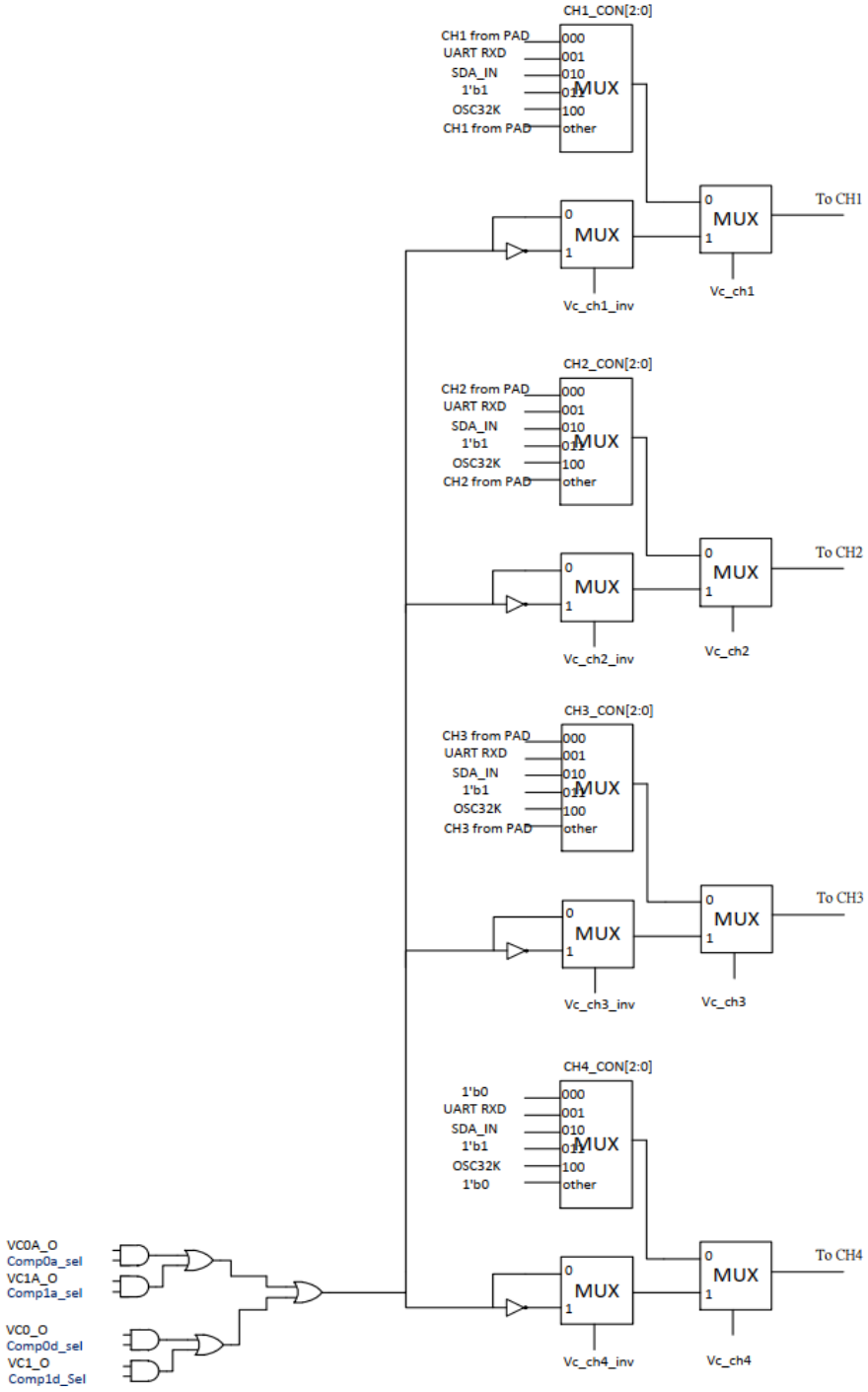


Fig. 7-2 Timer2 and EPWM CONFIG Block Diagram

- EPWM Brake configuration
  - SYSCFG_NMI_BK_SEL register

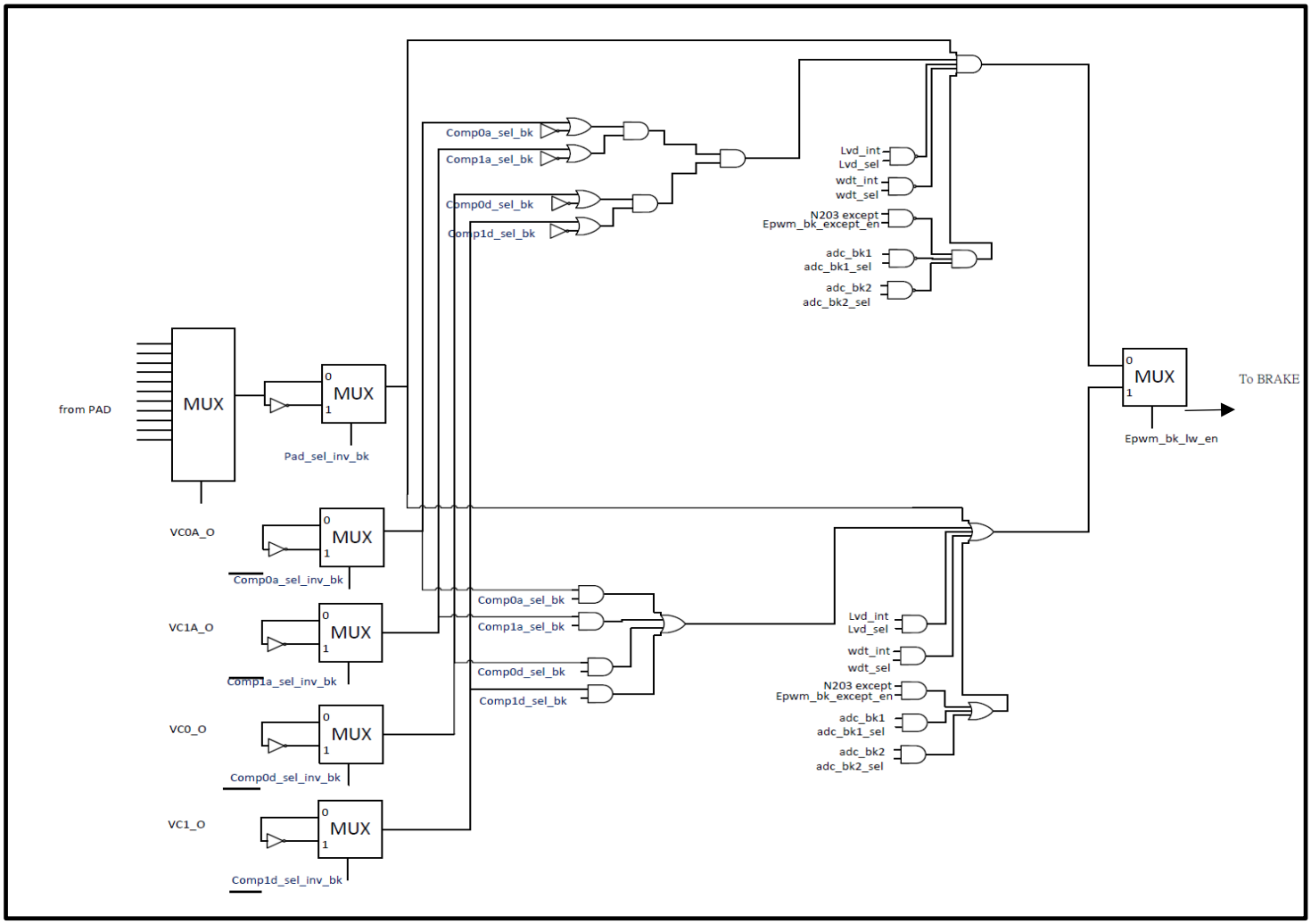


Fig. 7-3 EPWM BRAKE Block Diagram

- Debug Module configuration
  - Debug module control register(SYSCFG_DEBUGENCR)

7.2. Low Power Mode

PEC930 Supports sleep/deepsleep power-saving mode，Users can configure relevant system settings according to application needs. The process is as follows:

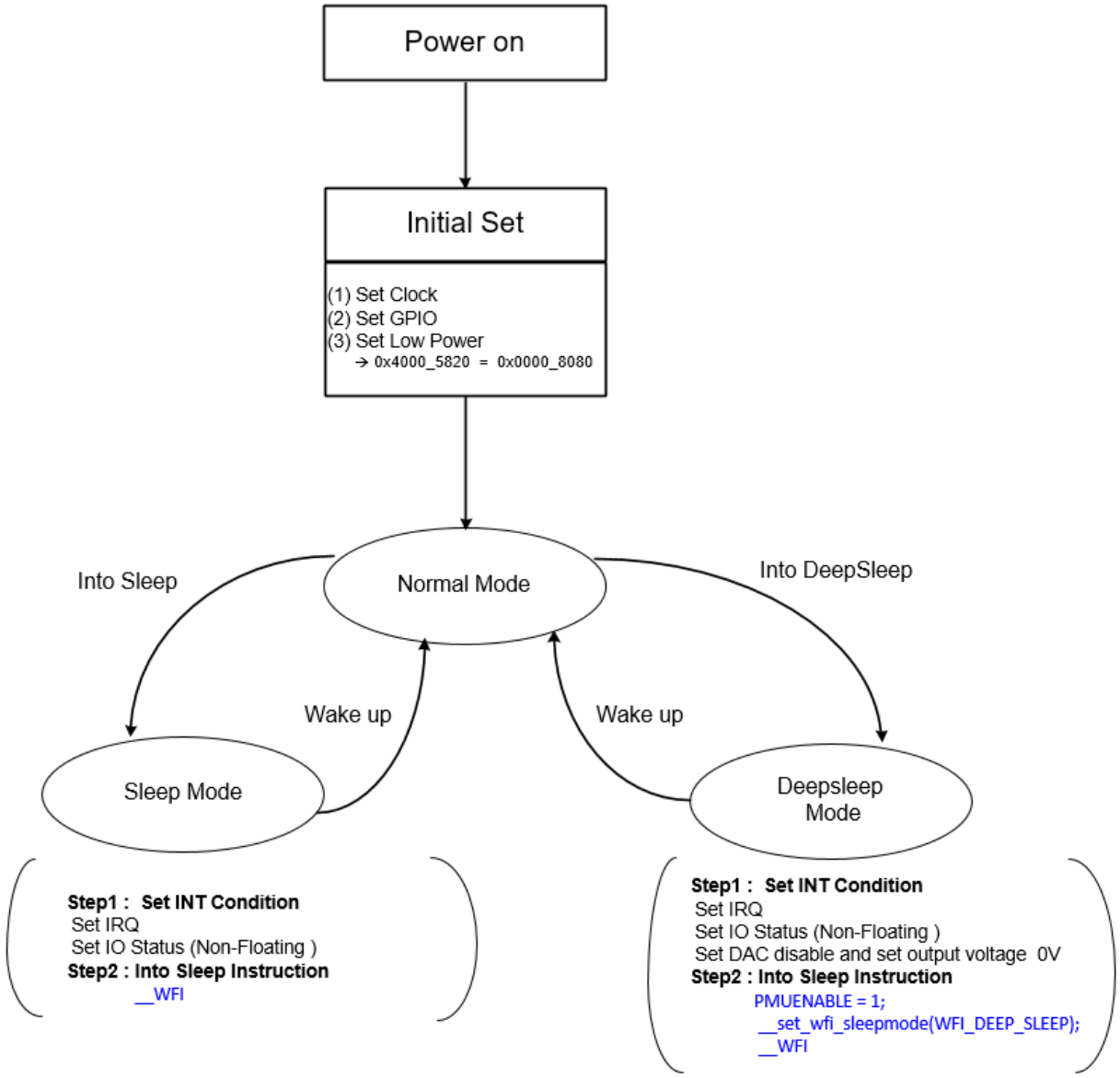


Fig. 7-4 Deepsleep / sleep configuration flowchart

- Set IRQ source, refer to the respective IP sections and related descriptions.

- Sleep Mode: Refer 9.3 chapter, enter sleep mode

- Deepsleep Mode: refer 9.3 chapter, refer enter deepsleep mode，need to set 0x4001F004 = 1 first, (Requires waiting 16 LIRCs before you can actually enter), refer 7.4.1 chapter, wake up: refer 2.3 chapter

  (a) Wake up @sleep: any interrupt source

  (b) Wake up @Deep sleep: gpio ,WDG(Always able to wake up) and LPTMR interrupt

  Refer 9.4 chapter, exit sleep/deepsleep mode

  (c) Sdk example: refer pwr_sleep_mode example

7.3. Register table

Address	Name	Description
0x4001F000	-	Reserved
0x4001F004	SYSCFG_PMUCR	Low-power active enable register
0x4001F008	-	Reserved
0x4001F00C	SYSCFG_MCOCR	MCO Clock output control register
0x4001F010	SYSCFG_SYSRSTSR	Reset flag register
0x4001F014	SYSCFG_REBOOT_UNLOCK	RETRIM password register
0x4001F018	SYSCFG_SYSRSTCR	Reset config register
0x4001F01C	SYSCFG_DEBUGENCR	Debug mode control register
0x4001F020	SYSCFG_SYSCLKCR	System clock config register
0x4001F024	SYSCFG_PRSTEN	Peripheral Reset Switch
0x4001F028	SYSCFG_PCLKEN	Peripheral Clock Switch
0x4001F02C	SYSCFG_ICEIOCR	JTAG Pin multiplexing enable register
0x4001F030	SYSCFG_RSTPINCR	Pin Reset enable register
0x4001F034	SYSCFG_TIM2_CON_SEL	Timer2 Channel setting register
0x4001F038	SYSCFG_EPWM_CON_SEL	EPWM Channel setting register
0x4001F03C	-	Reserved
0x4001F040	SYSCFG_PRSTEN1	Peripheral 1 Reset Switch
0x4001F044	SYSCFG_HCLKEN	Peripheral 1 Clock Switch
0x4001F048	SYSCFG_EVT_SEL	EVT Trigger source selection control bit
0x4001F04C	SYSCFG_NMICR	NMI/BK Trigger source selection control bit
0x4001F100	SYSCFG_CHIPID	Chip Version Number Register

Table 7-1 SYSCFG register table

7.4. Register Description

7.4.1. Low-power active enable register (SYSCFG_PMUCR)

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	PMUENABLE	R/W	0x0	Low-power control register, used in conjunction with SCR register Example SYSCFG_PMUCR= 1; __set_wfi_sleepmode (WFI_DEEP_SLEEP); __WFI Enter deepsleep mode

7.4.2. Clock output control register (SYSCFG_MCOCR)

Bit	Name	R/W	Reset	Description
31:24	-	R	0x0	Reserved
23:16	SOFT_KEY	R/W	0x0	writer “0x5a” ,generate soft reset
15:8	DIV_CNT	R/W	0x0	Frequency divider counter 0:Prohibited 1: /2 2~255: /(bit[15:8] +1)
7	DIV_SEL	R/W	0x0	DIV_SEL 1: Frequency divider output 0: Frequency output
6	DIV_EN	R/W	0x0	1: Frequency divider counter turn on 0: Frequency divider counter turn off，and clean counter 8’h0
5:3	-	R	0x0	Reserved
2:0	MCO_SEL	R/W	0x0	000: RC OSC 60M 001: External clock 010: System frequency division(SYSCFG_SYSCLKCR=3’b010) 011: RC OSC 32K 100: Resvered 101: Resvered 110: Reserved 111: 0

7.4.3. Reset flag register (SYSCFG_SYSRSTSR)

Bit	Name	R/W	Reset	Description
31:13	-	R	0x0	Reserved
12	RES_RB_FG	R/W1c	0x0	1: Retrim boot reset was performed Write 1 to the corresponding bit to clear it.
11	RES_LVD_FG	R/W1c	0x0	1: LVD voltage abnormality caused reset Write 1 to the corresponding bit to clear it
10	-	R	0x0	Reserved
9	RES_RESET_FG	R/W1c	0x0	1: RESET pin causes a reset Write 1 to the corresponding bit to clear it.
8	RES_POR_FG	R/W1c	0x1	1: POR causes reset Write 1 to the corresponding bit to clear it.
7:3	-	R	0x0	Reserved
2	RES_LOCK_FG	R/W1c	0x0	1: Reset LOCKUP Write 1 to the corresponding bit to clear it.
1	RES_WDG_FG	R/W1c	0x0	1: Watchdog causes reset Write 1 to the corresponding bit to clear it.
0	RES_SOFT_FG	R/W1c	0x0	1: System soft reset caused reset Write 1 to the corresponding bit to clear it.

7.4.4. RETRIM password register (SYSCFG_REBOOT_UNLOCK)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:0	RETRIMEN_PWD	R/W	0x0	Reboot Reset password，The reset password must be set to 0xAB56 before writing bit 10 of the “SYSCFG_SYSRSTCR” register to trigger a Reboot reset; otherwise, the Reboot reset will not be triggered.

7.4.5. Reset config register (SYSCFG_SYSRSTCR)

Bit	Name	R/W	Reset	Description
31:11	-	R	0x0	Reserved
10	RETRIMING_EN	R/W	0x0	Reboot enabled (active high) will trigger a system reboot when written to 1. During the reset process, the triggering value in the NVR will be reloaded.
9:0	-	R	0x0	Reserved

7.4.6. Debug mode control register (SYSCFG_DEBUGENCR)

Bit	Name	R/W	Reset	Description
31:16	Key	W	0x0	Bit 15:0 can only be written when writing 0x8A57; writing other values is invalid.
15	-	R	0x0	Reserved
14	WDG_ICE_EN	R/W	0x1	WDG Module debug mode has stopped working 1: active
13:12	-	R	0x0	Reserved
11	TMR2_ICE_EN	R/W	0x0	TMR2 Module debug mode has stopped working. 1: active
10:7	-	R	0x0	Reserved
6	EPWM_ICE_EN	R/W	0x0	EPWM Module debug mode has stopped working. 1: active
5	-	R	0x0	Reserved
4	LPTMR_ICE_EN	R/W	0x0	LPTMR Module debug mode has stopped working. 1: active
3	TMR1_ICE_EN	R/W	0x0	TMR1 Module debug mode has stopped working. 1: active
2	TMR0_ICE_EN	R/W	0x0	TMR0Module debug mode has stopped working. 1: active
1:0	-	R	0x0	Reserved

7.4.7. System clock config register (SYSCFG_SYSCLKCR)

Bit	Name	R/W	Reset	Description
31:26	-	R	0x0	Reserved
25:24	SysTick_CR	R/W	0x0	Systick Clock source selection. 2'b00: Internal 32kHz RC oscillation clock 2'b01: HCLK/2 2'b10: HCLK/4 2'b11: HCLK/8
23	RC32AON	R/W	0x0	1: RC32K always on(deepsleep mode)
22:12	-	R/W	0x0	Reserved
11:8	SYSCLK_DIV	R/W	0x9	Bit11~8Integer frequency divider division coefficient 0000: Prohibited 0001: RC60M /2 0010: RC60M /3 0011: RC60M /4 0100: RC60M /5 0101: RC60M /6 0110: RC60M /7 0111: RC60M /8 1000: RC60M /9 1001: RC60M /10 1010: RC60M /11 1011: RC60M /12 1100: RC60M /13 1101: RC60M /14 1110: RC60M /15 1111: RC60M /16
7:3	-	R	0x0	Reserved
2:0	SYSCLK_CR	R/W	0x2	Bit 2: Reserved Bit 1~0: System master clock selection. 2'b00: Internal 60MHz RC oscillation clock 2'b01: External PB2 input clock 2'b10: Integer divider output clock (by SYSCLK_DIV) 2'b11: Internal 32kHz RC oscillation clock

7.4.8. Peripheral Reset Switch (SYSCFG_PRSTEN)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15	ANA_RST_EN	R/W	0x0	1: ANA module reset 0: ANA module no reset
14	WDG_RST_EN	R/W	0x0	1: WDG module reset 0: WDG module no reset
13	COMP1_RST_EN	R/W	0x0	1: COMP1 module reset 0: COMP1 module no reset
12	I2C_RST_EN	R/W	0x0	1: I2C module reset 0: I2C module no reset
11	TMR2_RST_EN	R/W	0x0	1: TIMER2 module reset 0: TIMER2 module no reset
10	SPI_RST_EN	R/W	0x0	1: SPI module reset 0: SPI module no reset
9	-	R	0x0	Reserved
8	CMP0_RST_EN	R/W	0x0	1: COMP0 module reset 0: COMP0 module no reset
7	DSP_RST_EN	R/W	0x0	1: DSP module reset 0: DSP module no reset
6	EPWM_RST_EN	R/W	0x0	1: EPWM module reset 0: EPWM module no reset
5	ADC_RST_EN	R/W	0x0	1: ADC module reset 0: ADC module no reset
4	LPTMR_RST_EN	R/W	0x0	1: LPTMR module reset 0: LPTMR module no reset
3	TMR1_RST_EN	R/W	0x0	1: TIMER1 module reset 0: TIMER1 module no reset
2	TMR0_RST_EN	R/W	0x0	1: TIMER0 module reset 0: TIMER0 module no reset
1	-	R	0x0	Reserved
0	UART_RST_EN	R/W	0x0	1: UART module reset 0: UART module no reset

7.4.9. Peripheral Clock Switch (SYSCFG_PCLKEN)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15	ANA_CK_EN	R/W	0x0	1: ANA module clock enable 0: ANA module clock disable
14	WDG_CK_EN	R/W	0x1	1: WDG module clock enable 0: WDG module clock disable
13	COMP1_CK_EN	R/W	0x0	1: COMP1 module clock enable 0: COMP1 module clock disable
12	I2C_CK_EN	R/W	0x0	1: I2C module clock enable 0: I2C module clock disable
11	TMR2_CK_EN	R/W	0x0	1: TMR2 module clock enable 0: TMR2 module clock disable
10	SPI_CK_EN	R/W	0x0	1: SPI module clock enable 0: SPI module clock disable
9	-	R/W	0x0	Reserved
8	CMP0_CK_EN	R/W	0x0	1: CMP0 module clock enable 0: CMP0 module clock disable
7	DSP_CK_EN	R/W	0x0	1: DSP module clock enable 0: DSP module clock disable
6	EPWM_CK_EN	R/W	0x0	1: EPWM module clock enable 0: EPWM module clock disable
5	ADC_CK_EN	R/W	0x0	1: ADC module clock enable 0: ADC module clock disable
4	LPTMR_CK_EN	R/W	0x0	1: LPTMR module clock enable 0: LPTMR module clock disable
3	TMR1_CK_EN	R/W	0x0	1: TMR1 module clock enable 0: TMR1 module clock disable
2	TMR0_CK_EN	R/W	0x0	1: TMR0 module clock enable 0: TMR0 module clock disable
1	-	R	0x0	Reserved
0	UART_CK_EN	R/W	0x0	1: UART module clock enable 0: UART module clock disable

7.4.10. JTAG Pin multiplexing enable register (SYSCFG_ICEIOCR)

Bit	Name	R/W	Reset	Description
31:16	KEY	W	0x0	Bit [31: 16] = 0xE653，then can it be written bit [2:0]，Invalid when writing other values
15:3	-	R	0x0	Reserved
2	DBG_EN	R/W	0x1	CORE debug interface and internal function control 1: Enable 0: Disable
1	DBG_CTRL_EN	R/W	0x1	CORE debug halt and control function 1: Enable 0: Disable
0	PIN_JTAG_EN	R/W	0x1	JTAG Pin reuse control 1: Enable 0: Disable

7.4.11. Pin Reset enable register (SYSCFG_RSTPINCR)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:0	PIN_RST_HOLD	R/W	0x0	Reset Pin function enable Bit 15~0: Write 0xA563 to enable pin reset; write other values to disable pin reset function; Reading this register, bit 0 indicates the pin reset enable state, and the other bits are 0.

7.4.12. TIM2_CON_SEL register (SYSCFG_TIM2_CON_SEL)

Bit	Name	R/W	Reset	Description
31:30	-	R/W	0x0	Reserved
29:28	Comp1_0d_sel	R/W	0x0	TIM2 Input channel signal source selection: Comp digital output (after filter) 29: Comp1d_sel 28: Comp0d_sel 00: comp1d disable, 0: comp0d disable 01: comp1d disable, 0: comp0d enable 10: comp1d enable, 0: comp0d disable 11: comp1d enable, 0: comp0d enable
27:26	Vc_ch4	R/W	0x0	TIM2_CH4 Input channel signal source selection 27: Vc_ch4 26: Vc_ch4_inv [27:26] 00: CH4_CON ([14:12]) set 01: CH4_CON ([14:12]) set 10: COMP input (set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
25:24	Vc_ch3	R/W	0x0	TIM2_CH3 Input channel signal source selection 25: Vc_ch3 24: Vc_ch3_inv [25:24] 00: CH3_CON ([10:8]) set 01: CH3_CON ([10:8]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
23:22	Vc_ch2	R/W	0x0	TIM2_CH2 Input channel signal source selection 23: Vc_ch2 22: Vc_ch2_inv [23:22]

				00: CH2_CON ([6:4]) set 01: CH2_CON ([6:4]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
21:20	Vc_ch1	R/W	0x0	TIM2_CH1 Input channel signal source selection 21: Vc_ch1 20: Vc_ch1_inv) [21:20] 00: CH1_CON ([2:0]) set 01: CH1_CON ([2:0]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
19:18	Comp1_0a_sel	R/W	0x0	TIM2 Input channel signal source selection comp analog output 19: Comp1a_sel 18: Comp0a_sel 00: comp1a disable, 0: comp0a disable 01: comp1a disable, 0: comp0a enable 10: comp1a enable, 0: comp0a disable 11: comp1a enable, 0: comp0a enable
17:15	-	R	0x0	Reserved
14:12	CH4_CON	R/W	0x0	TIM2_CH4 Input channel signal source selection 000: reserved 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: reserved 110: reserved 111: reserved
11	-	R	0x0	Reserved

10:8	CH3_CON	R/W	0x0	TIM2_CH3 Input channel signal source selection 000: from PAD T2CAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD
7	-	R	0x0	Reserved
6:4	CH2_CON	R/W	0x0	TIM2_CH2 Input channel signal source selection 000: from PAD T2CAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD
3	-	R	0x0	Reserved
2:0	CH1_CON	R/W	0x0	TIM2_CH1 Input channel signal source selection 000: from PAD T2CAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD

7.4.13. EPWM_CON_SEL register (SYSCFG_EPWM_CON_SEL)

Bit	Name	R/W	Reset	Description
31	wdg_sel	R/W	0x0	EPWM BK Input channel signal source selection 0: wdg bk disable 1: wdg bk enable
30	Lvd_sel	R/W	0x0	EPWM BK Input channel signal source selection 0: lvd bk disable 1: lvd bk enable
29:28	Comp1_0d_sel	R/W	0x0	EPWM Input channel signal source selection comp digital output(go through Filter) 29: Comp1d_sel 28: Comp0d_sel 00: comp1d disable, 0: comp0d disable 01: comp1d disable, 0: comp0d enable 10: comp1d enable, 0: comp0d disable 11: comp1d enable, 0: comp0d enable
27:26	Vc_ch4	R/W	0x0	EPWM_CH4 Input channel signal source selection 27: Vc_ch4 26: Vc_ch4_inv [27:26] 00: CH4_CON ([14:12]) set 01: CH4_CON ([14:12]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)

25:24	Vc_ch3	R/W	0x0	EPWM_CH3 Input channel signal source selection 25: Vc_ch3 24: Vc_ch3_inv [25:24] 00: CH3_CON ([10:8]) set 01: CH3_CON ([10:8]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
23:22	Vc_ch2	R/W	0x0	EPWM_CH2 Input channel signal source selection 23: Vc_ch2 22: Vc_ch2_inv [23:22] 00: CH2_CON ([6:4]) set 01: CH2_CON ([6:4]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)
21:20	Vc_ch1	R/W	0x0	EPWM_CH1 Input channel signal source selection 21: Vc_ch1 20: Vc_ch1_inv) [21:20] 00: CH1_CON ([2:0]) set 01: CH1_CON ([2:0]) set 10: COMP input(set by Comp1_0a_sel) 11: COMP inv input(set by Comp1_0a_sel)

19:18	Comp1_0a_sel	R/W	0x0	EPWM Input channel signal source selection comp analog output 19: Comp1a_sel 18: Comp0a_sel 00: comp1a disable, 0: comp0a disable 01: comp1a disable, 0: comp0a enable 10: comp1a enable, 0: comp0a disable 11: comp1a enable, 0: comp0a enable
17:15	-	R	0x0	Reserved
14:12	CH4_CON	R/W	0x0	EPWM_CH4 Input channel signal source selection 000: reserved 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: reserved 110: reserved 111: reserved
11	-	R	0x0	Reserved
10:8	CH3_CON	R/W	0x0	EPWM_CH3 Input channel signal source selection 000: from PAD ECAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k

				101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD
7	-	R	0x0	Reserved
6:4	CH2_CON	R/W	0x0	EPWM_CH2 Input channel signal source selection 000: from PAD ECAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD
3	-	R	0x0	Reserved
2:0	CH1_CON	R/W	0x0	EPWM_CH1 Input channel signal source selection 000: from PAD ECAP1,2,3 (PA14/PA15/PB0/PB2/PB3/PB4) 001: uart0_rxd 010: sda_in 011: reserved 100: osc32k 101: the same 000 from PAD 110: the same 000 from PAD 111: the same 000 from PAD

7.4.14. Peripheral 1 Reset Switch (SYSCFG_PRSTEN1)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7	CRC_RES_EN	R/W	0x0	1: CRC module reset 0: CRC module no reset
6:2	-	R	0x0	Reserved
1	GPIOB_RES_EN	R/W	0x0	1: GPIOB module reset 0: GPIOB module no reset
0	GPIOA_RES_EN	R/W	0x0	1: GPIOA module reset 0: GPIOA module no reset

7.4.15. Peripheral 1 Clock Switch (SYSCFG_HCLKEN)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7	CRC_CK_EN	R/W	0x0	1: CRC module clock enable 0: CRC module clock disable
6:2	-	R	0x0	Reserved
1	GPIOB_CK_EN	R/W	0x1	1: GPIOB module clock enable 0: GPIOB module clock disable
0	GPIOA_CK_EN	R/W	0x1	1: GPIOA module clock enable 0: GPIOA module clock disable

7.4.16. EVT_SEL control register (SYSCFG_EVT_SEL)

Bit	Name	R/W	Reset	Description
31:15	-	R	0x0	Reserved
14	Evt_LVD_EN	R/W	0x0	evt source setting LVD bit 1: enable 0: disable
13	Evt_EPWM_EN	R/W	0x0	evt source setting EPWM bit 1: enable 0: disable
12	Evt_ADC_EN	R/W	0x0	evt source setting ADC bit 1: enable 0: disable
11	Evt_CMP1_EN	R/W	0x0	evt source setting CMP1 bit 1: enable 0: disable
10	Evt_CMP0_EN	R/W	0x0	evt source setting CMP0 bit 1: enable 0: disable
9	Evt_GPIOB_EN	R/W	0x0	evt source setting GPIOB bit 1: enable 0: disable
8	Evt_GPIOA_EN	R/W	0x0	evt source setting GPIOA bit 1: enable 0: disable

7	Evt_I2C_EN	R/W	0x0	evt source setting I2C bit 1: enable 0: disable
6	Evt_UART_EN	R/W	0x0	evt source setting Uart bit 1: enable 0: disable
5	Evt_SPI_EN	R/W	0x0	evt source setting SPI bit 1: enable 0: disable
4	Evt_WDG_EN	R/W	0x0	evt source setting Wdt bit 1: enable 0: disable
3	Evt_LPTMR_EN	R/W	0x0	evt source setting LPtimer bit 1: enable 0: disable
2	Evt_TMR2_EN	R/W	0x0	evt source setting timer2 bit 1: enable 0: disable
1	Evt_TMR1_EN	R/W	0x0	evt source setting timer1 bit 1: enable 0: disable
0	Evt_TMR0_EN	R/W	0x0	evt source setting timer0 bit 1: enable 0: disable

7.4.17. NMI_BK_SEL control register (SYSCFG_NMICR)

Bit	Name	R/W	Reset	Description
31:21		R/W	0x0	Reserved
20	Pad_sel_inv_bk	R/W	0x0	EPWM BKP Signal Selection 0: pad input to bk no inv 1: pad input to bk inv
19	Comp1a_sel_inv_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp1 analog IP input to bk no inv 1: comp1 analog IP input to bk inv
18	Comp0a_sel_inv_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp0 analog IP input to bk no inv 1: comp0 analog IP input to bk inv
17	Comp1d_sel_inv_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp1 digital input to bk no inv 1: comp1 digital input to bk inv
16	Comp0d_sel_inv_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp0 digital input to bk no inv 1: comp0 digital input to bk inv
15	Comp1a_sel_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp1a_bk disable 1: comp1a_bk enable
14	Comp0a_sel_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp0a_bk disable 1: comp0a_bk enable

13	Comp1d_sel_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp1d_bk disable 1: comp1d_bk enable
12	Comp0d_sel_bk	R/W	0x0	EPWM BKP Signal Selection 0: comp0d_bk disable 1: comp0d_bk enable
11	Epwm_bk_except_EN	R/W	0x0	EPWM BKP Signal Selection 1: enable epem_bk_exception(N203 exception) 0: Disable N203 exception list Instruction access fault Load access fault Store/AMO access fault Illegal instruction
10	adc_bk2_sel	R/W	0x0	EPWM BKP Signal Selection 1: adc_bk2 enable 0: adc_bk2 disable
9	adc_bk1_sel	R/W	0x0	EPWM BKP Signal Selection 1: adc_bk1 enable 0: adc_bk1 disable
8	Epwm_bk_low_EN	R/W	0x0	EPWM BKP Signal (Low/High) Action Selection 1: epem_bk_high (Normally 0) 0: epem_bk_low (Normally 1) Use with EPWM (EPWM_BDTR) bit13 BKP

7:5	NMI_EN	R/W	0x0	NMI interrupt enable (Write protection) 3'b101 = enable NMI interrupt Other = disable NMI interrupt
4:0	NMI_SEL	R/W	0x0	NMI Trigger Source Selection Control Bit The corresponding control bit selects the enable of the trigger source. The interrupt source can be selected from 15 peripheral interrupt numbers. 0x00: timer0 0x01: timer1 0x02: timer2 0x 03: lptimer 0x04: wdg 0x05: spi 0x06: uart 0x07: i2c 0x08: gpioa 0x09: apiob 0x0a: cmp0 0x0b: cmp1 0x0c: adc 0x0d: epwm 0x0e: lvd Other: Reserved Referece 3.12 section interrupt table (Vector Interrupt Controller)

7.4.18. Chip Version Number Register (SYSCFG_CHIPID)

Bit	Name	R/W	Reset	Description
31:0	CHIPID	R	0x00B5_700A	Chip Version Number Register (Read only) CHIPID[31:8]: Chip naming is fixed as follows 24'h00B570 CHIPID[7:0]: Chip version numbers, for example, 0A represents version A, 0B represents version B. Version numbers can be modified through metal plate alterations.

8. Flash memory controller (FMC)

8.1. Flash features

PEC930 embed a 32-bit wide flash memory of 32Kbytes available for storing programs and data.

Erase command is executed on 512-byte sector or whole chip basis. Flash can be programmed by using external programming tool. Program or erase command can be executed during runtime.

Flash reliably stores memory contents even after 100,000 program and erase cycles.

Basic features:

- Flash Array organized as one 256Kbit (8K x 32) Flash array.
- Fast erase and program:

  Chip erase:30 - 40 ms

  Sector erase:2 - 3 ms

  Word program:5 - 6.5 us
- Endurance 100K Cycles

8.2. Flash operations

Due to PEC930 flash memory controller, user can easily perform flash operations by setting relevant registers.

Flash operations:

- Read operation
- Program operation (Target cell should be 0xFFFFFFFF before excuting program operation)
- 512-byte sector erase
- Chip erase

8.3. Flash sector erase flow

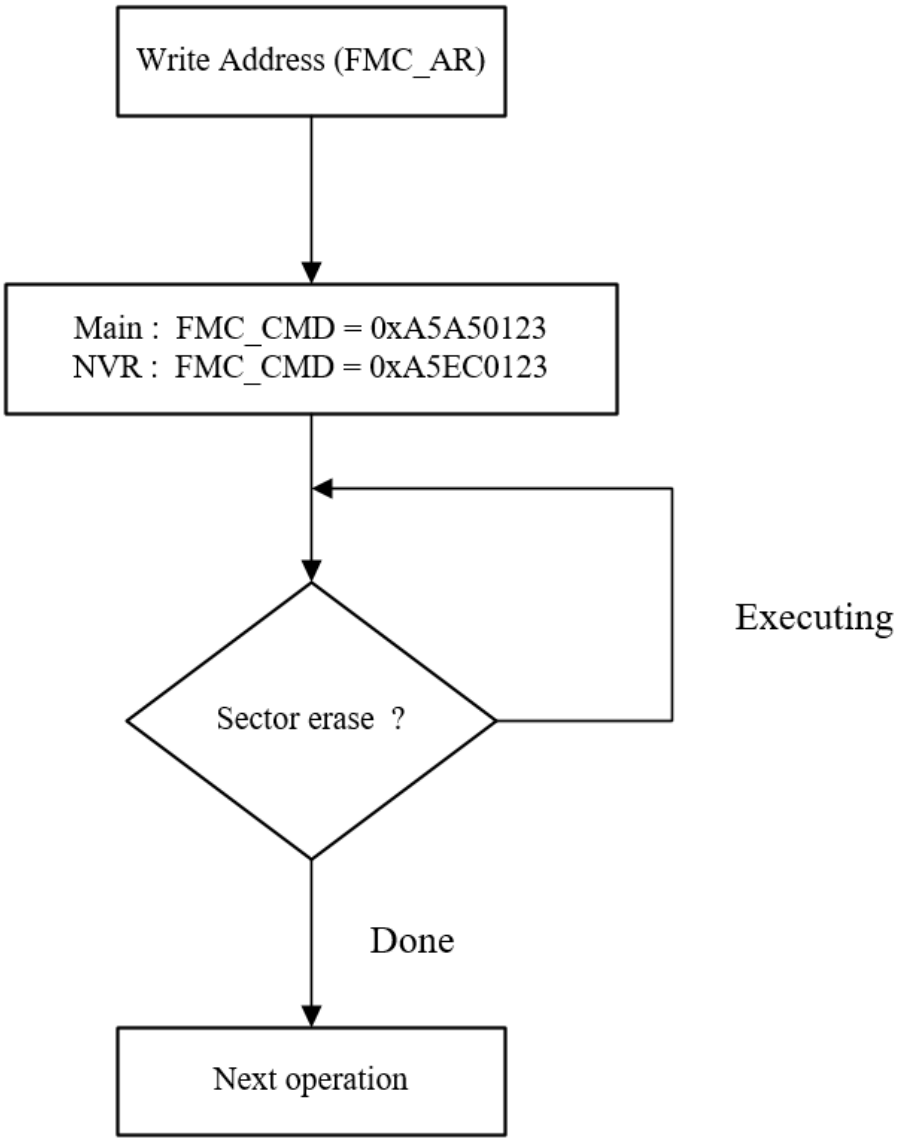


Fig. 8-1 Sector erase flow chart

8.4. Flash program flow

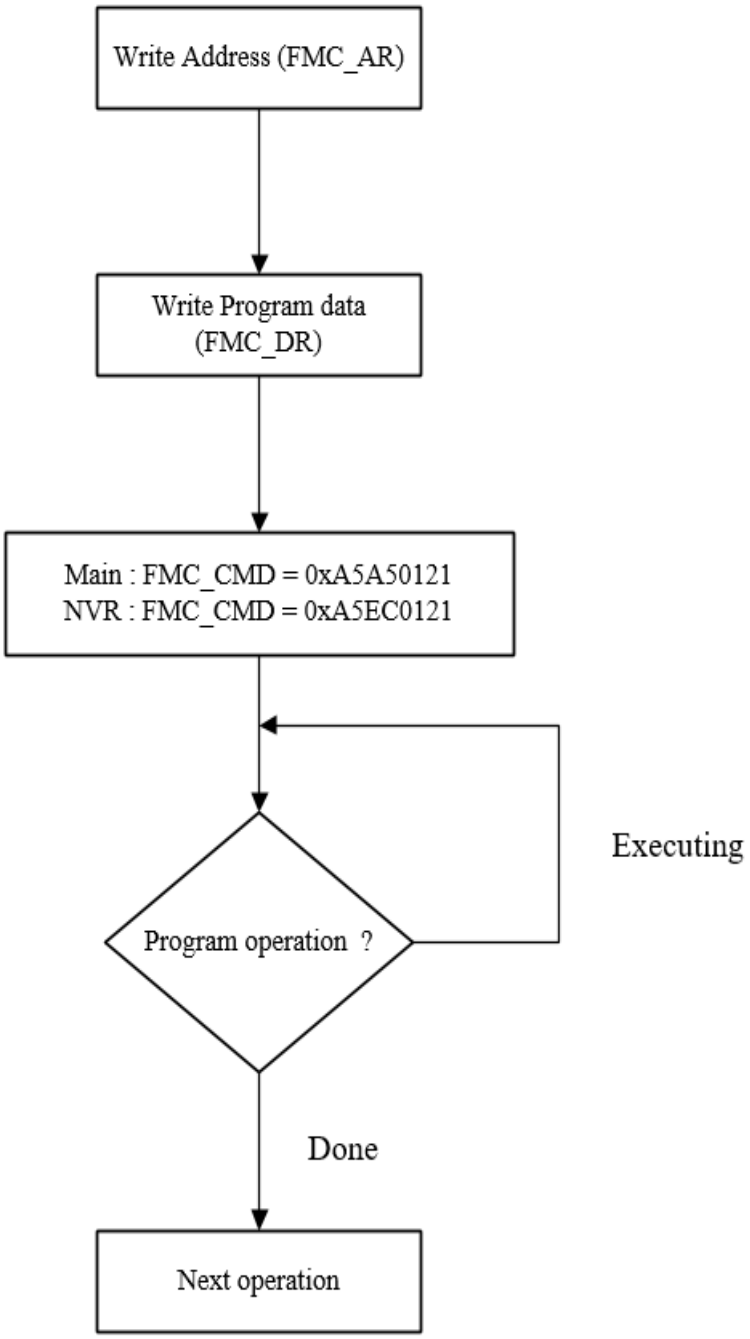


Fig. 8-2 Program flow chart

8.5. Flash erase / program clock

1Mhz erase / program clock is required for flash operations. Erase / program clock can be generated by setting FLDIV register. PEC930 flash controller does not support 0 wait cycle. The minimum default wait cycle is one .

8.6. Register table

Address	Name	Description
0x4000_F800	FMC_CMD	Flash command register
0x4000_F804	FMC_SR	Flash status Register
0x4000_F808	-	Reserved
0x4000_F80C	FMC_AR	Flash address register
0x4000_F810	FMC_DR	Flash data register
0x4000_F820	FMC_ACM	Flash access memory count register
0x4000_F824	-	Reserved
0x4000_F828	FMC_DIV	Flash clock divison register

Table 8-1 FLASH register table

8.7. Register description

8.7.1. Flash command register (FMC_CMD)

Bit	Name	R/W	Reset	Description
31:16	KEY	W	0x0	Security key for FMC_CMD write operation. For any write operation of this register to be valid, a security key **"0xA5A5"** must be written simultaneously. Security key **"0xA5A5"** is only valid for main array operations. NVR sectors operations requires another security key, which is "0xA5EC".
15:11	-	R	0x0	Reserved
10	ACMR	R/W	0x0	Access flash memory counter reset bit 0: Flash access counter is continuously enabled if ACE = 1, disabled if ACE = 0. 1: Flash access counter clear to 0
9	ACME	R/W	0x0	Access flash memory counter enable bit 0: Flash access counter disable 1: Flash access counter enable
8	UNLOCK	R/W	0x0	Lock bit for flash operation command 0: Flash operation command invalid 1: Flash operation command valid.
7:5	NWS	R/W	0x1	Flash wait cycle config 000: Doesn't support 0 wait cycle. 001: Wait cycle set to 1. 010: Wait cycle set to 2

				011: Wait cycle set to 3 100: Wait cycle set to 4 101: Wait cycle set to 5 110: Wait cycle set to 6 111: Wait cycle set to 7
4	-	R	0x0	Reserved
3:1	CMD	R/W	0x0	Flash command ID 000: Word program 001: Sectore erase 010: Chip erase 011: Reserved 1xx: Reserved
0	START	R/W	0x0	Flash operation start bit Erase operation 0: Erase operation idle 1: Start erase operation Read operation 0: Read operation done 1: Read operation in process (Automatically cleared to 0 when read operation is done)

8.7.2. Flash status register (FMC_ISR)

Bit	Name	R/W	Reset	Description
31:15	-	R	0x0	Reserved
14	IOSC60M_TRIM_ERR	R	0x0	0: iosc60m trim value valid 1: iosc60m trim value invalid
13	IOSC32K_TRIM_ERR	R	0x0	0: iosc32k trim value valid 1: iosc32k trim value invalid
12	VBUF_TRIM_ERR	R	0x0	0: vbuf trim value valid 1: vbuf trim value invalid
11	LDO_TRIM_ERR	R	0x0	0: ldo trim value valid 1: ldo trim value invalid
10	EXT_RSTN_TRIM_ERR	R	0x0	0: ext_rstn trim value valid 1: ext_rstn trim value invalid
9	PROTECT_R1_TRIM_ERR	R	0x0	0: protect_r1 trim value valid 1: protect_r1 trim value invalid
8	PROTECT_R2_TRIM_ERR	R	0x0	0: protect_r2 trim value valid 1: protect_r2 trim value invalid
7	OSCHF_TC_TRIM_ERR	R	0x0	0: oschf_tc trim value valid 1: oschf_tc trim value invalid
6	HCM	R/W1c	0x0	Access memory counter full flag Writing '1' to this bit will clear the flag. 0: Counter value is smaller than 0xFF00_0000 1: Counter reaches 0xFF00_0000.
5:4	-	R	0x0	Reserved

3	ADDR_ERR	R/W1c	0x0	FLAR error flag Writing '1' to this bit will clear the flag. 0: Flash address valid 1: Flash address invalid (out of range)
2	-	R	0x0	Reserved
1	KEY_ERR	R/W1c	0x0	FLCMD.KEY error flag Writing '1' to this bit will clear the flag. 0: KEYCODE is valid 1: KEYCODE is invalid
0	CMD_END	R/W1c	0x0	Flash operation done flag Writing '1' to this bit will clear the flag. 0: Flash operation still in process 1: Flash operation done

Note: If any trim_err_flag (bit [14:7]) is 1, please set SYSCFG_REBOOT_UNLOCK and SYSCFG_SYSRSTCR to relatch trim value.

8.7.3. Flash address register (FMC_AR)

Bit	Name	R/W	Reset	Description
31:22	-	R	0x0	Reserved
21	NVR	R/W	0x0	Flash NVR address select bit 0: Maps to main array 1: Maps to NVR array
20:15	-	R	0x0	Reserved
14:2	AR	R/W	0x0	PEC930 uses word addressing, each unique address corresponds to a "word" of data.
1:0	-	R	0x0	Reserved

8.7.4. Flash data register (FMC_DR)

Bit	Name	R/W	Reset	Description
31:0	DR	RW	0x0	It is the data that is to be written to the flash. 0 will be ignored only 1 will be wriiten. (Target cell should be erased to 0xFFFFFFFF before excuting program operation)

8.7.5. Flash access memory count register (FMC_ACM)

Bit	Name	R/W	Reset	Description
31:0	ACM	R	0x0	Counts every time MCU access to flash. When counter reaches 0xFF00_0000, FMC_ISR.HCM will be set to 1. At anytime when FMC_CMD.ACMR is set to 1, counter will be clear to 0.

8.7.6. Flash clock divison register (FMC_DIV)

Bit	Name	R/W	Reset	Description
31:9	-	R	0x0	Reserved
8	-	W	0x1	Must always remain 1.
7:0	DIV	RW	0x18	Divide factor for system clock to generate 1Mhz clock for erase and program. The formula is shown as below. $freq_{TICK} = \frac{freq_{SYS}}{FLDIV}$ Note: $freq_{TICK}$ is the frequency of the generated clock， $freq_{SYS}$ is the frequency of the system clock。

9. Vector Interrupt Controller

PEC930 RISC-V improved core interrupt controller (ECLIC) is integrated to achieve efficient exception and interrupt handling. ECLIC is designed to provide low-latency, vectorized, preemptive interrupts for RISC-V systems. When ECLIC is activated, it includes and replaces the existing RISC-V processor local interrupt controller. The ECLIC design has a basic design that requires minimal hardware, but it supports additional extensions to provide hardware acceleration. ECLIC is designed to support various software ABIs and interrupt models without compromising the implementation of high-performance processors through complex hardware. It is tightly coupled to the processor core. More detailed information about ECLIC can be found in the RISC-V technical reference manual.

9.1. Features

PEC930 interrupt controller has the following characteristics:

- Supports 15 hardware interrupts.
- 3 interrupt priority configuration bits – 8 interrupt priority levels(Reference N203 Spec).
- Supports exception preemption and tail-grabbing interrupts.
- Wakes the system from power-saving mode.

9.2. Interrupt Function Description

The PEC930 and the Improved Core Interrupt Controller (ECLIC) prioritize and handle all exceptions in machine mode.The PEC930 supports tail-biting interrupts, enabling back-to-back interrupts and significantly reducing the overhead of repeated state switching. The table below lists all interrupt types.

	IRQ Number	Exception or Interrupt	Priority	Program Addr Vector	Type
	0	reserved	(lowest)	Value of MTVT +4*0	reserved
	1	reserved		Value of MTVT +4*1	reserved
	2	reserved		Value of MTVT +4*2	reserved
	3	Machine Software interrupt		Value of MTVT +4*3	Level
	4	reserved		Value of MTVT +4*4	reserved
	5	reserved		Value of MTVT +4*5	reserved
	6	reserved		Value of MTVT +4*6	reserved
	7	Machine Timer interrupt		Value of MTVT +4*7	Level
	8	reserved		Value of MTVT +4*8	reserved
	9	reserved		Value of MTVT +4*9	reserved
	10	reserved		Value of MTVT +4*10	reserved
	11	reserved		Value of MTVT +4*11	reserved
	12	reserved		Value of MTVT +4*12	reserved
	13	reserved		Value of MTVT +4*13	reserved
	14	reserved		Value of MTVT +4*14	reserved
	15	reserved		Value of MTVT +4*15	reserved
	16	reserved		Value of MTVT +4*16	reserved
	17	reserved		Value of MTVT +4*17	reserved
	18	reserved		Value of MTVT +4*18	reserved
1	19	Timer0		Value of MTVT +4*19	Level

	IRQ Number	Exception or Interrupt	Priority	Program Addr Vector	Type
2	20	Timer1		Value of MTVT +4*20	Level
3	21	Timer2		Value of MTVT +4*21	Level
4	22	LPTIMER		Value of MTVT +4*22	Level
5	23	WDG		Value of MTVT +4*23	Level
6	24	SPI		Value of MTVT +4*24	Level
7	25	UART		Value of MTVT +4*25	Level
8	26	I2C-		Value of MTVT +4*26	Level
9	27	GPIOA		Value of MTVT +4*27	Level
10	28	GPIOB		Value of MTVT +4*28	Level
11	29	CMP0		Value of MTVT +4*29	Level
12	30	CMP1		Value of MTVT +4*30	Level
13	31	ADC		Value of MTVT +4*31	Level
14	32	EPWM		Value of MTVT +4*32	Level
15	33	LVD	(highest)	Value of MTVT +4*33	Level

Table 9-1 ECLIC interrupt number

Note: Reference 3.12

9.3. Enter sleep mode

The PEC9300 can enter a sleep state using the WFI instruction. When the processor executes the WFI instruction, it will:

- Immediately stop the execution of the current instruction stream.
- Wait for the processor core to complete any outstanding transactions that have not yet been completed, such as instruction fetch and data read/write operations, to ensure that all operations sent to the bus are completed.
    - Note: If a memory access error occurs while waiting for an operation on the bus to complete, the system will enter exception handling mode instead of sleeping.

Once all outstanding transactions have completed, the processor will safely enter an idle state, which can be called a "sleep" state.

Note: When entering deep sleep mode, the processor core will no longer be able to be debugged via the JTAG interface.

9.4. Exit sleep mode

PEC9300 to exit sleep mode are as follows:

- PEC9300 can be woken up un the following ways:
    1. Interrupt
    2. Event

9.4.1. Interrupt Wake-up

- If the mstatus.MIE field is configured to 1 (indicating that global interrupts are enabled), then:
  - When the IRQC (which arbitrates interrupts from external requests) sends an interrupt to the processor core, the processor core is awakened and enters the interrupt service routine to begin execution.
- If the mstatus.MIE field is configured to 0 (indicating that global interrupts are disabled), then:
  - If the wfe.WFE field of the CSR register is configured to 0, then:

    When IRQC (which arbitrates interrupts from external requests) sends an interrupt to the processor core, the processor core is awakened and continues to execute the previously stopped instruction stream sequentially (instead of entering the interrupt service routine).
  - If the wfe.WFE field of the CSR register is configured to 1, then wait for an Event to wake up.

9.4.2. Event Wake-up

An Event can wake up the processor core when the following conditions are met:

- If the mstatus.MIE field is configured to 0 (indicating that global interrupts are disabled) and the CSR register wfe.WFE field is configured to 1, then:
- When the processor core detects that the input signal rx_evt (called the Event signal) is high, the processor core is awakened and continues the instruction stream that was previously stopped (instead of entering the interrupt service routine).

9.5. Wait for Interrupt mechanism

The Wait for Interrupt mechanism refers to putting the processor core into a sleep mode and then waiting for an interrupt to wake it up. Upon waking, the processor core enters the corresponding interrupt handler.

As described in Sections 9.3 and 9.4, the Wait for Interrupt mechanism can be implemented directly using the WFI instruction (with the mstatus.MIE field configured to 1).

9.6. Wait for Event mechanism

The Wait for Event (Wait for Event) mechanism refers to putting the processor core into sleep mode and then waiting for an event to wake it up. Upon waking, the processor resumes the previously stopped program (instead of entering the interrupt handler).

As described in Sections 9.3 and 9.4, the Wait for Event mechanism can be implemented directly through the WFI instruction, in conjunction with the following instruction sequence:

Step 1: Configure the wfe.WFE field to 1

Step 2: Call the WFI instruction. After calling this instruction, the processor enters sleep mode. When the event wakes it up, it will continue execution.

Step 3: Restore the wfe.WFE field to 0

9.7. List of Relevant Registers

PEC930 the interrupt system provides registers, and the address spaces of each register are shown below:

Address	Register	Description
0x4001F048	EVT_SEL	EVT Trigger source selection control bit
0x4001F04C	NMI_BK_SEL	NMI / BK Trigger source selection control bit

Table 9-2 Interrupt controller register table

Note:

Reference 7.3.16

Reference 7.3.17

10. GPIO

There are up to 22 general purpose I/O ports (GPIO) for the device to implement logic input/output functions.

- PORT A: PA_0 ~ PA_15
- PORT: PB_0 ~ PB_5

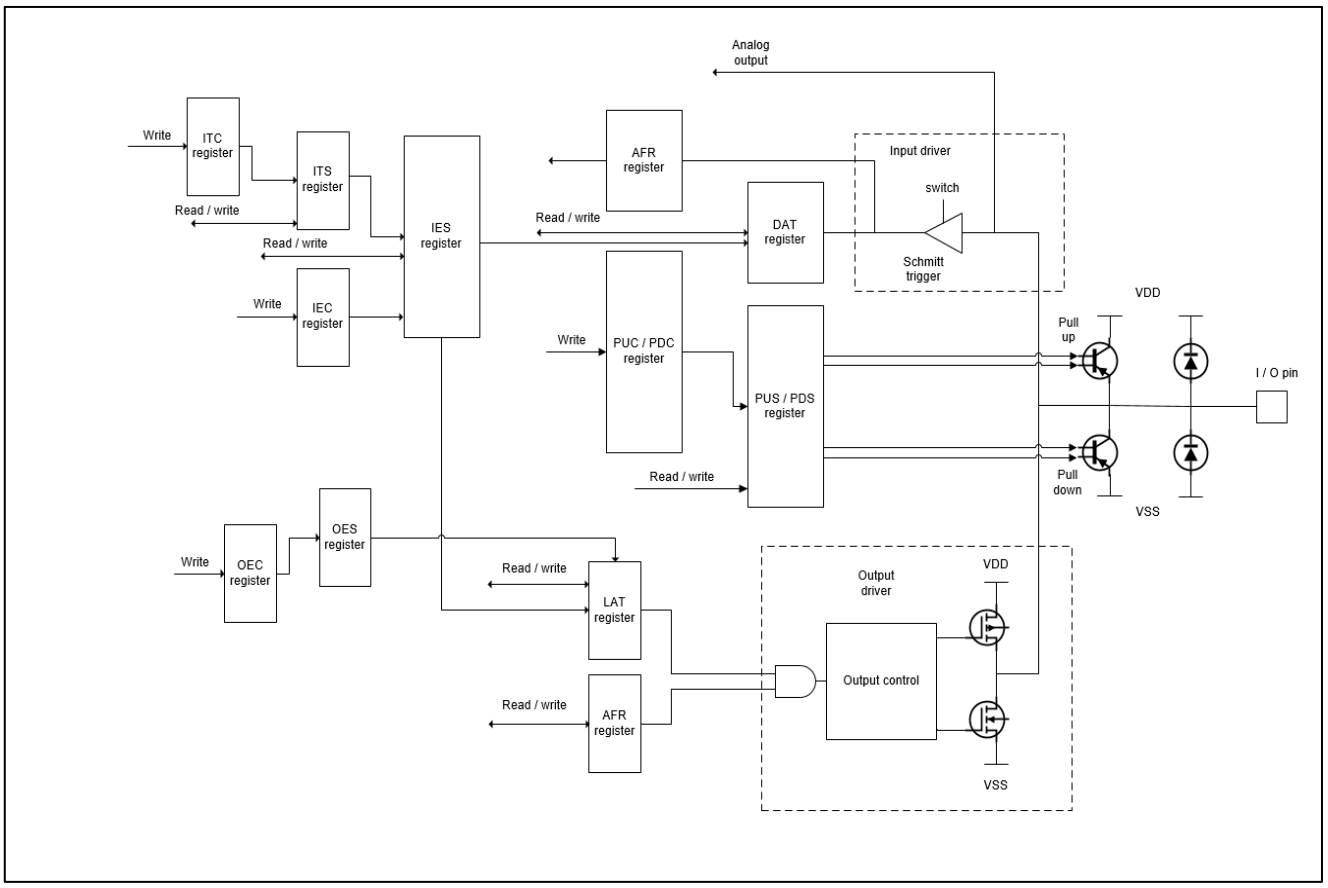


Fig. 10-1 GPIO block diagram

10.1. Features

When the I/O port is programmed as GPIO, the features are shown as below:

- All ports can be configured as input or output.
- All ports have external interrupt capability.
- Pull up and pull down resistors are disabled when configured as output.
- I/O ports are configured in input floating mode after reset.
- All GPIO pins have an internal weak pull-up and weak pull-down which can be chosen when configured as input.
- All ports can be programmed as open-drain or PMOS open-drain mode.
- Provides GPIO bit manipulation through bit masking.。
- Uses separate set and clear control registers to ensure thread-safe port operations.
- Configures I/O pins as GPIO or alternate functions (such as peripheral functions) via peripheral multiplexing registers.
- Supports port mapping operations, allowing control of individual bits or selected groups of bits within a port.

10.2. Input output configurations

ALL GPIO can be configured as input or output. Input or output is configured by setting PnOES (n = A or B) and PxOEC registers. For more details, please refer to the register description section below.

10.3. External interrupt

All GPIO ports interrupt can be configured by setting four pairs of registers, including GPIOn_IES, GPIOn_IEC, GPIOn_ITS0, GPIOn_ITC0，GPIOn_ITS1, GPIOn_ITC1, GPIOn_PLS and GPIOn_PLC. GPIOn_IES and GPIOn_IEC control whether interrupt is enabled or disabled. GPIOn_ITS0, GPIOn_ITC0, GPIOn_ITS1, GPIOn_ITC1 determines types of interrupt. GPIOn_PLS, GPIOn_PLC determines interrupt polarity.

Interrupt enable	Interrupt polarity	Interrupt type 0	Interrupt type 1	Interrupt response mode
0	-	-	-	Interrupt disable
1	0	0	0	Low active
1	0	1	0	Negative edge
1	1	0	0	High active
1	1	1	0	Positive edge
1	X	X	1	Signal toggle

Table 10-1 Interrupt response mode table

GPIOn_IST status register will be set to 1 after interrupt is triggered.Every I/O interrupt correspond to a bit of GPIOn_IST. Writing '1' to the correspond bit will clear the flag.

10.4. Input pull-up or pull down

When GPIO pin is configured as input, weak pull-up and pull-down resistors could be chosen (resistance value is around 10K). Pull up and pull down resistors are disabled when configured as output. GPIOn_PUS enables pull-up resistors, GPIOn_PUC disables pull-up resistors. GPIOn_PDS enables pull-down resistors, GPIOn_PDC disables pull-down resistors.

10.5. Output driving capability

Under 5V supply condition, all ports driving capability are up to 40mA.

10.6. Output open-drain

When configured as output, it is possible to use the output driver in open-drain mode or PMOS open-drain mode.‘0’ on the output register activates N-MOS, and ’1’ on the output register places the port in a high-impedance state. A external pull-up resistor is required to define a high logic level. Pull-up voltage can’t not be higher than Vdd + 0.6V. Open-drain function can be enable or disable by setting register GPIOn_ODS, GPIOn_ODC, GPIOn_PODS, GPIOn_PODC.

10.7. Schmitt trigger input function

Schmitt trigger input function can be enable or disable by setting register GPIOn_STE & GPIOn_STD.

10.8. Alternate function input output

I/O port is default as GPIO (except debug port & PB1 default as reset). Port can be configured as AFIO by setting register GPIOn_AFR, FN1_AFR & FN2_AFR. The detail alternate function assignments for each port are in the register description.

10.9. Notes

When switching the port pin mode under any condition, first set the corresponding enable register to 0. After completing the register configuration, set the enable bit back to 1.

10.10. Register table

Address	Name	Description
0x4001_0000 ~ 0x4001_08FC	-	Reserved
0x4001_1000	GPIOA_DAT	GPIO A input data register
0x4001_1004	GPIOA_LAT	GPIO A output data latch register
0x4001_1008	GPIOA_ITS1	GPIO A interrupt type set register 1
0x4001_100C	GPIOA_ITC1	GPIO A interrupt type clear register 1
0x4001_1010	GPIOA_OES	GPIO A output enable register
0x4001_1014	GPIOA_OEC	GPIO A output enable clear register
0x4001_1018	GPIOA_INES	GPIO A input enable register
0x4001_101C	GPIOA_INEC	GPIO A intput enable clear register
0x4001_1020	GPIOA_IES	GPIO A interrupt enable register
0x4001_1024	GPIOA_IEC	GPIO A interrupt clear register
0x4001_1028	GPIOA_ITS0	GPIO A interrupt type set register 0
0x4001_102C	GPIOA_ITC0	GPIO A interrupt type clear register 0
0x4001_1030	GPIOA_PLS	GPIO A interrupt polarity set register
0x4001_1034	GPIOA_PLC	GPIO A interrupt polarity clear register
0x4001_1038	GPIOA_IST	GPIO A interrupt status register
0x4001_103C	GPIOA_PUS	GPIO A pull-up set register
0x4001_1040	GPIOA_PUC	GPIO A pull-up clear register
0x4001_1044	GPIOA_ODS	GPIO A open-drain set register
0x4001_1048	GPIOA_ODC	GPIO A open-drain clear register
0x4001_104C	GPIOA_PDS	GPIO A pull-down set register
0x4001_1050	GPIOA_PDC	GPIO A pull-down clear register
0x4001_1054	GPIOA_PODS	GPIO A open sink set register
0x4001_1058	GPIOA_PODC	GPIO A open sink clear register
0x4001_105C	GPIOA_STE	GPIO A schmitt-trigger enable register
0x4001_1060	GPIOA_STD	GPIO A schmitt-trigger disable register
0x4001_1070	GPIOA_AFR	GPIO A AF control register
0x4001_1074	GPIOB_AFR	GPIO B AF control register
0x4001_1078	FN1_AFR	Peripheral Alternate Func control register 1
0x4001_107C	FN2_AFR	Peripheral Alternate Func control register 2

0x4001_11F8 ~ 0x4001_1BFC	-	Reserved
0x4001_2000	GPIOB_DAT	GPIO B input data register
0x4001_2004	GPIOB_LAT	GPIO B output data register
0x4001_2008	GPIOB_ITS1	GPIO B interrupt type set register 1
0x4001_200C	GPIOB_ITC1	GPIO B interrupt type clear register 1
0x4001_2010	GPIOB_OES	GPIO B output enable register
0x4001_2014	GPIOB_OEC	GPIO B output enable clear register
0x4001_2018	GPIOB_INES	GPIO B input enable register
0x4001_201C	GPIOB_INEC	GPIO B input enable clear register
0x4001_2020	GPIOB_IES	GPIO B interrupt enable register
0x4001_2024	GPIOB_IEC	GPIO B interrupt clear register
0x4001_2028	GPIOB_ITS0	GPIO B interrupt type set register 0
0x4001_202C	GPIOB_ITC0	GPIO B interrupt type clear register 0
0x4001_2030	GPIOB_PLS	GPIO B interrupt polarity set register
0x4001_2034	GPIOB_PLC	GPIO B interrupt polarity clear register
0x4001_2038	GPIOB_IST	GPIO B interrupt status register
0x4001_203C	GPIOB_PUS	GPIO B pull-up set register
0x4001_2040	GPIOB_PUC	GPIO B pull-up clear register
0x4001_2044	GPIOB_ODS	GPIO B open-drain set register
0x4001_2048	GPIOB_ODC	GPIO B open-drain clear register
0x4001_204C	GPIOB_PDS	GPIO B pull-down set register
0x4001_2050	GPIOB_PDC	GPIO B pull-down clear register
0x4001_2054	GPIOB_OSRS	GPIO B open sink set register
0x4001_2058	GPIOB_OSRC	GPIO B open sink clear register
0x4001_205C	GPIOB_STE	GPIO B schmitt-trigger set register
0x4001_2060	GPIOB_STD	GPIO B schmitt-trigger clear register
0x4001_2064 ~ 0x4001_2BFC	-	Reserved

Table 10-2 GPIO register table

10.11. Register description

10.11.1. GPIOn input data register (GPIOn_DAT)(n=A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_DAT	R/W	0x0	Contain the input value of the corresponding I/O port. Writing to this register actually writes to GPIOn_LAT. PORTA_DAT [15:0] correspond to PORT A PORTB_DAT [5:0] correspond to PORT B Read operation: 0: Port value is 0 1: Port value is 1 Write operation: 0: Port value is 0 1: Port value is 1

*1:When bit is reserved, its R/W attribute is R (read-only), and is always reads as 0. When bit is used, its R/W attribute becomes RW (read/write), allowing both read and write operations.

10.11.2. GPIOn output data latch register (GPIOn_LAT)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_LAT	R/W	0x0	Port output data The value is from the output data register, not the actually value of the port. PORTA_LAT [15:0] correspond to PORT A PORTB_LAT [5:0] correspond to PORT B

10.11.3. GPIOn interrupt type set register 1 (GPIOn_ITS1)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ITS1	R/W	0x0	Port interrupt type set PORTA_ITS1 [15:0] correspond to PORT A PORTB_ITS1 [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Interrupt type is determine by GPIOn_ITS0 1:I nterrupt type is level trigger, and can be trigger by high or low level. Note:ITS1 priority is higher than ITS0.

10.11.4. GPIOn interrupt type clear register 1 (GPIOn_ITC1)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ITC1	R/W	0x0	Port interrupt type clear PORTA_ITC1 [15:0] correspond to PORT A PORTB_ITC1 [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Interrupt type is determine by GPIOn_ITS0

10.11.5. GPIOn output enable set register (GPIOn_OES)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:10	-	*1	0x0	Reserved
15:0	PORTn_OES	R/W	0x0	Port output enable set PORTA_OES [15:0] correspond to PORT A PORTB_OES [5:0] correspond to PORT B Writing “0” to this bit has no effect. 0: Output mode disable 1: Output mode enable

10.11.6. GPIOn output enable clear register (GPIOn_OEC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_OEC	R/W	0x0	Port output enable clear PORTA_OEC [15:0] correspond to PORT A PORTB_OEC [5:0] correspond to PORT B This bit is always read as 0. Writing “0” to this bit has no effect.

10.11.7. GPIOn input enable set register (GPIOn_INES)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_INES	R/W	0xFF	Port input enable set PORTA_INES [15:0] correspond to PORT A PORTB_INES [5:0] correspond to PORT B Writing “0” to this bit has no effect. 0: Input mode disable 1: Input mode enable

10.11.8. GPIOn input enable clear register (GPIOn_INEC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_INEC	R/W	0x0	Port input enable clear PORTA_INES [15:0] correspond to PORT A PORTB_INES [5:0] correspond to PORT B This bit is always read as 0. Writing “0” to this bit has no effect.

10.11.9. GPIOn interrupt enable set register (GPIOn_IES)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_IES	R/W	0x0	Port interrupt enable set PORTA_IES [15:0] correspond to PORT A PORTB_IES [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Port interrupt disable 1: Port interrupt enable

10.11.10. GPIOn interrupt enable clear register (GPIOn_IEC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_IEC	R/W	0x0	Port interrupt enable clear PORTA_IEC [15:0] correspond to PORT A PORTB_IEC [5:0] correspond to PORT B Writing “0” to this register has no effect. Always read as 0.

10.11.11. GPIOn interrupt type set register 0 (GPIOn_ITS0)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ITS0	R/W	0x0	Port interrupt type set PORTA_ITS0 [15:0] correspond to PORT A PORTB_ITS0 [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Level trigger 1: Edge trigger

10.11.12. GPIOn interrupt type clear register 0 (GPIOn_ITC0)(n=A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ITC0	R/W	0x0	Port interrupt type clear PORTA_ITC0 [15:0] correspond to PORT A PORTB_ITC0 [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Port interrupt type is level trigger

10.11.13. GPIOn interrupt polarity set register (GPIOn_PLS)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PLS	R/W	0x0	Port interrupt polarity set PORTA_ITC1 [15:0] correspond to PORT A PORTB_ITC1 [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Low level or negedge trigger interrupt 1: High level or posedge trigger interrupt

10.11.14. GPIOn interrupt polarity clear register (GPIOn_PLC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PLC	R/W	0x0	Port interrupt type clear PORTA_PLC [15:0] correspond to PORT A PORTB_PLC [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Low level or negedge trigger interrupt

10.11.15. GPIOn interrupt flag status clear register (GPIOn_IST)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_IST	R/W1c	0x0	Port interrupt flag status PORTA_IST [15:0] correspond to PORT A PORTB_IST [5:0] correspond to PORT B Writing “0” to this register has no effect. Writing '1' to this register will clear the flag. 0: No interrupt occured 1: Interrupt occured

10.11.16. GPIOn input pull-up set register (GPIOn_PUS)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PUS	R/W	0x0 or 0x6	Port input pull-up set PORTA_PUS [15:0] correspond to PORT A, reset value is 0x0 PORTB_PUS [5:0] correspond to PORT B, reset value is 0x6 Writing “0” to this register has no effect. 0: Input pull-up disable 1: Input pull-up enable

10.11.17. GPIOn input pull-up clear register (GPIOn_PUC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PUC	R/W	0x0	Port input pull-up clear PORTA_PUC [15:0] correspond to PORT A PORTB_PUC [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Input pull-up disable

10.11.18. GPIOn output open-drain set register (GPIOn_ODS)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ODS	R/W	0x0	Port output open-drain set PORTA_PUS [15:0] correspond to PORT A PORTB_PUS [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Output open-drain disable 1: Output open-drain enable

10.11.19. GPIOn output open-drain clear register (GPIOn_ODC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_ODC	R/W	0x0	Port output open-drain clear PORTA_ODC [15:0] correspond to PORT A PORTB_ODC [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Output open-drain disable

10.11.20. GPIOn input pull-down set register (GPIOn_PDS)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PDS	R/W	0x0	Port input pull-up set PORTA_PDS [15:0] correspond to PORT A PORTB_PDS [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Input pull-down disable 1: Input pull-down enable

10.11.21. GPIOn input pull-down clear register (GPIOn_PDC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PDC	R/W	0x0	Port input pull-down clear PORTA_PUC [15:0] correspond to PORT A PORTB_PUC [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Input pull-down disable

10.11.22. GPIOn output PMOS open-drain set register (GPIOn_PODS)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PODS	R/W	0x0	Port output PMOS open-drain set PORTA_PODS [15:0] correspond to PORT A PORTB_PODS [5:0] correspond to PORT B Writing “0” to this register has no effect. 0: Output PMOS open-drain disable 1: Output PMOS open-drain enable

10.11.23. GPIOn output PMOS open-drain clear register (GPIOn_PODC)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_PODC	R/W	0x0	Port output PMOS open-drain clear PORTA_ODC [15:0] correspond to PORT A PORTB_ODC [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Output PMOS open-drain disable

10.11.24. GPIOn Schmitt-Trigger enable register (GPIOn_STE)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	Portn_STE	R/W	0x1	Schmitt-Trigger function enable PORTA_STE [15:0] correspond to PORT A PORTB_STE [5:0] correspond to PORT B Writing “0” to this register has no effect. 1: Schmitt-Trigger function enable

10.11.25. GPIOn Schmitt-Trigger disable register (GPIOn_STD)(n = A, B)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:6	-	*1	0x0	Reserved
15:0	PORTn_STD	R/W	0x0	Schmitt-Trigger function disable PORTA_STD [15:0] correspond to PORT A PORTB_STD [5:0] correspond to PORT B Always read as 0. Writing “0” to this register has no effect. 1: Schmitt-Trigger function disable

10.11.26. GPIOA alternate function control register (GPIOA_AFR)

Bit	Name	R/W	Reset	Description
31:24	-	R	0x0	Reserved
23:21	PA15_AFR	R/W	0x0	000: GPIO 001: TXD 010: RXD 011: MOSI 100: MISO 101: Reserved 110: SCL 111: Reserved
20:18	PA14_AFR	R/W	0x0	000: GPIO 001: Reserved 010: Reserved 011: SCK 100: Reserved 101: Reserved 110: MCO (Microcontroller Clock Output) 111: Reserved
17:15	PA5_AFR	R/W	0x0	000: GPIO 001: CS 010: Reserved 011: Reserved 100: Reserved 101: TIM2CH4 output 110: EPWM2N output 111: Reserved
14:12	PA4_AFR	R/W	0x0	000: GPIO 001: MISO 010: MOSI 011: Reserved 100: Reserved 101: TIM2CH3 output 110: EPWM2P output 111: Reserved
11:9	PA3_AFR	R/W	0x0	000: GPIO

				001: MOSI 010: MISO 011: Reserved 100: Reserved 101: TIM2CH2 output 110: EPWM2N output 111: Reserved
8:6	PA2_AFR	R/W	0x0	000: GPIO 001: SCK 010: Reserved 011: Reserved 100: Reserved 101: TIM2CH1 output 110: EPWM2P output 111: Reserved
5:3	PA1_AFR	R/W	0x0	000: GPIO 001: TXD 010: RXD 011: SCL 100: SDA 101: Reserved 110: EPWM1N output 111: Reserved
2:0	PA0_AFR	R/W	0x0	000: GPIO 001: RXD 010: TXD 011: SDA 100: SCL 101: Reserved 110: EPWM1P output 111: Reserved

10.11.27. GPIOB alternate function control register (GPIOB_AFR)

Bit	Name	R/W	Reset	Description
31:18	-	R	0x0	Reserved
17:15	PB5_AFR	R/W	0x0	000: GPIO 001: RXD 010: TXD 011: CS 100: TIM2CH1 output 101: SDA 110: EPWM1N output 111: Reserved
14:12	PB4_AFR	R/W	0x0	000: GPIO 001: TXD 010: RXD 011: MISO 100: MOSI 101: Reserved 110: EPWM1P output 111: Reserved
11:9	PB3_AFR	R/W	0x0	000: GPIO 001: TXD 010: SCL 011: MOSI 100: MISO 101: Reserved 110: EPWM2P output 111: Reserved
8:6	PB2_AFR	R/W	0x0	000: GPIO 001: SDA 010: SCL 011: SCK 100: TIM2CH1 output 101: Reserved 110: EPWM2N output 111: Reserved

5:3	PB1_AFR	R/W	0x0	000: GPIO 001: SCL 010: SDA 011: CS 100: TIM2CH1 output 101: Reserved 110: MCO (Microcontroller Clock Output) 111: Reserved
2:0	PB0_AFR	R/W	0x0	000: GPIO 001: RXD 010: TXD 011: MISO 100: MOSI 101: Reserved 110: SDA 111: Reserved

10.11.28. Peripheral alternate function control register 1 (FN1_AFR)

Bit	Name	R/W	Reset	Description
31:17	-	R	0x0	Reserved
16:13	EPETR	R/W	0x0	EPWM ETR remapping 0000: Reserved 0001: Remap to PA14 0010: Remap to PA15 0011: Remap to PB0 0100: Remap to PB1 0101: Remap to PB2 0110: Remap to PB3 0111: Remap to PB4 1000: Remap to PB5
12:10	ECAP2	R/W	0x0	ECAP2 remapping 000: Reserved 001: Remap to PA14 010: Remap to PA15 011: Remap to PB0 100: Remap to PB2 101: Remap to PB3 110: Remap to PB4
9:7	ECAP1	R/W	0x0	ECAP1 remapping 000: Reserved 001: Remap to PA14 010: Remap to PA15 011: Remap to PB0 100: Remap to PB2 101: Remap to PB3 110: Remap to PB4
6:4	ECAP0	R/W	0x0	ECAP0 remapping 000: Reserved 001: Remap to PA14 010: Remap to PA15 011: Remap toPB0 100: Remap to PB2 101: Remap to PB3

				110: Remap to PB4
3:0	BKIN	R/W	0x0	BK_IN remapping 0000: Reserved 0001: Remap to PA6 0010: Remap to PA7 0011: Remap to PA8 0100: Remap to PA9 0101: Remap to PA10 0110: Remap to PA11 0111: Remap to PA12 1000: Remap to PA13 1001: Remap to PB1 1010: Remap to PB3 1011: Remap to PB5

10.11.29. Peripheral alternate function control register 1 (FN2_AFR)

Bit	Name	R/W	Reset	Description
31:21	-	R	0x0	Reserved
20	I2C_PULL7	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PB3 pull high 1: Enable PB3 pull high
19	I2C_PULL6	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PB5 pull high 1: Enable PB5 pull high
18	I2C_PULL5	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PA0 pull high 1: Enable PA0 pull high
17	I2C_PULL4	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PA1pull high 1: Enable PA1 pull high
16	I2C_PULL3	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PA15 pull high 1: Enable PA15 pull high
15	I2C_PULL2	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PB0 pull high 1: Enable PB0 pull high
14	I2C_PULL1	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PB1 pull high 1: Enable PB1 pull high
13	I2C_PULL0	R/W	0x0	I2C pull High (pull high resistor value is 10Kohm) This bit is only valid when I2C is enabled. 0: Disable PB2 pull high 1: Enable PB2 pull high

12:9	T2ETR	R/W	0x0	TIM2 ETR remapping 0000: Reserved 0001: Remap to PA14 0010: Remap to PA15 0011: Remap to PB0 0100: Remap to PB1 0101: Remap to PB2 0110: Remap to PB3 0111: Remap to PB4 1000: Remap to PB5
8:6	TCAP2	R/W	0x0	TIM2 CAP2 remapping 000: Reserved 001: Remap to PA14 010: Remap to PA15 011: Remap to PB0 100: Remap to PB2 101: Remap to PB3 110: Remap to PB4
5:3	TCAP1	R/W	0x0	TIM2 CAP1 remapping 000: Reserved 001: Remap to PA14 010: Remap to PA15 011: Remap to PB0 100: Remap to PB2 101: Remap to PB3 110: Remap to PB4
2:0	TCAP0	R/W	0x0	TIM2 CAP0 remapping 000:Reserved 001:Remap to PA14 010:Remap to PA15 011:Remap to PB0 100:Remap to PB2 101:Remap to PB3 110:Remap to PB4

11. Basic timer(TIMx, x=0, 1, LP)

11.1. TIM0/1

Timer TIM0/1 is a 16-bit basic timer, operating in up-counting mode.

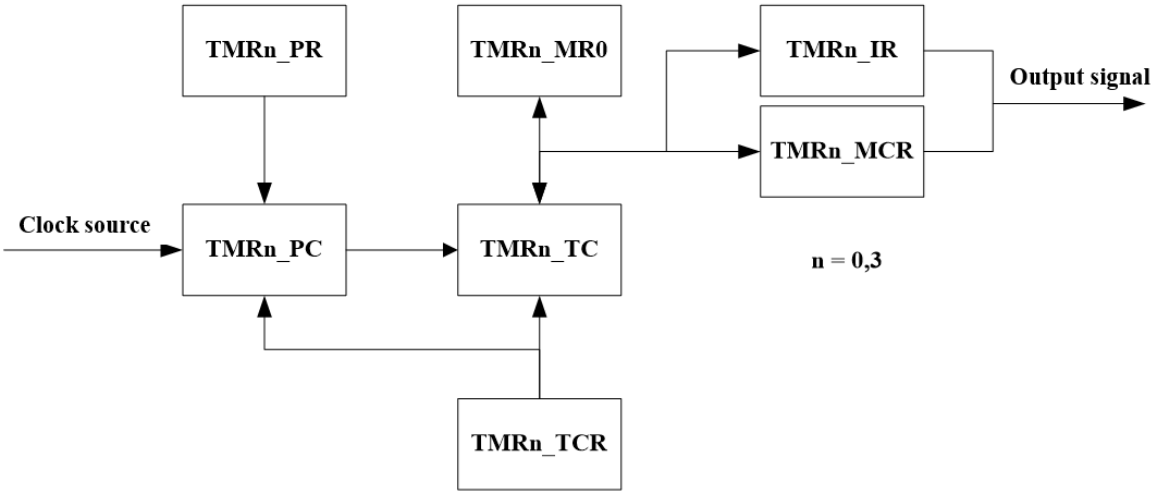


Fig. 11-1 Functional Block Diagram of Timer TIM0/1

The TIM0_PC register performs prescaling on the clock source based on the division factor set in the TIM0_PR register. The TIM0_TC register (Timer Counter) counts up using the clock generated by the TIM0_PC register. When the value in the TIM0_TC register matches the value in the TIM0_MR0 register, the TIM0_IR register asserts a compare match interrupt signal (by setting the MR0 bit in the TIM0_IR register). ◦ Based on the parameters configured in the TIM0_MCR register, Timer TIM0 generates the corresponding output signal (interrupt, timer reset, or timer stop).

The TIM0_TCR register is used to enable and reset Timer TIM0.

11.2. TIM0/1 Register table

Address	Name	Description
0x4000_0000	TIM0_IR	TIM0 interrupt Register
0x4000_0004	TIM0_TCR	TIM0 Control Register
0x4000_0008	TIM0_TC	TIM0 Counter Register
0x4000_000C	TIM0_PR	TIM0 Prescaler Ratio Register
0x4000_0010	TIM0_PC	TIM0 Prescaler Counter Register
0x4000_0014	TIM0_MCR	TIM0 Match Control Register
0x4000_0018	TIM0_MR0	TIM0 Match Value Register 0

Table 11-1 TIM0 register table

Address	Name	Description
0x4000_0800	TIM1_IR	TIM1 interrupt Register
0x4000_0804	TIM1_TCR	TIM1 Control Register
0x4000_0808	TIM1_TC	TIM1 Counter Register
0x4000_080C	TIM1_PR	TIM1 Prescaler Ratio Register
0x4000_0810	TIM1_PC	TIM1 Prescaler Counter Register
0x4000_0814	TIM1_MCR	TIM1 Match Control Register
0x4000_0818	TIM1_MR0	TIM1 Match Value Register 0

Table 11-2 TIM1 register table

11.3. LPTIM

The LPTIM timer is a 16-bit Low Power Timer, clocked by the internal 32 KHz oscillator or the system clock. The LPTIM timer is also capable of operating in low-power modes, and its output signal can be configured to wake the device from these modes.

The LPTIM timer contains synchronization logic between its two clock domains. When the LPTIM control registers are updated via the system clock, a subsequent read operation returns the updated value immediately; however, this updated value only takes effect after the timer synchronizes across two counter clock domain cycles.

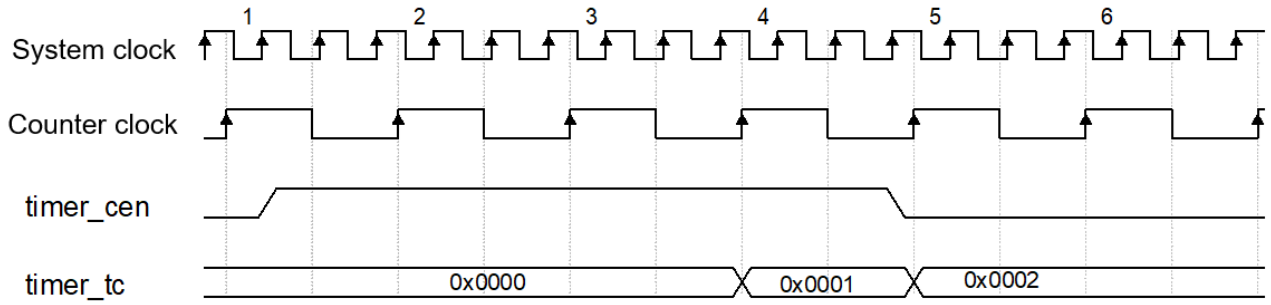


Fig. 11-2 LPTIM Enable Signal Timing Diagram

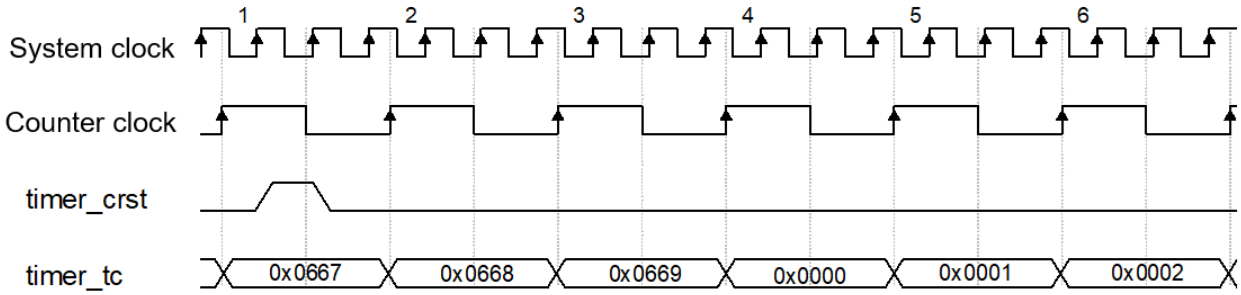


Fig. 11-3 LPTIM Reset Signal Timing Diagram

When configuring the LPTIM timer, all parameters should be set up first, and finally, the counter should be enabled.

11.4. LPTIM Register table

Address	Name	Description
0x4000_C800	LPTIM_IR	LPTIM Interrupt Register
0x4000_C804	LPTIM_TCR	LPTIM Control Register
0x4000_C808	LPTIM_TC	LPTIM Counter Register
0x4000_C80C	LPTIM_PR	LPTIM Prescaler Ratio Register
0x4000_C810	LPTIM_PC	LPTIM Prescaler Counter Register
0x4000_C814	LPTIM_MCR	LPTIM Match Control Register
0x4000_C818	LPTIM_MR0	LPTIM Match Value Register 0

Table 11-3 LPTIM register table

11.5. TIM0, 1, LPTIM Register discription

11.5.1. TIMn Interrupt Register(TIMn_IR(n = 0, 1, LPTIM))

The TIM_IR register contains the Match interrupt flag 0 (MR0) for the TIMn module. The corresponding Match interrupt flag 0 (MR0) is triggered whenever the value in the TIMn_TC register matches the value in the TIMn_MR0 register, regardless of the setting of the relevant interrupt enable bit (a parameter within the TIMn_MCR register).

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	MR0	R/W1c	0x0	MR0: Match interrupt flag 0 0: The value of the TIMn_TC (Timer Counter) register has not matched the value of the TIMn_MR0 (Match 0 Register) register. 1: The value of the TIMn_TC register has matched the value of the TIMn_MR0 register. This flag is cleared by writing '1' to this bit.

11.5.2. TIMn Control Register(TIMn_TCR(n = 0, 1, LPTIM))

Bit	Name	R/W	Reset	Description
31:7	-	R	0x0	Reserved
6:4	TIM0_TRIG	R/W	0x0	Bits [6:4] TIMn_TRIG (n = 0, 1): 0x0: Normal start of Timer n timing function (n=0, 1). 0x1: EPWM1P (CH1 output) Rising edge triggers Timer n timing function (AS CEn = 1). 0x2: EPWM2P (CH2 output) Rising edge triggers Timer n timing function (AS CEn = 1). 0x3: EPWM3P (CH3 output) Rising edge triggers Timer n timing function (AS CEn = 1). 0x4: EPWM1P (CH1 output) Falling edge triggers Timer n timing function (AS CEn = 1). 0x5: EPWM2P (CH2 output) Falling edge triggers Timer n timing function (AS CEn = 1). 0x6: EPWM3P (CH3 output) Falling edge triggers Timer n timing function (AS CEn = 1). 0x7: Reserved.
3:2	CLKS	R	0x0	Bits [3:2] CLKS: LPTIM clock selection control 00: System clock Others: No effect

1	CRst	R/W	0x0	Bit [1] CRst: Timer module software reset control bit When this bit is written to '1', a reset operation is performed on the Timer Counter register and the Prescaler Counter register. For LPTIM, due to synchronization logic, there is a delay from writing this register until the reset is complete. The hardware automatically clears this bit to '0' after the reset synchronization is complete. Read: 0: The timer is not in the process of being reset 1: The Timer Counter register and Prescaler Counter register are in the process of being reset Write: 0: No effect 1: Resets the Timer Counter register and Prescaler register. The hardware automatically clears this bit to '0' after the reset is complete
0	CEn	R/W	0x0	Bit [0] CEn: Timer module operation enable control bit 0: Timer counting function disabled 1: Timer counting function started (Base on Trigger Source)

LPTIM

Bit	Name	R/W	Reset	Description
31:4	-	R	0x0	Reserved
3:2	CLKS	R/W	0x0	Bits [3:2] CLKS: LPTIM clock selection control 00: System clock Other: OSC_32KHz clock
1	CRst	R/W	0x0	Bit [1] CRst: Timer module software reset control bit When this bit is written to '1', a reset operation is performed on the Timer Counter register and the Prescaler Counter register. For LPTIM, due to synchronization logic, there is a delay from writing this register until the reset is complete. The hardware automatically clears this bit to '0' after the reset synchronization is complete. Read: 0: The timer is not in the process of being reset 1: The Timer Counter register and Prescaler Counter register are in the process of being reset Write: 0: No effect 1: Resets the Timer Counter register and Prescaler register. The hardware automatically clears this bit to '0' after the reset is complete
0	CEn	R/W	0x0	Bit [0] CEn: Timer module operation enable control bit 0: Timer counting function disabled 1: Timer counting function started (Base on Trigger Source)

11.5.3. TIMn Counter Register(TIMn_TC(n = 0, 1, LPTIM))

The current count value of the timer, ranging from 0x0000 to 0xFFFF, which wraps around.

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:0	TC	R/W	0x0	Bits [15:0] TC: Timer current value This register is read-only in LPTIM. The value range is 0x0000 to 0xFFFF.

11.5.4. TIMn Prescaler Ratio Register(TIMn_PR(n = 0, 1, LPTIM))

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	PR	R/W	0x0	Bits [7:0] PR: Timer prescaler factor The value range is 0x00 to 0xFF. The formula for system clock division is: fTIM = fSYS / (PR + 1) Where: fTIM is the clock frequency of the timer counting operation fSYS is the system clock frequency of the chip PR is the prescaler factor The value range is 0x00 to 0xFF

11.5.5. TIMn Prescaler Counter Register(TIMn_PC(n = 0, 1, LPTIM))

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	PC	R/W	0x0	Bits [7:0] PC: Timer prescaler counter current value The value range is 0x00 to 0xFF

11.5.6. TIMn Match Control Register(TIMn_MCR(n = 0, 1, LPTIM))

Sets the timer match control mode.

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2	MR0STOP	R/W	0x0	Bit [2] MR0S: Counter stop control bit when TIMn_TC and TIMn_MR0 match 0: TIMn_TC does not stop counting 1: TIMn_TC stops counting, and the counter is disabled (CEn bit in the TIMn_TCR register is cleared to '0')
1	MR0RST	R/W	0x0	Bit [1] MR0R: Counter reset control bit when TIMn_TC and TIMn_MR0 match 0: No reset generated 1: Reset generated
0	MR0INT	R/W	0x0	Bit [0] MR0I: Interrupt control bit when TIMn_TC and TIMn_MR0 match 0: No interrupt generated 1: Interrupt generated

11.5.7. TIMn Match Value Register 0(TIMn_MR0(n = 0, 1, LPTIM))

Bit	Name	R/W	Reset	Description
31:16	-	R	0	Reserved
15:0	MATCH	R/W	0	Bits [15:0] MATCH: Timer compare value The value range is 0x0000 to 0xFFFF.

12. Advanced Timer and General Purpose Timer(EPWM, TIM2)

12.1 EPWM Overview

The Advanced Timer (EPWM) consists of a 20-bit auto-reload counter, which is driven by a programmable prescaler. It is suitable for various purposes, including measuring the pulse width of input signals (Input Capture), or generating output waveforms (Output Compare, PWM, complementary PWM with embedded dead time, etc.).

By using the timer prescaler and the RCC clock control prescaler, it is possible to adjust the pulse width and waveform period from a few microseconds to several milliseconds.

12.2 EPWM Characteristics

The functions of the EPWM timer include:

- 20-bit up, down, or up/down auto-reload counter
- 6-bit programmable (modifiable in real-time) prescaler, with the counter clock frequency division factor ranging from 1 to 65535
- Up to 4 independent channels:
  - Input Capture
  - Output Compare
  - PWM Generation (edge-aligned or center-aligned modes)
  - One-pulse mode output
- Complementary outputs with programmable dead time
- Synchronization circuits for controlling the timer and inter-timer communication using external signals

- Repetition counter that allows updating timer registers after a specified number of counter periods
- Brake input signal that can place the timer output signal in a reset state or a known state
- Interrupt generation upon the occurrence of the following events
  - Update: Counter overflow/underflow, counter initialization (via software or internal/external trigger)
  - Trigger events (counter start, stop, initialization, or counting triggered by internal/external sources)
  - Input Capture
  - Output Compare
  - Brake signal input:
    (1) External BKIN level signal (high level or low level)
    (2) Output of Analog Comparator 0, 1 (output high or output low)
    (3) ADC threshold comparison conversion result event occurs (two channels selectable)
    (4) Low voltage detection below selected threshold.
    (5) WDG overflow event occurs
    (6) CPU exception event occurs
- Supports incremental (quadrature) encoder for positioning
- Trigger input as an external clock or current management based on period

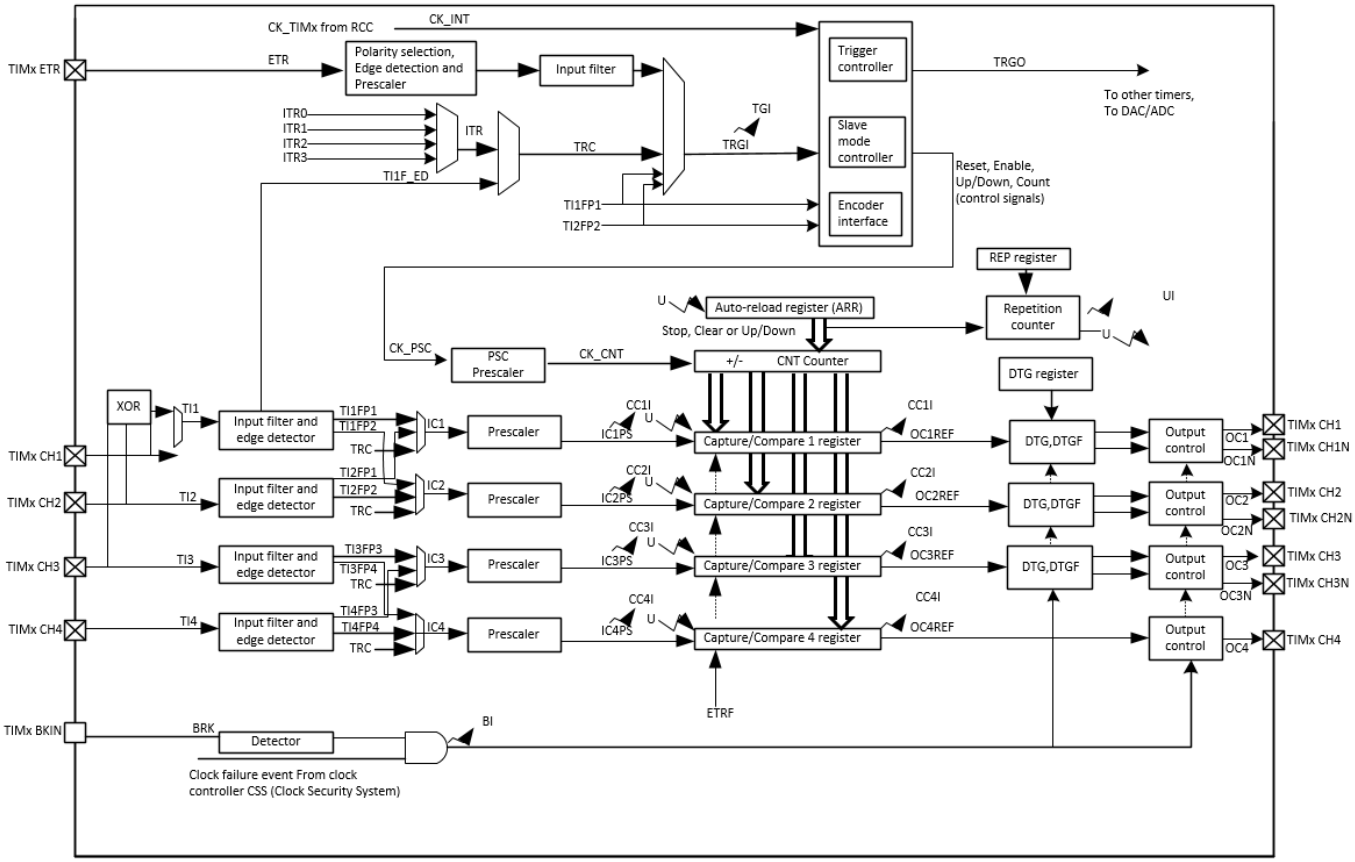


Fig 12-1 Advanced Timer block diagram

Note: Depending on the control bit setting, the contents of the preload register are transferred to the active (or shadow) register during an Update (U) event.

Event

Interrupt

12.3 EPWM Function Description

12.3.1 Time-Base Unit

The main part of the programmable advanced control timer is a 20-bit counter and its associated auto-reload register. This counter can perform up-counting, down-counting, or up/down bidirectional counting. The counter clock is obtained by dividing the clock source via the prescaler.

The counter, auto-reload register, and prescaler register can be read and written by software, even while the counter is running.

The Time-Base Unit includes:

- Counter Register (EPWM_CNT)
- Prescaler Register (EPWM_PSC)
- Auto-Reload Register (EPWM_ARR)
- Repetition Counter Register (EPWM_RCR)

The auto-reload register is preloaded. Writing or reading the auto-reload register accesses the preload register. Depending on the setting of the Auto-Reload Preload Enable bit (ARPE) in the EPWM_CR1 register, the contents of the preload register are transferred to the shadow register either immediately or at every Update Event (UEV). An Update Event is generated when the counter reaches the overflow condition (or the underflow condition during down-counting) and the UDIS bit in the EPWM_CR1 register is equal to '0'. An Update Event can also be generated by software. The generation of the Update Event under each configuration will be described in detail later.

The counter is driven by the clock output of the prescaler, CK_CNT. CK_CNT is only active when the Counter Enable bit (CEN) in the EPWM_CR1 register is set. (See the controller's slave mode description for more details on enabling the counter.)

Note: that the counter begins counting one clock cycle after the CEN bit in the EPWM_CR1 register is set.

The prescaler decription

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a 16-bit register (in the EPWM_PSC register) controlling the 20-bit counter. Since this control register is buffered, it can be changed at runtime. The new prescaler parameters are adopted at the next update event

Figure 12-2 and Figure 12-3 show examples of changing counter parameters while the prescaler is running.

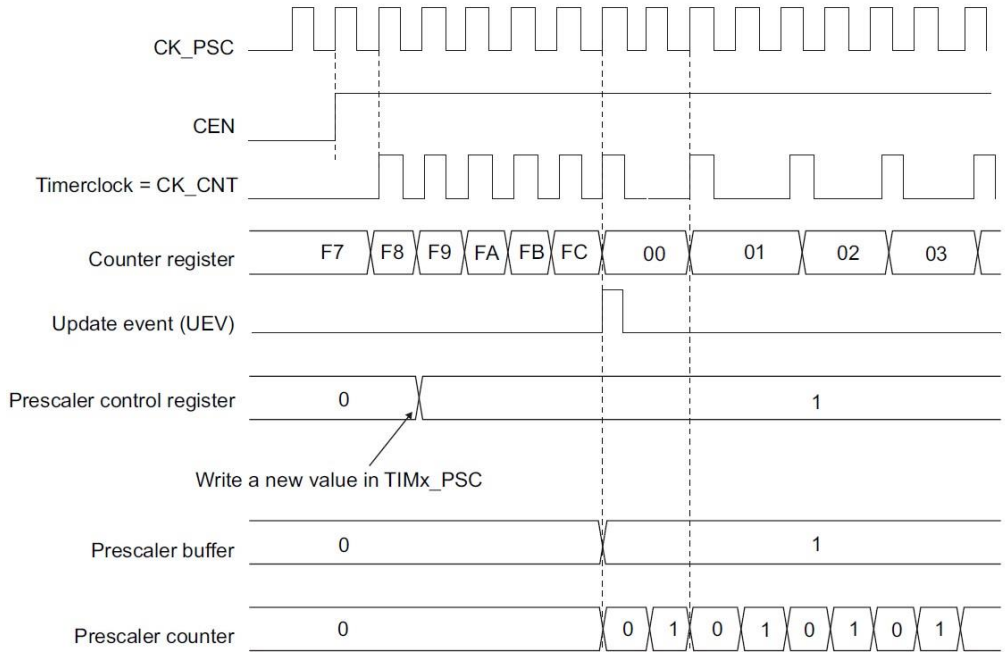


Fig 12-2 Counter timing diagram when the prescaler parameter changes from 1 to 2

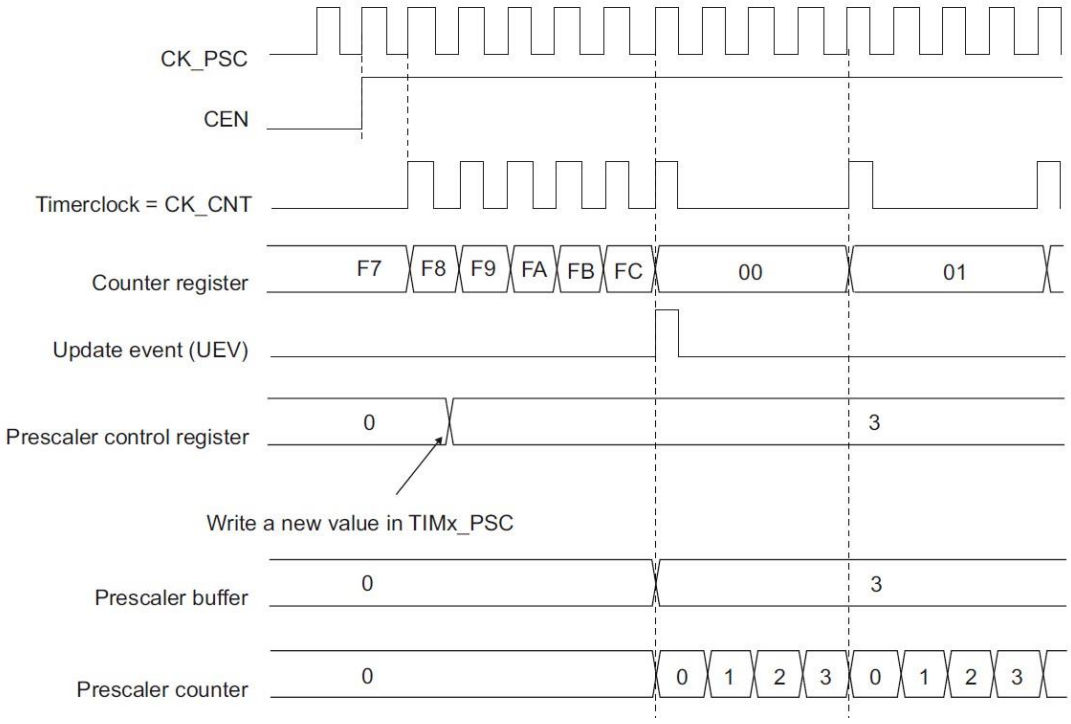


Fig 12-3 Counter timing diagram when the prescaler parameter changes from 1 to 4

12.3.2 Counter modes

12.3.2.1 Up-counting mode

In up-counting mode, the counter counts from 0 to the auto-reload value (contents of the EPWM_ARR counter), then restarts from 0 and generates a counter overflow event.

If the repetition counter function is used, an update event (UEV) is generated when the up-counting reaches the set repetition count number (EPWM_RCR); otherwise, an update event is generated every time the counter overflows.

Setting the UG bit in the EPWM_EGR register (by software or using the slave mode controller) can also generate an update event.

Setting the UDIS bit in the EPWM_CR1 register can disable update events; this avoids updating the shadow registers when writing new values to the preload registers. No update events will be generated until the UDIS bit is cleared to '0'. However, when an update event should have occurred, the counter is still cleared to '0', and the prescaler counter is also cleared to 0 (but the prescaler value remains unchanged). Furthermore, if the URS bit (update request selection) in the EPWM_CR1 register is set, setting the UG bit will generate an update event UEV, but the hardware does not set the UIF flag (i.e., no interrupt is generated). This is to avoid generating both update and capture interrupts simultaneously when clearing the counter in capture mode.

When an update event occurs, all registers are updated, and the hardware simultaneously (depending on the URS bit) sets the update flag bit (the UIF bit in the EPWM_SR register).

- The repetition counter is reloaded with the contents of the EPWM_RCR register.
- The auto-reload shadow register is updated with the value of the preload register (EPWM_ARR).
- The buffer of the prescaler is updated with the value of the preload register (contents of the EPWM_PSC register).

The following figures show some examples of the counter behavior at different clock frequencies when EPWM_ARR=0x36.

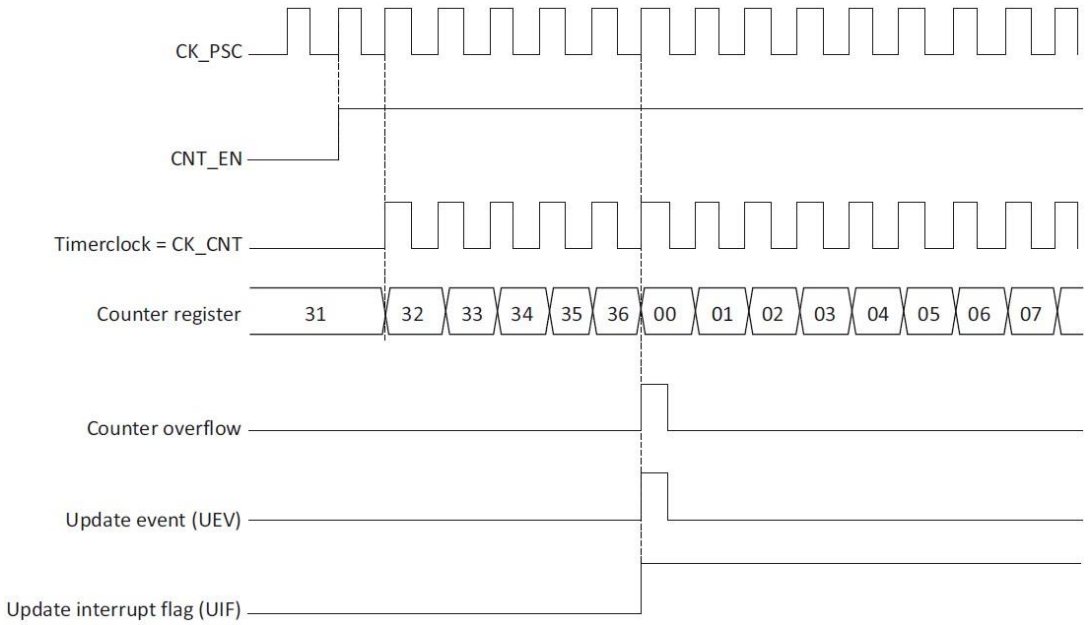


Fig 12-4 Counter timing diagram with internal clock division factor of 1

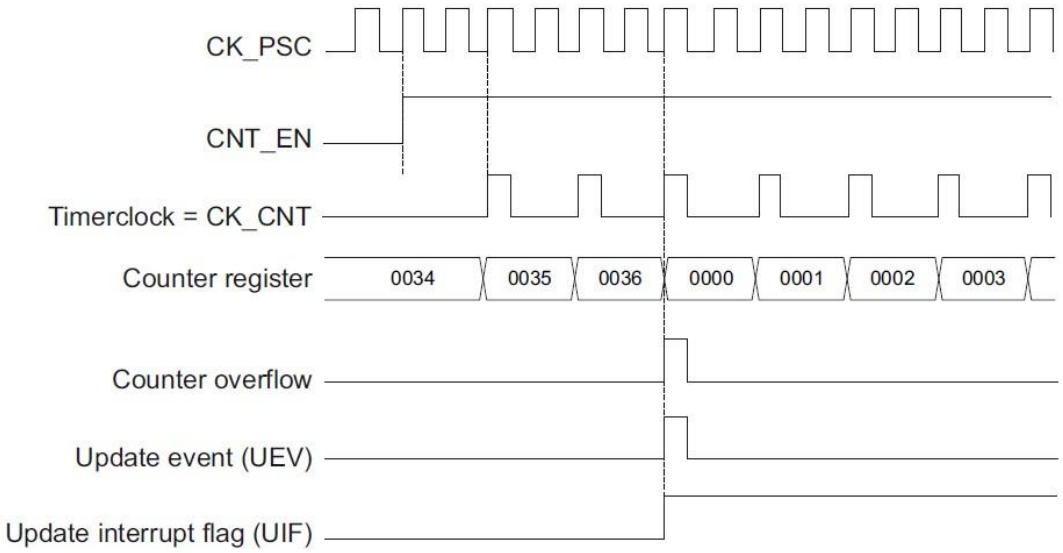


Fig 12-5 Counter timing diagram with internal clock division factor of 2

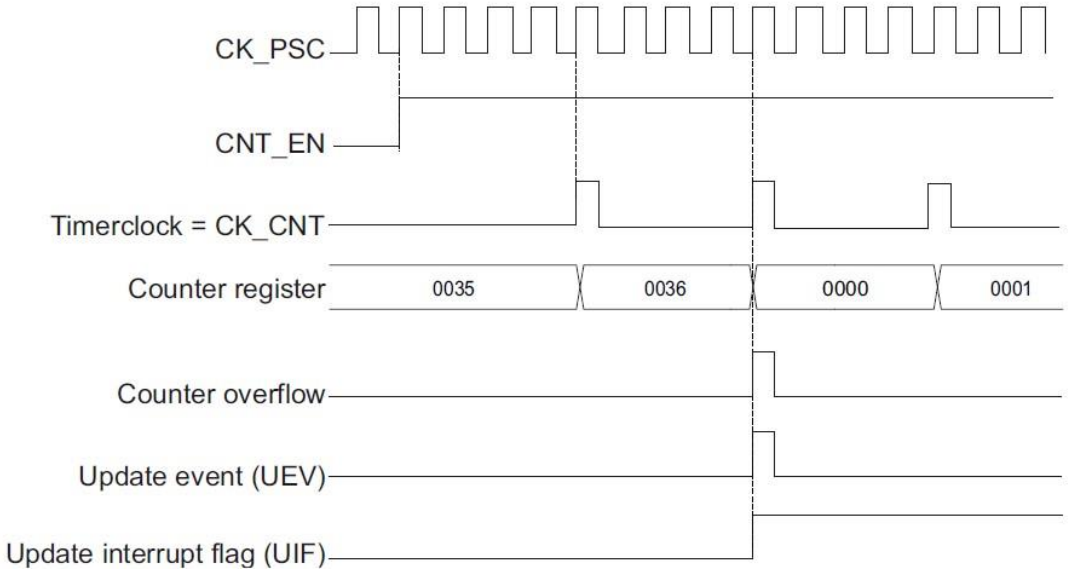


Fig 12-6 Counter timing diagram with internal clock division factor of 4

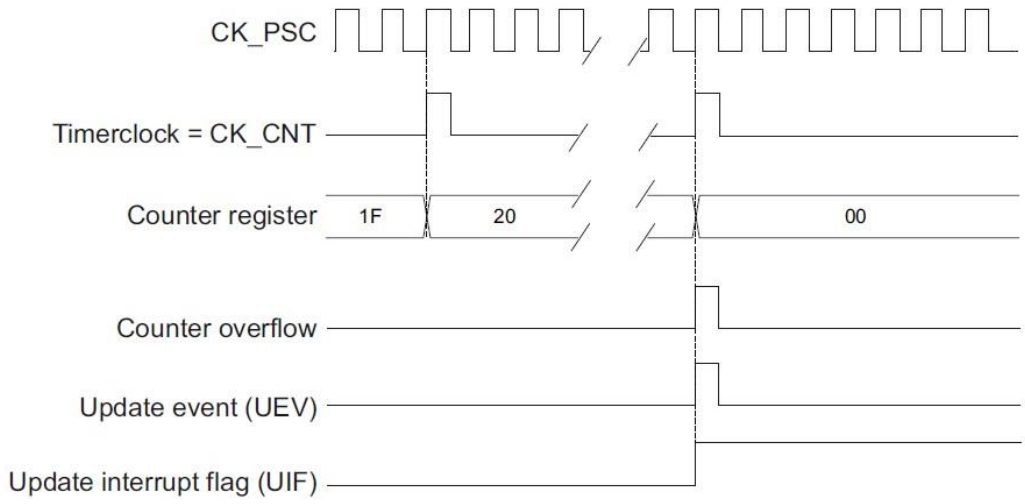


Fig 12-7 Counter timing diagram with internal clock division factor of N

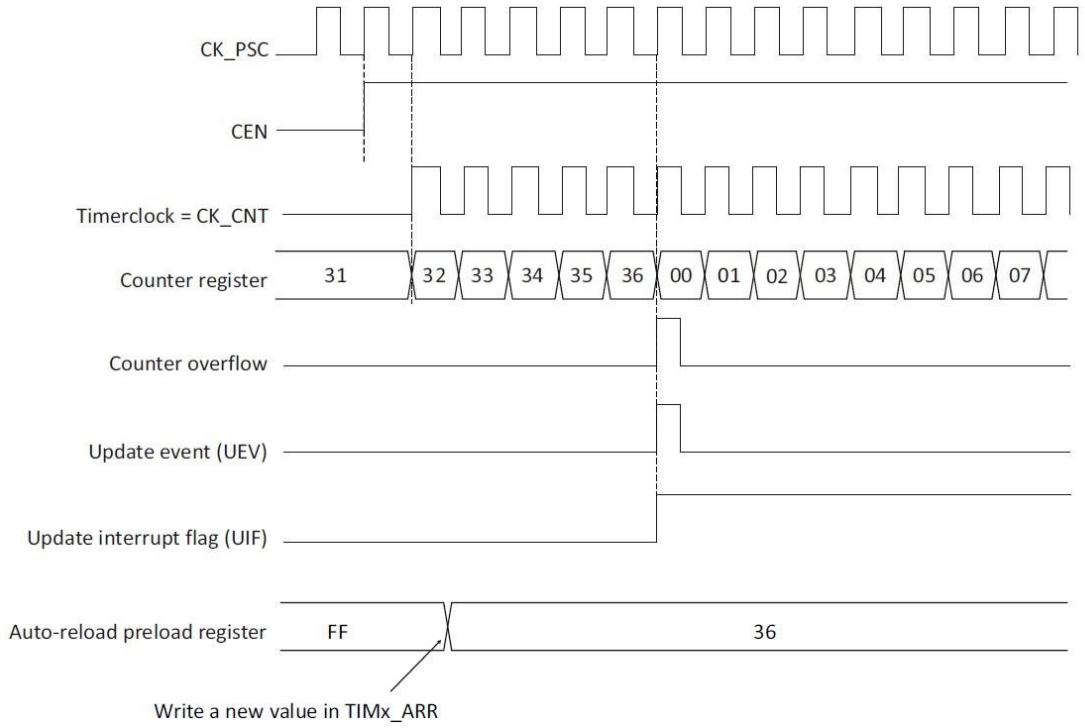


Fig 12-8 Counter timing diagram: Update event when ARPE=0 (EPWM_ARR not preloaded)

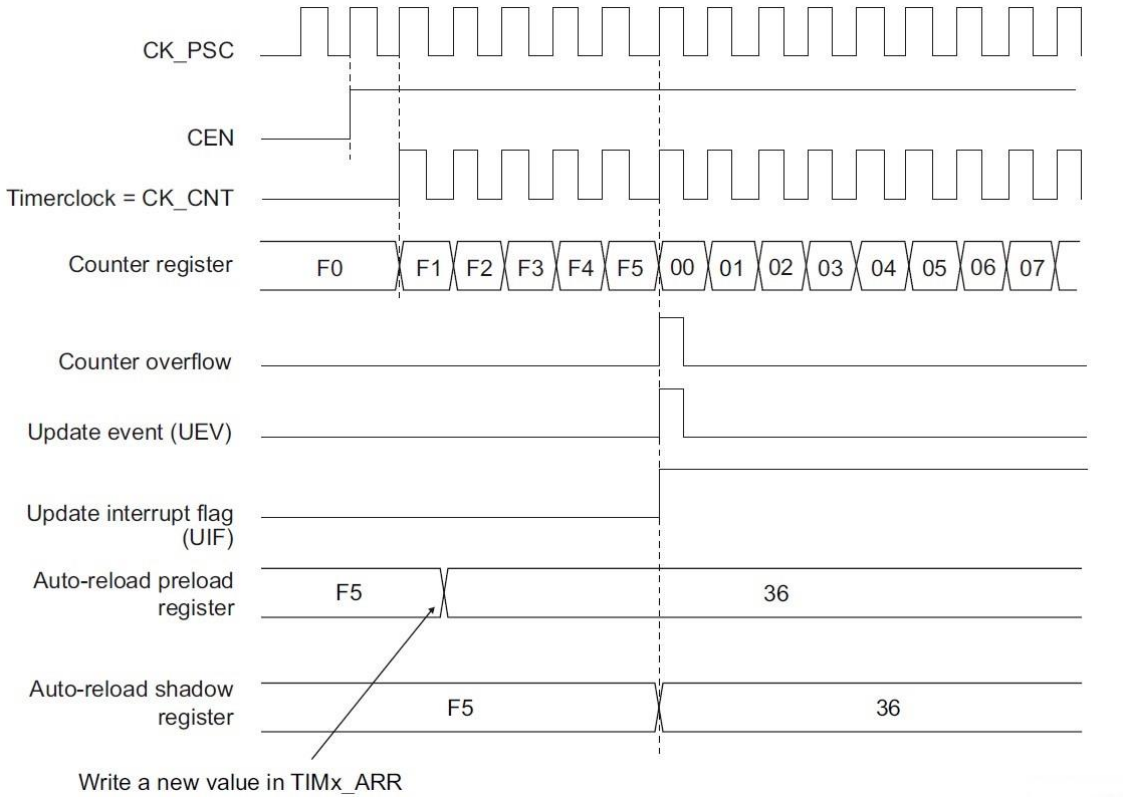


Fig 12-9 Counter timing diagram: Update event when ARPE=1 (EPWM_ARR preloaded)

12.3.2.2 Down-counting mode

In down-counting mode, the counter counts down from the auto-reload value (contents of the EPWM_ARR counter) to 0, and then restarts from the auto-reload value and generates a counter underflow event.

If the repetition counter function is used, an update event (UEV) is generated when the down-counting is repeated for the number of times set in the repetition counter register (EPWM_RCR), otherwise, an update event is generated every time the counter underflows.

Setting the UG bit in the EPWM_EGR register (by software or using the slave mode controller) can also generate an update event.

Setting the UDIS bit in the EPWM_CR1 register can disable the UEV event. This avoids updating the shadow registers when writing new values to the preload registers. Therefore, no update event is generated until the UDIS bit is cleared to 0. However, the counter will still restart counting from the current auto-reload value, and the prescaler counter will restart from 0 (but the prescaler division factor remains unchanged).

Furthermore, if the URS bit (update request selection) in the EPWM_CR1 register is set, setting the UG bit will generate an update event UEV but will not set the UIF flag (thus no interrupt is generated), this is to avoid generating both update and capture interrupts simultaneously when a capture event occurs and the counter is cleared.

When an update event occurs, all registers are updated, and the update flag bit (the UIF bit in the EPWM_SR register) is also set (depending on the URS bit setting).

- The repetition counter is reloaded with the contents of the EPWM_RCR register
- The prescaler's buffer is loaded with the preload value (contents of the EPWM_PSC register)
- The current auto-reload register is updated with the preload value
  (contents of the EPWM_ARR register)

Note: The auto-reload register is updated before the counter reloads, so the next period will be the expected value.

The following figures show some examples of the counter operation at different clock frequencies when EPWM_ARR=0x36.

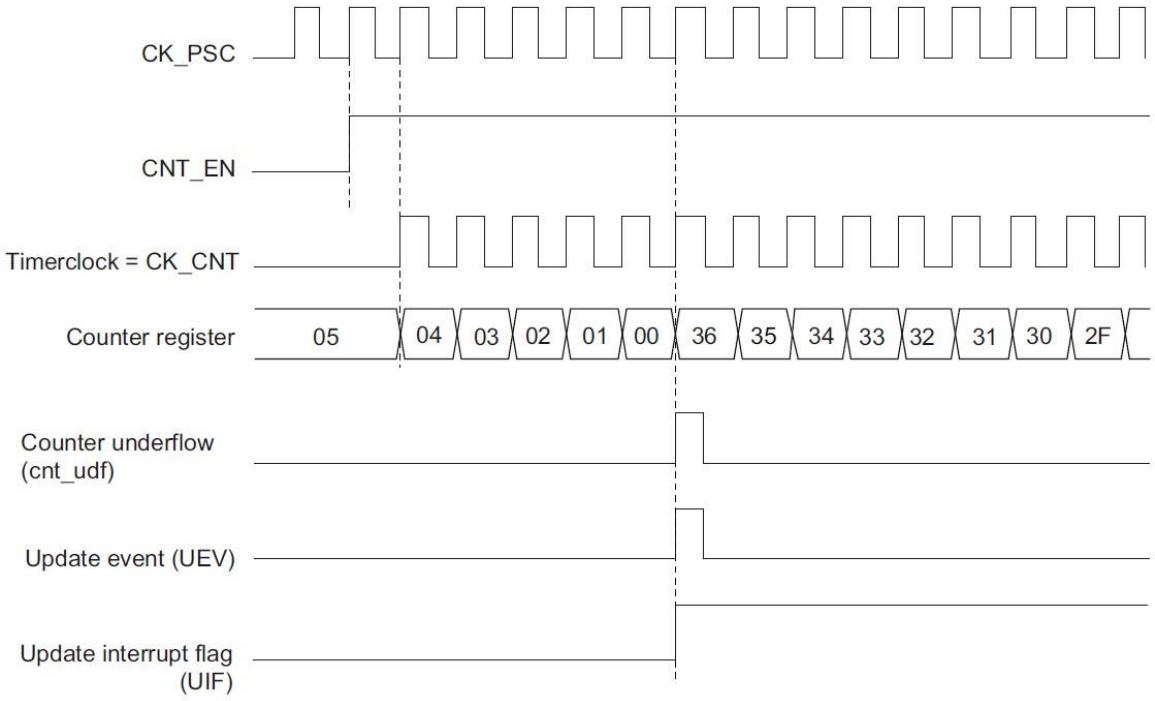


Fig 12-10 Counter timing diagram with internal clock division factor of 1

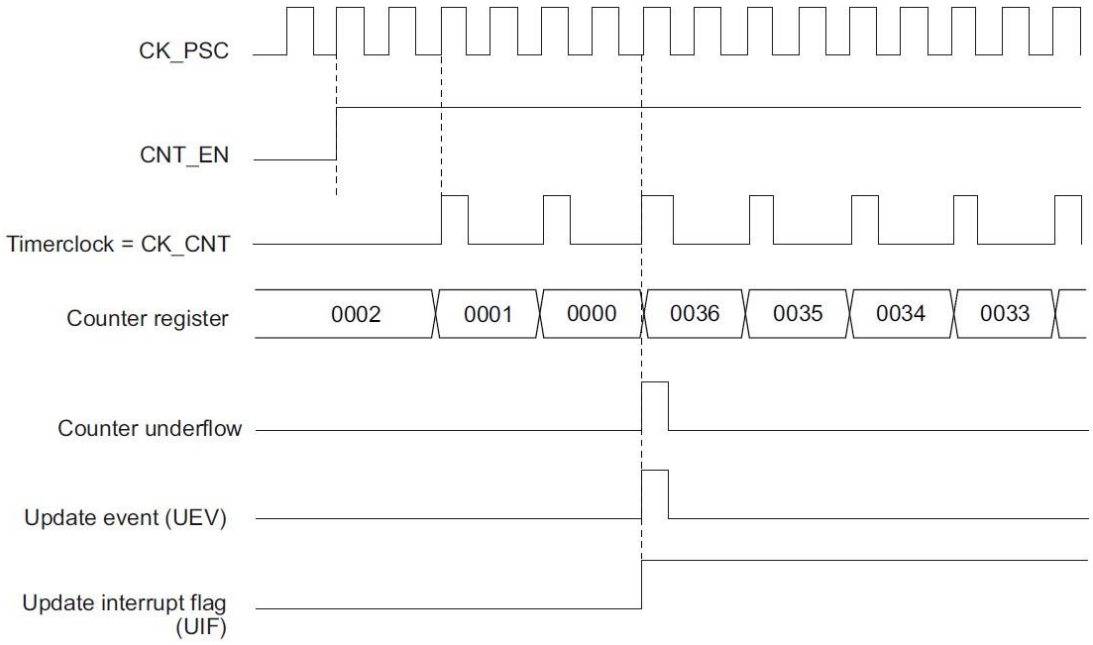


Fig 12-11 Counter timing diagram with internal clock division factor of 2

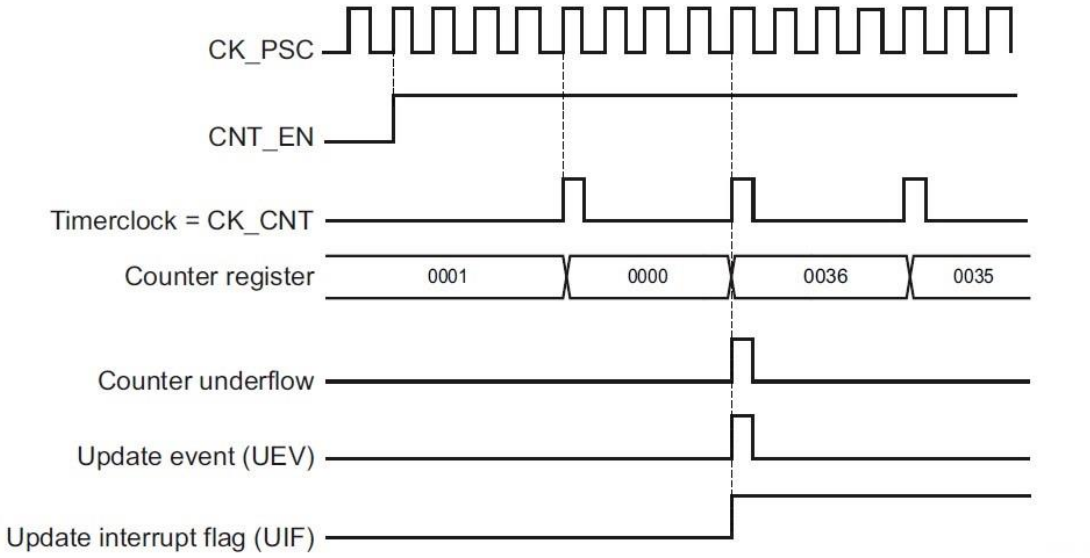


Fig 12-12 Counter timing diagram with internal clock division factor of 4

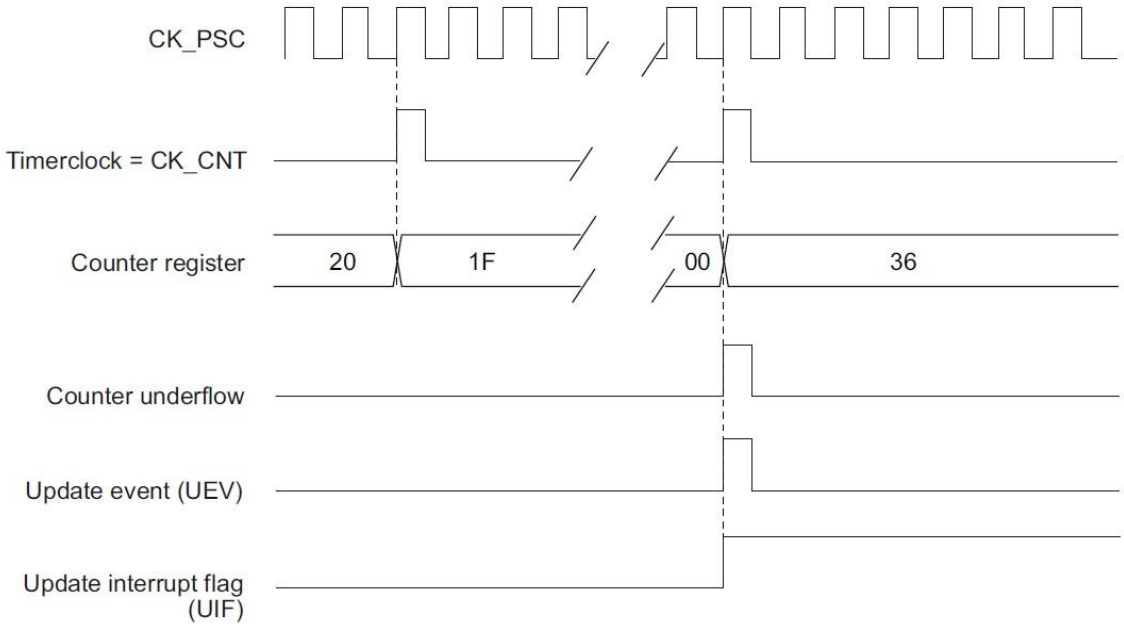


Fig 12-13 Counter timing diagram with internal clock division factor of N

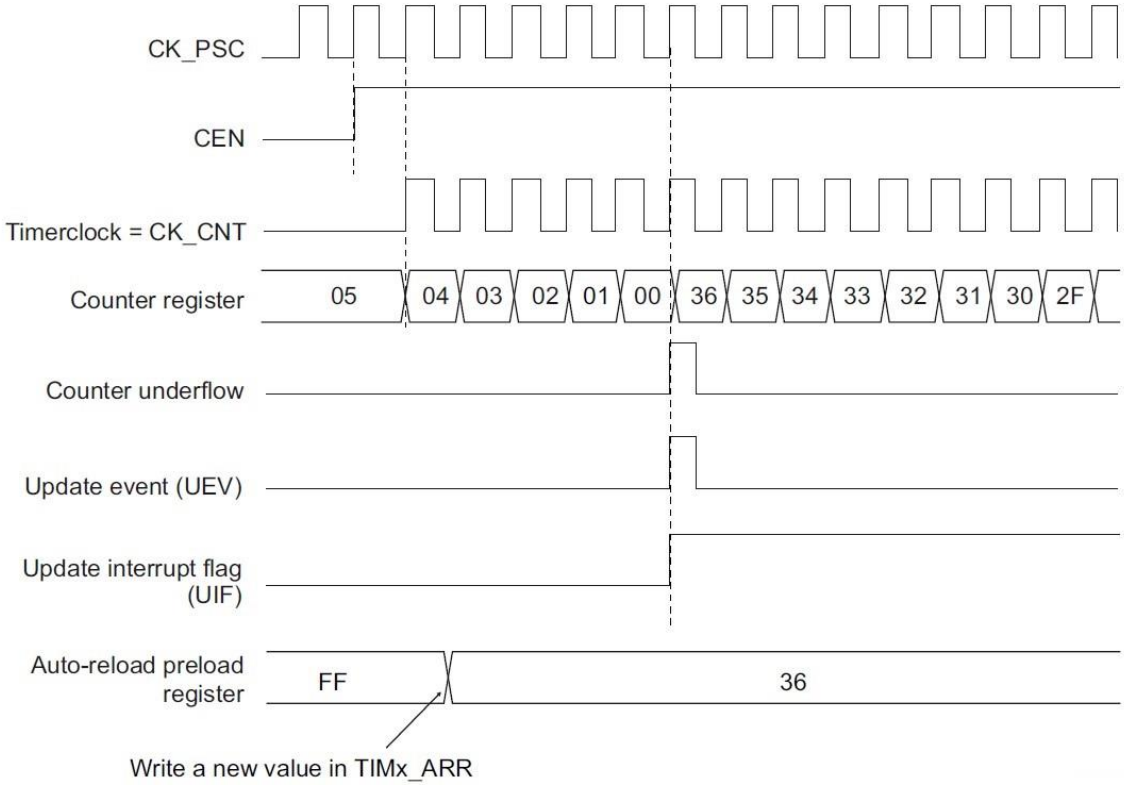


Fig 12-14 Counter timing diagram: Update event when the repetition counter is not used

12.3.2.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (EPWM_ARR register) - 1, generating a counter overflow event, then counts down to 1 and generates a counter underflow event; then it restarts counting from 0.

In this mode, the DIR direction bit in the EPWM_CR1 register cannot be written. It is updated by hardware and indicates the current counting direction. An update event can be generated at each counter overflow and each counter underflow; an update event can also be generated by setting the UG bit in the EPWM_EGR register (by software or using the slave mode controller). Then, the counter restarts counting from 0, and the prescaler also restarts counting from 0.

Setting the UDIS bit in the EPWM_CR1 register can disable the UEV event. This avoids updating the shadow registers when writing new values to the preload registers. Therefore, no update event is generated until the UDIS bit is cleared to 0. However, the counter continues counting up or down, according to the current auto-reload value.

Furthermore, if the URS bit (update request selection) in the EPWM_CR1 register is set, setting the UG bit will generate an update event UEV but will not set the UIF flag (thus no interrupt is generated), this is to avoid generating both update and capture interrupts simultaneously when a capture event occurs and the counter is cleared.

When an update event occurs, all registers are updated, and the update flag bit (the UIF bit in the EPWM_SR register) is also set (depending on the URS bit setting).

- The repetition counter is reloaded with the contents of the EPWM_RCR register
- The buffer of the prescaler is loaded with the preload value (contents of the EPWM_PSC register)
- The current auto-reload register is updated with the preload value (contents of the EPWM_ARR register)

Note: If the update is generated by a counter overflow, the auto-reload is updated before the counter reloads, so the next period will be the expected value (the counter is loaded with the new value).

The following are some examples of the counter operation at different clock frequencies:

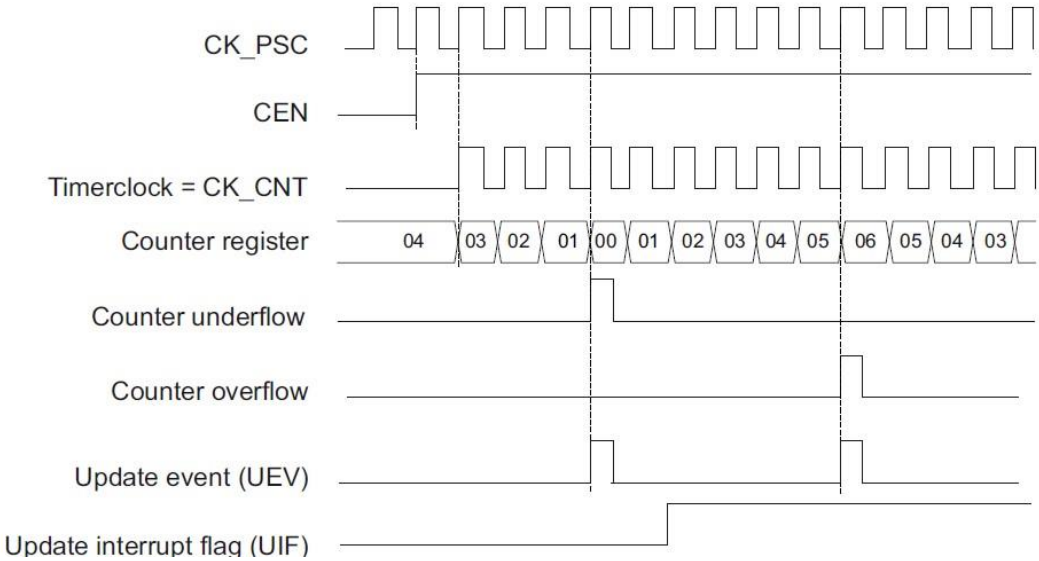


Fig 12-15 Counter timing diagram: Internal clock division factor of 1, EPWM_ARR=0x6

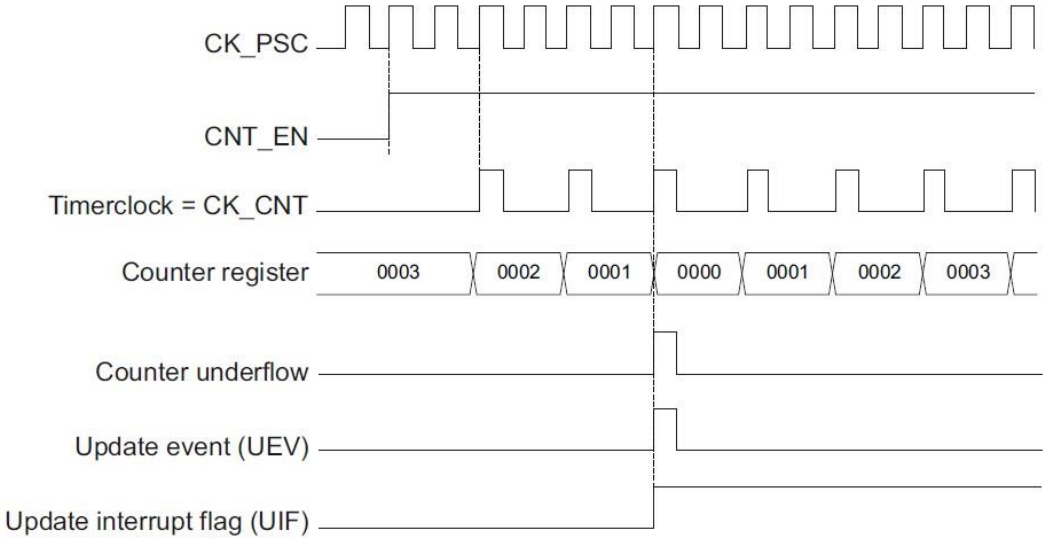


Fig 12-16 Counter timing diagram: Internal clock division factor of 2

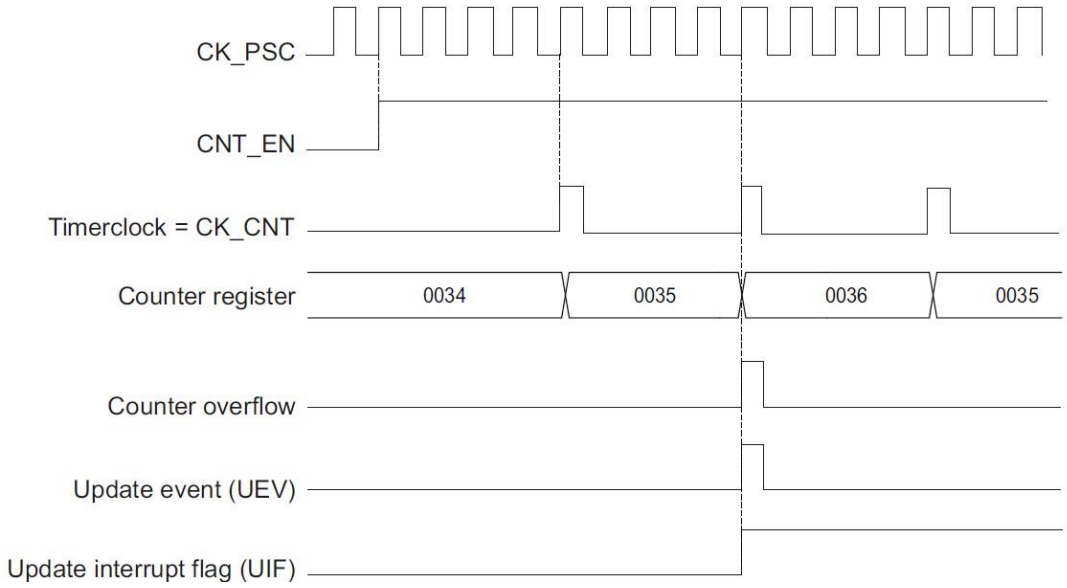


Fig 12-17 Counter timing diagram: Internal clock division factor of 4, EPWM_ARR=0x36

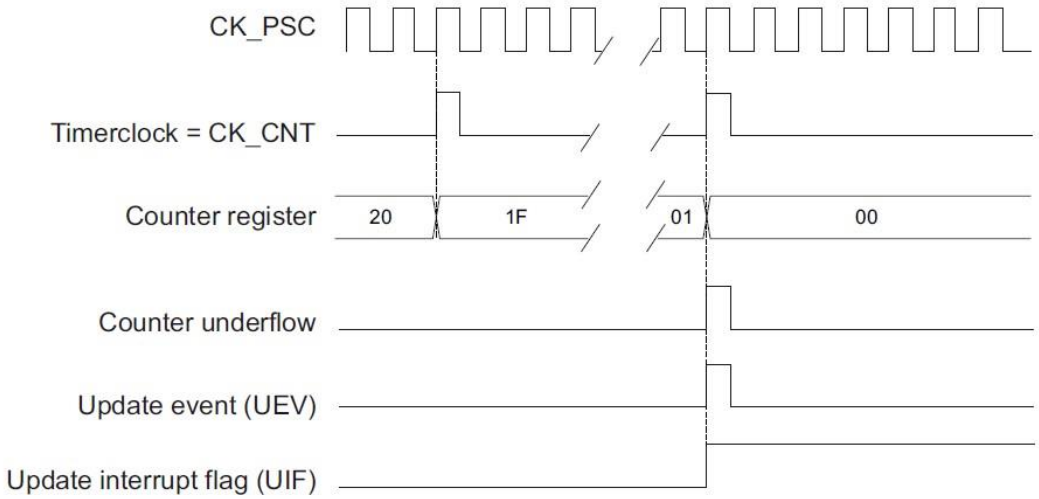


Fig 12-18 Counter timing diagram: Internal clock division factor of N

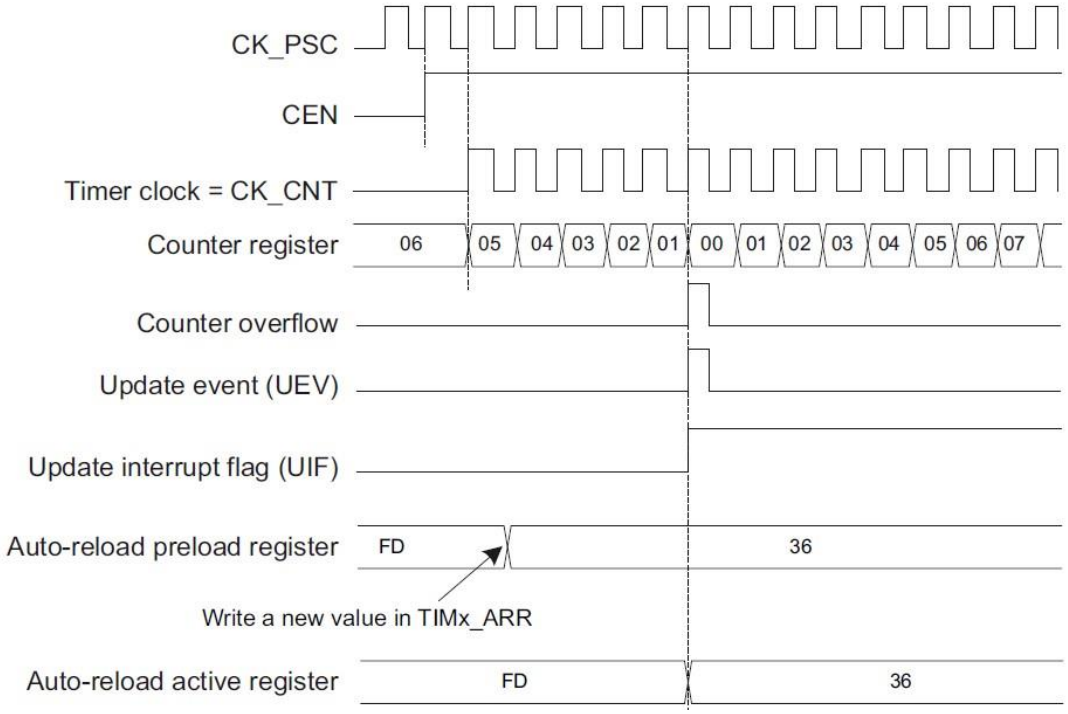


Fig 12-19 Counter timing diagram: Update event when ARPE=1 (Counter underflow)

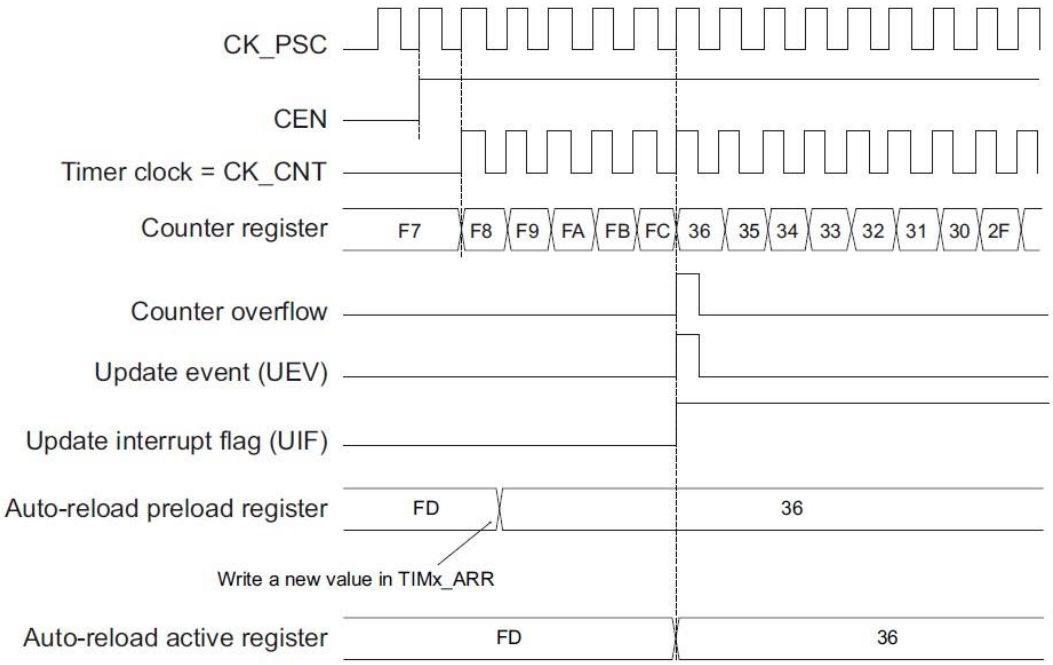


Fig 12-20 Counter timing diagram: Update event when ARPE=1 (Counter overflow)

12.3.3 Repeat counter

Section 12.3.1 Time-base unit explains how the update event (UEV) is generated when the counter overflows/underflows; however, in fact, it is only generated when the repetition counter reaches 0. This feature is very useful for generating PWM signals.

This means that every N counter overflows or underflows, data is transferred from the preload registers to the shadow registers (EPWM_ARR auto-reload register, EPWM_PSC prescaler register, and also the capture/compare registers EPWM_CCRx in compare mode), where N is the value in the EPWM_RCR repetition counter register.

The repetition counter decrements when any of the following conditions are met:

- Each time the counter overflows in up-counting mode
- Each time the counter underflows in down-counting mode
- Each time the counter overflows and underflows in center-aligned mode. Although this limits the maximum PWM repetition period to 128, it allows updating the duty cycle twice per PWM period. In center-aligned mode, because the waveform is symmetric, if the compare register is refreshed only once per PWM period, the maximum resolution is 2xTck.

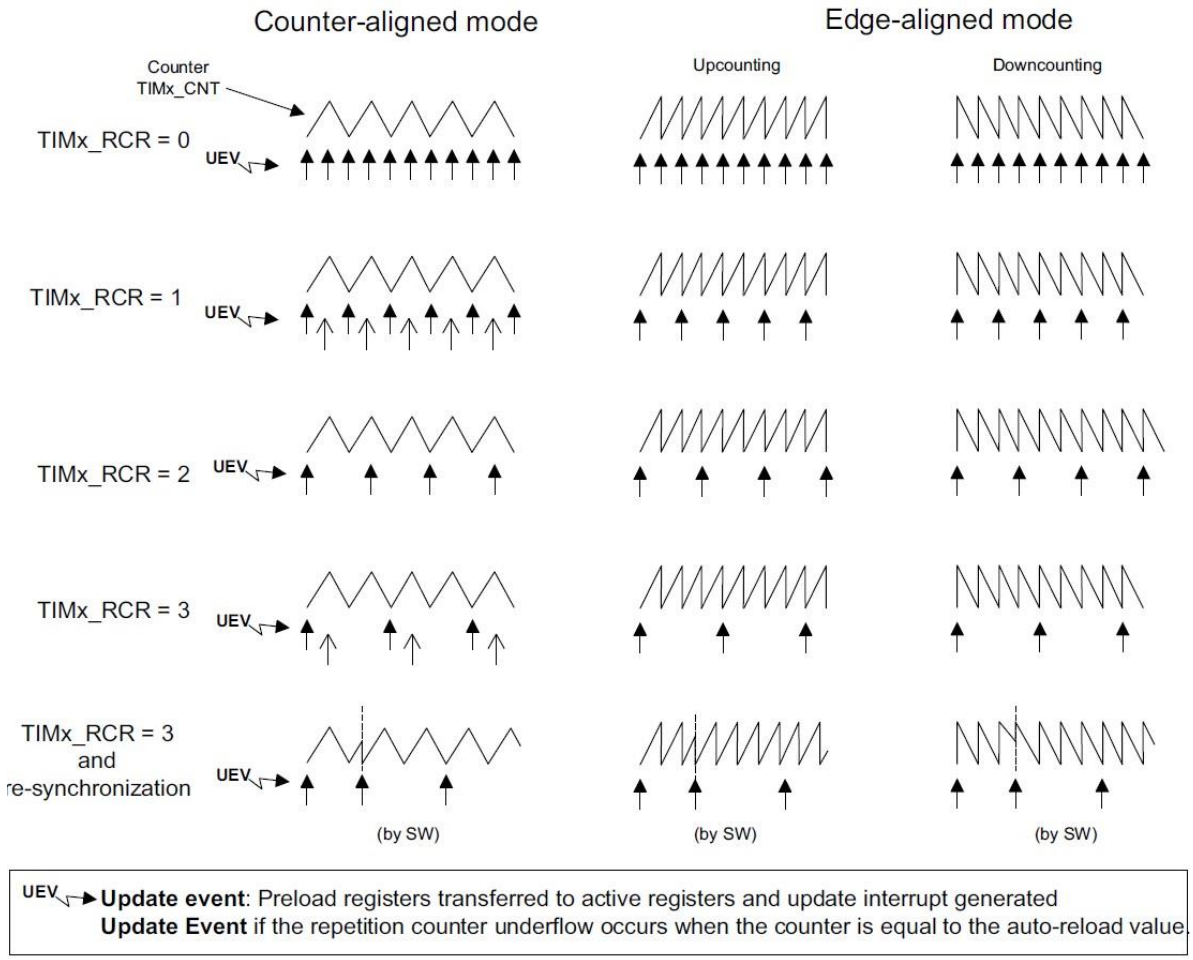


Fig 12-21 Examples of update rates in different modes and EPWM_RCR register settings

The repetition counter is auto-loaded, and the repetition rate is defined by the value of the EPWM_RCR register (see Figure 12-21). When an update event is generated by software (by setting the UG bit in EPWM_EGR) or by the hardware slave mode controller, the update event occurs immediately regardless of the value of the repetition counter, and the contents of the EPWM_RCR register are reloaded into the repetition counter.

12.3.4 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: External input pin
- External clock mode 2: External trigger input ETR(EPETR)
- Internal trigger input (ITRx): Using one timer as the prescaler for another timer. For example, timer EPWM can be configured as the prescaler for another timer TIM2, see Section 13.3.15.1 for details.
- If the slave mode controller is disabled (SMS=000), then the CEN, DIR (in the EPWM_CR1 register), and UG bits (in the EPWM_EGR register) are the actual control bits and can only be modified by software (the UG bit is still automatically cleared). As long as the CEN bit is written to '1', the prescaler clock is provided by the internal clock CK_INT.

The figure below shows the operation of the control circuit and the up-counter in normal mode, without a prescaler.

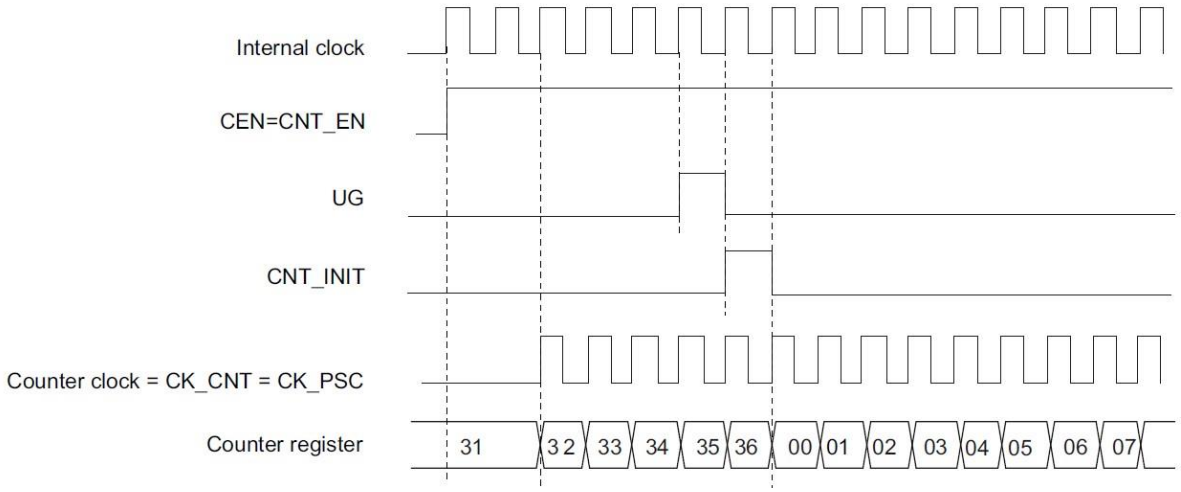


Fig 12-22 Control circuit in normal mode, internal clock division factor of 1

12.3.4.1 External clock source mode 1

This mode is selected when SMS=111 in the EPWM_SMCR register. The counter can count at each rising or falling edge of the selected input.

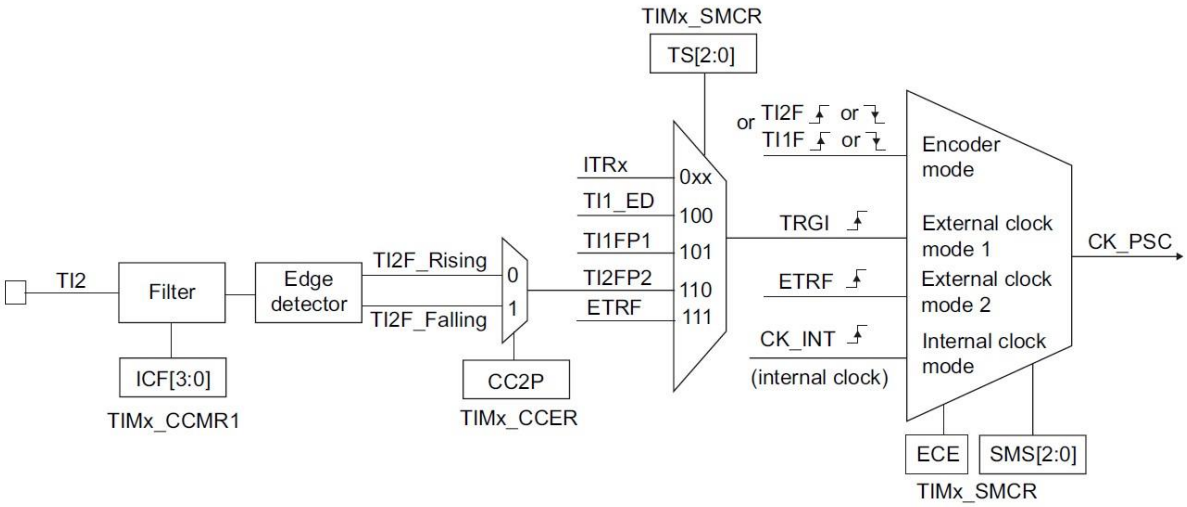


Fig 12-23 Example of TI2 external clock connection

For example, to configure the up-counter to count on the rising edge of the TI2 input, use the following steps:

1. Configure CC2S=01 in the EPWM_CCMR1 register to configure channel 2 to detect the rising edge of the TI2 input.
2. Configure IC2F[3:0] in the EPWM_CCMR1 register to select the input filter bandwidth (if no filter is needed, keep IC2F=0000).
3. Configure CC2P=0 in the EPWM_CCER register to select rising edge polarity.
4. Configure SMS=111 in the EPWM_SMCR register to select timer external clock mode 1.
5. Configure TS=110 in the EPWM_SMCR register to select TI2 as the trigger input source.
6. Set CEN=1 in the EPWM_CR1 register to enable the counter.

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

When a rising edge occurs on TI2, the counter counts once, and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter depends on the resynchronization circuit at the TI2 input.

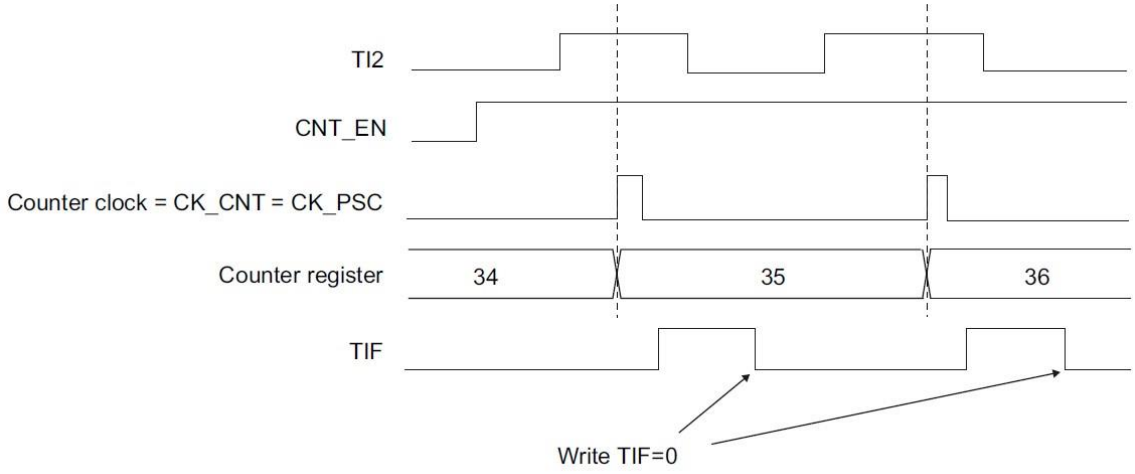


Fig 12-24 Control circuit in external clock mode 1

12.3.4.2 External clock source mode 2

The method to select this mode is: set ECE=1 in the EPWM_SMCR register. The counter can count at each rising or falling edge of the external trigger ETR. The figure below is the block diagram of the external trigger input.

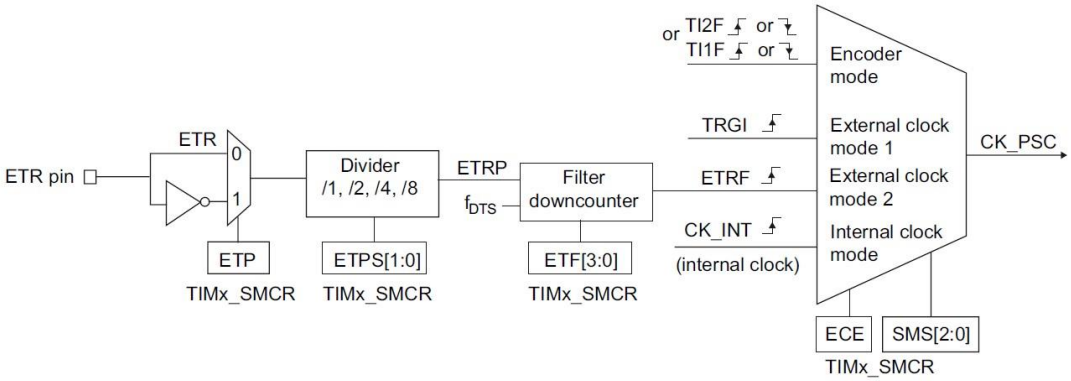


Fig 12-25 External trigger input block diagram

For example, to configure the up-counter to count once every 2 rising edges on ETR, use the following steps:

1. No filter is needed in this example, set ETF[3:0]=0000 in the EPWM_SMCR register.
2. Set the prescaler, set ETPS[1:0]=01 in the EPWM_SMCR register.
3. Select rising edge detection for ETR, set ETP=0 in the EPWM_SMCR register.
4. Enable external clock mode 2, write ECE=1 in the EPWM_SMCR register.
5. Enable the counter, write CEN=1 in the EPWM_CR1 register.

The counter counts once every 2 ETR rising edges. The delay between the rising edge on ETR and the actual clock of the counter depends on the resynchronization circuit at the ETRP signal input.

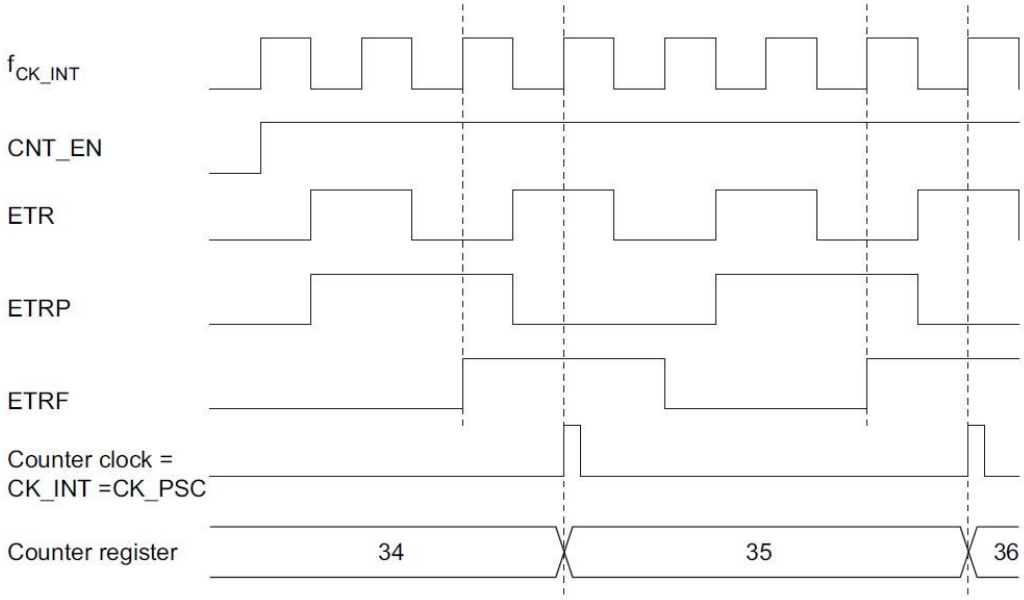


Fig 12-26 Control circuit in external clock mode 2

12.3.5 Capture/compare channels

Each capture/compare channel is built around a capture/compare register (including a shadow register), including a capture input section (digital filtering, multiplexing, and prescaler), and an output section (comparator and output control). The input capture channel is configured by SYSCFG_EPWM_CON_SEL.

Figures 12-27 to 12-30 give an overview of a capture/compare channel.

The input stage samples the corresponding TIx input signal and generates a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx), which can be used as a trigger input for the slave mode controller or as a capture command. The signal passes through the prescaler into the capture register (ICxPS).

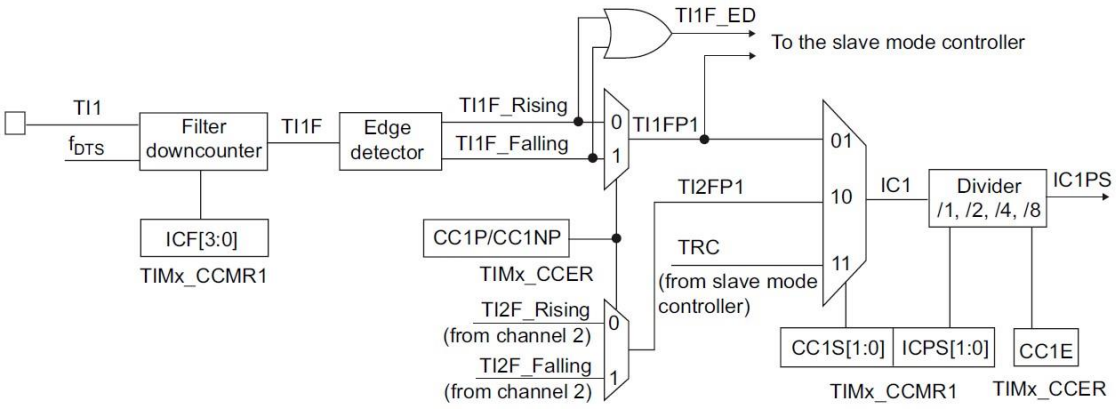


Fig 12-27 Capture/compare channel (e.g., Channel 1 input stage)

The output stage generates an intermediate waveform OCxRef (active high) as a reference, and the end of the chain determines the polarity of the final output signal.

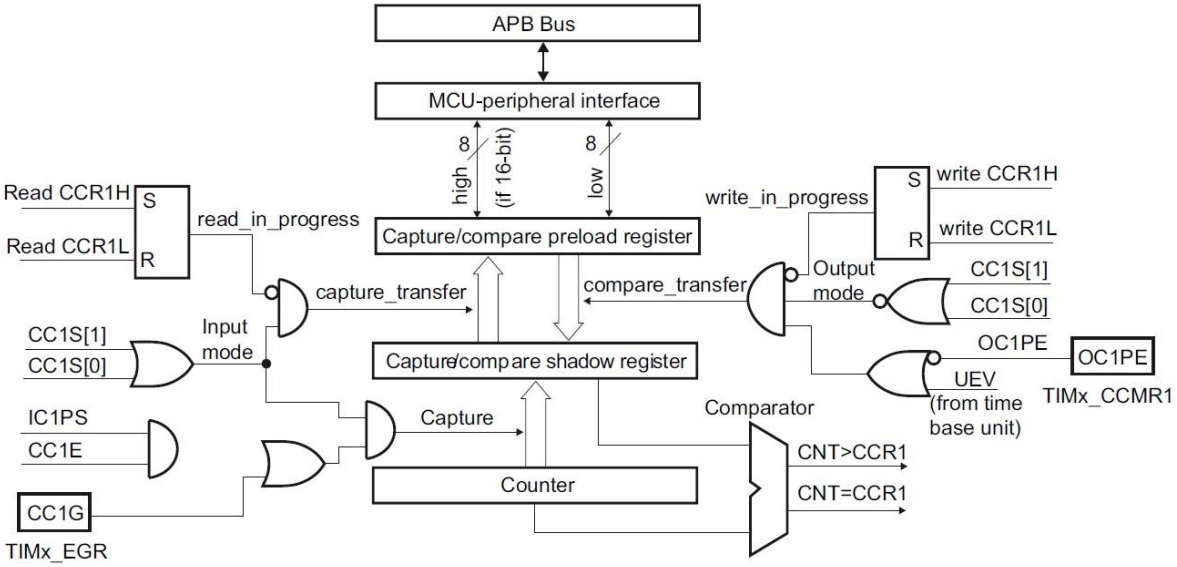


Fig 12-28 Capture/compare channel 1 main circuit

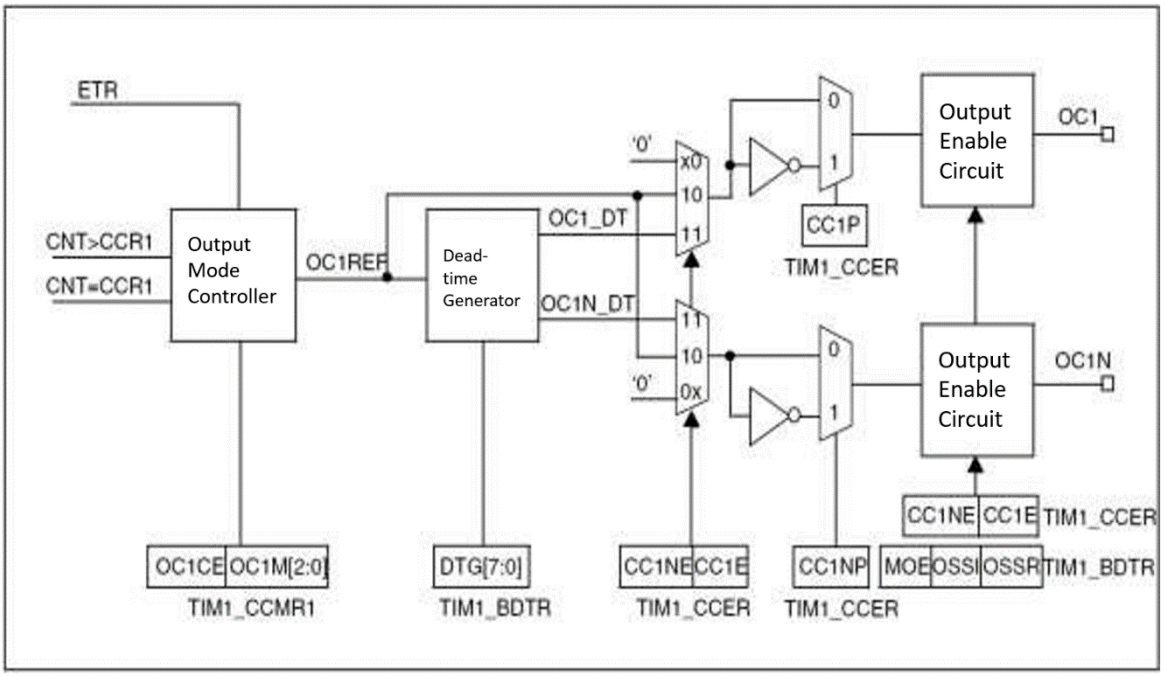


Fig 12-29 Output stage of capture/compare channel (Channels 1 to 3)

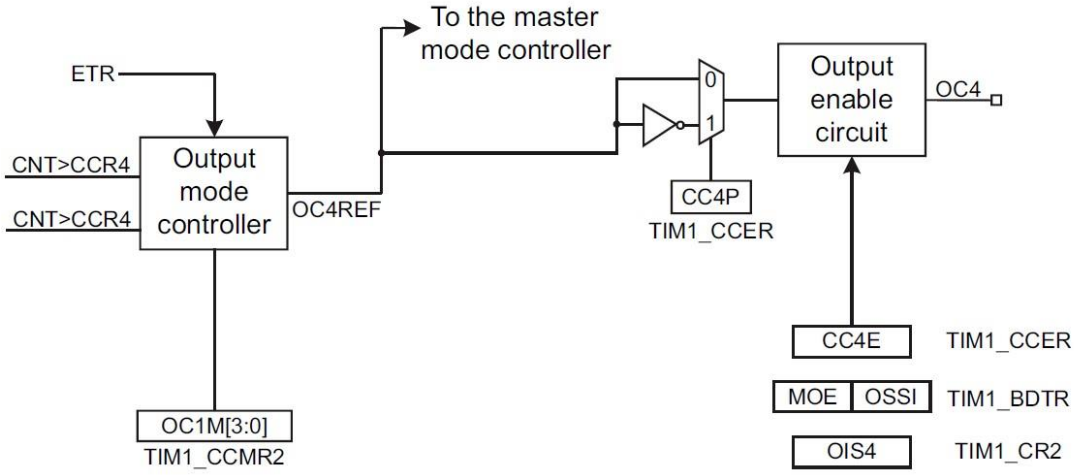


Fig 12-30 Output stage of capture/compare channel (Channel 4)

The capture/compare module consists of a preload register and a shadow register. The read and write operations access only the preload register. In capture mode, captures occur in the shadow register and are then copied into the preload register. In compare mode, the content of the preload register is copied into the shadow register, and then the content of the shadow register is compared with the counter.

12.3.6 Input capture mode

In input capture mode, the current value of the counter is latched into the capture/compare register (EPWM_CCRx) when a corresponding edge is detected on the ICx signal. When a capture event occurs, the corresponding CCxIF flag (EPWM_SR register) is set to 1, and an interrupt is generated if the interrupt is enabled. If the CCxIF flag is already high when the capture event occurs, the over-capture flag CCxOF (EPWM_SR register) is set to 1. Writing CCxIF=0 clears the CCxIF, or reading the captured data stored in the EPWM_CCRx register also clears the CCxIF. Writing CCxOF=0 clears the CCxOF.

The following example illustrates how to capture the counter value into the EPWM_CCR1 register on the rising edge of the TI1 input, the steps are as follows:

- Select the active input: EPWM_CCR1 must be connected to the TI1 input, so write CC1S=01 in the EPWM_CCMR1 register. As long as CC1S is not '00', the channel is configured as an input, and the EPWM_CCR1 register becomes read-only.
- Configure the input filter for the required bandwidth according to the characteristics of the input signal (i.e., when the input is TIx, the input filter control bits are the ICxF bits in the EPWM_CCMRx register). Assuming the input signal jitters for up to 5 internal clock cycles, we must configure the filter bandwidth to be longer than 5 clock cycles; therefore we can sample 8 times continuously (at fDTS frequency) to confirm a real edge transition on TI1, i.e., write IC1F=0011 in the EPWM_CCMR1 register.

- Select the active transition edge of the TI1 channel, write CC1P=0 (rising edge) in the EPWM_CCER register.
- Configure the input prescaler. In this example, we wish the capture to occur at every valid level transition, so the prescaler is disabled (write IC1PS=00 in the EPWM_CCMR1 register).
- Set CC1E=1 in the EPWM_CCER register to allow capturing the counter value into the capture register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the EPWM_IER register.

When an input capture occurs:

- The value of the counter is transferred to the EPWM_CCR1 register when a valid level transition occurs.
- The CC1IF flag is set (interrupt flag). If at least 2 consecutive captures occur and CC1IF has not been cleared, CC1OF is also set to 1
- If the CC1IE bit is set, an interrupt is generated.

To handle capture overflow, it is recommended to read the data before reading the capture overflow flag. This is to avoid missing capture overflow information that may be generated after reading the capture overflow flag and before reading the data.

Note: Setting the corresponding CCxG bit in the EPWM_EGR register can generate an input capture interrupt via software.

12.3.7 PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except for the following differences:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode. For example, you can measure the period (in the EPWM_CCR1 register) and the duty cycle (in the EPWM_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on the CK_INT frequency and prescaler value).
- Select the active input for EPWM_CCR2: write CC2S=10 in the EPWM_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1 (used for capturing data in EPWM_CCR1 and clearing the counter): write CC1P=0 (active on rising edge).
- Select the active input for EPWM_CCR2: write CC2S=10 in the EPWM_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2 (used for capturing data in EPWM_CCR2): write CC2P=1 (active on falling edge).
- Select the valid trigger input signal: write TS=101 in the EPWM_SMCR register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write SMS=100 in the EPWM_SMCR register.
- Enable capture: write CC1E=1 and CC2E=1 in the EPWM_CCER register.

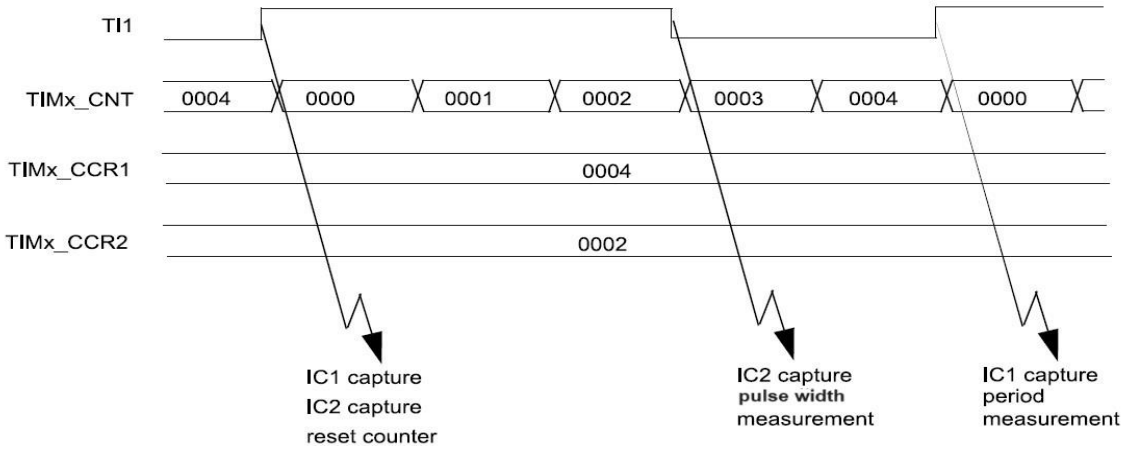


Fig 12-31 PWM input mode timing settings

Since only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only be used with the EPWM_CH1/EPWM_CH2 signals.

12.3.8 Forced output mode

In output mode (CCxS=00 in the EPWM_CCMRx register), the output compare signal (OCxREF and corresponding OCx/OCxN) can be directly forced to an active or inactive state by software, independently of the comparison result between the output compare register and the counter.

Setting OCxM=101 in the corresponding EPWM_CCMRx register forces the output compare signal (OCxREF/OCx) to the active state. Thus, OCxREF is forced high (OCxREF is always active high), and OCx receives a signal with opposite polarity to CCxP.

For example: CCxP=0 (OCx active high), then OCx is forced high.

Setting OCxM=100 in the EPWM_CCMRx register can force the OCxREF signal low.

In this mode, the comparison between the EPWM_CCRx shadow register and the counter is still performed, and the corresponding flags are modified. Therefore, corresponding interrupts will still be generated. This will be introduced in the output compare mode section below.

12.3.9 Output compare mode

This function is used to control an output waveform or indicate that a given period of time has elapsed. When the counter matches the content of the capture/compare register, the output compare function performs the following operations:

- Output the value defined by the output compare mode (OCxM bits in the EPWM_CCMRx register) and output polarity (CCxP bit in the EPWM_CCER register) to the corresponding pin. Upon compare match, the output pin can keep its level (OCxM=000), be set to the active level (OCxM=001), be set to the inactive level (OCxM=010), or toggle (OCxM=011).
- Set the flag bit in the interrupt status register (CCxIF bit in the EPWM_SR register).
- If the corresponding interrupt mask is set (CCxIE bit in the EPWM_IER register), an interrupt is generated.

The OCxPE bit in EPWM_CCMRx selects whether the EPWM_CCRx register uses the preload register. In output compare mode, the update event UEV has no effect on the OCxREF and OCx output. The synchronization precision can reach one counter clock cycle. Output compare mode (in one-pulse mode) can also be used to output a single pulse.

Configuration steps for output compare mode:

1. Select the counter clock (internal, external, prescaler).
2. Write the corresponding data to the EPWM_ARR and EPWM_CCRx registers.
3. If an interrupt request is required, set the CCxIE bit.
4. Select the output mode, for example:
   - Request to toggle the OCx output pin when the counter matches CCRx, set OCxM=011
   - Set OCxPE = 0 to disable the preload register
   - Set CCxP = 0 to select active high polarity
5. Set the CEN bit in the EPWM_CR1 register to start the counter.

   Set CCxE = 1 to enable the outputThe EPWM_CCRx register can be updated by software at any time to control the output waveform, provided that the preload register is not used (OCxPE='0', otherwise the shadow register of EPWM_CCRx can only be updated when the next update event occurs). The figure below gives an example.

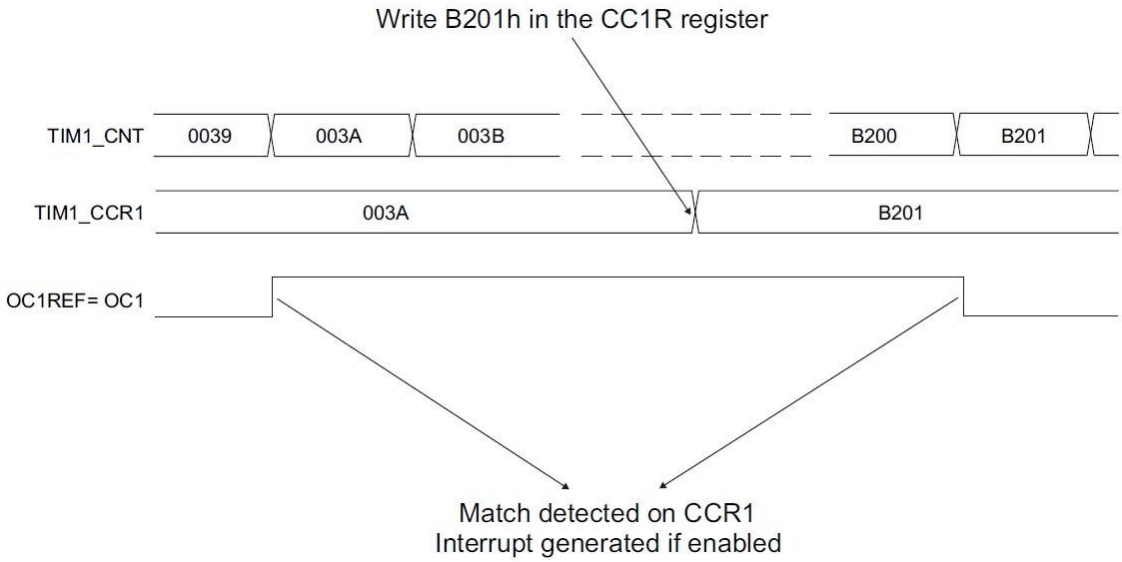


Fig 12-32 Output compare mode, toggle OC1

12.3.10 PWM output mode

Pulse width modulation mode can generate a signal with a frequency determined by the EPWM_ARR register and a duty cycle determined by the EPWM_CCRx register.

Writing '110' (PWM mode 1) or '111' (PWM mode 2) to the OCxM bits in the EPWM_CCMRx register can independently configure each OCx output channel to generate a PWM signal. The corresponding preload register must be enabled by setting the OCxPE bit in the EPWM_CCMRx register, and finally, the auto-reload preload register must be enabled by setting the ARPE bit in the EPWM_CR1 register (in up-counting or center-aligned mode).

The preload register is transferred to the shadow register only when an update event occurs; therefore, before the counter starts counting, all registers must be initialized by setting the UG bit in the EPWM_EGR register.

The polarity of OCx can be set by software using the CCxP bit in the EPWM_CCER register; it can be set to active high or active low. The OCx output enable is controlled by the combination of CCxE, CCxNE, MOE, OSSI, and OSSR bits (in the EPWM_CCER and EPWM_BDTR registers). See the description of the EPWM_CCER register for details.

In PWM mode (mode 1 or mode 2), EPWM_CNT and EPWM_CCRx are always compared to determine whether EPWM_CCRx <= EPWM_CNT or EPWM_CNT <= EPWM_CCRx is met (depending on the counting direction of the counter).

If the EPWM_CR1.ASYMEN bit is enabled to turn on the PWM output asymmetric mode, the EPWM_CNT down-counting will use EPWM_CCDRx as the comparison reference.

According to the state of the CMS bits in the EPWM_CR1 register, the timer can generate edge-aligned PWM signals or center-aligned PWM signals.

12.3.10.1 PWM edge-aligned mode

Up-counting configuration

Up-counting is performed when the DIR bit in the EPWM_CR1 register is low.

The following is an example of PWM mode 1. When EPWM_CNT < EPWM_CCRx, the PWM reference signal OCxREF is high, otherwise it is low. If the compare value in EPWM_CCRx is greater than the auto-reload value (EPWM_ARR), then OCxREF remains at '1'. If the compare value is 0, then OCxREF remains at '0'. The figure below shows an example of edge-aligned PWM waveforms when EPWM_ARR = 0x8.

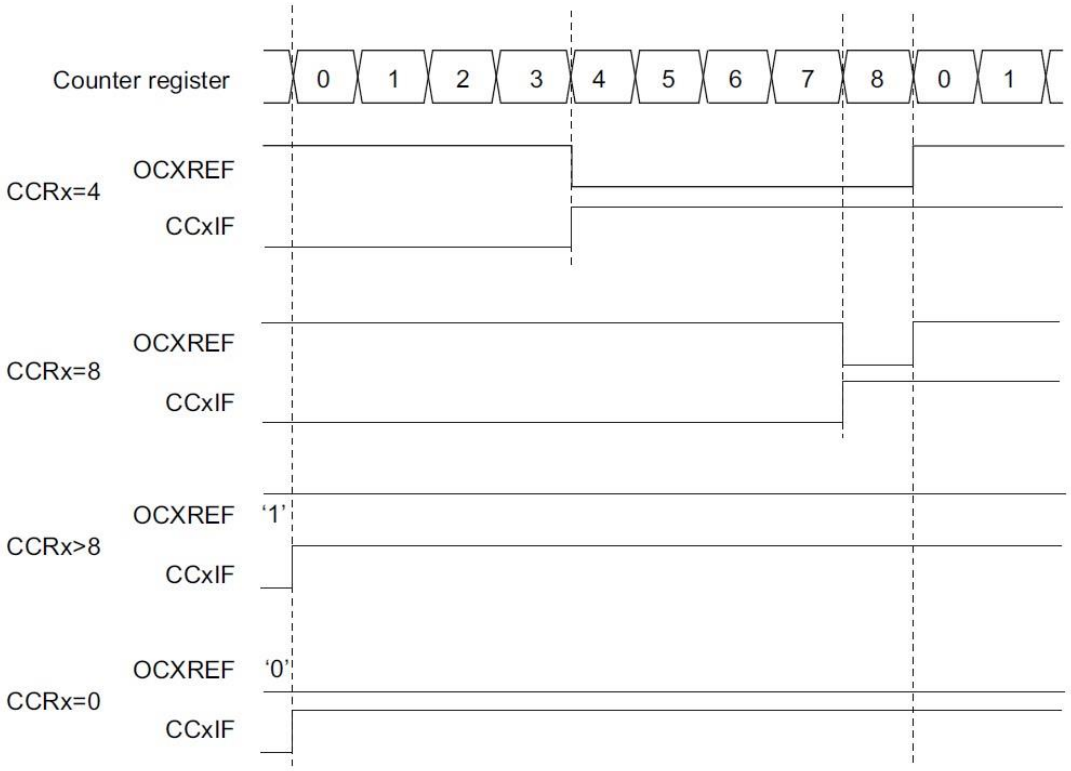


Fig 12-33 Edge-aligned PWM waveforms (ARR = 8)

Down-counting configuration

Down-counting is performed when the DIR bit in the EPWM_CR1 register is high. In PWM mode 1, the reference signal OCxREF is low when EPWM_CNT > EPWM_CCRx, otherwise it is high. If the compare value in EPWM_CCRx is greater than the auto-reload value in EPWM_ARR, then OCxREF remains at '1'. A 0% PWM waveform cannot be generated in this mode.

12.3.10.2 PWM center-aligned mode

Symmetric mode

Center-aligned mode is selected when the CMS bits in the EPWM_CR1 register are not '00' (all other configurations have the same effect on the OCxREF/OCx signals). Depending on the CMS bit settings, the compare flag can be set when the counter counts up, when the counter counts down, or when the counter counts both up and down. The count direction bit (DIR) in the EPWM_CR1 register is updated by hardware and should not be modified by software. Set ASYMEN=0 for symmetric PWM output.

The figure below shows some examples of center-aligned PWM waveforms:

- EPWM_ARR = 8
- PWM mode 1, ASYMEN=0, symmetric PWM output
- CMS=01 in the EPWM_CR1 register, in center-aligned mode 1, the compare flag is set when the counter counts down.

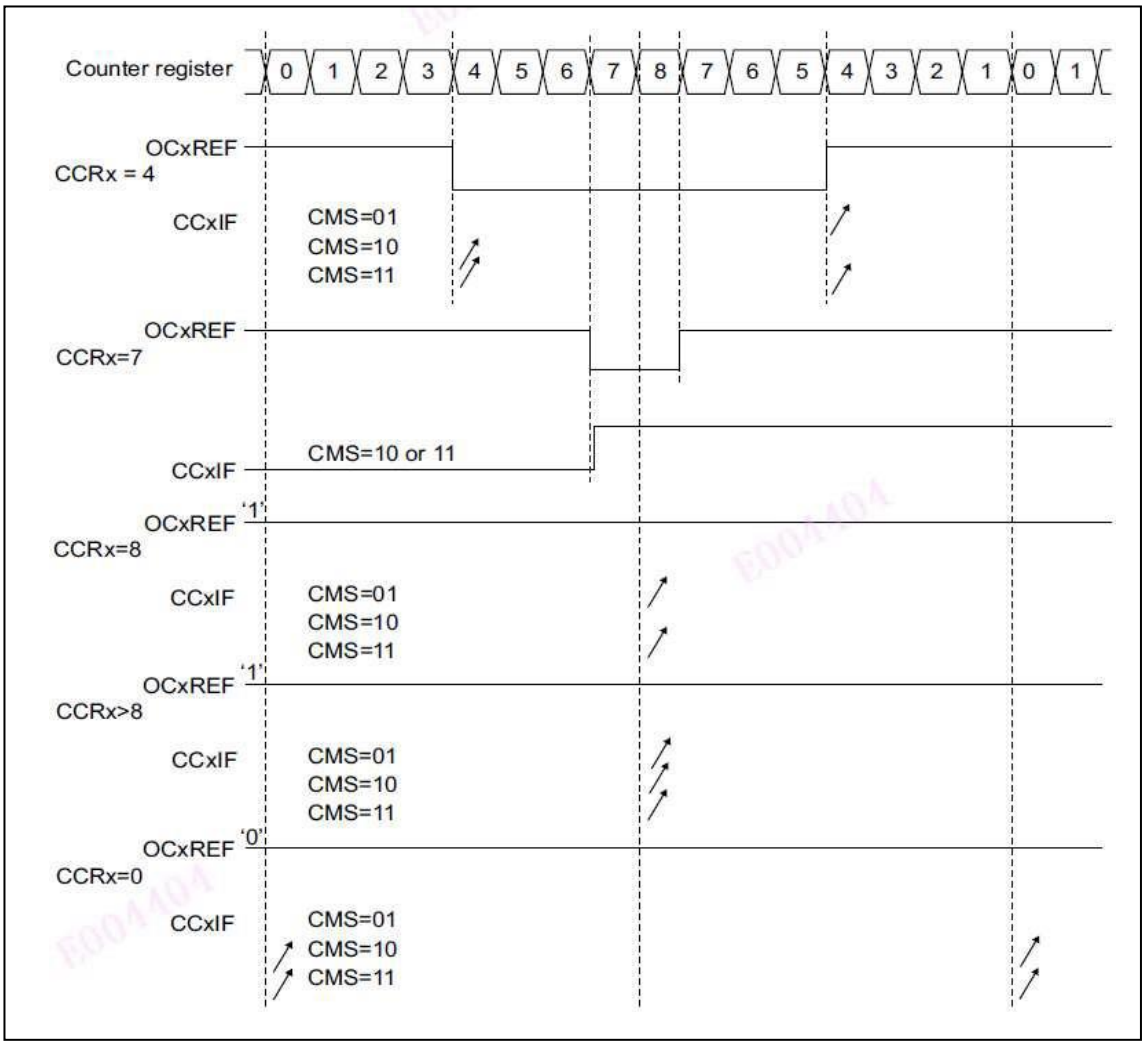


Fig 12-34 Center-aligned symmetric PWM waveforms (ARR=8)

Hints on using center-aligned mode:

- When entering center-aligned mode, the current up/down counting configuration is used. This means whether the counter counts up or down depends on the current value of the DIR bit in the EPWM_CR1 register. Furthermore, software cannot modify the DIR and CMS bits at the same time.
- It is not recommended to write to the counter while running in center-aligned mode, as this may produce unpredictable results. Specifically:
  - If the value written to the counter is greater than the auto-reload value (EPWM_CNT > EPWM_ARR), the direction is not updated. For example, if the counter is counting up, it will continue to count up
  - If 0 or the value of EPWM_ARR is written to the counter, the direction is updated, but no update event UEV is generated.

The safest way to use center-aligned mode is to generate a software update (set the UG bit in the EPWM_EGR register) before starting the counter, and do not modify the counter value while counting is in progress.

Asymmetric mode

Center-aligned mode is selected when the CMS bits in the EPWM_CR1 register are not '00' (all other configurations have the same effect on the OCxREF/OCx signals). Depending on the CMS bit settings, the compare flag can be set when the counter counts up, when the counter counts down, or when the counter counts both up and down. The count direction bit (DIR) in the EPWM_CR1 register is updated by hardware and should not be modified by software. Set ASYMEN=1 for asymmetric PWM output.

The figure below shows some examples of center-aligned PWM waveforms

- EPWM_ARR=900
- PWM mode 1, ASYMEN=1, PWM output is asymmetric
- EPWM_CCRn=300, EPWM_CCDR1=600

CMS=10 in the EPWM_CR1 register, in center-aligned mode 2, the compare flag CCnIF is set when the counter counts up.

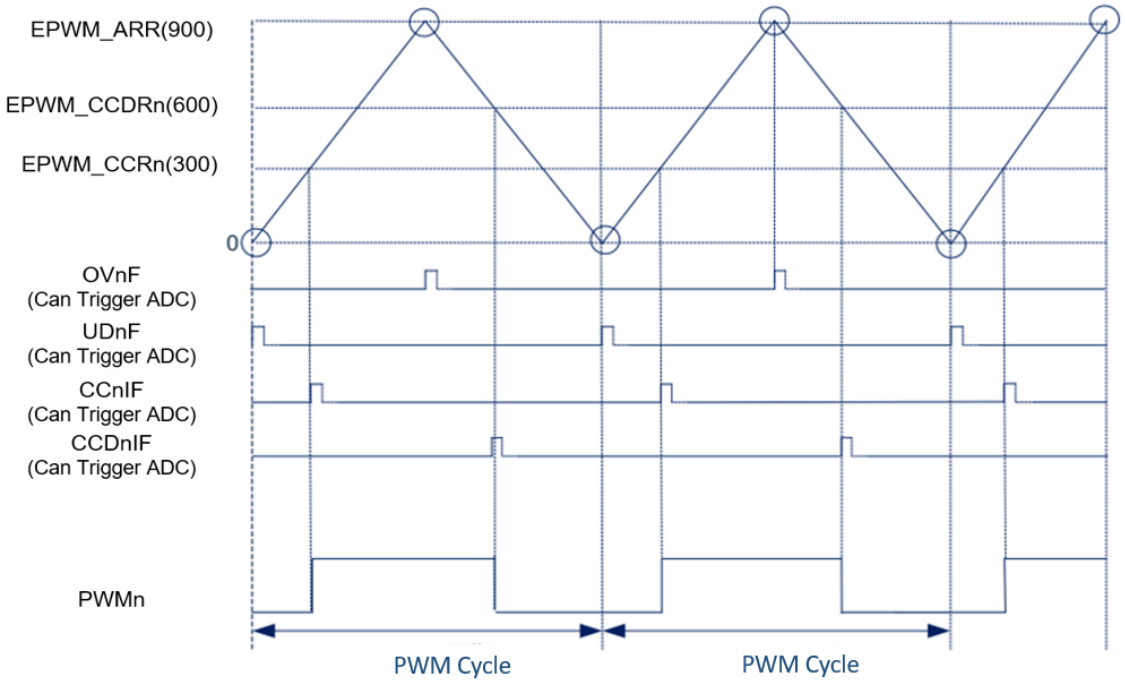


Fig 12-35 Center-aligned asymmetric PWM waveforms (EPWM_ARR = 900)

12.3.11 Complementary outputs and dead-time insertion

The advanced control timer (EPWM) can output two complementary signals and manage the instantaneous switch-off and switch-on of the outputs. This duration is usually called dead-time. Users should adjust the dead-time according to the connected output devices and their characteristics (level translation delay, power switch delay, etc.).

By configuring the CCxP and CCxNP bits in the EPWM_CCER register, the polarity can be selected independently for each output (main output OCx or complementary output OCxN).

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: CCxE and CCxNE bits in the EPWM_CCER register, and MOE, OISx, OISxN, OSSI, and OSSR bits in the EPWM_BDTR and EPWM_CR2 registers. See Table 12-4 Control bits for complementary output channels OCx and OCxN with brake function for details. In particular, dead-time is activated when transitioning to the IDLE state (MOE falling to 0).

Setting both the CCxE and CCxNE bits will insert dead-time. If a brake circuit exists, the MOE bit must also be set. Each channel has a 10-bit dead-time generator. The reference signal OCxREF can generate 2 outputs OCx and OCxN. If OCx and OCxN are active high:

- The OCx output signal is the same as the reference signal, except its rising edge is delayed relative to the rising edge of the reference signal.
- The OCxN output signal is opposite to the reference signal, except its rising edge is delayed relative to the falling edge of the reference signal.

If the delay is greater than the currently active output width (OCx or OCxN), the corresponding pulse will not be generated.

The following figures show the relationship between the output signals of the dead-time generator and the current reference signal OCxREF. (Assuming CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1, and CCxNE = 1)

- DTAE = 0/1: (Symmetric and asymmetric with dead-time)

When EPWM_BDTR.DTAE is set to 0, it is a symmetric output, and the dead-time for both rising and falling edges is determined by EPWM_BDTR.DTG. When EPWM_BDTR.DTAE is set to 1, it is an asymmetric output, and the dead-time for the falling edge is determined by EPWM_BDTR.DTGF.

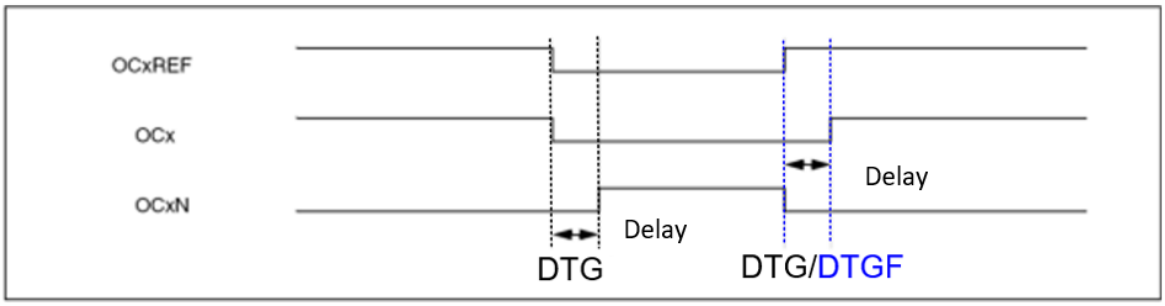


Fig 12-36 Complementary output with symmetric and asymmetric dead-time insertion

- Dead-time delay greater than pulse duration results in no pulse on the complementary output.

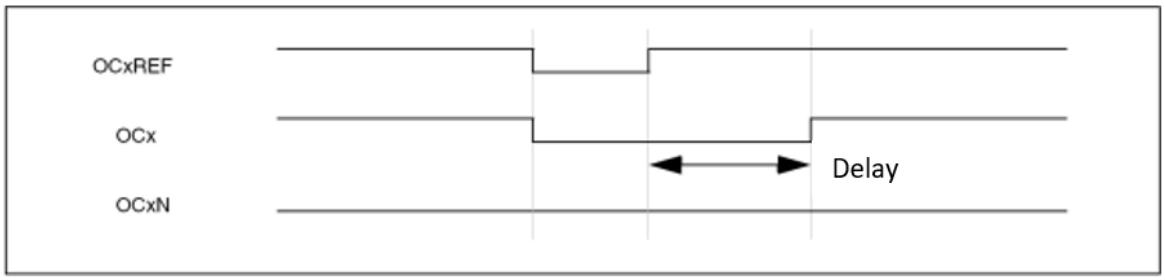


Fig 12-37 Dead-time waveform delay greater than negative pulse

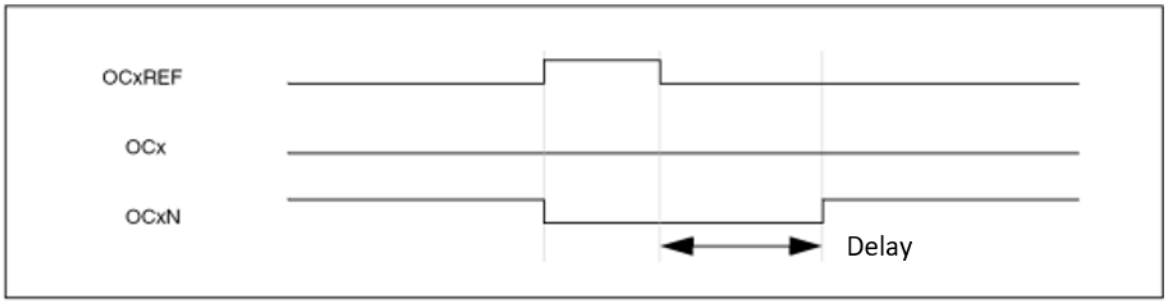


Fig 12-38 Dead-time waveform delay greater than positive pulse

The dead-time delay for each channel is the same and is configured by the DTG bits in the EPWM_BDTR register.

See the delay calculation in the EPWM Brake and Dead-time Register (EPWM_BDTR) for details.

12.3.11.1 Redirecting OCxREF to OCx or OCxN

In output mode (forced, output compare, or PWM), OCxREF can be redirected to the OCx or OCxN output by configuring the CCxE and CCxNE bits in the EPWM_CCER register.

This function allows sending a specific waveform (such as PWM or static active level) on a specific output while the complementary output is at an inactive level. Another function is to allow both outputs to be at the inactive level simultaneously, or to be at the active level and complementary output with dead-time.

Note: When only OCxN is enabled (CCxE=0, CCxNE=1), it is not inverted and becomes high immediately when OCxREF is active. For example, if CCxNP=0, then OCxN=OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE=CCxNE=1), OCx is active when OCxREF is high; whereas OCxN is the opposite, becoming active when OCxREF is low.

12.3.12 Using the brake function

When using the brake function, the output enable signals and inactive levels are modified according to the corresponding control bits (MOE, OSSI, and OSSR bits in the EPWM_BDTR register, OISx and OISxN bits in the EPWM_CR2 register). However, the OCx and OCxN outputs cannot be at the active level simultaneously at any time. See Table 12-4 Control bits for complementary output channels OCx and OCxN with brake function for details.

The brake source can be either the brake input pin or a clock failure event. The clock failure event is generated by the clock security system in the reset and clock controller.

After a system reset, the brake circuit is disabled and the MOE bit is low. Setting the BKE bit in the EPWM_BDTR register enables the brake function. The polarity of the brake input signal can be selected by configuring the BKP bit in the same register. BKE and BKP can be modified simultaneously. When writing to the BKE and BKP bits, there is a delay of 1 APB clock cycle before the actual write occurs, so it is necessary to wait 1 APB clock cycle before correctly reading back the written bits.

Because the MOE falling edge can be asynchronous, a resynchronization circuit is placed between the actual signal (at the output) and the synchronous control bit (in the EPWM_BDTR register). This resynchronization circuit creates a delay between the asynchronous signal and the synchronous signal. Specifically, if you write MOE=1 when it is low, you must insert a delay (dummy instruction) before reading it to get the correct value. This is because you are writing an asynchronous signal but reading a synchronous signal.

When a brake occurs (selected level appears on the brake input), the following actions occur:

- The MOE bit is cleared asynchronously, putting the outputs in the inactive state, idle state, or reset state (selected by the OSSI bit). This feature is valid even when the MCU oscillator is off.
- Once MOE=0, each output channel outputs the level defined by the OISx bits in the EPWM_CR2 register. If OSSI=0, the timer releases the output enable, otherwise the output enable is always high.

- When using complementary outputs:
  - The outputs are first put in the reset state, i.e., the inactive state (depending on polarity). This is an asynchronous operation, valid even if the timer has no clock.
  - If the timer clock is still present, the dead-time generator will be reactivated, driving the output ports according to the levels indicated by the OISx and OISxN bits after the dead-time. Even in this case, OCx and OCxN cannot be driven to the active level simultaneously. Note that due to the resynchronization of MOE, the dead-time is slightly longer than usual (approximately 2 ck_tim clock cycles).
  - If OSSI=0, the timer releases the output enable, otherwise it keeps the output enable; or once one of CCxE or CCxNE becomes high, the output enable becomes high.
- If the BIE bit in the EPWM_IER register is set, an interrupt is generated when the brake status flag (BIF bit in the EPWM_SR register) is '1'.
- If the AOE bit in the EPWM_BDTR register is set, the MOE bit is automatically set at the next update event UEV; for example, this can be used for regulation. Otherwise, MOE remains low until it is set to '1' again; in this case, this feature can be used for safety purposes, you can connect the brake input to the alarm output of a power driver, a thermal sensor, or other safety devices.

Note: The brake input is level active. Therefore, when the brake input is active, MOE cannot be set (automatically or by software) at the same time. Also, the status flag BIF cannot be cleared.

The brake is generated by the BRK input, its active polarity is programmable and it is enabled by the BKE bit in the EPWM_BDTR register.

In addition to the brake input and output management, a write protection is implemented in the brake circuit to ensure the safety of the application. It allows the user to freeze several configuration parameters (dead-time duration, OCx/OCxN polarity and disabled state, OCxM configuration, brake enable and polarity).

The user can select one of the three protection levels via the LOCK bits in the EPWM_BDTR register, see Section 12.5-18 EPWM Brake and Dead-time Register (EPWM_BDTR). The LOCK bits can only be written once after MCU reset.

The figure below shows an example of output response to a brake.

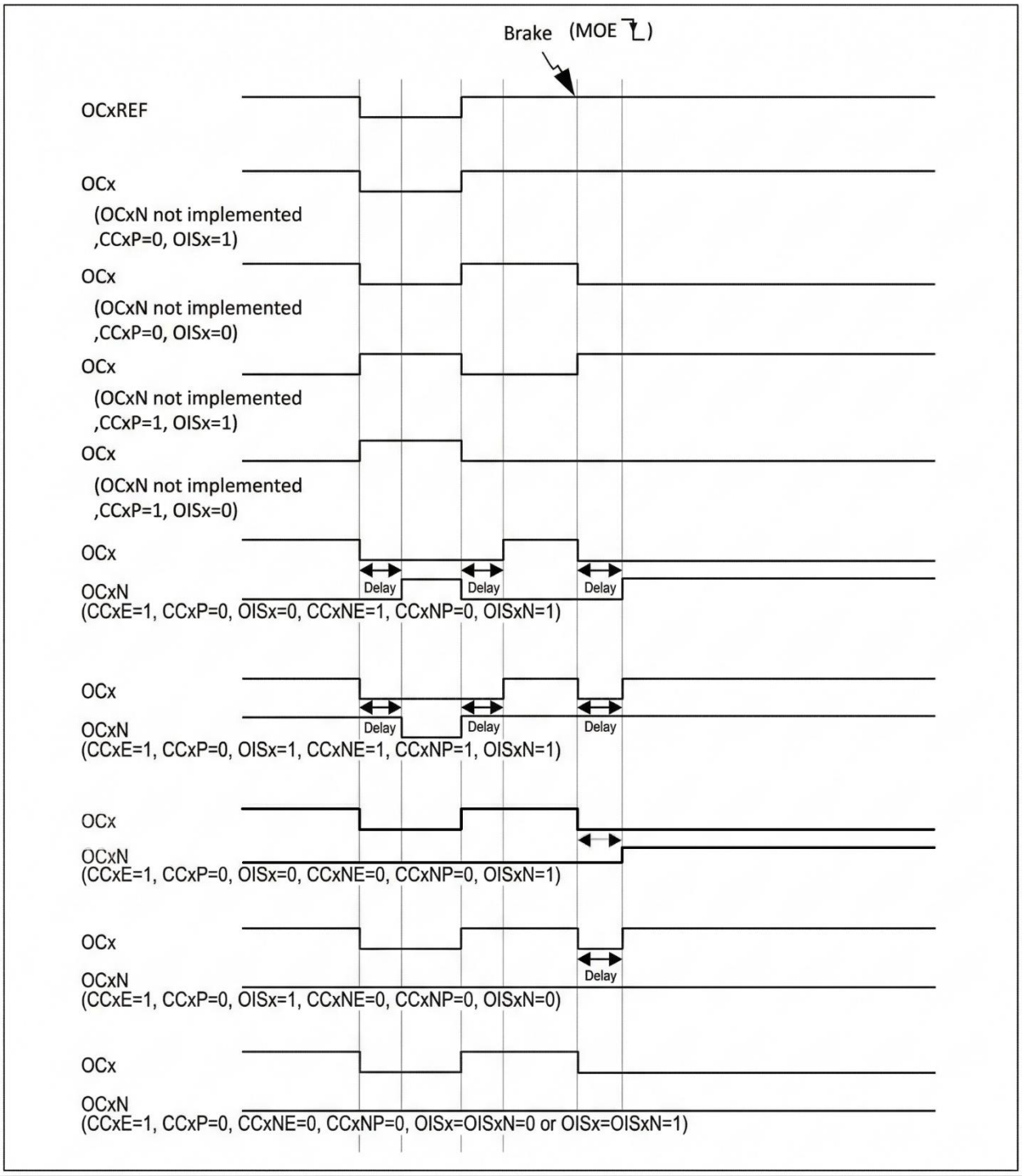


Fig 12-39 Output behavior in response to a brake

12.3.13 Clearing the OCxREF signal on an external event

For a given channel, setting the corresponding OCxCE bit in the EPWM_CCMRx register to '1' allows the OCxREF signal to be driven low by a high level on the ETRF input. The OCxREF signal will remain low until the next update event UEV occurs.

This function can only be used in output compare and PWM modes, and cannot be used in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator for current control. In this case, ETR must be configured as follows:

1. The external trigger prescaler must be disabled: ETPS[1:0]=00 in the EPWM_SMCR register.
2. External clock mode 2 must be disabled: ECE=0 in the EPWM_SMCR register.
3. The external trigger polarity (ETP) and external trigger filter (ETF) can be configured as needed.

The figure below shows the behavior of the OCxREF signal corresponding to different values of OCxCE when the ETRF input becomes high. In this example, the timer EPWM is configured in PWM mode.。

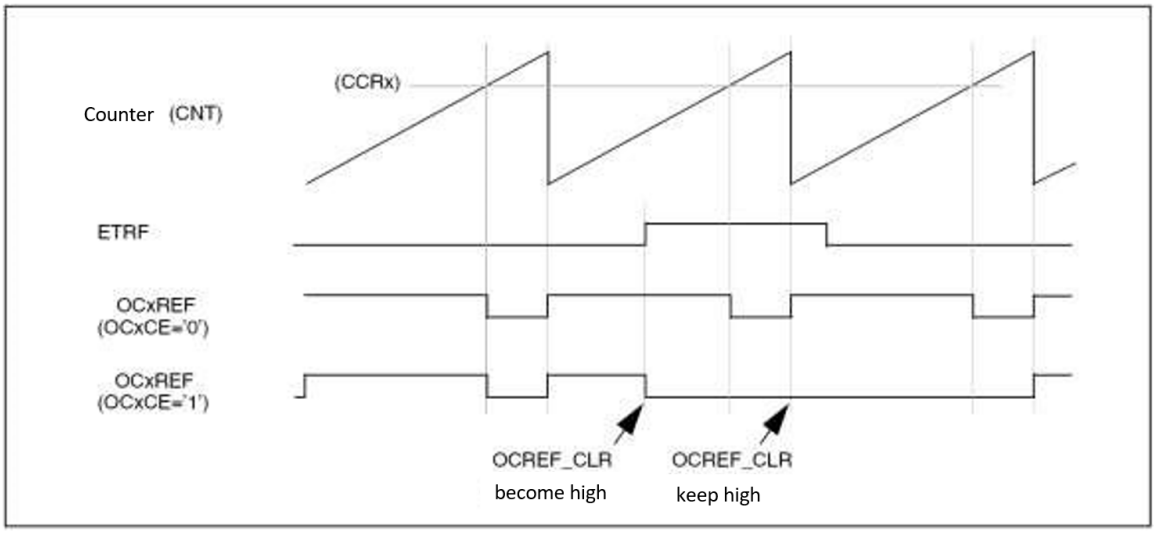


Fig 12-40 Clearing the OCxREF of EPWMGenerating six-step PWM output

12.3.14 Six step PWM generation

When complementary outputs are required on a channel, the preload bits are OCxM, CCxE, and CCxNE. When a COM commutation event occurs, these preload bits are transferred to the shadow register bits. This allows you to pre-program the configuration for the next step and change the configuration of all channels simultaneously at the same time. COM can be generated by software by setting the COM bit in the EPWM_EGR register, or by hardware on the rising edge of TRGI.

When a COM event occurs, a flag bit (COMIF bit in the EPWM_SR register) is set. If the COMIE bit in the EPWM_IER register is set, an interrupt is generated.

The figure below shows the OCx and OCxN outputs under three different configurations when a COM event occurs.

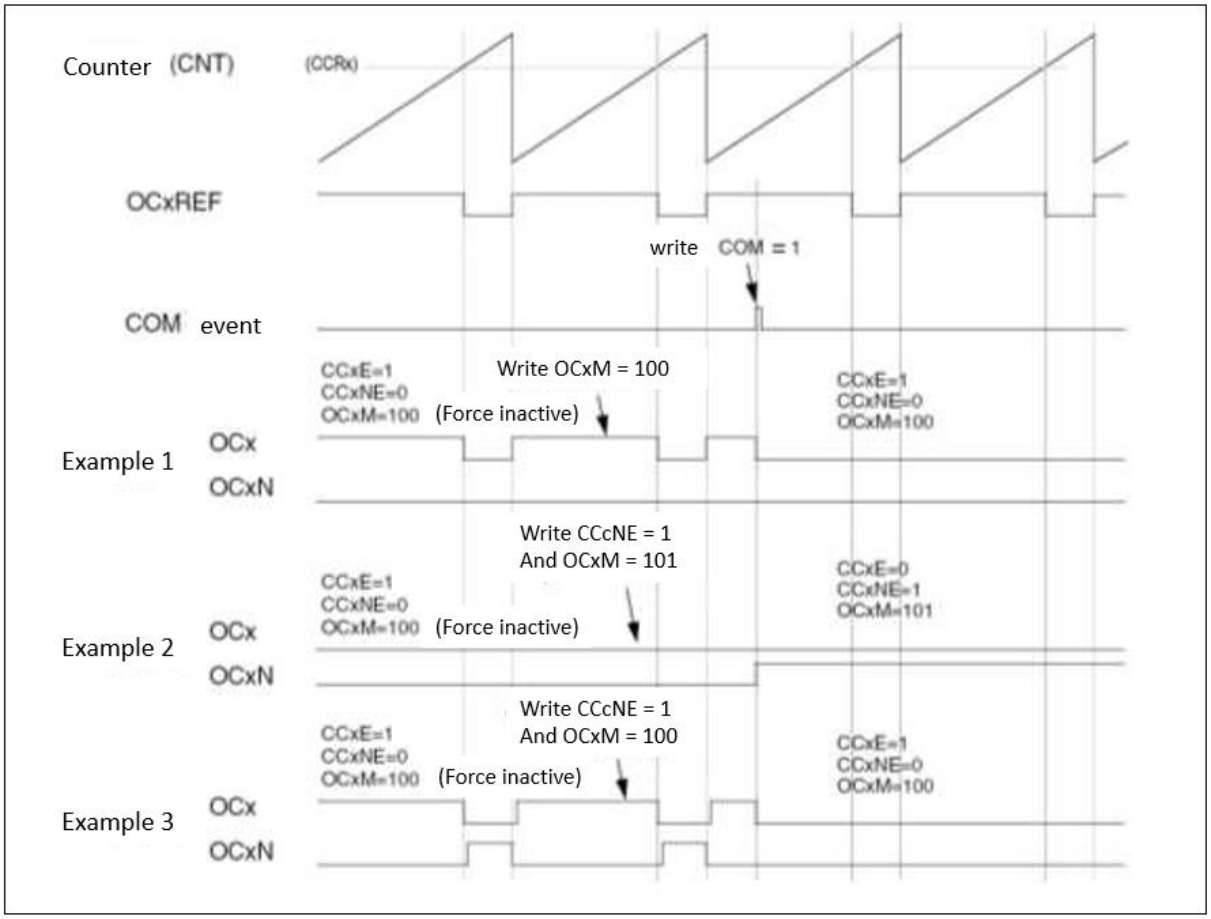


Fig 12-41 Example of six-step PWM generation using COM (OSSR=1)

12.3.15 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. This mode allows the counter to start in response to a stimulus and generate a pulse with a programmable width after a programmable delay.

The counter can be started via the slave mode controller to generate waveforms in output compare mode or PWM mode. Setting the OPM bit in the EPWM_CR1 register selects one-pulse mode, allowing the counter to stop automatically when the next update event UEV occurs.

A pulse can be generated only when the compare value is different from the counter initial value. Before starting (when the timer is waiting for a trigger), the configuration must be as follows:

- Up-counting mode: Counter CNT < CCRx <= ARR (specifically, 0 < CCRx)
- Down-counting mode: Counter CNT > CCRx

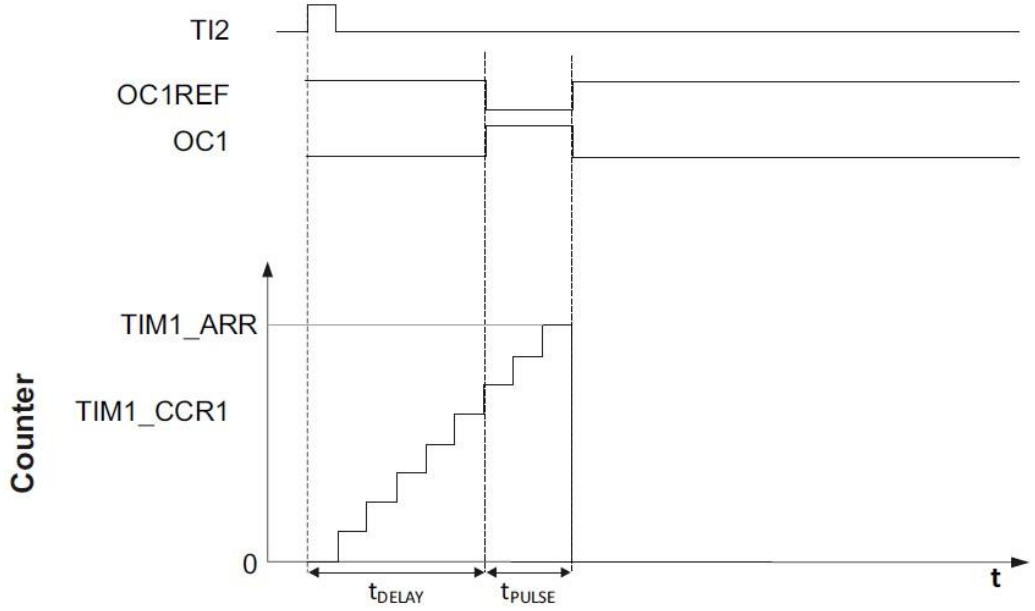


Fig 12-42 Example of one-pulse mode

For example, you need to generate a positive pulse of length tPULSE on OC1 after a delay of tDELAY, starting when a rising edge is detected on the TI2 input pin.

Assume TI2FP2 is used as Trigger 1:

- Set CC2S=01 in the EPWM_CCMR1 register to map TI2FP2 to TI2.
- Set CC2P=0 in the EPWM_CCER register to make TI2FP2 detect a rising edge.
- Set TS=110 in the EPWM_SMCR register to select TI2FP2 as the trigger for the slave mode controller (TRGI).
- Set SMS=110 in the EPWM_SMCR register (trigger mode), TI2FP2 is used to start the counter.

The OPM waveform is defined by the values written to the compare registers (taking into account the clock frequency and the counter prescaler)

- tDELAY is defined by the value in the EPWM_CCR1 register.
- tPULSE is defined by the difference between the auto-reload value and the compare value (EPWM_ARR - EPWM_CCR1).
- Assume we want to generate a waveform from 0 to 1 when a compare match occurs, and from 1 to 0 when the counter reaches the preload value; first, set OC1M=111 in the EPWM_CCMR1 register to enter PWM mode 2; optionally enable the preload register as needed: set OC1PE=1 in EPWM_CCMR1 and ARPE in the EPWM_CR1 register; then write the compare value in the EPWM_CCR1 register, write the auto-reload value in the EPWM_ARR register, set the UG bit to generate an update event, and then wait for an external trigger event on TI2. In this example, CC1P=0.

In this example, the DIR and CMS bits in the EPWM_CR1 register should be set low. Since only one pulse is needed, OPM=1 must be set in the EPWM_CR1 register to stop counting at the next update event (when the counter rolls over from the auto-reload value to 0).

12.3.15.1 Special case: OCx fast enable

In one-pulse mode, the edge detection logic on the TIx input pin sets the CEN bit to start the counter. Then the comparison between the counter and the compare value generates the output transition. However, these operations require a certain number of clock cycles, thus limiting the minimum achievable delay tDELAY.

If a waveform with minimum delay is required, the OCxFE bit in the EPWM_CCMRx register can be set; in this case, OCxREF (and OCx) responds directly to the stimulus and no longer depends on the comparison result, the output waveform is the same as the waveform upon compare match. OCxFE only works when the channel is configured in PWM1 and PWM2 modes.

12.3.16 Encoder interface mode

The method to select the encoder interface mode is: if the counter counts only on the edges of TI2, set SMS=001 in the EPWM_SMCR register; if it counts only on the edges of TI1, set SMS=010; if the counter counts on the edges of both TI1 and TI2, set SMS=011.

The polarity of TI1 and TI2 can be selected by setting the CC1P and CC2P bits in the EPWM_CCER register; the input filter can also be programmed if needed.

The two inputs TI1 and TI2 are used to interface with an incremental encoder. Refer to Table 12-1. Assuming the counter has been enabled (CEN=1 in the EPWM_CR1 register), the counter is clocked by each valid transition on TI1FP1 or TI2FP2. TI1FP1 and TI2FP2 are the signals of TI1 and TI2 after passing through the input filter and polarity control; if there is no filtering and no inversion, then TI1FP1=TI1, TI2FP2=TI2. The counting pulses and direction signals are generated according to the transition sequence of the two input signals. Based on the transition sequence of the two input signals, the counter counts up or down, while the hardware sets the DIR bit in the EPWM_CR1 register accordingly. Regardless of whether the counter counts based on TI1, TI2, or both TI1 and TI2, a transition on any input (TI1 or TI2) will recalculate the DIR bit.

Encoder interface mode is basically equivalent to using an external clock with direction selection. This means the counter only counts continuously between 0 and the auto-reload value in the EPWM_ARR register (counting from 0 to ARR or ARR to 0 depending on the direction). Therefore, EPWM_ARR must be configured before counting starts; similarly, the capturer, comparator, prescaler, repetition counter, trigger output features, etc., still work as usual. Encoder mode and external clock mode 2 are not compatible and therefore cannot operate simultaneously.

In this mode, the counter is automatically modified according to the speed and direction of the incremental encoder, so the content of the counter always indicates the position of the encoder. The counting direction corresponds to the direction of rotation of the connected sensor. The table below lists all possible combinations, assuming TI1 and TI2 do not switch simultaneously.

Active Edge	Level of relative signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 Signal	TI1FP1 Signal	TI2FP2 Signal	TI2FP2 Signal
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down Counting	Up Counting	No Count	No Count
	Low	Up Counting	Down Counting	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up Counting	Down Counting
	Low	No Count	No Count	Down Counting	Up Counting
Counting on both TI1 and TI2	High	Down Counting	Up Counting	Up Counting	Down Counting
	Low	Up Counting	Down Counting	Down Counting	Up Counting

Table 12-1 Relationship between counting direction and encoder signals

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are generally used to convert the differential outputs of the encoder to digital signals, which greatly increases noise immunity. The third signal output by the encoder indicates the mechanical zero point, which can be connected to an external interrupt input and trigger a counter reset

The figure below is an example of counter operation, showing the generation of counting signals and direction control. It also shows how input jitter is suppressed when both edges are selected; jitter may occur when the sensor position is close to a transition point. In this example, we assume the configuration is as follows:

- CC1S='01' (EPWM_CCMR1 register, IC1FP1 mapped to TI1)
- CC2S='01'(EPWM_CCMR2 register, IC2FP2 mapped to TI2)
- CC1P='0'(EPWM_CCER register，IC1FP1 non-inverted，IC1FP1=TI1)
- CC2P='0'(EPWM_CCER register，IC2FP2 non-inverted，IC2FP2=TI2)
- SMS='011'(EPWM_SMCR register，all inputs are active on both rising and falling edges)
- CEN='1'(EPWM_CR1 register, counter enabled)

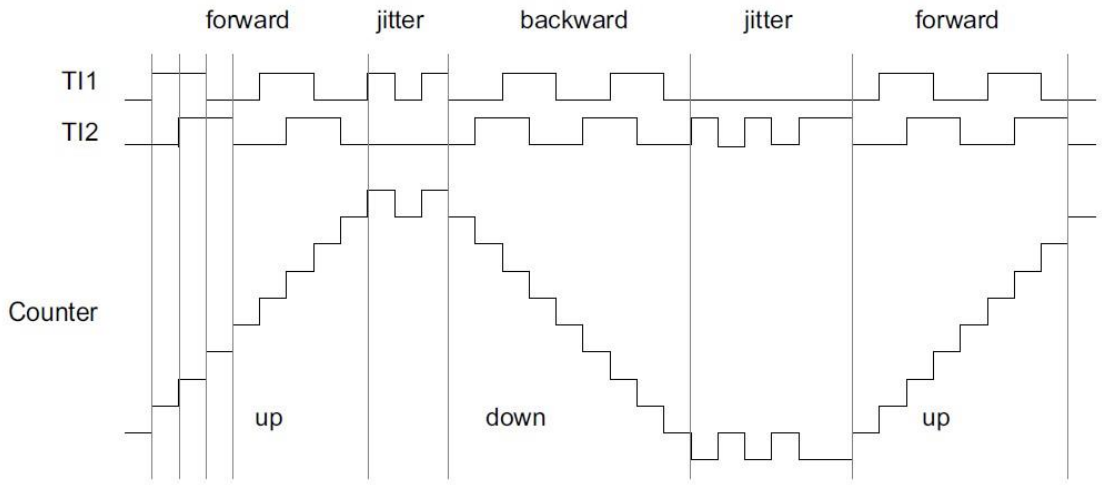


Fig 12-43 Example of counter operation in encoder mode

The figure below shows an example of counter operation when the polarity of IC1FP1 is inverted (CC1P='1', other configurations are the same as the example above).

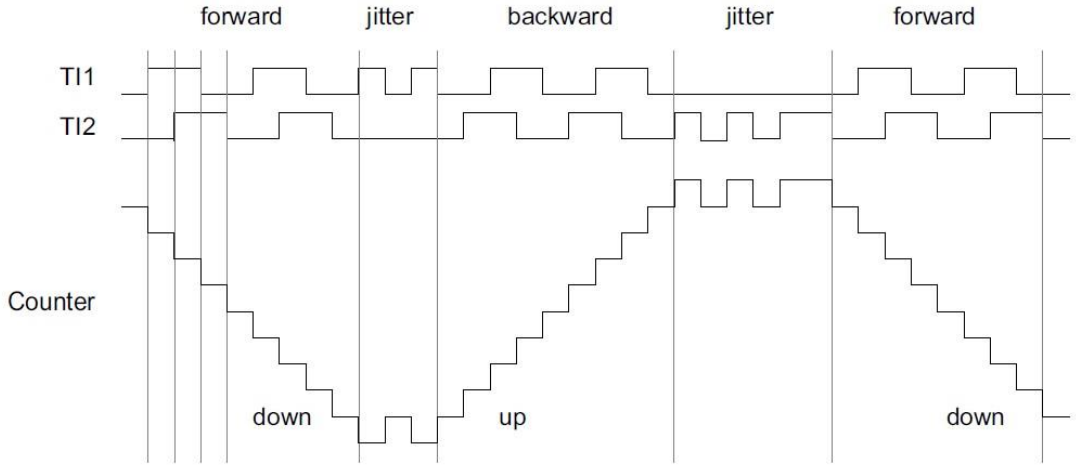


Fig 12-44 Example of encoder interface mode with IC1FP1 inverted

When the timer is configured in encoder interface mode, it provides information on the current position of the sensor. Using a second timer configured in capture mode, the interval between two encoder events can be measured to obtain dynamic information (speed, acceleration, deceleration). The encoder output indicating the mechanical zero point can be used for this purpose. Based on the interval between two events, the counter can be read at fixed times. If possible, you can latch the value of the counter into a third input capture register (the capture signal must be periodic and can be generated by another timer).

12.3.17 Timer input XOR function

The TI1S bit in the EPWM_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate. The 3 inputs of the XOR gate are EPWM_CH1, EPWM_CH2, and EPWM_CH3.

The XOR output can be used for all timer input functions, such as trigger or input capture.

12.3.18 Synchronization of EPWM timer and external trigger

The EPWM timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode, and Trigger mode.

12.3.18.1 Slave mode: Reset mode

When a trigger input event occurs, the counter and its prescaler can be re-initialized; meanwhile, if the URS bit in the EPWM_CR1 register is low, an update event (UEV) is also generated; then all preload registers (EPWM_ARR, EPWM_CCRx) are updated.

In the following example, the rising edge on the TI1 input causes the up-counter to be cleared:

- Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filter is needed, so keep IC1F=0000). The capture prescaler is not used in trigger operations, so it does not need to be configured. The CC1S bit selects the input capture source, i.e., CC1S=01 in the EPWM_CCMR1 register. Set CC1P=0 in the EPWM_CCER register to determine the polarity (detect rising edge only).
- Set SMS=100 in the EPWM_SMCR register to configure the timer in reset mode; set TS=101 in the EPWM_SMCR register to select TI1 as the input source.
- Set CEN=1 in the EPWM_CR1 register to start the counter.

The counter starts counting based on the internal clock, and then operates normally until a rising edge appears on TI1; at this time, the counter is cleared and restarts counting from 0. At the same time, the trigger flag (TIF bit in the EPWM_SR register) is set, and an interrupt request is generated according to the setting of the TIE (interrupt enable) bit in the EPWM_IER register.

he figure below shows the behavior when the auto-reload register EPWM_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter depends on the resynchronization circuit at the TI1 input.

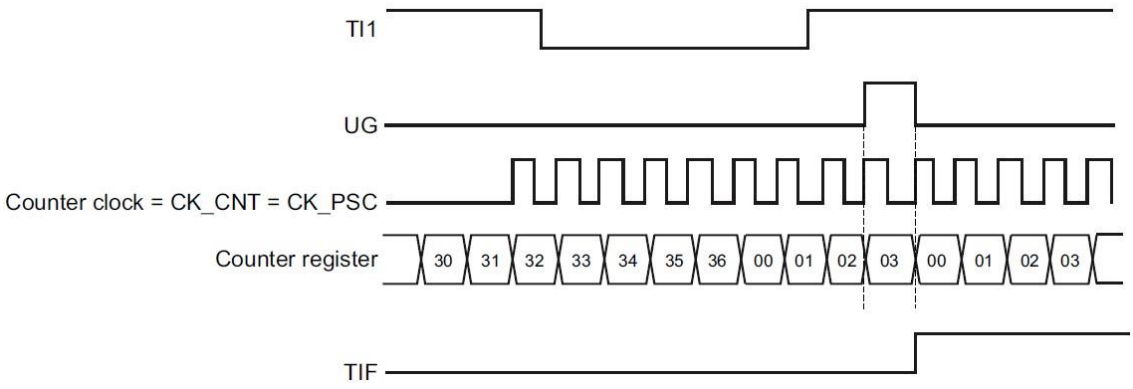


Fig 12-45 Control circuit in reset mode

12.3.18.2 Slave mode: Gated mode

Enable the counter according to the level of the selected input.

In the following example, the counter counts up only when TI1 is low:

- Configure channel 1 to detect the low level on TI1. Configure the input filter bandwidth (in this example, no filtering is needed, so keep IC1F=0000). The capture prescaler is not used in trigger operations, so it does not need to be configured. The CC1S bit is used to select the input capture source, set CC1S=01 in the EPWM_CCMR1 register. Set CC1P=1 in the EPWM_CCER register to determine the polarity (detect low level only).
  Set SMS=101 in the EPWM_SMCR register to configure the timer in gated mode; set TS=101 in the EPWM_SMCR register to select TI1 as the input source.
- Set CEN=1 in the EPWM_CR1 register to start the counter. In gated mode, if CEN=0, the counter cannot start, regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock, and once TI1 becomes high, it stops counting. The TIF flag in EPWM_SR is set when the counter starts or stops.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronization circuit at the TI1 input.

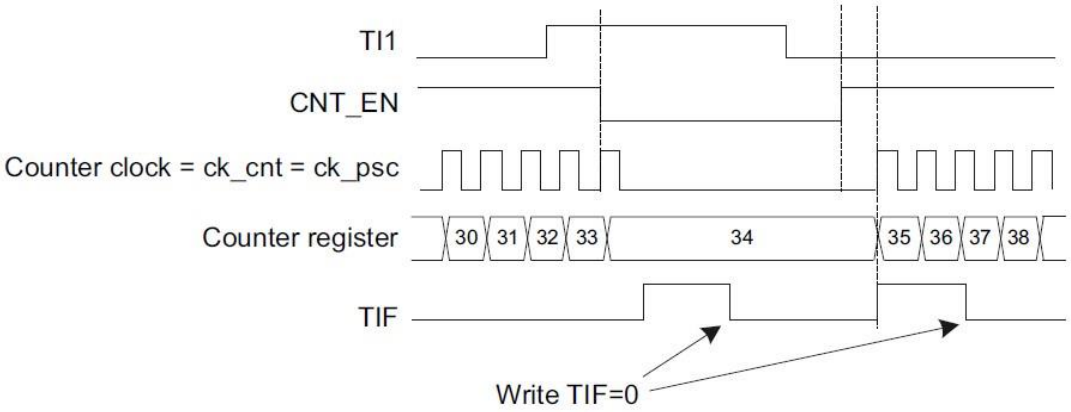


Fig 12-46 Control circuit in gated mode

12.3.18.3 Slave mode: Trigger mode

The counter is enabled by a selected event on the input.

In the following example, the counter starts counting up on the rising edge of the TI2 input:

- Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is needed, keep IC2F=0000). The capture prescaler is not used in trigger operations and does not need to be configured. The CC2S bit is used only to select the input capture source, set CC2S=01 in the EPWM_CCMR1 register. Set CC2P=1 in the EPWM_CCER register to determine the polarity (detect low level only).
  Set SMS=110 in the EPWM_SMCR register to configure the timer as trigger mode; set TS=110 in the EPWM_SMCR register to select TI2 as the input source.

When a rising edge occurs on TI2, the counter starts counting driven by the internal clock, and the TIF flag is set.

The delay between the rising edge of TI2 and the start of the counter depends on the resynchronization circuit at the TI2 input.

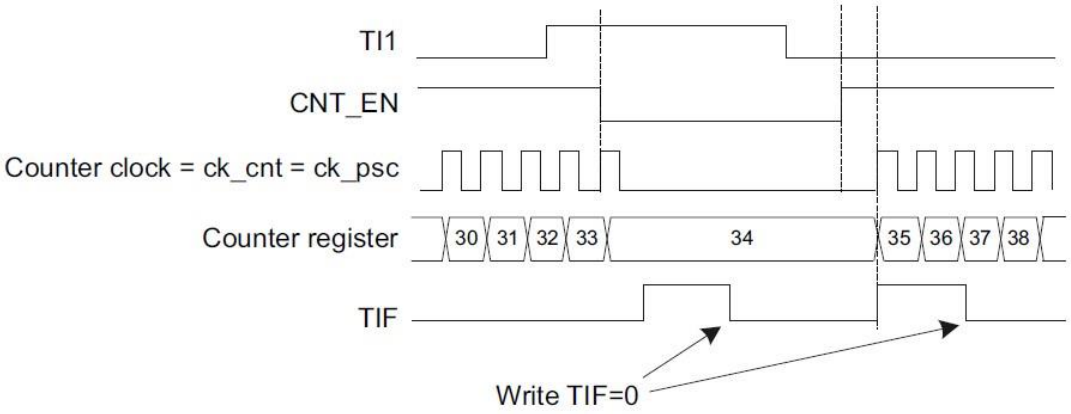


Fig 12-47 Control circuit in trigger mode

12.3.18.4 Slave mode: External clock mode 2 + Trigger mode

External clock mode 2 can be used in combination with another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as the input for the external clock, and another input can be selected as the trigger input in reset mode, gated mode, or trigger mode.
It is not recommended to use the TS bits in the EPWM_SMCR register to select ETR as TRGI.
In the following example, once a rising edge appears on TI1, the counter counts up once for every rising edge on ETR:

1. Configure the external trigger input circuit via the EPWM_SMCR register:
   - ETF=0000:No filter
   - ETPS=00:No prescaler
   - ETP=0: Detect rising edge of ETR, set ECE=1 to enable external clock mode 2.
2. Configure channel 1 to detect the rising edge of TI1 as follows:
   - IC1F=0000:No filter
   - Capture prescaler is not used in trigger operation, no configuration needed
   - Set CC1S=01 in the EPWM_CCMR1 register to select the input capture source
   - Set CC1P=0 in the EPWM_CCER register to determine the polarity (detect rising edge only)
3. Set SMS=110 in the EPWM_SMCR register to configure the timer in trigger mode. Set TS=101 in the EPWM_SMCR register to select TI1 as the input source.

When a rising edge appears on TI1, the TIF flag is set, and the counter starts counting on the rising edge of ETR. The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the ETRP input.

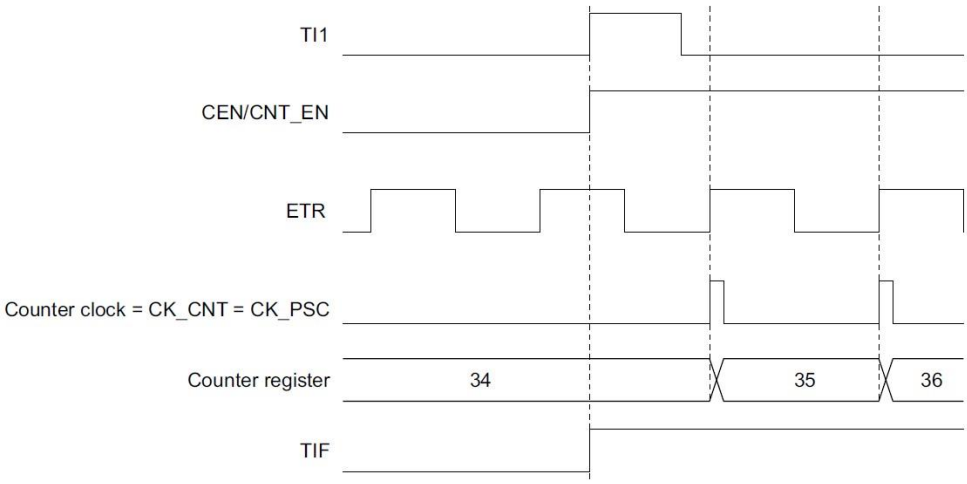


Fig 12-48 Control circuit in external clock mode 2 + trigger mode

12.3.18.5 Timer synchronization

All TIM timers are connected internally for timer synchronization or chaining. See the next chapter 13.3.15 Timer synchronization for details.

12.3.18.6 Debug mode

When the microcontroller enters debug mode (CPU core stopped), depending on the setting of DBG_EPWM_STOP in the DBG module, the EPWM counter can either continue normal operation or stop.

12.4 EPWM register table

These peripheral registers can be accessed by word (32-bit). EPWM base address 0x4000 C000.

Address	Name	Description
0x4000C000	EPWM_CR1	EPWM Control Register 1
0x4000C004	EPWM_CR2	EPWM Control Register 2
0x4000C008	EPWM_SMCR	EPWM Slave Mode Control Register
0x4000C00C	EPWM_IER	EPWM Interrupt Enable Register
0x4000C010	EPWM_SR	EPWM Status Register
0x4000C014	EPWM_EGR	EPWM Event Generation Register
0x4000C018	EPWM_CCMR1	EPWM Capture/Compare Mode Register 1
0x4000C01C	EPWM_CCMR2	EPWM Capture/Compare Mode Register 2
0x4000C020	EPWM_CCER	EPWM Capture/Compare Enable Register
0x4000C024	EPWM_CNT	EPWM Counter Register
0x4000C028	EPWM_PSC	EPWM Prescaler Register
0x4000C02C	EPWM_ARR	EPWM Auto-Reload Register
0x4000C030	EPWM_RCR	EPWM Repetition Counter Register
0x4000C034	EPWM_CCR1	EPWM Capture/Compare Register 1
0x4000C038	EPWM_CCR2	EPWM Capture/Compare Register 2
0x4000C03C	EPWM_CCR3	EPWM Capture/Compare Register 3
0x4000C040	EPWM_CCR4	EPWM Capture/Compare Register 4
0x4000C044	EPWM_BDTR	EPWM Brake and Dead-Time Register
0x4000C050	EPWM_CCDR1	EPWM Capture/Compare Down Register 1
0x4000C054	EPWM_CCDR2	EPWM Capture/Compare Down Register 2
0x4000C058	EPWM_CCDR3	EPWM Capture/Compare Down Register 3
0x4000C05C	EPWM_CCDR4	EPWM Capture/Compare Down Register 4

Table 12-2 EPWM register table

12.5 EPWM register description

12.5.1 EPWM Control Register 1(EPWM_CR1)

Bit	Name	R/W	Reset	Description
31:11	Reserved	--	0x0	Always read as 0.
10	ASYMEN	R/W	0x0	EPWM Center-Aligned Asymmetric Counting Enable 0: Symmetric Counting Enable 1: Asymmetric Counting Enable
9:8	CKD	R/W	0x0	Clock division These 2 bits define the division ratio between the timer clock (CK_INT) frequency, dead-time and the sampling clock used by the dead-time generator and digital filters (ETR,TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved, do not use this configuration
7	ARPE	R/W	0x0	Auto-reload preload enable 0: EPWM_ARR register is not buffered 1: EPWM_ARR register is buffered
6:5	CMS	R/W	0x0	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1 The counter counts up and down alternatively. Output compare interrupt flags of channels configured as output (CCxS=00 in EPWM_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2 The counter counts up and down alternatively. Output compare interrupt flags of channels configured as output (CCxS=00 in EPWM_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3 The counter counts up and down alternatively. Output compare interrupt flags of channels configured as output (CCxS=00 in EPWM_CCMRx register) are set both when the counter is counting up and down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).

4	DIR	R/W	0x0	Direction 0: Counter counts up; 1: Counter counts down. Note: This bit is read-only when the counter is configured in Center-aligned mode or Encoder mode.
3	OPM	R/W	0x0	One pulse mode 0: Counter is not stopped at update event; 1: Counter stops at the next update event (clearing the CEN bit).
2	URS	R/W	0x0	Update request source This bit selects the UEV event source by software 0: The update interrupt is generated by any of the following events if set to 0: -Counter overflow/underflow -Setting UG bit - Update generated by slave mode controller 1: The update interrupt is generated only by counter overflow/underflow if set to 1.
1	UDIS	R/W	0x0	Update disable This bit enables/disables the UEV event generation by software 0: UEV enabled. The Update (UEV) event is generated by any of the following events: -Counter overflow/underflow -Setting UG bit -Update generated by slave mode controller Buffered registers are loaded with their preload values. 1: UEV disabled. No update event is generated, shadow registers (ARR, PSC, CCRx) keep their value. If the UG bit is set or if a hardware reset is received from the slave mode controller, the counter and the prescaler are re-initialized.
0	CEN	R/W	0x0	Counter enable 0: Counter disabled; 1: Counter enabled. Note: External clock, gated mode and encoder mode work only if the CEN bit has been previously set by software.

12.5.2 EPWM Control Register 2(EPWM_CR2)

Bit	Name	R/W	Reset	Description
31:15	Reserved	--	0x0	Always read as 0.
14	OIS4	R/W	0x0	Output Idle state 4 (OC4 output). Refer to OIS1 bit.
13	OIS3N	R/W	0x0	Output Idle state 3 (OC3N output). Refer to OIS1N bit.
12	OIS3	R/W	0x0	Output Idle state 3 (OC3 output). Refer to OIS1 bit.
11	OIS2N	R/W	0x0	Output Idle state 2 (OC2N output). Refer to OIS1N bit.
10	OIS2	R/W	0x0	Output Idle state 2 (OC2 output). Refer to OIS1 bit.
9	OIS1N	R/W	0x0	Output Idle state 1 (OC1N output) 0: OC1N=0 after dead-time when MOE=0; 1: OC1N=1 after dead-time when MOE=0. Note: This bit cannot be modified as soon as the LOCK level 1, 2 or 3 has been programmed (EPWM_BKR register).
8	OIS1	R/W	0x0	Output Idle state 1 (OC1 output) 0: OC1=0 after dead-time when MOE=0 (if OC1N is implemented); 1: OC1=1 after dead-time when MOE=0 (if OC1N is implemented). Note: This bit cannot be modified as soon as the LOCK level 1, 2 or 3 has been programmed (EPWM_BKR register).
7	TI1S	R/W	0x0	TI1 selection 0: EPWM_CH1 pin connected to TI1 input; 1: EPWM_CH1, EPWM_CH2 and EPWM_CH3 pins connected to TI1 input after XOR.

6:4	MMS	R/W	0x0	Master mode selection These 3 bits are used to select the information to be sent in master mode to slave timers for synchronization (TRGO). Possible combinations are as follows: 000: Reset - The UG bit from the EPWM_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode), the signal on TRGO is delayed compared to the actual reset. 001: Enable - The Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logical OR between the CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in EPWM_SMCR register). 010: Update - The update event is selected as trigger output (TRGO). For instance, a master timer clock can be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output sends a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or compare match occurs. 100: Compare - OC1REF signal is used as trigger output (TRGO). 101: Compare - OC2REF signal is used as trigger output (TRGO). 110: Compare - OC3REF signal is used as trigger output (TRGO). 111: Compare - OC4REF signal is used as trigger output (TRGO).
3	Reserved	--	0x0	Always read as 0.
2	CCUS	R/W	0x0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit only. 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	--	0x0	Reserved
0	CCPC	R/W	0x0	Capture/compare preloaded control 0: The CCxE, CCxNE and OCxM bits are not preloaded. 1: The CCxE, CCxNE and OCxM bits are preloaded; after having been written, they are updated only when the COM bit is set. Note: This bit acts only on channels that have a complementary output.

12.5.3 EPWM Slave Mode Control Register(EPWM_SMCR)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0	Always read as 0.
15	ETP	R/W	0x0	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations 0: ETR is not inverted, active at high level or rising edge 1: ETR is inverted, active at low level or falling edge
14	ECE	R/W	0x0	External clock enable This bit enables External clock mode 2 0: External clock mode 2 disabled 1: External clock mode 2 enabled The counter is clocked by any active edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: It is possible to use the slave modes simultaneously with the external clock mode 2: Reset mode, Gated mode and Trigger mode; nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.
13:12	ETPS [1:0]	R/W	0x0	External trigger prescaler The frequency of the external trigger signal ETRP must be at most 1/4 of the EPWMCLK frequency. A prescaler can be used to reduce the frequency of ETRP when a faster external clock is input. 00: Prescaler off 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8 外

11:8	ETF [3:0]	R/W	0x0	External trigger filter These bits define the sampling frequency for the ETRP signal and the bandwidth of the digital filter for ETRP. The digital filter is essentially an event counter that outputs a transition after N events are recorded. 0000: No filter, sampling is done at fDTS 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
7	MSM	R/W	0x0	Master/slave mode 0: No effect; 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer (through TRGO) and its slaves. This is useful if we want to synchronize several timers to a single external event.
6:4	TS[2:0]	R/W	0x0	Trigger selection These 3 bits select the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF)

				See Table 12-3 for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. SMS=000) to avoid incorrect edge detections during the transition.
3	Reserved	--	0x0	Always read as 0.
2:0	SMS [2:0]	R/W	0x0	Slave mode selection When external signals are selected, the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control register descriptions) 000: Slave mode disabled - if CEN=1 then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset Mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated Mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger Mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Slave Timer	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
EPWM	Tim2_trgo	irq_timer0	irq_timer1	irq_lptimer

Table 12-3 EPWM Internal Trigger Connections

12.5.4 EPWM Interrupt Enable Register(EPWM_IER)

Bit	Name	R/W	Reset	Description
31:14	Reserved	--	0x0	Always read as 0.。
13	CCD4IE	R/W	0x0	Capture/Compare 4 interrupt enable (ASYMEN =1 and down-counting mode valid) 0: Capture/Compare 4 interrupt disabled; 1: Capture/Compare 4 interrupt enabled.
12	UDIE	R/W	0x0	Underflow interrupt enable 0: Underflow interrupt disabled; 1: Underflow interrupt enabled.
11	OVIE	R/W	0x0	Overflow interrupt enable 0: Overflow interrupt disabled; 1: Overflow interrupt enabled.
10	CCD3IE	R/W	0x0	Capture/Compare down 3 interrupt enable (ASYMEN =1 and down-counting mode valid) 0: Capture/Compare 3 interrupt disabled; 1: Capture/Compare 3 interrupt enabled.
9	CCD2IE	R/W	0x0	Capture/Compare down 2 interrupt enable (ASYMEN =1 and down-counting mode valid) 0: Capture/Compare 2 interrupt disabled; 1: Capture/Compare 2 interrupt enabled.
8	CCD1IE	R/W	0x0	Capture/Compare down 1 interrupt enable (ASYMEN =1 and down-counting mode valid) 0: Capture/Compare 1 interrupt disabled; 1: Capture/Compare 1 interrupt enabled.
7	BIE	R/W	0x0	Brake interrupt enable 0: Brake interrupt disabled; 1: Brake interrupt enabled.
6	TIE	R/W	0x0	Trigger interrupt enable 0: Trigger interrupt disabled; 1: Trigger interrupt enabled.
5	COMIE	R/W	0x0	COM interrupt enable 0: COM interrupt disabled; 1: COM interrupt enabled.
4	CC4IE	R/W	0x0	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled; 1: Capture/Compare 4 interrupt enabled.

3	CC3IE	R/W	0x0	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled; 1: Capture/Compare 3 interrupt enabled.
2	CC2IE	R/W	0x0	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled; 1: Capture/Compare 2 interrupt enabled.
1	CC1IE	R/W	0x0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled; 1: Capture/Compare 1 interrupt enabled.
0	UIE	R/W	0x0	Update interrupt enable 0: Update interrupt disabled; 1: Update interrupt enabled.

12.5.5 EPWM Status Register(EPWM_SR)

Bit	Name	R/W	Reset	Description
31:19	Reserved	--	0x0	Always read as 0.
18	CC4DIF	R/W1c	0x0	Capture/Compare 4 down interrupt flag 1: EPWM_CNT value matches EPWM_CCDR4 value in down-counting mode CCD4IF bit goes high 0: No match occurred (write 0 to clear)
17	UDIF	R/W1c	0x0	Underflow interrupt flag 1: EPWM_CNT down-counting mode underflow bit 0: No underflow bit
16	OVIF	R/W1c	0x0	Overflow interrupt flag 1: EPWM_CNT up-counting mode overflow bit 0: No overflow bit
15	CC3DIF	R/W1c	0x0	Capture/Compare down 3 interrupt flag 1: EPWM_CNT value matches EPWM_CCDR3 value in down-counting mode CCD3IF bit goes high 0: No match occurred (write 0 to clear)
14	CC2DIF	R/W1c	0x0	Capture/Compare down 2 interrupt flag 1: EPWM_CNT value matches EPWM_CCDR2 value in down-counting mode CCD2IF bit goes high 0: No match occurred (write 0 to clear)
13	CC1DIF	R/W1c	0x0	Capture/Compare down 1 interrupt flag 1: EPWM_CNT value matches EPWM_CCDR1 value in down-counting mode CCD1IF bit goes high 0: No match occurred (write 0 to clear)
12	CC4OF	R/W1c	0x0	Capture/Compare 4 overcapture flag Refer to CC1OF description.
11	CC3OF	R/W1c	0x0	Capture/Compare 3 overcapture flag Refer to CC1OF description.
10	CC2OF	R/W1c	0x0	Capture/Compare 2 overcapture flag Refer to CC1OF description.

9	CC1OF	R/W1c	0x0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture mode. Write 0 to clear this bit. 0: No overcapture occurred; 1: The counter value has been captured in EPWM_CCR1 register while CC1IF flag was already set.
8	Reserved	--	0x0	Always read as 0.
7	BIF	R/W1c	0x0	Brake interrupt flag This bit is set by hardware as soon as the brake input goes active. It can be cleared by software if the brake input is not active. 0: No brake event occurred; 1: An active level has been detected on the brake input.
6	TIF	R/W1c	0x0	Trigger interrupt flag This bit is set by hardware when a trigger event occurs (active edge detected on TRGI input when the slave mode controller is in a mode other than gated mode, or any edge in gated mode). It is cleared by software. 0: No trigger event occurred; 1: Trigger interrupt pending.
5	COMIF	R/W1c	0x0	COM interrupt flag This bit is set by hardware as soon as a COM event occurs (when Capture/Compare control bits: CCxE, CCxNE, OCxM have been updated). It is cleared by software. 0: No COM event occurred; 1: COM interrupt pending.
4	CC4IF	R/W1c	0x0	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	R/W1c	0x0	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	R/W1c	0x0	Capture/Compare 2 interrupt flag Refer to CC1IF description.

1	CC1IF	R/W1c	0x0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set by hardware when the counter value matches the compare value, except in center-aligned mode (refer to CMS bits in EPWM_CR1 register). It is cleared by software. 0: No match occurred 1: The content of the counter EPWM_CNT matches the content of the EPWM_CCR1 register When the content of EPWM_CCR1 is greater than the content of EPWM_APR, the CC1IF bit goes high when the counter overflows in up-counting or up/down-counting modes, or underflows in down-counting mode. If channel CC1 is configured as input: This bit is set by hardware when a capture event occurs. It is cleared by software or by reading the EPWM_CCR1 register. 0: No input capture occurred 1: The counter value has been captured (copied) to EPWM_CCR1 (an edge has been detected on IC1 which matches the selected polarity)
0	UIF	R/W1c	0x0	Update interrupt flag This bit is set by hardware when an update event occurs. It is cleared by software. 0: No update event occurred; 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - If UDIS=0 in EPWM_CR1 register, when repetition counter underflow or overflow (update event generated when repetition counter = 0). - If URS=0 and UDIS=0 in EPWM_CR1 register, when CNT is re-initialized by software using the UG bit in EPWM_EGR register. - If URS=0 and UDIS=0 in EPWM_CR1 register, when CNT is re-initialized by a trigger event. (Refer to 12.5.3 EPWM slave mode control register (EPWM_SMCR)).

12.5.6 EPWM Event Generation Register (EPWM_EGR)

Bit	Name	R/W	Reset	Description
31:8	Reserved	--	0x0	Always read as 0.
7	BG	W	0x0	Brake generation This bit is set by software in order to generate a brake event. It is automatically cleared by hardware. 0: No action; 1: A brake event is generated. The MOE bit is cleared and the BIF flag is set. If the interrupt is enabled, an interrupt is generated.
6	TG	W	0x0	Trigger generation This bit is set by software in order to generate a trigger event. It is automatically cleared by hardware. 0: No action; 1: The TIF flag is set in EPWM_SR register. If the interrupt is enabled, an interrupt is generated.
5	COMG	W	0x0	Capture/Compare control update generation This bit is set by software. It is automatically cleared by hardware. 0: No action; 1: When CCPC=1, it allows updating CCxE, CCxNE and OCxM bits. Note: This bit acts only on channels that have a complementary output.
4	CC4G	W	0x0	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	W	0x0	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	W	0x0	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	W	0x0	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event. It is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in EPWM_CCR1 register. The CC1IF flag is set, corresponding interrupt is sent if enabled. If CC1IF was already high, CC1OF flag is set.

0	UG	W	0x0	Update generation This bit is set by software. It is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generate an update of the registers. Note that the prescaler counter is cleared too (but the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (upcounting), else it takes the value of EPWM_ARR (downcounting) if DIR=1.

12.5.7 EPWM Capture/Compare Mode Register 1(EPWM_CCMR1)

The channel can be used in input (capture mode) or output (compare mode). The direction of the channel is defined by the corresponding CCxS bits. The other bits of this register act differently in input and output modes. OCxx describes the function of the channel in output mode, ICxx describes the function of the channel in input mode. Therefore, it must be noted that the same bit has a different function in output and input modes.

- **Output Compare Mode**

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15	OC2CE	R/W	0x0	Output Compare 2 clear enable
14:12	OC2M	R/W	0x0	Output Compare 2 mode
11	OC2PE	R/W	0x0	Output Compare 2 preload enable
10	OC2FE	R/W	0x0	Output Compare 2 fast enable
9:8	CC2S	R/W	0x0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC2S can be written only when the channel is OFF (CC2E=0 in EPWM_CCER register).
7	OC1CE	R/W	0x0	Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF input; 1: OC1REF is cleared as soon as a high level is detected on ETRF input.

6:4	OC1M	R/W	0x0	Output Compare 1 mode These 3 bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active high or active low level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register EPWM_CCR1 and the counter EPWM_CNT has no effect on the outputs; 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter EPWM_CNT matches the capture/compare register 1 (EPWM_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter EPWM_CNT matches the capture/compare register 1 (EPWM_CCR1). 011: Toggle - OC1REF toggles when EPWM_CCR1=EPWM_CNT. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In up-counting, channel 1 is active as long as EPWM_CNT<EPWM_CCR1 else inactive. In down-counting, channel 1 is inactive (OC1REF=0) as long as EPWM_CNT>EPWM_CCR1 else active (OC1REF=1). 111: PWM mode 2 - In up-counting, channel 1 is inactive as long as EPWM_CNT<EPWM_CCR1 else active. In down-counting, channel 1 is active as long as EPWM_CNT>EPWM_CCR1 else inactive. Note 1: These bits cannot be modified as soon as the LOCK level 3 has been programmed (LOCK bit in EPWM_BDTR register) and CC1S=00 (the channel is configured as output). Note 2: In PWM mode 1 or 2, the OC1REF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode.

3	OC1PE	R/W	0x0	Output Compare 1 preload enable 0: Preload register on EPWM_CCR1 disabled. EPWM_CCR1 can be written at anytime, the new value is taken into account immediately. 1: Preload register on EPWM_CCR1 enabled. Read/Write operations access the preload register. EPWM_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits cannot be modified as soon as the LOCK level 3 has been programmed (LOCK bit in EPWM_BDTR register) and CC1S=00 (the channel is configured as output). Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM=1 in EPWM_CR1 register). Otherwise the behavior is not guaranteed.
2	OC1FE	R/W	0x0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even when the trigger is ON. The minimum delay to activate CC1 output when an active edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	R/W	0x0	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC1S bit is writable only when the channel is OFF (CC1E = 0 in TIM2_CCER).

- **Input Capture Mode:**

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:12	IC2F	R/W	0x0	Input capture 2 filter
11:10	IC2PSC	R/W	0x0	Input capture 2 prescaler
9:8	CC2S	R/W	0x0	Capture/Compare 2 selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC2 channel is configured as output; 01: CC2 channel is configured as input, IC2 is mapped on TI2; 10: CC2 channel is configured as input, IC2 is mapped on TI1; 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC2S can be written only when the channel is OFF (CC2E=0 in EPWM_CCER register).
7:4	IC1F	R/W	0x0	Input capture 1 filter These bits define the sampling frequency for the TI1 input and the length of the digital filter. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001:fSAMPLING = fCK_INT，N=2 0010:fSAMPLING = fCK_INT，N=4 0011:fSAMPLING = fCK_INT，N=8 0100:fSAMPLING = fDTS / 2，N=6 0101:fSAMPLING = fDTS / 2，N=8 0110:fSAMPLING = fDTS / 4，N=6 0111:fSAMPLING = fDTS / 4，N=8 1000:fSAMPLING = fDTS / 8，N=6 1001:fSAMPLING = fDTS / 8，N=8 1010:fSAMPLING = fDTS / 16，N=5 1011:fSAMPLING = fDTS / 16，N=6 1100:fSAMPLING = fDTS / 16，N=8 1101:fSAMPLING = fDTS / 32，N=5 1110:fSAMPLING = fDTS / 32，N=6 1111:fSAMPLING = fDTS / 32，N=8

3:2	IC1PSC	R/W	0x0	Input capture 1 prescaler These 2 bits define the prescaler ratio on CC1 input (IC1). The prescaler is reset as soon as CC1E=0 (in EPWM_CCER register). 00: no prescaler, capture is done each time an edge is detected on the capture input; 01: capture is done once every 2 events; 10: capture is done once every 4 events; 11: capture is done once every 8 events.
1:0	CC1S	R/W	0x0	Capture/Compare 1 selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC1 channel is configured as output; 01: CC1 channel is configured as input, IC1 is mapped on TI1; 10: CC1 channel is configured as input, IC1 is mapped on TI2; 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC1S can be written only when the channel is OFF (CC1E=0 in EPWM_CCER register).

12.5.8 EPWM Capture/Compare Mode Register 2(EPWM_CCMR2)

- **Output Compare Mode**

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15	OC4CE	R/W	0x0	Output Compare 4 clear enable
14:12	OC4M	R/W	0x0	Output Compare 4 mode
11	OC4PE	R/W	0x0	Output Compare 4 preload enable
10	OC4FE	R/W	0x0	Output Compare 4 fast enable
9:8	CC4S[1:0]	R/W	0x0	Capture/Compare 4 selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC4 channel is configured as output; 01: CC4 channel is configured as input, IC4 is mapped on TI4; 10: CC4 channel is configured as input, IC4 is mapped on TI3; 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC4S can be written only when the channel is OFF (CC4E=0 in EPWM_CCER register).
7	OC3CE	R/W	0x0	Output Compare 3 clear enable
6:4	OC3M	R/W	0x0	Output Compare 3 mode
3	OC3PE	R/W	0x0	Output Compare 3 preload enable
2	OC3FE	R/W	0x0	Output Compare 3 fast enable
1:0	CC3S	R/W	0x0	Capture/Compare 3 selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output; 01: CC3 channel is configured as input, IC3 is mapped on TI3; 10: CC3 channel is configured as input, IC3 is mapped on TI4; 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC3S can be written only when the channel is OFF (CC3E=0 in EPWM_CCER register).

- **Input Caupre Mode:**

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:12	IC4F	R/W	0x0	Input capture 4 filter
11:10	IC4PSC	R/W	0x0	Input capture 4 prescaler
9:8	CC4S	R/W	0x0	Capture/Compare 4 selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC4 channel is configured as output; 01: CC4 channel is configured as input, IC4 is mapped on TI4; 10: CC4 channel is configured as input, IC4 is mapped on TI3; 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC4S can be written only when the channel is OFF (CC4E=0 in EPWM_CCER register).
7:4	IC3F	R/W	0x0	Input capture 3 filter
3:2	IC3PSC	R/W	0x0	Input capture 3 prescaler
1:0	CC3S	R/W	0x0	Capture/Compare 3 Selection These 2 bits define the direction of the channel (input/output) as well as the used input: 00: CC3 channel is configured as output; 01: CC3 channel is configured as input, IC3 is mapped on TI3; 10: CC3 channel is configured as input, IC3 is mapped on TI4; 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected (selected through TS bit in EPWM_SMCR register). Note: CC3S can be written only when the channel is OFF (CC3E=0 in EPWM_CCER register).

12.5.9 EPWM Capture/Compare Enable Register (EPWM_CCER)

Bit	Name	R/W	Reset	Description
31:14	Reserved	--	0x0	Always read as 0.
13	CC4P	R/W	0x0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	R/W	0x0	Capture/Compare 4 output enable Refer to CC1E description.
11	CC3NP	R/W	0x0	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
10	CC3NE	R/W	0x0	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
9	CC3P	R/W	0x0	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	R/W	0x0	Capture/Compare 3 output enable Refer to CC1E description.
7	CC2NP	R/W	0x0	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
6	CC2NE	R/W	0x0	Capture/Compare 3 complementary output polarity Refer to CC1NP description.
5	CC2P	R/W	0x0	Capture/Compare 3 output polarity Refer to CC1P description.
4	CC2E	R/W	0x0	Capture/Compare 3 output enable Refer to CC1E description.
3	CC1NP	R/W	0x0	Capture/Compare 1 complementary output polarity 0: OC1N active high; 1: OC1N active low. Note: This bit cannot be modified as soon as the LOCK level 2 or 3 has been programmed (LOCK bit in EPWM_BDTR register) and CC1S=00 (channel configured as output).
2	CC1NE	R/W	0x0	Capture/Compare 1 complementary output enable 0: Off - OC1N is not active. OC1N level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits. 1: On - OC1N signal is output on the corresponding output pin. Its output level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits.

1	CC1P	R/W	0x0	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high; 1: OC1 active low. CC1 channel configured as input: This bit selects whether IC1 or inverted IC1 is used as trigger or capture signal. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted. Note: This bit cannot be modified as soon as the LOCK level 2 or 3 has been programmed (LOCK bit in EPWM_BDTR register).
0	CC1E	R/W	0x0	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 output level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin. Its output level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can occur in the TIM2_CCR1 register. 0: Capture disabled 1: Capture enabled

Table 12-4 Control bits for complementary OCx and OCxN channels with brake function

Control bits					Output state	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx Output state	OCxN Output state
1	X	0	0	0	Output disabled (disconnected from timer) OCx=0，OCx_EN=0	Output disabled (disconnected from timer)OCxN=0，OCxN_EN=0
		0	0	1	Output disabled (disconnected from timer) OCx=0，OCx_EN=0	OCxREF + polarity， OCxN= OCxREF xor CCxNP， OCxN_EN=1
		0	1	0	OCxREF + polarity， OCx= OCxREF xor CCxP， OCx_EN=1	Output disabled (disconnected from timer)OCxN=0，OCxN_EN=0
		0	1	1	OCxREF + polarity + dead-time， OCx_EN=1	OCxREF inverted + polarity + dead-time，OCxN_EN=1
		1	0	0	Output disabled (disconnected from timer)OCx=CCxP， OCx_EN=0	Output disabled (disconnected from timer)OCxN=CCxNP， OCxN_EN=0
		1	0	1	Off state (output enabled with inactive level)OCx=CCxP， OCx_EN=1	OCxREF + polarity， OCxN= OCxREF xor CCxNP， OCxN_EN=1
		1	1	0	OCxREF + polarity， OCx= OCxREF xor CCxP， OCx_EN=1	Off state (output enabled with inactive level) OCxN=CCxNP，OCxN_EN=1
		1	1	1	OCxREF + polarity + dead-time， OCx_EN=1	OCxREF inverted + polarity + dead-time，OCxN_EN=1

0	0	X	0	0	Output disabled (disconnected from timer) Asynchronously: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0; If clock is present: after a dead-time OCx=OISx, OCxN=OISxN, assuming OISx and OISxN do not correspond to the active level of OCx and OCxN. 输出禁止(与定时器断开) 异步地 OCx=CCxP，OCx_EN=0，OCxN=CCxNP，
	0		0	1	
	0		1	0	
	0		1	1	
	1		0	0	Off state (output enabled with inactive level) Asynchronously: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1; If clock is present: after a dead-time OCx=OISx, OCxN=OISxN, assuming OISx and OISxN do not correspond to the active level of OCx and OCxN.
	1		0	1	
	1		1	0	
	1		1	1	

If both outputs of a channel are not used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP and CCxNP must be cleared.

Note: The state of the external I/O pins connected to the complementary OCx and OCxN channels depends on the OCx and OCxN channel state and the GPIO and AFIO registers.

12.5.10 EPWM Counter Register (EPWM_CNT)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CNT	R/W	0x0	Counter value

12.5.11 EPWM Prescaler Register (EPWM_PSC)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	PSC	R/W	0x0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event; update events include the counter being cleared by the UG bit in the EPWM_EGR register or by the slave mode controller working in reset mode.

12.5.12 EPWM Auto Reload Register (EPWM_ARR)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	ARR	R/W	0x0	Auto-reload value ARR contains the value to be loaded in the actual auto-reload register. Refer to Section 12.3.1 for details on ARR update and behavior. The counter is blocked when the auto-reload value is null.

12.5.13 EPWM Repetition Counter Register (EPWM_RCR)

Bit	Name	R/W	Reset	Description
31:8	Reserved	--	0x0	Always read as 0.
7:0	REP[7:0]	R/W	0x0	Repetition counter value Once the preload function is enabled, these bits allow the user to set the update rate of the compare registers (i.e. periodic transfer from preload to active registers); if the update interrupt is enabled, it also affects the update interrupt generation rate. Each time the downcounter REP_CNT reaches 0, an update event is generated and the REP_CNT counter restarts counting from the REP value. As REP_CNT reloads the REP value only when a periodic update event U_RC occurs, the new value written to the EPWM_RCR register is taken into account only at the next periodic update event. This means that in PWM mode, (REP+1) corresponds to: - The number of PWM periods in edge-aligned mode; The number of PWM half-periods in center-aligned mode;

12.5.14 EPWM Capture/Compare Register 1 (EPWM_CCR1)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCR1	R/W	0x0	Capture/Compare 1 value If CC1 channel is configured as output: CCR1 contains the value to be loaded in the active capture/compare 1 register (preload value). If the preload function is not selected in the EPWM_CCMR1 register (OC1PE bit), the value written is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 1 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter EPWM_CNT and signaled on OC1 output. If CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

12.5.15 EPWM Capture/Compare Register 2(EPWM_CCR2)

Bit	Name	Reset	R/W	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCR2	R/W	0x0	Capture/Compare 2 value If CC2 channel is configured as output: CCR2 contains the value to be loaded in the active capture/compare 2 register (preload value). If the preload function is not selected in the EPWM_CCMR1 register (OC2PE bit), the value written is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 2 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter EPWM_CNT and signaled on OC2 output. If CC2 channel is configured as input: CCR2 contains the counter value transferred by the last input capture 2 event (IC2).

12.5.16 EPWM Capture/Compare Register 3(EPWM_CCR3)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCR3	R/W	0x0	Capture/Compare 3 value If CC3 channel is configured as output: CCR3 contains the value to be loaded in the active capture/compare 3 mregister (preload value). If the preload function is not selected in the EPWM_CCMR2 register (OC3PE bit), the value written is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 3 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter EPWM_CNT and signaled on OC3 output. If CC3 channel is configured as input: CCR3 contains the counter value transferred by the last input capture 3 event (IC3).

12.5.17 EPWM Capture/Compare Register 4(EPWM_CCR4)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCR4[19:0]	R/W	0x0	Capture/Compare 4 value If CC4 channel is configured as output: CCR4 contains the value to be loaded in the active capture/compare 4 register (preload value). If the preload function is not selected in the EPWM_CCMR2 register (OC4PE bit), the value written is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 4 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter EPWM_CNT and signaled on OC4 output. If CC4 channel is configured as input: CCR4 contains the counter value transferred by the last input capture 4 event (IC4).

12.5.18 EPWM Brake and Dead-Timer Register(EPWM_BDTR)

Note: As the AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] bits can be write-protected depending on the LOCK configuration, it is necessary to configure them in the very first write to the EPWM_BDTR register.

Bit	Name	R/W	Reset	Description
31:25	Reserved	--	0x0	Always read as 0.
24	DTAE	R/W	0x0	Asymmetric dead-time enable control 0: Dead-time for rising and falling edges is the same, defined by DTG 1: Rising edge dead-time is defined by DTG[7:0], Falling edge dead-time is defined by DTGF[7:0].
23:16	DTGF	R/W	0x0	Complementary channel OCxREFC falling edge dead-time configuration DTGF[7:5]=0xxDeadTime = (DTGF[7:0] + 0) × tDTS × 1 DTGF[7:5]=10xDeadTime = (DTGF[5:0] + 64) × tDTS × 2 DTGF[7:5]=110: DeadTime = (DTGF[4:0] + 32) × tDTS × 8 DTGF[7:5]=111: DeadTime = (DTGF[4:0] + 32) × tDTS × 16
15	MOE	R/W	0x0	Main output enable This bit is cleared asynchronously by hardware as soon as the brake input is active. It can be set by software or automatically depending on the AOE bit. It acts only on the channels which are configured in output. 0: OC and OCN outputs are disabled or forced to idle state; 1: OC and OCN outputs are enabled if their respective enable bits are set (CCxE, CCxNE in EPWM_CCER register). See 12.5.9 EPWM and TIM2 capture/compare enable register (EPWM_CCER) for more details on OC/OCN enable.

14	AOE	R/W	0x0	Automatic output enable 0: MOE can be set only by software; 1: MOE can be set by software or automatically at the next update event (if the brake input is not active). Note: This bit cannot be modified as soon as the LOCK level 1 has been programmed (LOCK bit in EPWM_BDTR register).
13	BKP	R/W	0x0	Brake polarity 0: Brake input is active low; 1: Brake input is active high. Note: This bit cannot be modified as soon as the LOCK level 1 has been programmed (LOCK bit in EPWM_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.
12	BKE	R/W	0x0	Brake enable 0: Brake inputs (BRK and CCS clock failure event) disabled; 1: Brake inputs (BRK and CCS clock failure event) enabled. Note: This bit cannot be modified as soon as the LOCK level 1 has been programmed (LOCK bit in EPWM_BDTR register). Note: Any write operation to this bit takes a delay of 1 APB clock cycle to become effective.

11	OSSR	R/W	0x0	Off-state selection for Run mode This bit is used when MOE=1 on channels configured as outputs. It does not exist on timers with no complementary outputs. See 12.5.9 EPWM and TIM2 capture/compare enable register (EPWM_CCER) for more details on OC/OCN enable. 0: When the timer is not working, OC/OCN outputs are disabled (OC/OCN enable output signal=0); 1: When the timer is not working, as soon as CCxE=1 or CCxNE=1, the OC/OCN outputs are enabled with their inactive level. OC/OCN enable output signal=1. Note: This bit cannot be modified as soon as the LOCK level 2 has been programmed (LOCK bit in EPWM_BDTR register).
10	OSSI	R/W	0x0	Off-state selection for Idle mode This bit is used when MOE=0 on channels configured as outputs. See 11.5.9 EPWM and TIM8 capture/compare enable register (EPWM_CCER) for more details on OC/OCN enable. 0: When the timer is not working, OC/OCN outputs are disabled (OC/OCN enable output signal=0); 1: When the timer is not working, as soon as CCxE=1 or CCxNE=1, the OC/OCN outputs are enabled with their idle level. OC/OCN enable output signal=1. Note: This bit cannot be modified as soon as the LOCK level 2 has been programmed (LOCK bit in EPWM_BDTR register).

9:8	LOCK[1:0]	R/W	0x0	Lock Configuration This bit-field offers a write protection against software errors. 00: LOCK OFF - No write protection; 01: LOCK Level 1 - DTG, BKE, BKP, AOE bits in EPWM_BDTR register and OISx/OISxN bits in EPWM_CR2 register cannot be written; 10: LOCK Level 2 - LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in EPWM_CCER register, as long as the related channel is configured in output through the CCxS bits) as well as OSSR/OSSI bits cannot be written; 11: LOCK Level 3 - LOCK Level 2 + CC Control bits (OCxM/OCxPE bits in EPWM_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) cannot be written; Note: The LOCK bits can be written only once after the reset. Once the EPWM_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG[7:0]	R/W	0x0	Dead-time generator setup These bits define the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration: DTG[7:5]=0xx => DT=DTG[7:0]xTdtg, Tdtg = tDTS DTG[7:5]=10x => DT=(64+DTG[5:0])xTdtg, Tdtg = 2xtDTS DTG[7:5]=110 => DT=(32+DTG[4:0])xTdtg, Tdtg = 8xtDTS DTG[7:5]=111 => DT=(32+DTG[4:0])xTdtg, Tdtg = 16xtDTS Example: if tDTS = 125ns (8MHz), dead-time possible values are: 0 to 15875ns, if step is 125ns; 16us to 31750ns, if step is 250ns; 32us to 63us, if step is 1us; 64us to 126us, if step is 2us; Note: This bit-field cannot be modified as soon as the LOCK level 1, 2 or 3 has been programmed (LOCK bit in EPWM_BDTR register).

12.5.19 EPWM Capture/Compare Down Register 1 (EPWM_CCDR1)

Bit	Name	R/W	Reset	Description
31:20	Reserved	R	0x0	Reserved
19:0	CCDR1	R/W	0x0	EPWM capture/compare down value (Compare channel 1 value)

				(Valid when asymmetric counting in EPWM center-aligned mode is enabled via EPWM_CR1.ASYMEN)

12.5.20 EPWM Capture/Compare Down Register 2 (EPWM_CCDR2)

Bit	Name	R/W	Reset	Description
31:20	Reserved	R	0x0	Reserved
19:0	CCDR2	R/W	0x0	EPWM capture/compare down value (Compare channel 2 value) (Valid when asymmetric counting in EPWM center-aligned mode is enabled via EPWM_CR1.ASYMEN)

12.5.21 EPWM Capture/Compare Down Register 3 (EPWM_CCDR3)

Bit	Name	R/W	Reset	Description
31:20	Reserved	R	0x0	Reserved
19:0	CCDR3	R/W	0x0	EPWM capture/compare down value (Compare channel 3 value) (Valid when asymmetric counting in EPWM center-aligned mode is enabled via EPWM_CR1.ASYMEN)

12.5.22 EPWM Capture/Compare Down Register 4 (EPWM_CCDR4)

Bit	Name	R/W	Reset	Description
31:20	Reserved	R	0x0	Reserved
19:0	CCDR4	R/W	0x0	EPWM capture/compare down value (Compare channel 4 value) (Valid when asymmetric counting in EPWM center-aligned mode is enabled via EPWM_CR1.ASYMEN)

12.6 TIM2 Overview

The general-purpose timer consists of a 16-bit auto-reload counter driven by a programmable prescaler.
It may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).
Pulse lengths and waveform periods can be adjusted from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers.
The timers are completely independent, and do not share any resources. They can be synchronized together.

12.7 TIM2 Characteristics

General-purpose TIM2 timer features include:

- 16-bit up, down, up/down auto-reload counter
- 16-bit programmable (can be modified on the fly) prescaler used to divide the counter clock frequency by any factor between 1 and 65536 - 4 independent channels:
- 4 independent channels:
  - Input capture
  - Output compare
  - PWM generation (Edge- or Center-aligned modes)
  - One-pulse mode output
- Synchronization circuit to control the timer with external signals and to interconnect the timers
- Interrupt generation on the following events:
  - Update: counter overflow/underflow, counter initialization (by software or internal/external trigger)
  - Trigger event (counter start, stop, initialization or count by internal/external trigger)
  - Input capture
  - Output compare
- Supports incremental (quadrature) encoder and hall sensor circuitry for positioning purposes
- Trigger input as external clock or cycle-by-cycle current management

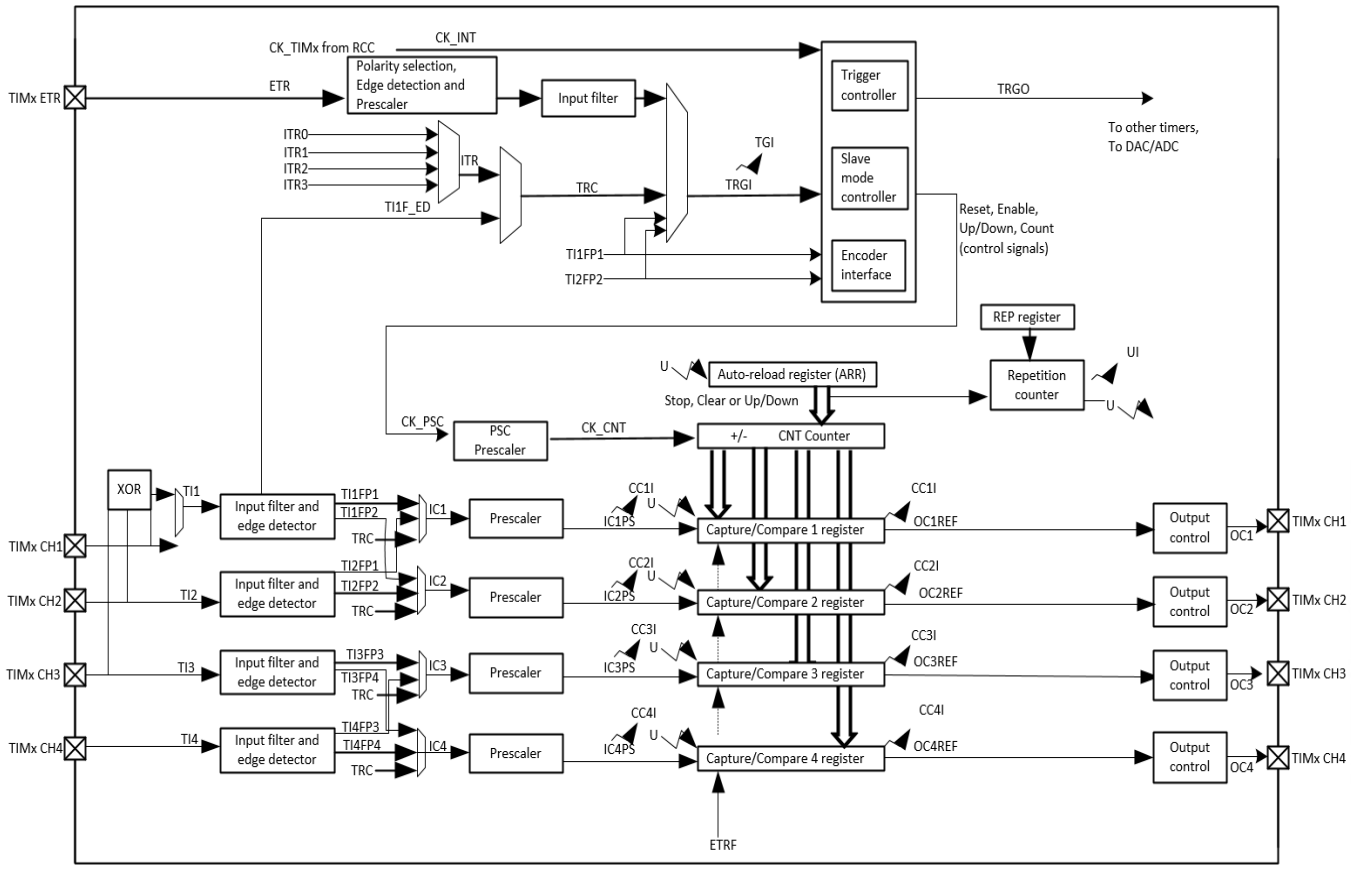


Fig 12-49 General-purpose timer block diagram

Note: Preload register content is transferred to the active register at U (Update) event depending on control bit settings

Event

Interrupt

12.8 TIM2 functional description

12.8.1 Time-base unit

The main block of the programmable general-purpose timer is a 16-bit counter with its related auto-reload register. The counter can count up, down or both up and down. The counter clock is yielded by a prescaler.
The counter, the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running. The time-base unit includes:

- Counter register (TIM2_CNT)
- Prescaler register (TIM2_PSC)
- Auto-Reload register(TIM2_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The content of the preload register is transferred into the shadow register permanently or at each update event (UEV), depending on the auto-reload preload enable bit (ARPE) in TIM2_CR1 register. The update event is generated when the counter reaches the overflow (or underflow when downcounting) and if the UDIS bit equals 0 in the TIM2_CR1 register. It can also be generated by software. The generation of the update event is described in detail for each configuration.
The counter is clocked by the prescaler output CK_CNT, which is enabled only when the counter enable bit (CEN) in TIM2_CR1 register is set. (Refer to the slave mode controller description for details on counter enabling).
Note: The actual counter enable signal CNT_EN is set one clock cycle after CEN.

Prescaler description

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIM2_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.
Figure 12-50 and Figure 12-51 give some examples of the counter behavior when the prescaler ratio is changed on the fly.

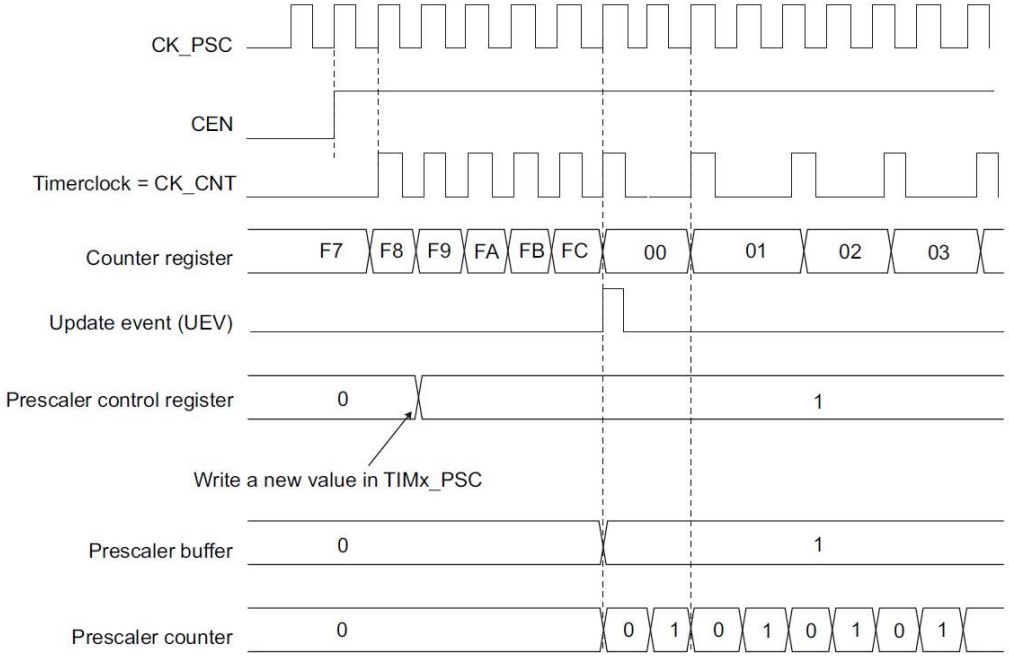


Fig 12-50 Counter timing diagram with prescaler division change from 1 to 2

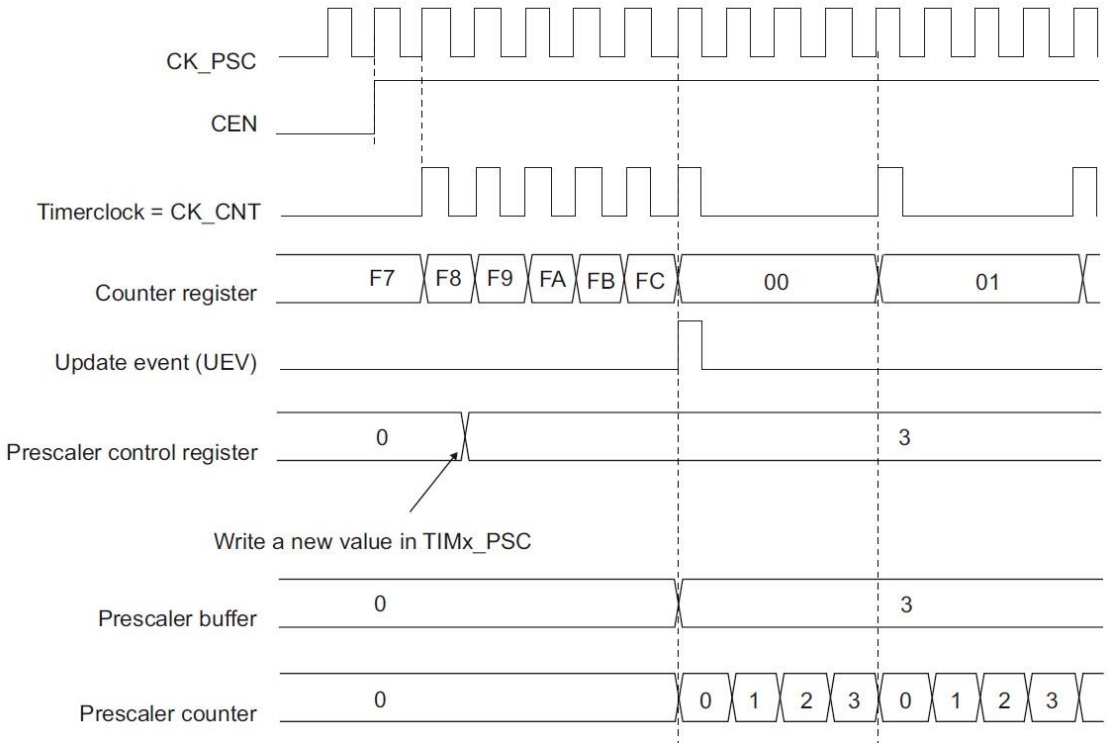


Fig 12-51 Counter timing diagram with prescaler division change from 1 to 4

12.8.2 Counter modes

12.8.2.1 Up-counting mode

In up-counting mode, the counter counts from 0 to the auto-reload value (content of the TIM2_ARR counter), then restarts from 0 and generates a counter overflow event.
An update event can be generated at each counter overflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller).
The update event can be disabled by setting the UDIS bit in the TIM2_CR1 register. This avoids updating the shadow registers while writing new values in the preload registers. No update event occurs until the UDIS bit is written to 0. However, the counter and the prescaler counter are still cleared to 0 (but the prescale factor does not change) when the update event should have occurred. In addition, if the URS bit (update request selection) was set in the TIM2_CR1 register, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.
When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The auto-reload shadow register is updated with the preload value (TIM2_ARR).

The following figures show some examples of the counter behavior for different clock frequencies when TIM2_ARR=0x36.

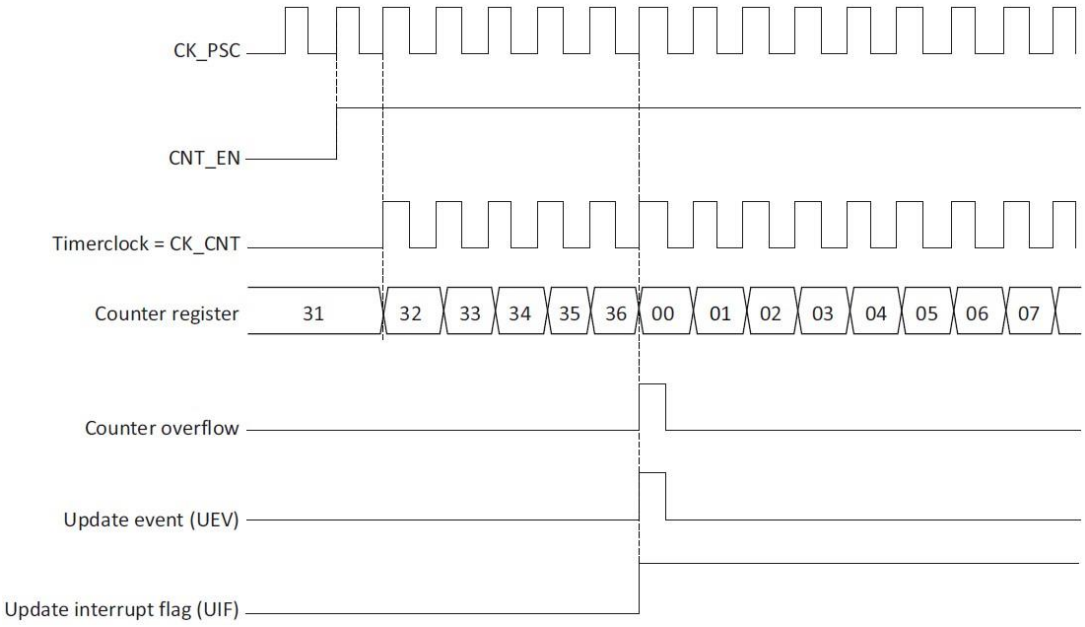


Fig 12-52 Counter timing diagram with internal clock divided by 1

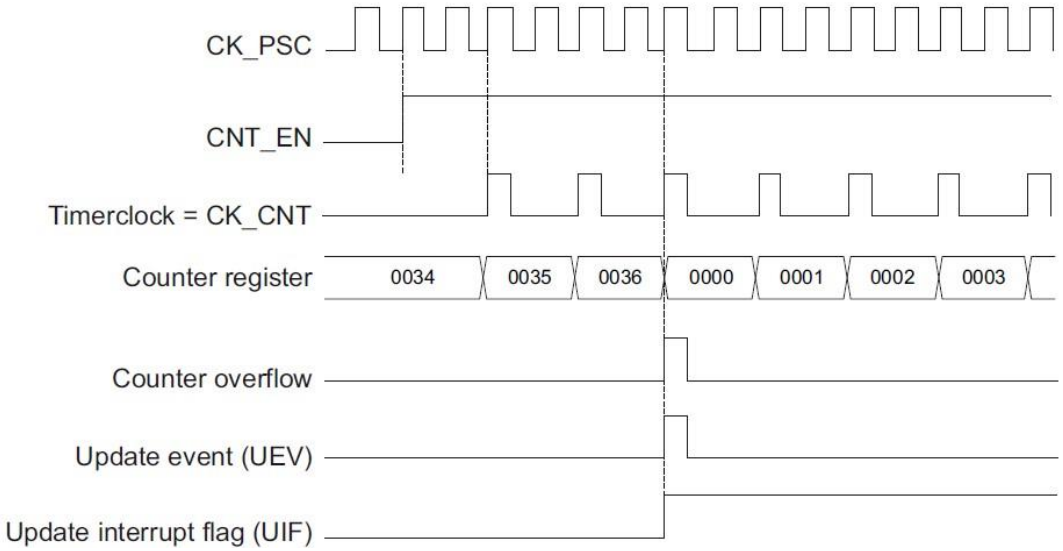


Fig 12-53 Counter timing diagram with internal clock divided by 2

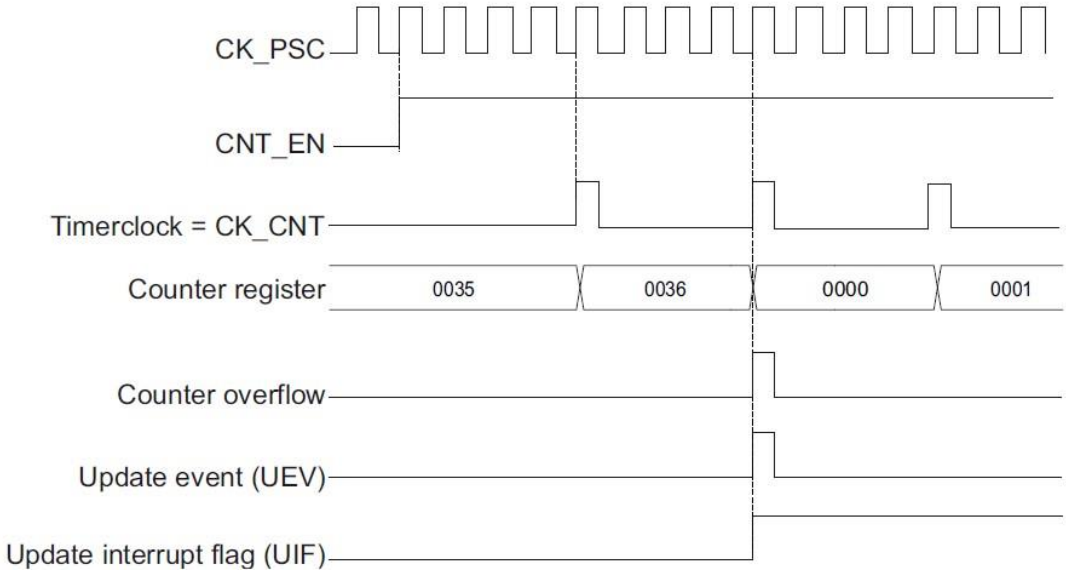


Fig 12-54 Counter timing diagram with internal clock divided by 4

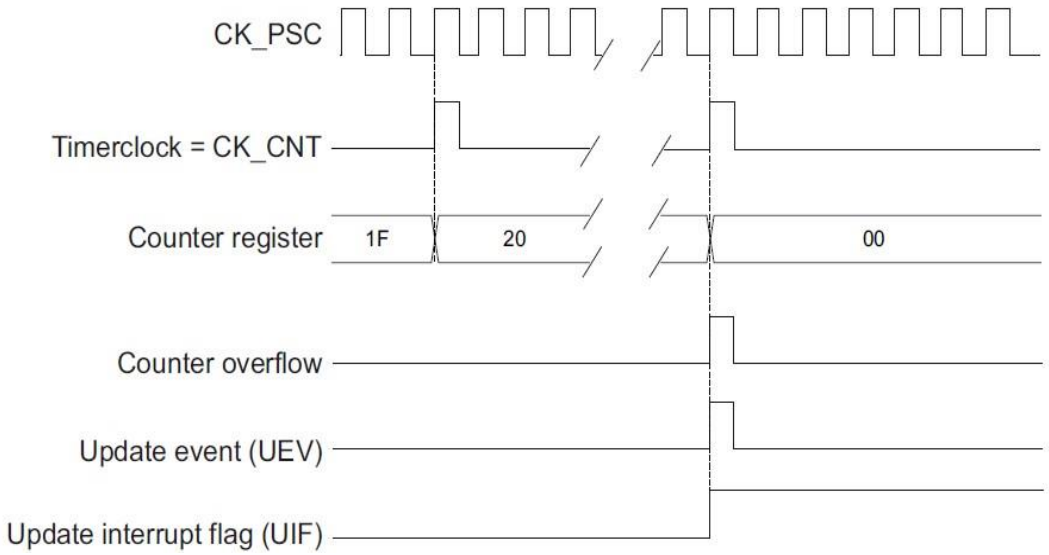


Fig 12-55 Counter timing diagram with internal clock divided by N

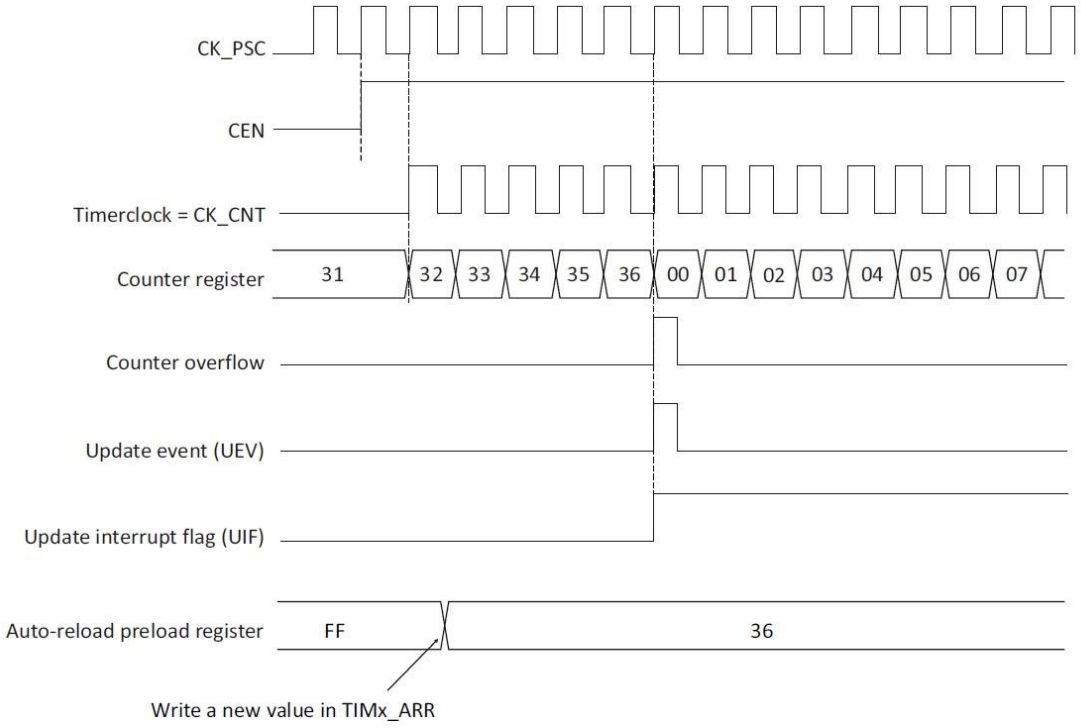


Fig 12-56 Counter timing diagram, update event when ARPE=0 (TIM2_ARR not preloaded)

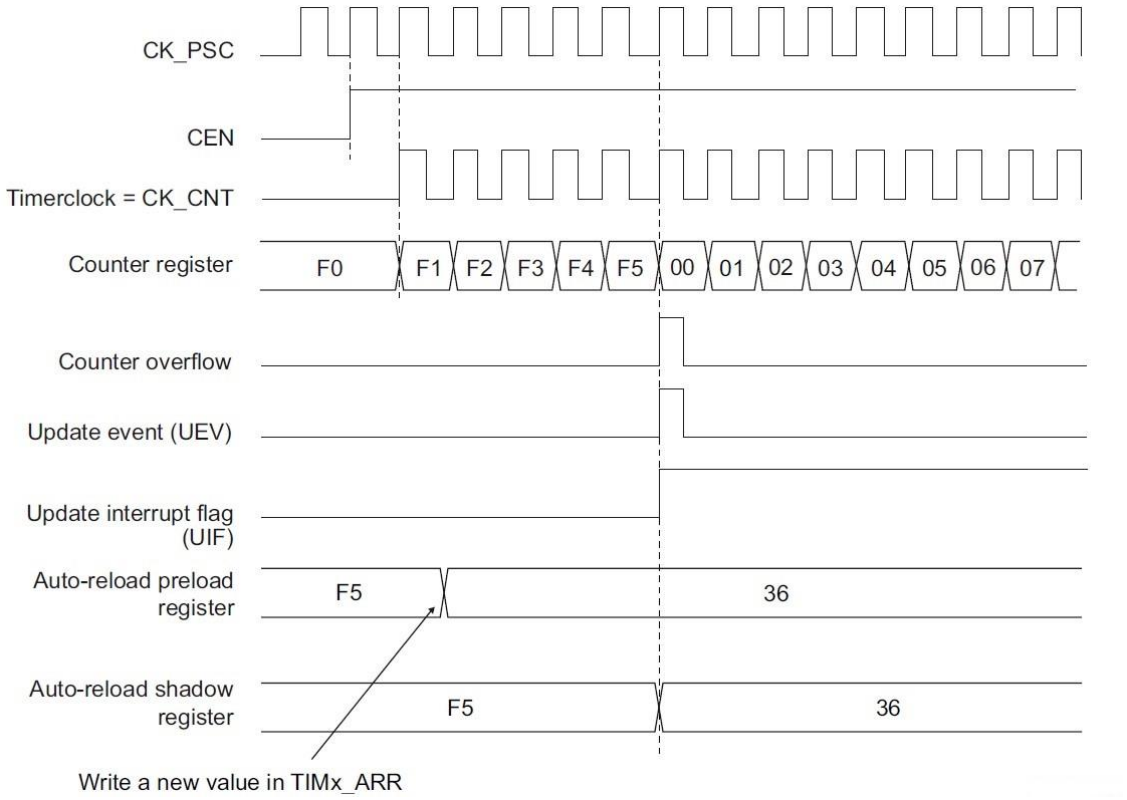


Fig 12-57 Counter timing diagram, update event when ARPE=1 (TIM2_ARR not preloaded)

12.8.2.2 Down-counting mode

In down-counting mode, the counter counts from the auto-reload value (content of the TIM2_ARR value) down to 0, then restarts from the auto-reload value and generates a counter underflow event.
An update event can be generated at each counter underflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller).
The update event UEV can be disabled by setting the UDIS bit in the TIM2_CR1 register. This avoids updating the shadow registers while writing new values in the preload registers. No update event occurs until the UDIS bit is written to 0. However, the counter restarts from the current auto-reload value, whereas the prescaler's counter restarts from 0 (but the prescale factor does not change).
In addition, if the URS bit (update request selection) was set in the TIM2_CR1 register, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.
When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The active auto-reload register is updated with the preload value (content of the TIM2_ARR register). Note: The auto-reload is updated before the counter is reloaded, so that the next period is the expected one.

The following figures show some examples of the counter behavior for different clock frequencies when TIM2_ARR=0x36.

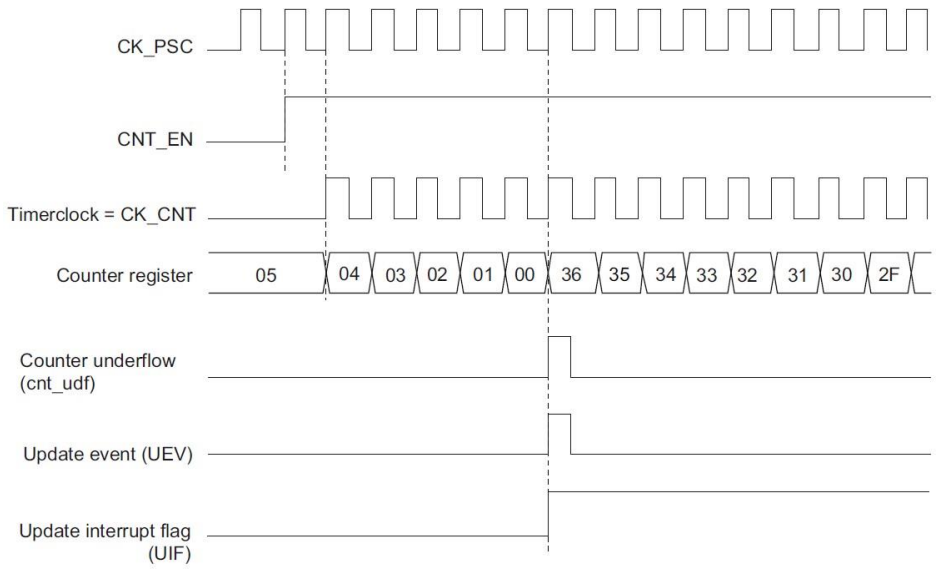


Fig 12-58 Counter timing diagram with internal clock divided by 1

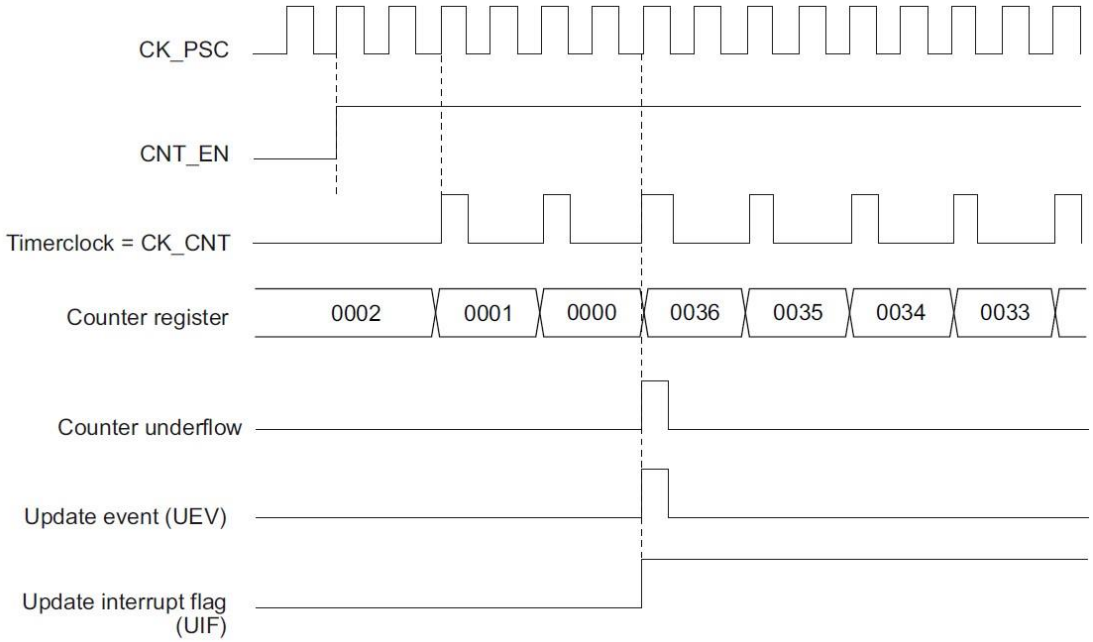


Fig 12-59 Counter timing diagram with internal clock divided by 2

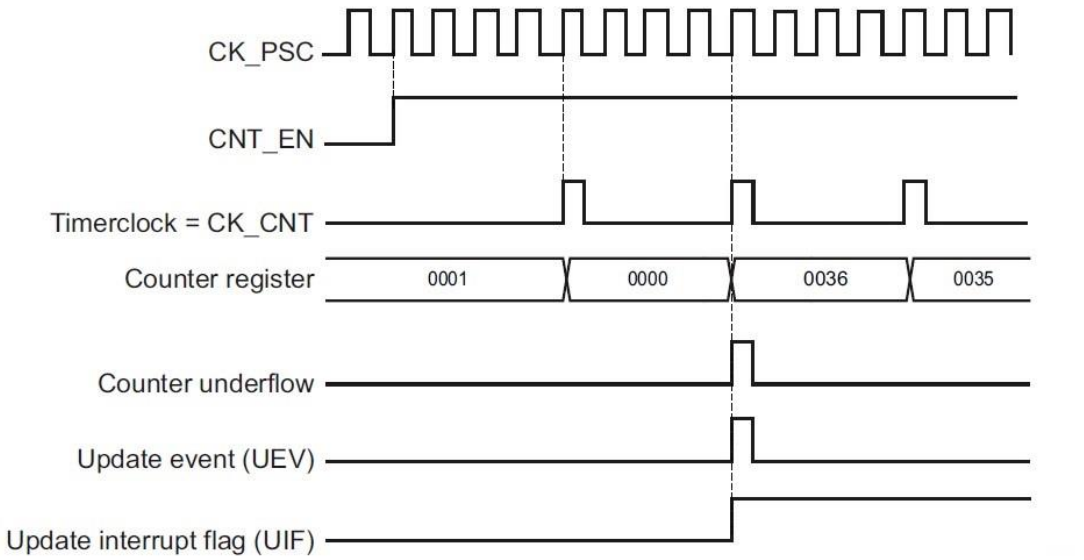


Fig 12-60 Counter timing diagram with internal clock divided by 4

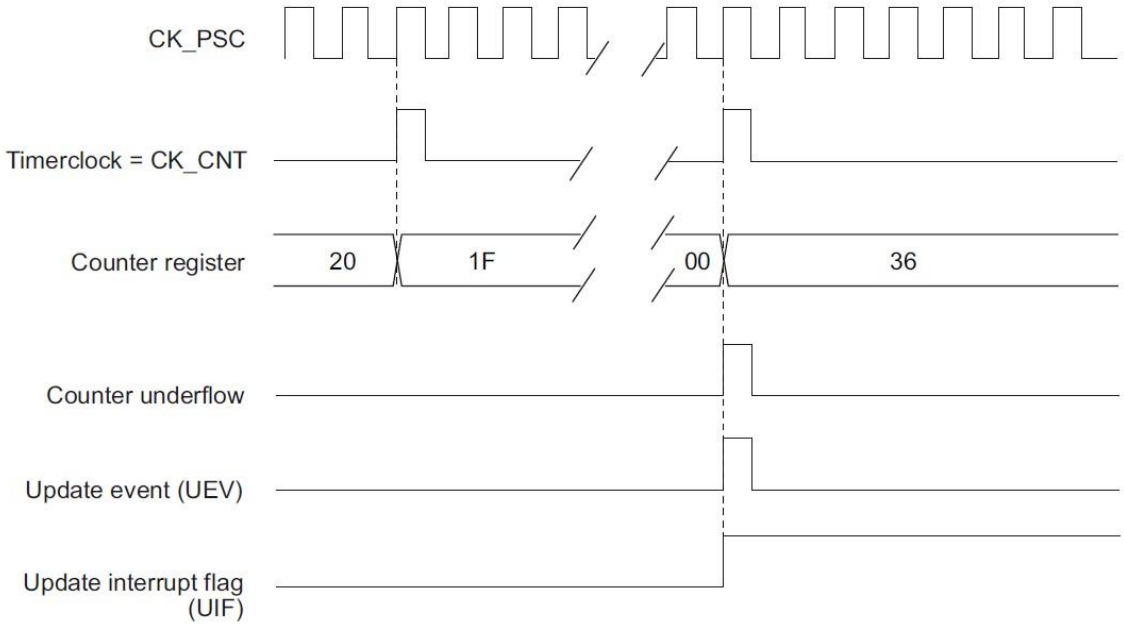


Fig 12-61 Counter timing diagram with internal clock divided by N

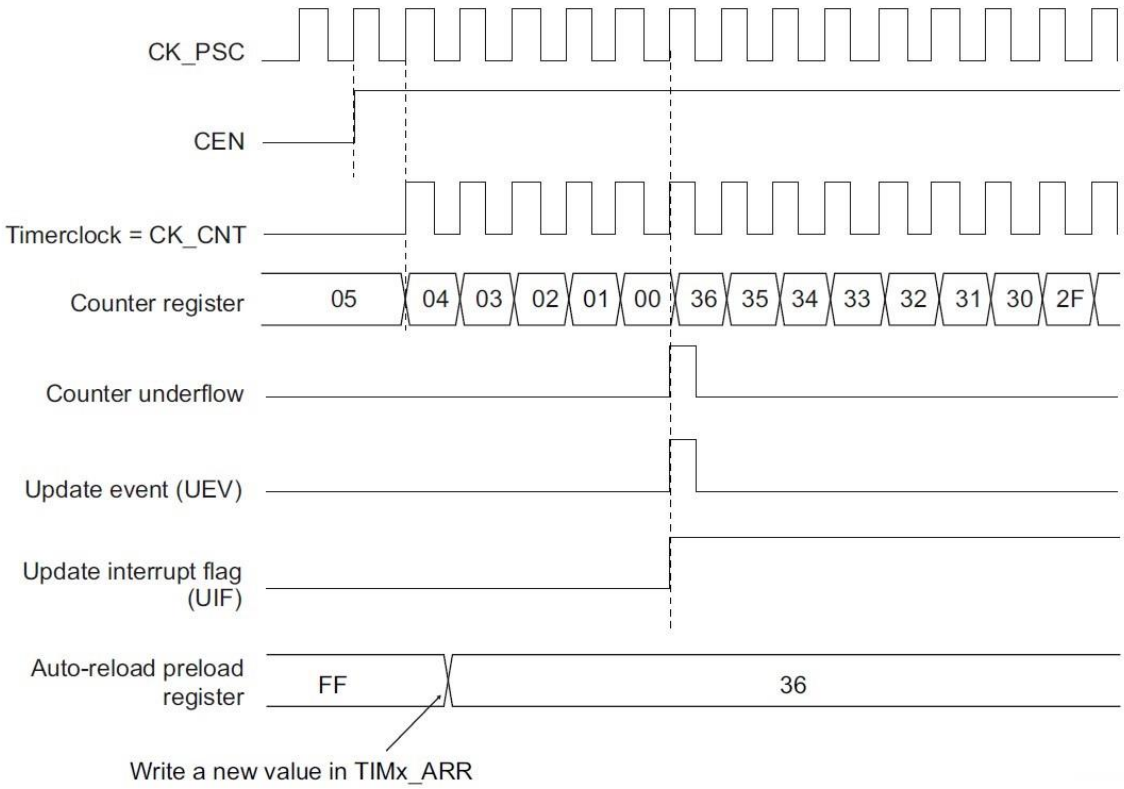


Fig 12-62 Counter timing diagram, update event when repetition counter is not used

12.8.2.3 Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIM2_ARR register) - 1, generates a counter overflow event, then counts down to 1 and generates a counter underflow event. Then it restarts counting from 0.

In this mode, the direction bit DIR in the TIM2_CR1 register cannot be written. It is updated by hardware and gives the current direction of the counter. The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIM2_EGR register (by software or by using the slave mode controller). In this case, the counter restarts counting from 0 and the prescaler counter restarts from 0.

The update event UEV can be disabled by setting the UDIS bit in the TIM2_CR1 register. This avoids updating the shadow registers while writing new values in the preload registers. No update event occurs until the UDIS bit is written to 0. However, the counter continues counting up and down, based on the current auto-reload value.

In addition, if the URS bit (update request selection) was set in the TIM2_CR1 register, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt is sent). This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (UIF bit in TIM2_SR register) is set (depending on the URS bit).

- The buffer of the prescaler is reloaded with the preload value (content of the TIM2_PSC register).
- The active auto-reload register is updated with the preload value (content of the TIM2_ARR register).

Note: If the update source is a counter overflow, the auto-reload is updated before the counter is reloaded, so that the next period is the expected one (the counter is loaded with the new value). Here are some examples of the counter behavior for different clock frequencies:

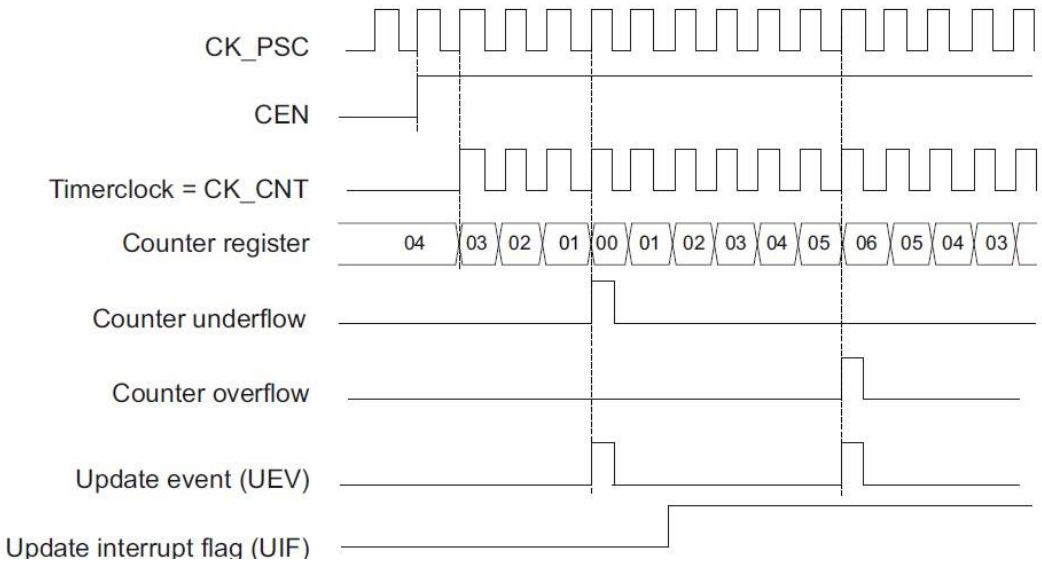


Fig 12-63 Counter timing diagram, internal clock divided by 1, TIM2_ARR=0x6

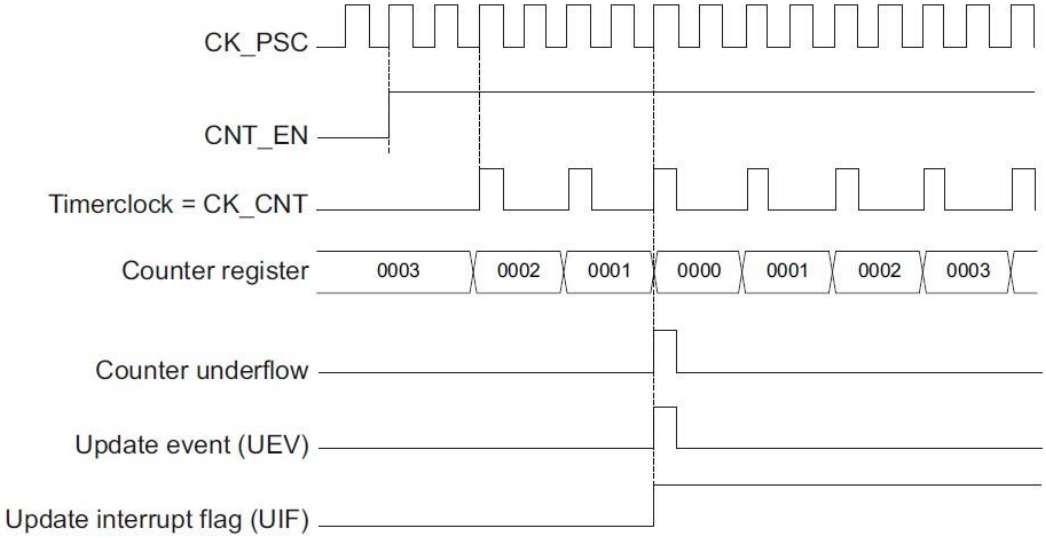


Fig 12-64 Counter timing diagram, internal clock divided by 2

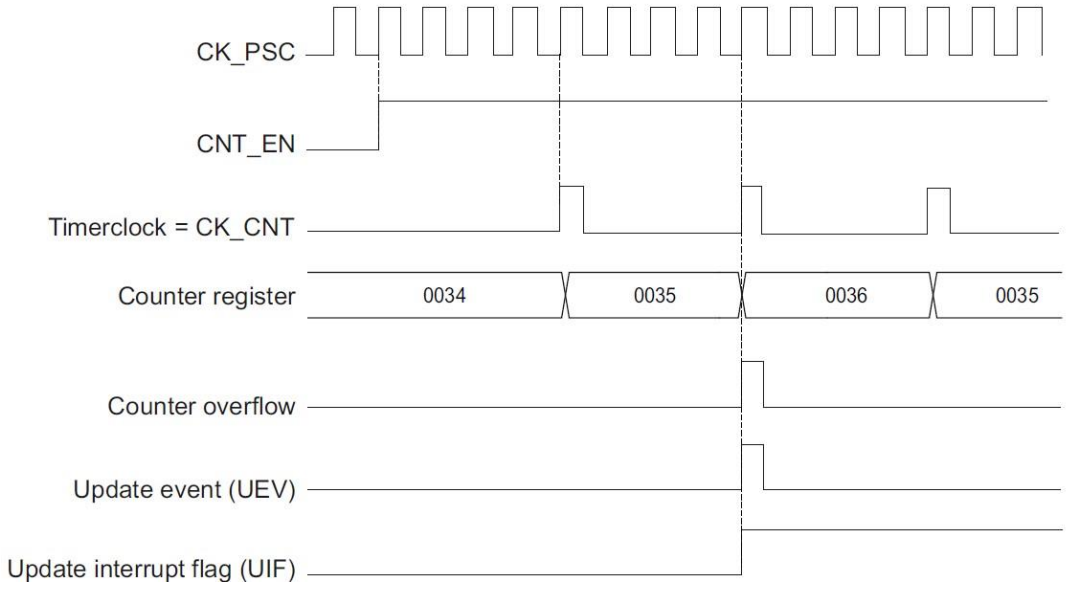


Fig 12-65 Counter timing diagram, internal clock divided by 4, TIM2_ARR=0x36

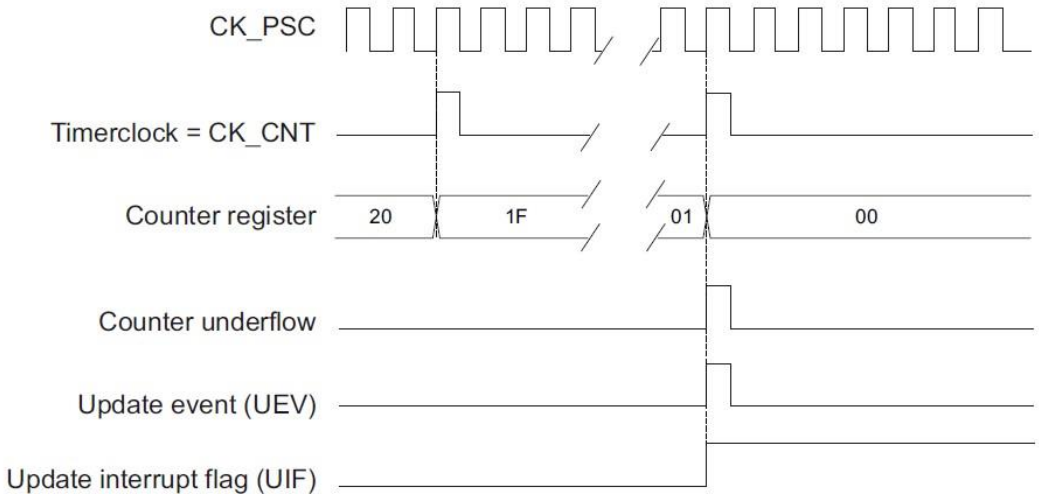


Fig 12-66 Counter timing diagram, internal clock divided by N

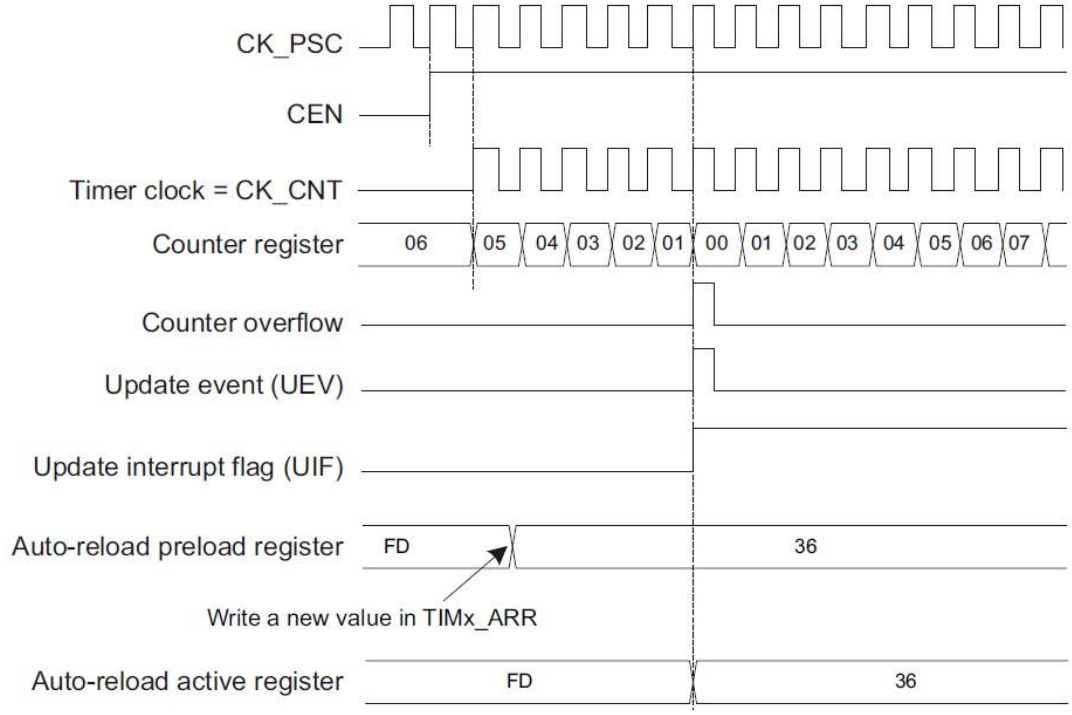


Fig 12-67 Counter timing diagram, update event with ARPE=1 (counter underflow)

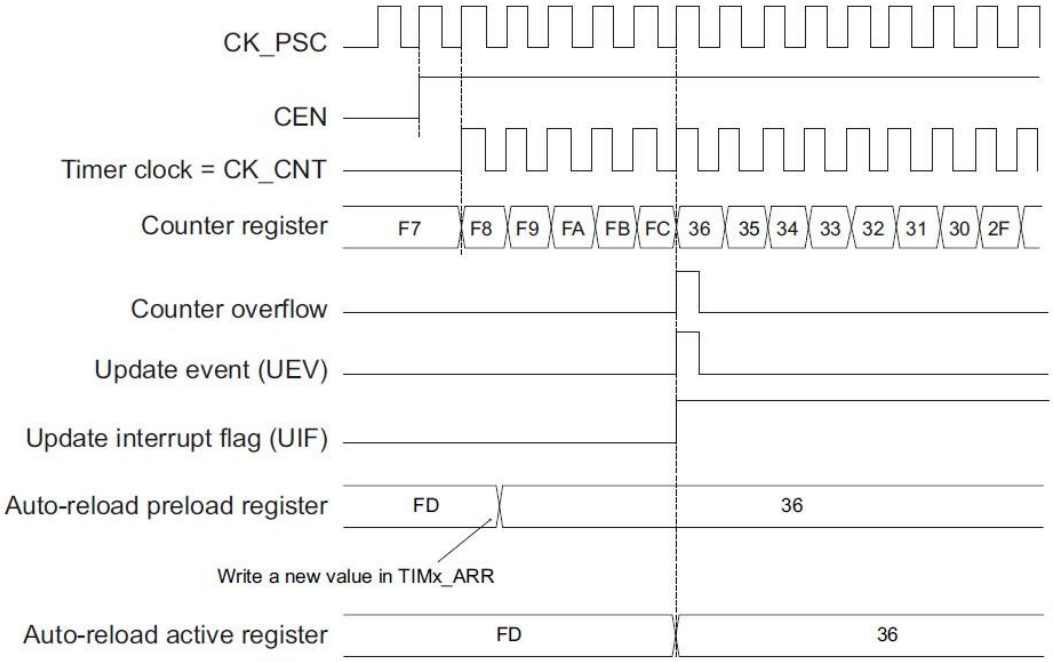


Fig 12-68 Counter timing diagram, update event with ARPE=1 (counter overflow)

12.8.3 Clock selection

The counter clock can be provided by the following clock sources:

- Internal clock (CK_INT)
- External clock mode 1: external input pin
- External clock mode 2: external trigger input (ETR)
  Internal trigger inputs (ITRx): using one timer as prescaler for another timer, for example, Timer 1 can be configured to act as a prescaler for Timer 2.

12.8.3.1 Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000 in TIM2_SMCR register), then the CEN, DIR (in TIM2_CR1 register) and UG bits (in TIM2_EGR register) are the actual control bits and can be changed only by software (except UG which remains cleared automatically). As soon as the CEN bit is written to 1, the prescaler is clocked by the internal clock CK_INT.

The figure below shows the behavior of the control circuit and the up-counter in normal mode, without prescaler.

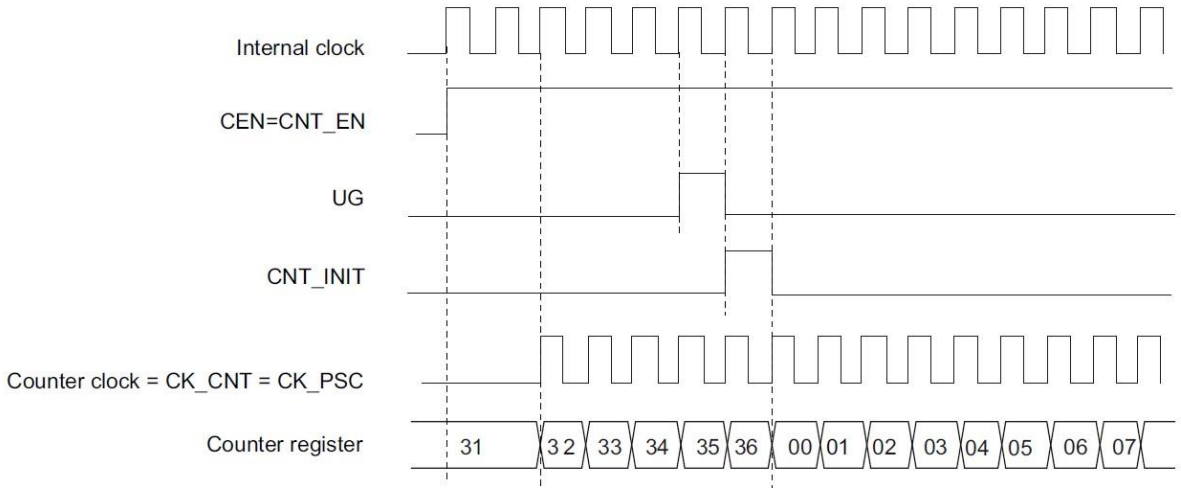


Fig 12-69 Control circuit in normal mode, internal clock divided by

12.8.3.2 External clock source mode 1

This mode is selected when SMS=111 in the TIM2_SMCR register. The counter can count at each rising or falling edge on a selected input.

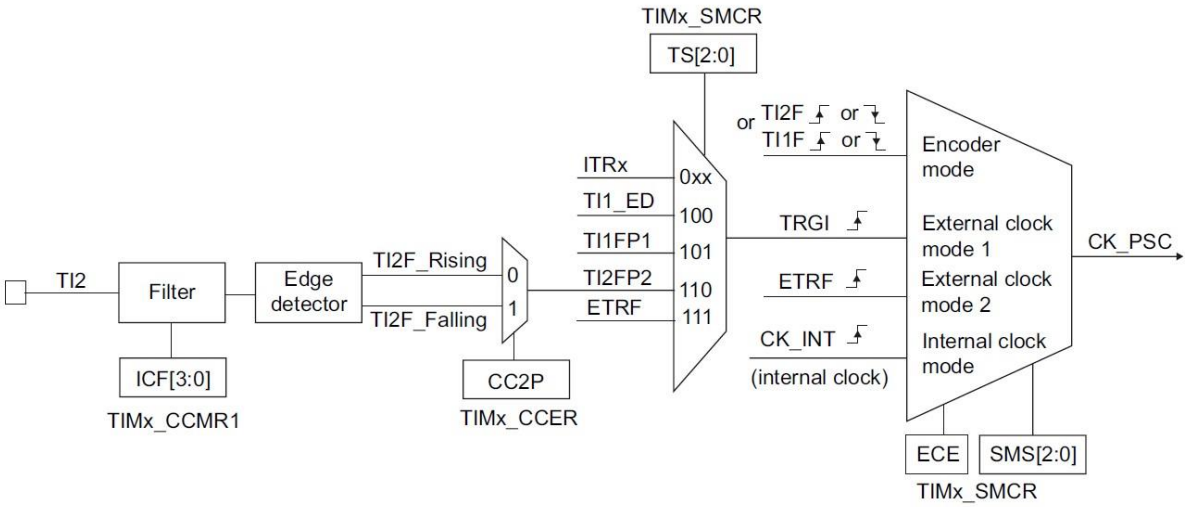


Fig 12-70 Example of TI2 external clock connection

For example, to configure the up-counter to count in response to a rising edge on the TI2 input, use the following steps:

1. Configure TIM2_CCMR1 register CC2S='01' to configure channel 2 to detect rising edge on TI2 input.
2. Configure IC2F[3:0] in TIM2_CCMR1 register to select the input filter bandwidth (keep IC2F=0000 if no filter is needed).

Note: The capture prescaler is not used for triggering, so it does not need to be configured.

3. Configure TIM2_CCER register CC2P='0' to select rising edge polarity.
4. Configure SMS='111' in TIM2_SMCR register to select timer external clock mode 1.
5. Configure TS='110' in TIM2_SMCR register to select TI2 as trigger input source.
6. Set CEN='1' in TIM2_CR1 register to enable the counter.

When a rising edge occurs on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge on TI2 and the actual clock of the counter depends on the resynchronization circuit on TI2 input.

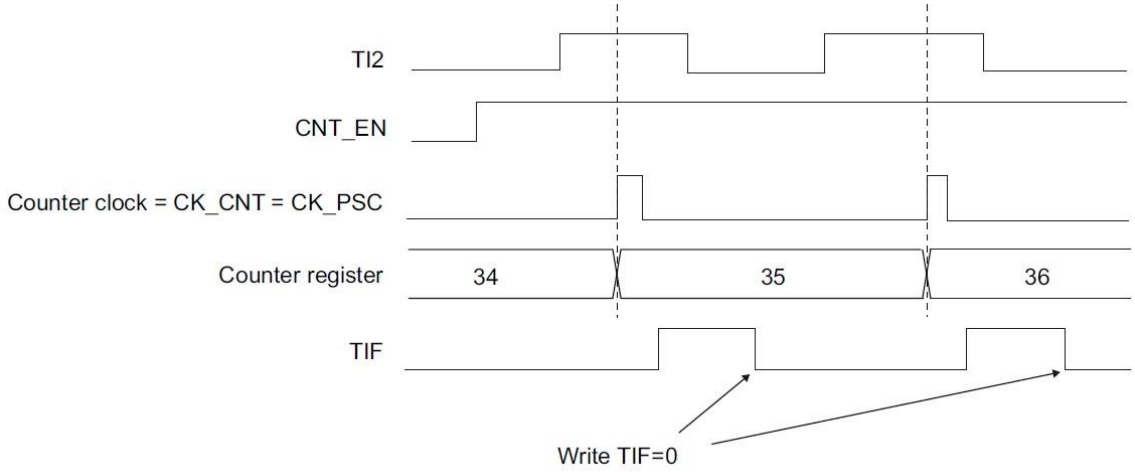


Fig 12-71 Control circuit in external clock mode 1

12.8.3.3 External clock source mode 2

This mode is selected by setting ECE=1 in the TIM2_SMCR register. The counter can count at each rising or falling edge on the external trigger ETR. The figure below shows the block diagram of the external trigger input:

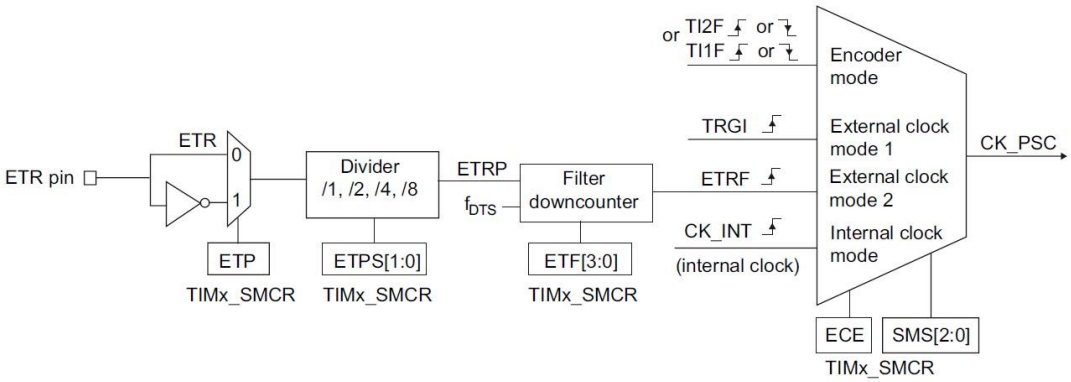


Fig 12-72 External trigger input block diagram

For example, to configure the up-counter to count every 2 rising edges on ETR, use the following steps:

1. As no filter is needed in this example, set ETF[3:0]=0000 in the TIM2_SMCR register.
2. Set the prescaler by setting ETPS[1:0]=01 in the TIM2_SMCR register.
3. Select rising edge detection on ETR by setting ETP=0 in the TIM2_SMCR register.
4. Enable external clock mode 2 by setting ECE=1 in the TIM2_SMCR register.
5. Enable the counter by setting CEN=1 in the TIM2_CR1 register.

The counter counts once every 2 ETR rising edges.

The delay between the rising edge on ETR and the actual clock of the counter depends on the resynchronization circuit on the ETRP signal.

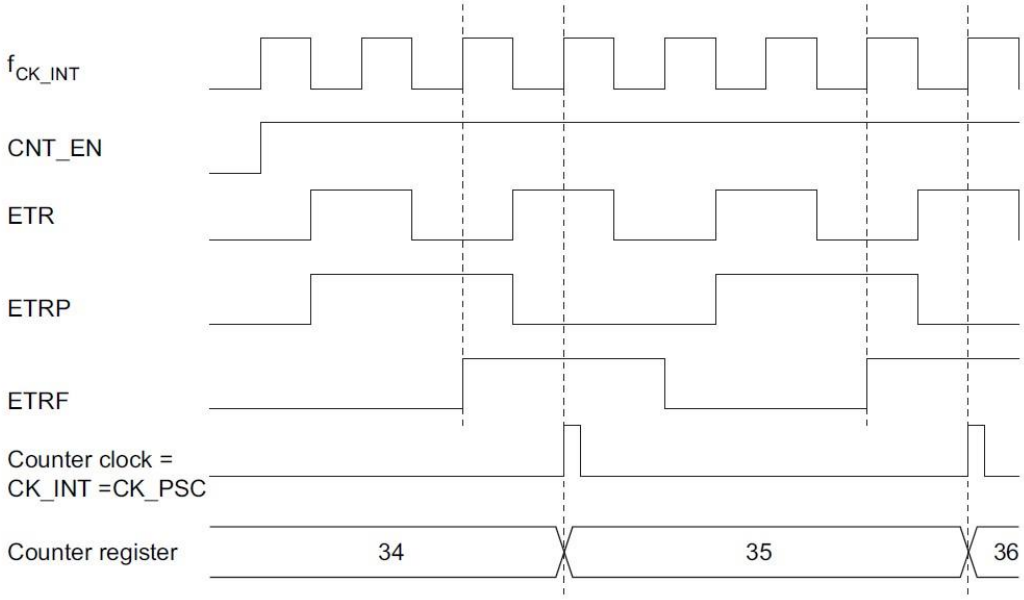


Fig 12-73 Control circuit in external clock mode 2

12.8.4 Capture/compare channels

Each capture/compare channel is built around a capture/compare register (including a shadow register), including the input stage (digital filtering, multiplexing, and prescaler), and the output stage (comparator and output control).

The input capture channel is configured by SYSCFG_TIM2_CON_SEL.

The following figures provide an overview of a capture/compare channel.

- The input stage samples the corresponding TIx input signal and generates a filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx), which can be used as a trigger input for the slave mode controller or as a capture command. The signal passes through the prescaler into the capture register (ICxPS).

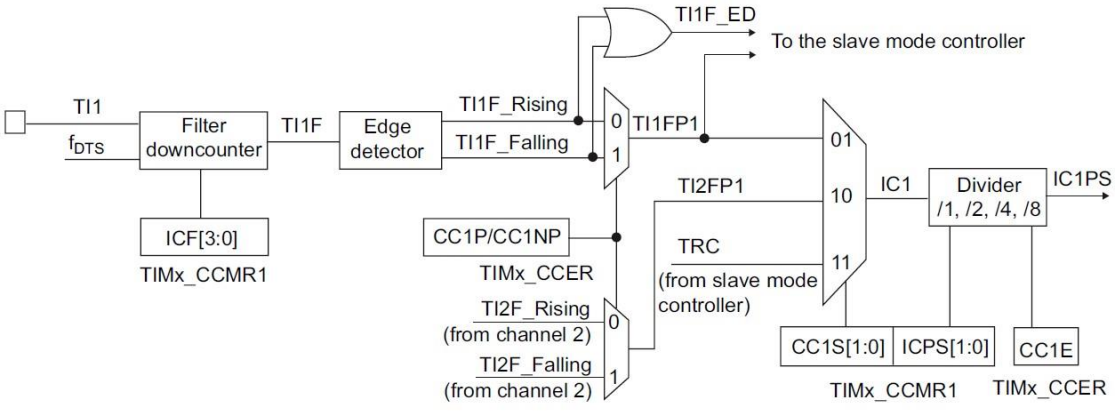


Fig 12-74 Capture/compare channel (e.g., Channel 1 input stage)

The output stage generates an intermediate waveform OCxRef (active high) as a reference, and the end of the chain determines the polarity of the final output signal.

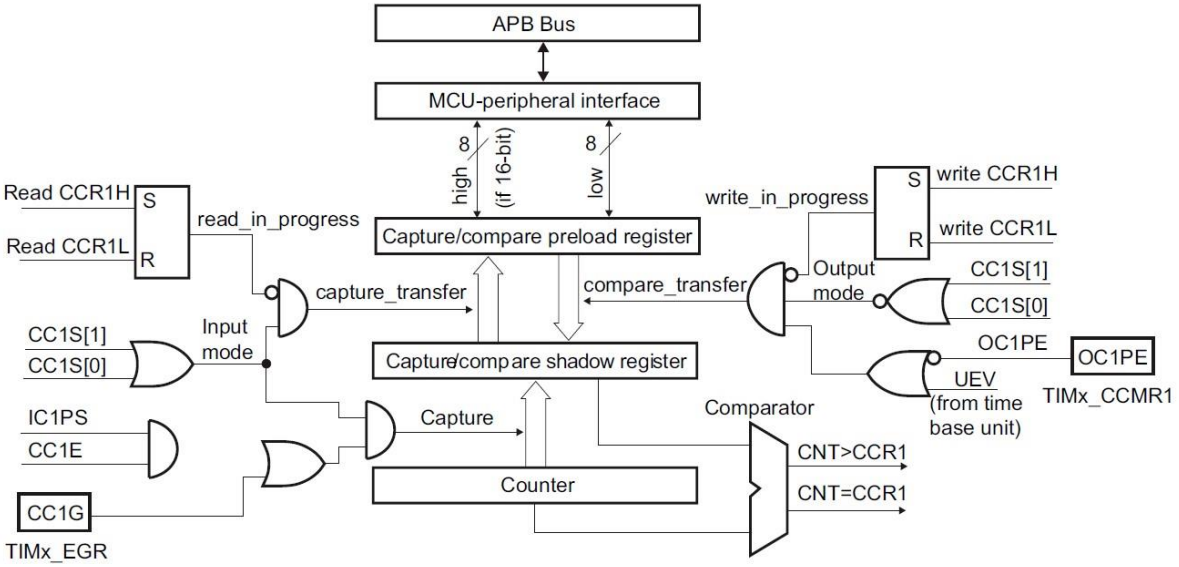


Fig 12-75 Capture/compare channel 1 main circuit

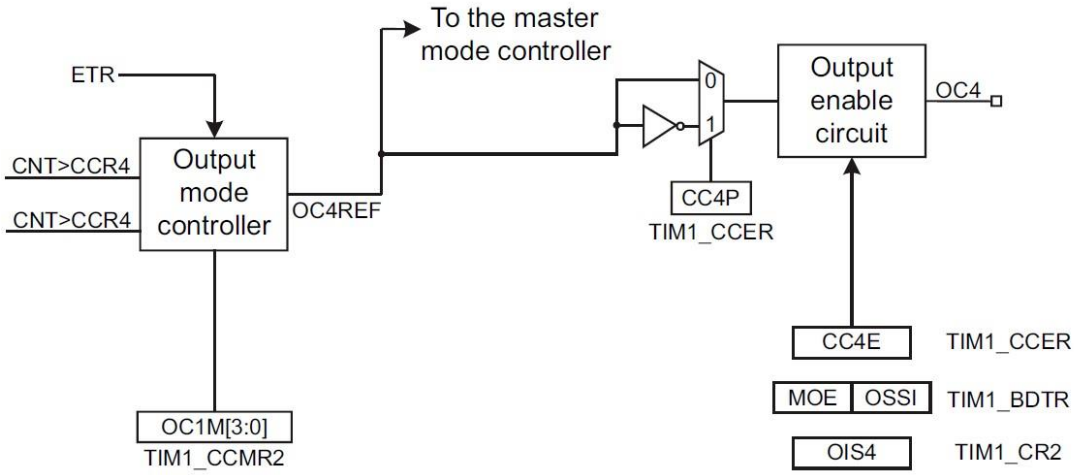


Fig 12-76 Output stage of capture/compare channel (Channel 1)

The capture/compare module consists of a preload register and a shadow register. Read and write operations access only the preload register. In capture mode, captures occur in the shadow register and are then copied into the preload register. In compare mode, the content of the preload register is copied into the shadow register, and then the content of the shadow register is compared with the counter.

12.8.5 Input capture mode

- In Input capture mode, the Capture/Compare register (TIM2_CCRx) is used to latch the value of the counter after a transition detected on the corresponding ICx signal. When a capture event occurs, the corresponding CCxIF flag (TIM2_SR register) is set to '1' and an interrupt is generated if enabled. If the capture event occurs when CCxIF flag was already high, then the overcapture flag CCxOF (TIM2_SR register) is set to '1'. CCxIF can be cleared by writing it to 0 or by reading the captured data stored in the TIM2_CCRx register. CCxOF is cleared by writing it to 0.

  The following example shows how to capture the counter value in TIM2_CCR1 when a rising edge on TI1 input occurs:

  - Select the active input: TIM2_CCR1 must be connected to the TI1 input, so write CC1S=01 in the TIM2_CCMR1 register. As soon as CC1S is different from '00', the channel is configured as input and the TIM2_CCR1 register becomes read-only. ◦
  - Program the input filter duration you need with respect to the signal capability (bits ICxF in the TIM2_CCMRx register if the input is TIx). Let's imagine that, if the input signal is unstable during at most 5 internal clock cycles, we must program a filter duration longer than these 5 clock cycles. We can validate a transition on TI1 when 8 consecutive samples (at fDTS frequency) are detected (write IC1F=0011 in the TIM2_CCMR1 register).
  - Select the edge of the active transition on the TI1 channel by writing CC1P=0 (rising edge) in the TIM2_CCER register.
  - Program the input prescaler. In our example, we wish the capture to be performed at each valid transition, so the prescaler is disabled (write IC1PS=00 in the TIM2_CCMR1 register).
  - Enable capture from the counter into the capture register by setting the CC1E bit in the TIM2_CCER register (CC1E=1).
  - If needed, enable the related interrupt request by setting the CC1IE bit in the TIM2_IER register. When an input capture occurs:
  - The value of the counter is transferred to the TIM2_CCR1 register.
  - The CC1IF flag is set (interrupt flag). CC1OF is also set to '1' if at least two consecutive captures occurred whereas CC1IF was not cleared. ◦
  - 如An interrupt is generated depending on the CC1IE bit.

In order to handle the overcapture, it is recommended to read the data before reading the overcapture flag. This is to avoid missing an overcapture information which could happen after reading the overcapture flag and before reading the data.

Note: The IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIM2_EGR register.

12.8.6 PWM input mode

This mode is a particular case of input capture mode. The procedure is the same except:

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- Selection of the TIxFP signal which is selected as trigger input and the slave mode controller is configured in reset mode.

For example, you can measure the period (in TIM2_CCR1 register) and the duty cycle (in TIM2_CCR2 register) of the PWM applied on TI1 using the following procedure (depending on CK_INT frequency and prescaler value):

- Select the active input for TIM2_CCR1: write the CC1S bits to 01 in the TIM2_CCMR1 register (select TI1).
- Select the active polarity for TI1FP1 (used both for capture in TIM2_CCR1 and counter clear): write the CC1P bit to 0 (active on rising edge).
- Select the active input for TIM2_CCR2: write the CC2S bits to 10 in the TIM2_CCMR1 register (select TI1).
- Select the active polarity for TI1FP2 (used for capture in TIM2_CCR2): write the CC2P bit to 1 (active on falling edge).
- Select the valid trigger input: write the TS bits to 101 in the TIM2_SMCR register (select TI1FP1).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIM2_SMCR register.
- Enable the captures: write the CC1E and CC2E bits to 1 in the TIM2_CCER register.

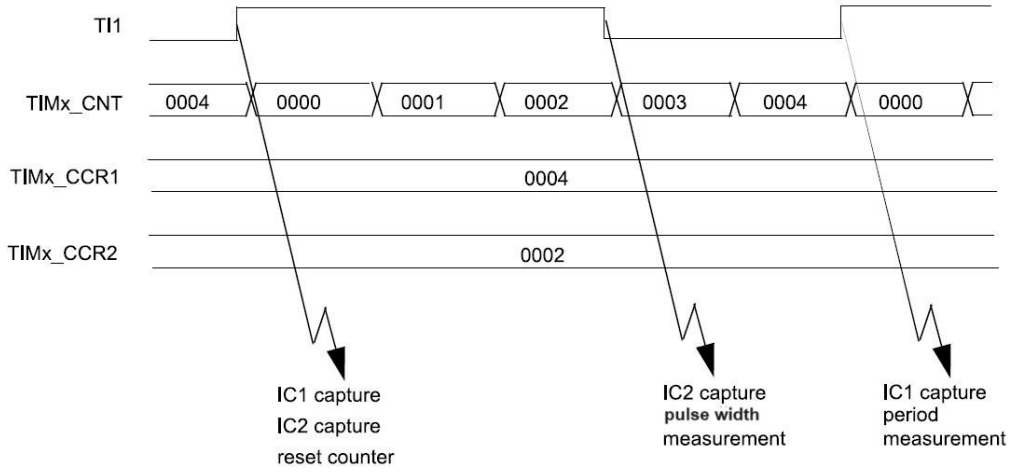


Fig 12-77 PWM input mode timing

Note: Since only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can be used only with the TIM2_CH1 / TIM2_CH2 signals.

12.8.7 Forced output mode

In output mode (CCxS bits = 00 in the TIM2_CCMRx register), each output compare signal (OCxREF and then OCx) can be forced to active or inactive level directly by software, independently of any comparison between the output compare register and the counter.

To force an output compare signal (OCxREF/OCx) to its active level, you just write 101 in the OCxM bits in the corresponding TIM2_CCMRx register. Thus OCxREF is forced high (OCxREF is always active high) and OCx gets opposite value to CCxP polarity bit.

For example: CCxP=0 (OCx active high) => OCx is forced to high level.

The OCxREF signal can be forced low by writing the OCxM bits to 100 in the TIM2_CCMRx register. However, the comparison between the TIM2_CCRx shadow register and the counter is still performed and allows the flag to be set. Interrupt requests can be sent accordingly. This is described in the Output compare mode section below.

12.8.8 Output compare mode

This function is used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the capture/compare register and the counter, the output compare function:

- Assigns the corresponding output pin to the programmable value defined by the output compare mode (OCxM bits in the TIM2_CCMRx register) and the output polarity (CCxP bit in the TIM2_CCER register). The output pin can keep its level (OCxM=000), be set active (OCxM=001), be set inactive (OCxM=010) or can toggle (OCxM=011) on match.
- Sets a flag in the interrupt status register (CCxIF bit in the TIM2_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIM2_IER register).

Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIM2_IER register).

The timing resolution is one count of the counter. Output compare mode can also be used to output a single pulse (in One Pulse mode).

Procedure:

1. Select the counter clock (internal, external, prescaler).
2. Write the desired data in the TIM2_ARR and TIM2_CCRx registers.
3. Set the CCxIE bit if an interrupt request is to be generated.
4. Select the output mode. For example: write OCxM=011 to toggle OCx output pin when CNT matches CCRx, write OCxPE=0 to disable preload register, write CCxP=0 to select active high polarity, and write CCxE=1 to enable the output.
5. Enable the counter by setting the CEN bit in the TIM2_CR1 register.

The TIM2_CCRx register can be updated at any time by software to control the output waveform, provided that the preload register is not enabled (OCxPE='0', else TIM2_CCRx shadow register is updated only at the next update event). The figure below shows an example.

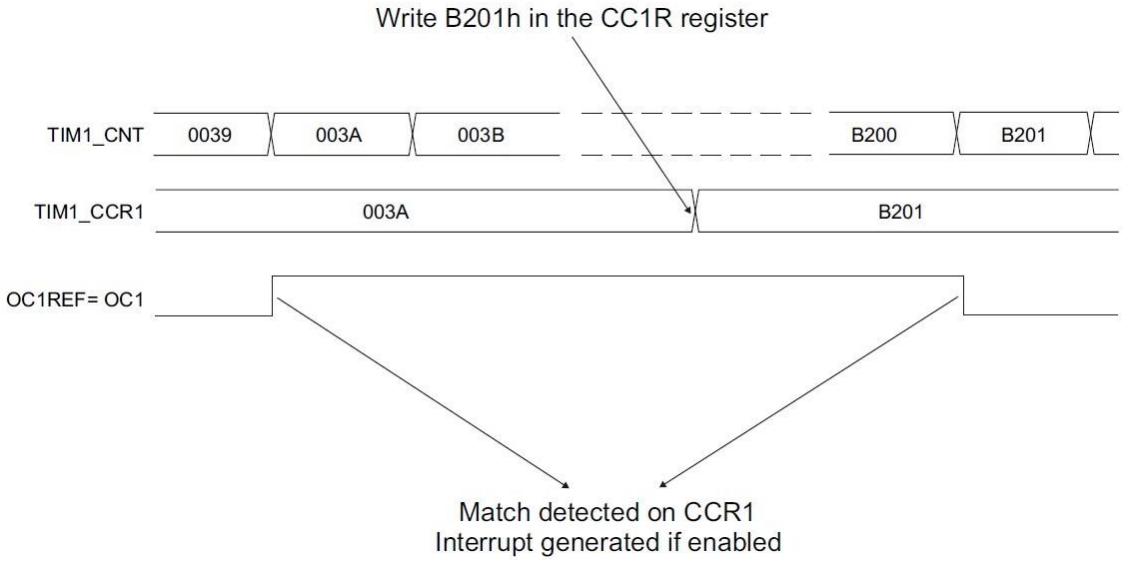


Fig 12-78 Output compare mode, toggle on OC1

12.8.9 PWM mode

Pulse width modulation mode allows you to generate a signal with a frequency determined by the value of the TIM2_ARR register and a duty cycle determined by the value of the TIM2_CCRx register.

The PWM mode can be selected independently on each channel (one PWM per OCx output) by writing '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bits in the TIM2_CCMRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIM2_CCMRx register, and eventually the auto-reload preload register (in up-counting or center-aligned modes) by setting the ARPE bit in the TIM2_CR1 register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, you have to initialize all the registers by setting the UG bit in the TIM2_EGR register. The OCx polarity can be programmed using the CCxP bit in the TIM2_CCER register. It can be active high or active low. The OCx output is enabled by the CCxE bit in the TIM2_CCER register. Refer to the TIM2_CCER register description for details.

In PWM mode (1 or 2), TIM2_CNT and TIM2_CCRx are always compared to determine whether TIM2_CCRx <= TIM2_CNT or TIM2_CNT <= TIM2_CCRx (depending on the direction of the counter). However, to be compliant with the OCREF_CLR functionality (OCxREF can be cleared by an external event on the ETR signal until the next PWM period), the OCxREF signal is asserted only:

- When the result of the comparison changes, or
- When the output compare mode (OCxM bits in TIM2_CCMRx register) switches from the "frozen" configuration (no comparison, OCxM='000') to one of the PWM modes (OCxM='110' or '111').

This allows the PWM output to be forced by software during the operation.

The timer is able to generate PWM in edge-aligned mode or center-aligned mode depending on the CMS bits in the TIM2_CR1 register.

12.8.9.1 PWM edge-aligned mode

12.8.9.1.1 Up-cointing configuration

Up-counting is active when the DIR bit in the TIM2_CR1 register is low.

Refer to the following example of PWM mode 1. The reference PWM signal OCxREF is high as long as TIM2_CNT < TIM2_CCRx, otherwise it becomes low. If the compare value in TIM2_CCRx is greater than the auto-reload value (in TIM2_ARR), then OCxREF is held at '1'. If the compare value is 0, OCxREF is held at '0'. The following figure shows an example of edge-aligned PWM waveforms with TIM2_ARR=8.

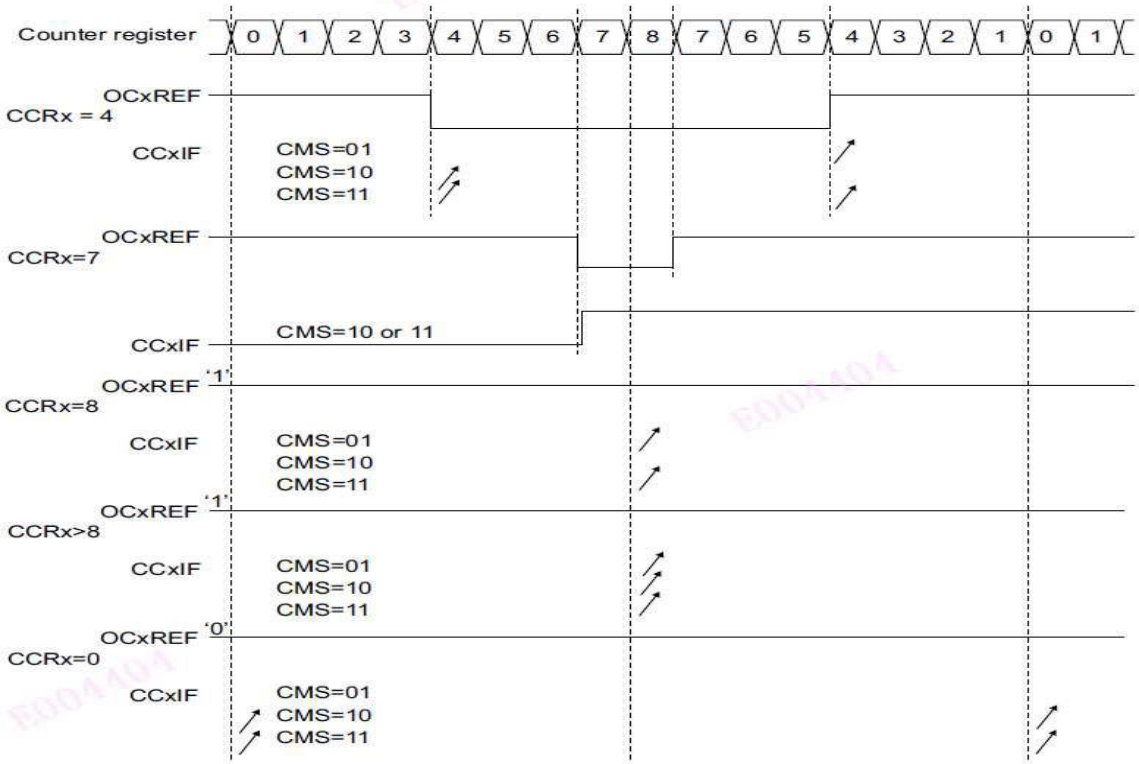


Fig 12-79 Edge-aligned PWM waveforms (ARR=8)

12.8.9.1.2 Down-counting configuration

Down-counting is active when the DIR bit in the TIM2_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low as long as TIM2_CNT > TIM2_CCRx, otherwise it becomes high. If the compare value in TIM2_CCRx is greater than the auto-reload value in TIM2_ARR, then OCxREF is held at '1'. 0% PWM is not possible in this mode.

12.8.9.2 PWM center-aligned mode

Center-aligned mode is active when the CMS bits in TIM2_CR1 register are different from '00' (all other configurations have the same effect on the OCxREF/OCx signals). Center-aligned mode is active when the CMS bits in TIM2_CR1 register are different from '00' (all other configurations have the same effect on the OCxREF/OCx signals). The direction bit (DIR) in the TIM2_CR1 register is updated by hardware and must not be changed by software

The figure below shows some examples of center-aligned PWM waveforms:

- TIM2_ARR=8
- PWM mode 1
- CMS=01 in TIM2_CR1 register (compare flag set when counter counts down in center-aligned mode 1)

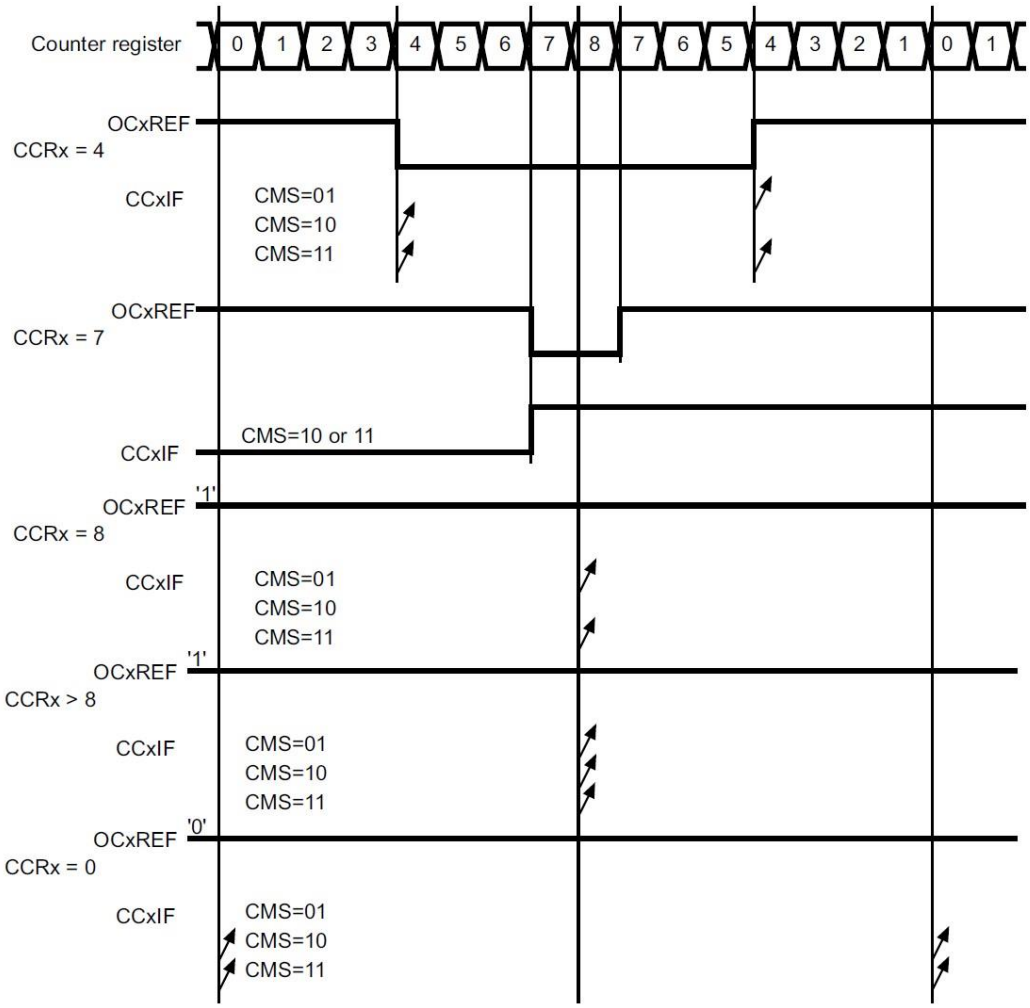


Fig 12-80 Center-aligned PWM waveforms (ARR=8)

12.8.9.2.1 Hints on using center-aligned mode

When starting center-aligned mode, the current up/down configuration is used. It means that the counter counts up or down depending on the value written in the DIR bit in the TIM2_CR1 register. Moreover, the DIR and CMS bits must not be changed at the same time by the software.
Writing to the counter while running in center-aligned mode is not recommended as it can lead to unpredictable results. In particular:

- The direction is not updated if you write a value in the counter that is greater than the auto-reload value (TIM2_CNT>TIM2_ARR). For example, if the counter was counting up, it continues to count up.
- The direction is updated if you write 0 or write the TIM2_ARR value in the counter but no Update Event UEV is generated.

The safest way to use center-aligned mode is to generate an update by software (setting the UG bit in the TIM2_EGR register) just before starting the counter and not to write the counter while it is running.

12.8.9.3 One-pulse mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.
Starting the counter can be done through the slave mode controller. Generating the waveform can be done in output compare mode or PWM mode. You select One-pulse mode by setting the OPM bit in the TIM2_CR1 register. This makes the counter stop automatically at the next update event UEV.
A pulse can be generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:
In up-counting: CNT < CCRx <= ARR (in particular, 0 < CRx)
In down-counting: CNT > CCRx

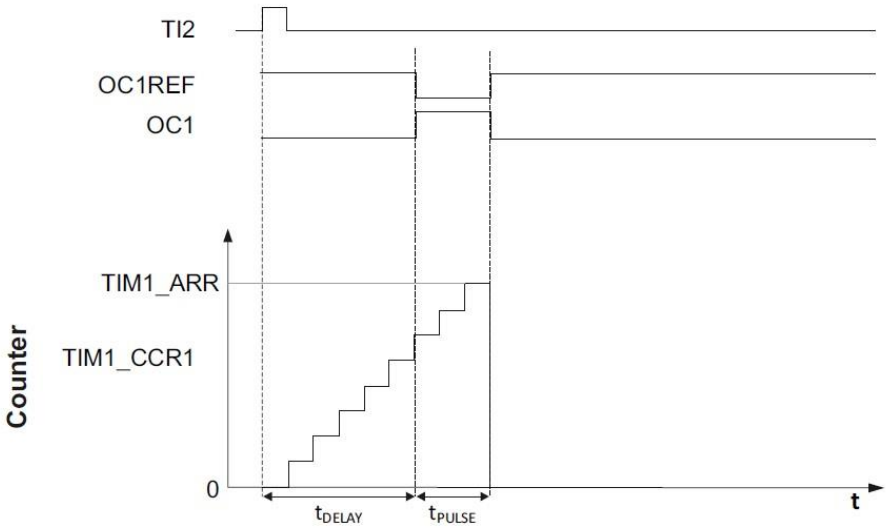


Fig 12-81 Example of one-pulse mode

For example, you need to generate a positive pulse on OC1 with a length of tPULSE and after a delay of tDELAY as soon as a rising edge is detected on the TI2 input pin.
Let's use TI2FP2 as Trigger 1:

- Map TI2FP2 to TI2: write CC2S='01' in the TIM2_CCMR1 register.
- Make TI2FP2 active on rising edge: write CC2P='0' in the TIM2_CCER register.
- Select TI2FP2 as the trigger for the slave mode controller (TRGI): write TS='110' in the TIM2_SMCR register.
- Select trigger mode to start the counter: write SMS='110' in the TIM2_SMCR register.

The OPM waveform is defined by writing the compare registers (taking into account the clock frequency and the counter prescaler):

- tDELAY is defined by the value written in the TIM2_CCR1 register.
- tPULSE is defined by the difference between the auto-reload value and the compare value (TIM2_ARR - TIM2_CCR1).
- Let's say you want to generate a waveform from '0' to '1' when a comparison match occurs and a waveform from '1' to '0' when the counter reaches the preload value; first, write OC1M='111' in the TIM2_CCMR1 register to enter PWM mode 2. Optionally enable the preload registers: write OC1PE='1' in the TIM2_CCMR1 register and ARPE in the TIM2_CR1 register. Then write the compare value in the TIM2_CCR1 register, the auto-reload value in the TIM2_ARR register, generate an update by setting the UG bit and wait for an external trigger event on TI2. In this example, CC1P='0'.

In this example, the DIR and CMS bits in the TIM2_CR1 register should be low.
Since only one pulse is needed, the OPM bit in the TIM2_CR1 register must be set to '1' to stop the counter at the next update event (when the counter rolls over from the auto-reload value to 0).

12.8.9.4 Special case: OCx fast enable

In One-pulse mode, the edge detection on TIx input sets the CEN bit which enables the counter. Then the comparison between the counter and the compare value makes the output toggle. But these operations take some clock cycles and thus limit the minimum delay tDELAY obtainable.
If you want to output a waveform with the minimum delay, you can set the OCxFE bit in the TIM2_CCMRx register. Then OCxREF (and OCx) are forced in response to the stimulus, without depending on the comparison result. Its level is the same as if a compare match had occurred. OCxFE acts only if the channel is configured in PWM1 or PWM2 mode.

12.8.10 Clearing the OCxREF signal on an external event

The OCxREF signal for a given channel can be driven low by a high level on the ETRF input by setting the corresponding OCxCE bit in the TIM2_CCMRx register to '1'. The OCxREF signal remains low until the next update event, UEV, occurs.

This function can only be used in output compare and PWM modes, and does not work in forced mode.

For example, the OCxREF signal can be connected to the output of a comparator to be used for current handling. In this case, the ETR must be configured as follows:

1. The external trigger prescaler must be kept off: bits ETPS[1:0]='00' in the TIM2_SMCR register.
2. External clock mode 2 must be disabled: bit ECE='0' in the TIM2_SMCR register.
3. The external trigger polarity (ETP) and the external trigger filter (ETF) can be configured according to the application needs.

The figure below shows the behavior of the OCxREF signal when the ETRF input becomes high, for different values of OCxCE. In this example, the timer TIM2 is programmed in PWM mode.

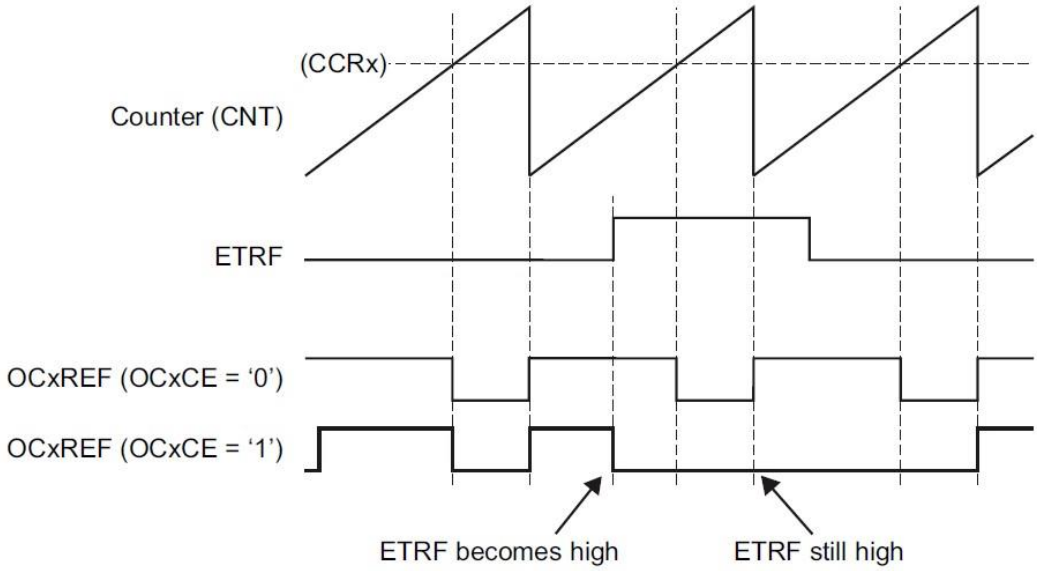


Fig 12-82 Clearing TIM2 OCxREF

12.8.11 Encoder interface mode

Select Encoder interface mode by writing: SMS='001' in the TIM2_SMCR register if the counter is counting on TI2 edges only, SMS='010' if it is counting on TI1 edges only and SMS='011' if it is counting on both TI1 and TI2 edges.

Select Encoder interface mode by writing: SMS='001' in the TIM2_SMCR register if the counter is counting on TI2 edges only, SMS='010' if it is counting on TI1 edges only and SMS='011' if it is counting on both TI1 and TI2 edges.

The two inputs TI1 and TI2 are used to interface to an incremental encoder. Refer to Table 12-5. Assuming that it is enabled (CEN bit='1' in the TIM2_CR1 register), the counter is clocked by each valid transition on TI1FP1 or TI2FP2 (TI1 and TI2 after input filter and polarity selection, TI1FP1=TI1 if not filtered and not inverted, TI2FP2=TI2 if not filtered and not inverted). The sequence of transitions of the two inputs is evaluated and generates count pulses and a direction signal. Depending on the sequence the counter counts up or down, and the DIR bit in the TIM2_CR1 register is modified by hardware accordingly. The DIR bit is recalculated at each transition on any input (TI1 or TI2), whatever the counter is counting on TI1, on TI2 or on both TI1 and TI2.

Encoder interface mode acts basically as an external clock with direction selection. This means that the counter just counts continuously between 0 and the auto-reload value in the TIM2_ARR register (0 to ARR or ARR to 0 depending on the direction). So you must configure TIM2_ARR before starting. In the same way, the capturer, the comparator, the prescaler, the trigger output features, etc. work as usual.

In this mode, the counter is modified automatically following the speed and the direction of the incremental encoder and its content, therefore, always represents the encoder's position. The count direction corresponds to the rotation direction of the connected sensor. The table below summarizes the possible combinations, assuming TI1 and TI2 do not switch at the same time.

Active Edge	Level of relative signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 Signal	TI1FP1 Signal	TI2FP2 Signal	TI2FP2 Signal
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down Counting	Up Counting	No Count	No Count
	Low	Up Counting	Down Counting	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up Counting	Down Counting
	Low	No Count	No Count	Down Counting	Up Counting
Counting on both TI1 and TI2	High	Down Counting	Up Counting	Up Counting	Down Counting
	Low	Up Counting	Down Counting	Down Counting	Up Counting

Table 12-5 Counting direction versus encoder signals

An external incremental encoder can be connected directly to the MCU without external interface logic. However, comparators are normally used to convert the encoder's differential outputs to digital signals. This greatly increases noise immunity. The third encoder output which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a counter reset.

The figure below gives an example of counter operation, showing count signal generation and direction control. It also shows how input jitter is compensated where both edges are selected. This might occur if the sensor is positioned near to one of the switching points.

In this example, we assume that the configuration is the following:

- CC1S='01' (TIM2_CCMR1 register, IC1FP1 mapped on TI1)
- CC2S='01'(TIM2_CCMR2 register，IC2FP2 mapped on TI2)
- CC1P='0'(TIM2_CCER register，IC1FP1 non-inverted，IC1FP1=TI1)
- CC2P='0' (TIM2_CCER register, IC2FP2 non-inverted, IC2FP2=TI2)
- SMS='011' (TIM2_SMCR register, all inputs are active on both rising and falling edges)
- CEN='1' (TIM2_CR1 register, Counter enabled)

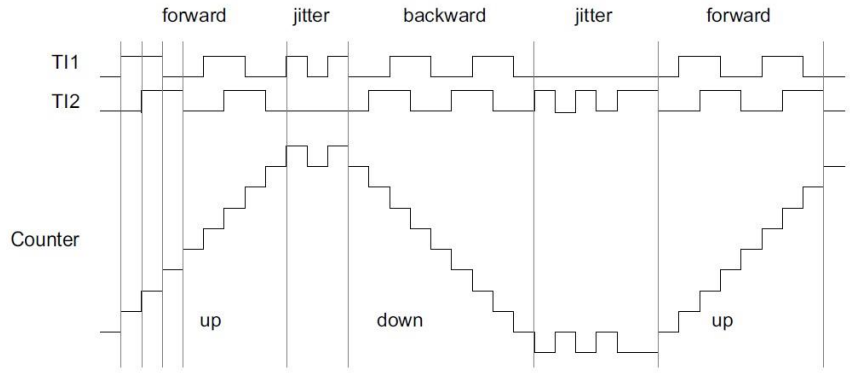


Fig 12-83 Example of counter operation in encoder interface mode

The figure below shows an example of counter behavior when the IC1FP1 polarity is inverted (same configuration as above except CC1P='1').

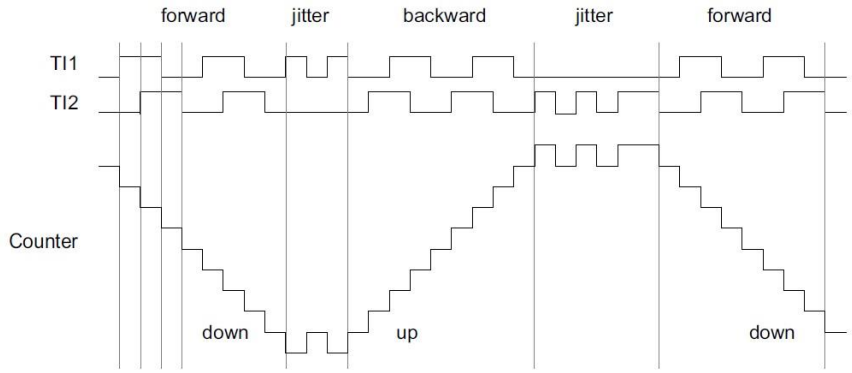


Fig 12-84 Example of encoder interface mode with IC1FP1 inverted

When the timer is configured in encoder interface mode, it provides information on the sensor's current position. You can obtain dynamic information (speed, acceleration, deceleration) by measuring the period between two encoder events using a second timer configured in capture mode. The output of the encoder which indicates the mechanical zero can be used for this purpose. Depending on the interval between two events, the counter can be read at a fixed time. If possible, you can latch the counter value into a third input capture register (the capture signal must be periodic and can be generated by another timer).

12.8.12 Timer input XOR function

The TI1S bit in the TIM2_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate, combining the 3 inputs TIM2_CH1, TIM2_CH2 and TIM2_CH3.
The TI1S bit in the TIM2_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate, combining the 3 inputs TIM2_CH1, TIM2_CH2 and TIM2_CH3.

12.8.13 Timers and external trigger synchronization

The TIM2 timer can be synchronized with an external trigger in several modes: Reset mode, Gated mode and Trigger mode.

12.8.13.1 Slave mode: Reset mode

The counter and its prescaler can be re-initialized in response to an event on a trigger input. Moreover, if the URS bit from the TIM2_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIM2_ARR, TIM2_CCRx) are updated.

In the following example, the up-counter is cleared in response to a rising edge on TI1 input:

- Configure channel 1 to detect rising edges on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC1S bits select the input capture source only, CC1S=01 in the TIM2_CCMR1 register. Write CC1P=0 in TIM2_CCER register to validate the polarity (and detect rising edges only).
  Configure the timer in reset mode by writing SMS=100 in TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in TIM2_SMCR register.
- Start the counter by writing CEN=1 in the TIM2_CR1 register.

The counter starts counting on the internal clock, then behaves normally until the rising edge on TI1 occurs. At this point, the counter is cleared and restarts counting from 0. In the meantime, the trigger flag (TIF bit in the TIM2_SR register) is set and an interrupt request can be sent depending on the TIE (interrupt enable) bit in TIM2_IER register. The figure below shows this behavior when the auto-reload register TIM2_ARR=0x36. The delay between the rising edge on TI1 and the actual reset of the counter depends on the resynchronization circuit on TI1 input.

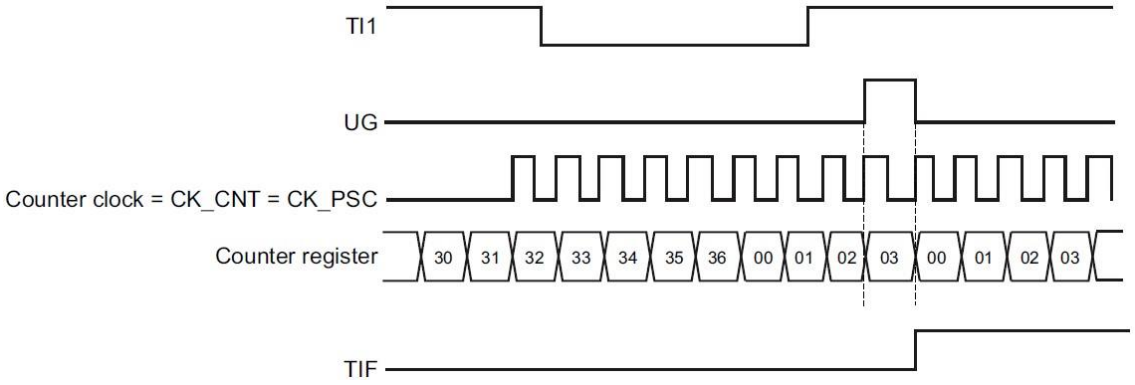


Fig 12-85 Control circuit in reset mode

12.8.13.2 Slave mode: Gated mode

The counter is enabled depending on the level of a selected input.

In the following example, the counter counts up only when TI1 is low:

- Configure channel 1 to detect low level on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC1S bits select the input capture source only, write CC1S=01 in the TIM2_CCMR1 register. Write CC1P=1 in the TIM2_CCER register to validate the polarity (detect low level only).
- Configure the timer in gated mode by writing SMS=101 in the TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in the TIM2_SMCR register.
- Enable the counter by writing CEN=1 in the TIM2_CR1 register. In gated mode, the counter does not start if CEN=0, whatever the trigger input level.

The counter starts counting on the internal clock as long as TI1 is low and stops as soon as TI1 becomes high. The TIF flag in the TIM2_SR register is set both when the counter starts or stops.

The delay between the rising edge on TI1 and the actual stop of the counter depends on the resynchronization circuit on TI1 input.

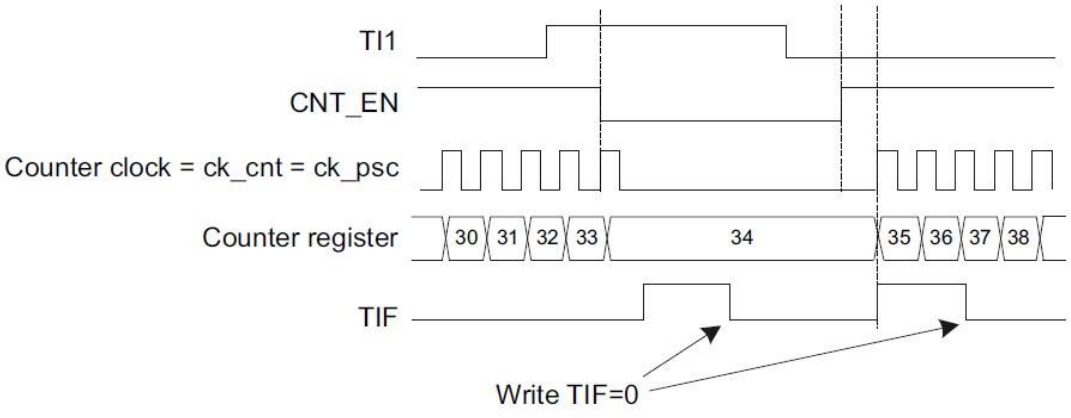


Fig 12-86 Control circuit in gate mode

12.8.13.3 Slave mode: Trigger mode

The counter is started in response to an event on a selected input.

In the following example, the up-counter starts in response to a rising edge on TI2 input:

- Configure channel 2 to detect rising edge on TI2. Configure the input filter duration (in this example, we don't need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so you don't need to configure it. The CC2S bits are used only for input capture source selection, write CC2S=01 in the TIM2_CCMR1 register. Write CC2P=1 in the TIM2_CCER register to validate the polarity (detect low level only).
- Configure the timer in trigger mode by writing SMS=110 in the TIM2_SMCR register. Select TI2 as the input source by writing TS=110 in the TIM2_SMCR register. ◦

When a rising edge occurs on TI2, the counter starts counting on the internal clock and the TIF flag is set.

The delay between the rising edge on TI2 and the actual start of the counter depends on the resynchronization circuit on TI2 input.

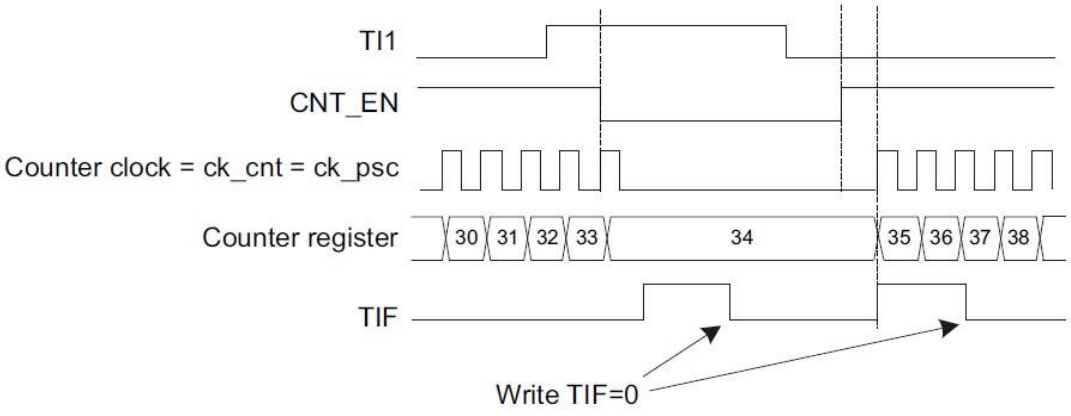


Fig 12-87 Control circuit in trigger mode

12.8.13.4 Slave mode: External clock mode 2 + Trigger mode

External clock mode 2 can be used in addition to another slave mode (except external clock mode 1 and encoder mode). In this case, the ETR signal is used as external clock input, and another input can be selected as trigger input when in reset mode, gated mode or trigger mode. It is not recommended to select ETR as TRGI through the TS bits of TIM2_SMCR register.

In the following example, the counter counts up at each rising edge on ETR after a rising edge on TI1:

1. Configure the external trigger input circuit via the TIM2_SMCR register:
   - ETF=0000:No filter
   - ETPS=00: Prescaler disabled
   - ETP=0: Detection of rising edges on ETR, set ECE=1 to enable external clock mode 2
2. Configure channel 1 to detect rising edge on TI as follows:
   - IC1F=0000: No filter
   - The capture prescaler is not used for triggering, so it does not need to be configured
   - Write CC1S=01 in TIM2_CCMR1 register to select the input capture source
   - Write CC1P=0 in TIM2_CCER register to validate the polarity (detect rising edge only)
3. 3.Configure the timer in trigger mode by writing SMS=110 in the TIM2_SMCR register. Select TI1 as the input source by writing TS=101 in the TIM2_SMCR register.

When a rising edge occurs on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR. The delay between the rising edge on the ETR signal and the actual reset of the counter depends on the resynchronization circuit on the ETRP input.

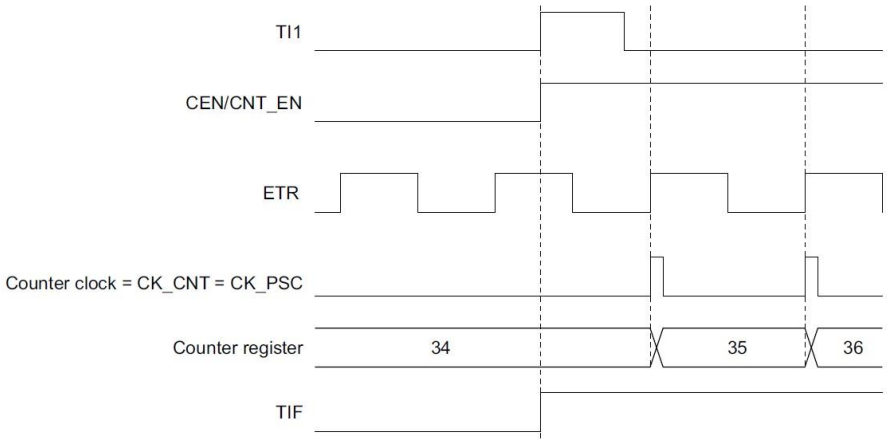


Fig 12-88 Control circuit in external clock mode 2 + trigger mode

12.8.14 Timer synchronization

All TIM2 timers are linked together internally for timer synchronization or chaining. When one timer is in Master mode, it can reset, start, stop or clock the counter of another timer configured in Slave mode.

The figure below presents an overview of the trigger selection and the master mode selection blocks. ◦

12.8.14.1 Using one timer as prescaler for another timer

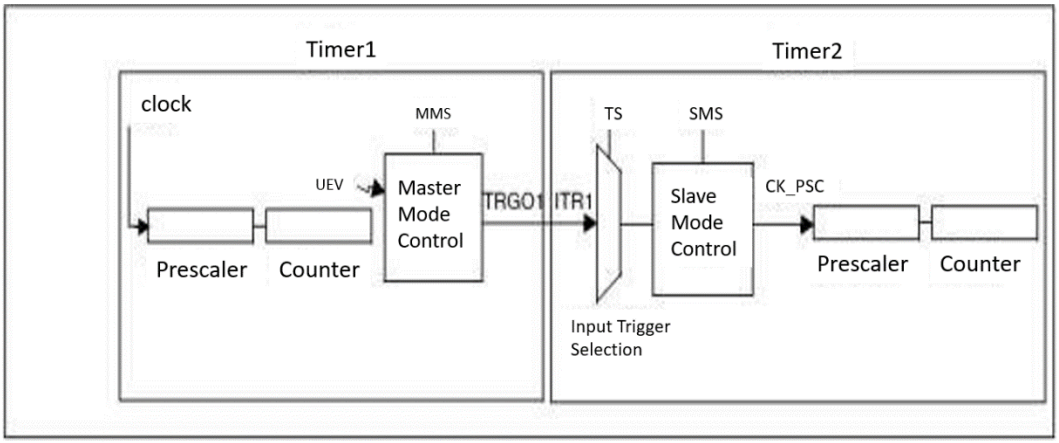


Fig 12-89 Master/Slave timer example

For example, you can configure Timer 1 to act as a prescaler for Timer 2. Refer to Figure 12-85, do the following:

- Configure Timer 1 in master mode so that it outputs a periodic trigger signal on each update event UEV. Writing MMS='010' in the EPWM_CR2 register causes a rising edge to be output on TRGO1 each time an update event is generated.
- Connect the TRGO1 output of Timer 1 to Timer 2, write TS='000' in the TIM2_SMCR register to configure Timer 2 in slave mode using ITR1 as internal trigger.
- Then put the slave mode controller in external clock mode 1 (write SMS=111 in the TIM2_SMCR register); this way Timer 2 is clocked by the rising edge of the periodic Timer 1 signal (which corresponds to the timer 1 counter overflow).
- Finally, both timers must be enabled by setting their respective CEN bits (TIM2_CR1 register).

Note: If OCx is selected on Timer 1 as trigger output (MMS=1xx), its rising edge is used to clock the counter of Timer 2.

12.8.14.2 Using one timer to enable another timer

In this example, the enable of Timer 2 is controlled with the output compare of Timer 1. Refer to Figure 12-86 for connections. Timer 2 counts on the divided internal clock only when OC1REF of Timer 1 is high. Both timers clock frequencies are divided by 3 by the prescaler (fCK_CNT=fCK_INT/3).

- Configure Timer 1 in master mode, sending its Output Compare Reference (OC1REF) as trigger output (MMS=100 in EPWM_CR2 register).
- Configure Timer 1 OC1REF waveform (EPWM_CCMR1 register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 in TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Set CEN=1 in TIM2_CR1 register to enable Timer 2.
- Set CEN=1 in EPWM_CR1 register to start Timer 1.

Note: The clock of the Timer 2 is not synchronized with the clock of the Timer 1, this mode only affects the Timer 2 counter enable signal.

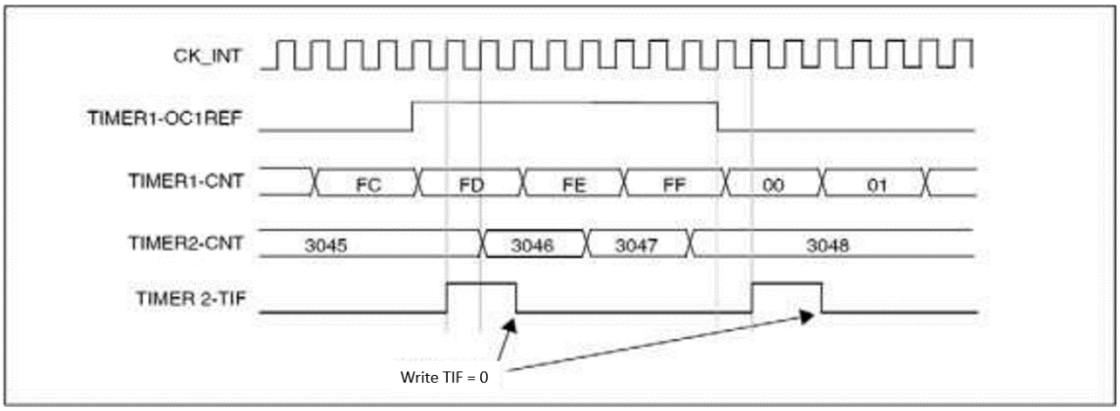


Fig 12-90 Timer 1 OC1REF controlling Timer 2

In the example of Figure 12-91, before Timer 2 starts, their counters and prescalers are not initialized, so they start counting from the current value. You can reset both timers before starting Timer 1 to make them start from a given value, i.e., write any desired value in the timer counter. Writing the UG bit in the TIM2_EGR register resets the timer. In the next example, we need to synchronize Timer 1 and Timer 2. Timer 1 is in master mode and starts from 0, Timer 2 is in slave mode and starts from 0xE7; both timers have the same prescaler ratio. Writing '0' to CEN bit in EPWM_CR1 will disable Timer 1, and Timer 2 stops immediately.

- Configure Timer 1 in master mode, sending Output Compare 1 Reference signal (OC1REF) as trigger output (MMS=100 in EPWM_CR2 register).
- Configure Timer 1 OC1REF waveform (EPWM_CCMR1 register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 in TIM2_SMCR register).
- Configure Timer 2 in gated mode (SMS=101 in TIM2_SMCR register).
- Set UG='1' in EPWM_EGR register to reset Timer 1.
- Set UG='1' in TIM2_EGR register to reset Timer 2.
- Write '0xE7' to Timer 2 counter (TIM2_CNT) to initialize it to 0xE7.
- Set CEN='1' in TIM2_CR1 register to enable Timer 2.
- Set CEN='1' in EPWM_CR1 register to start Timer 1.
- Set CEN='0' in EPWM_CR1 register to stop Timer 1.

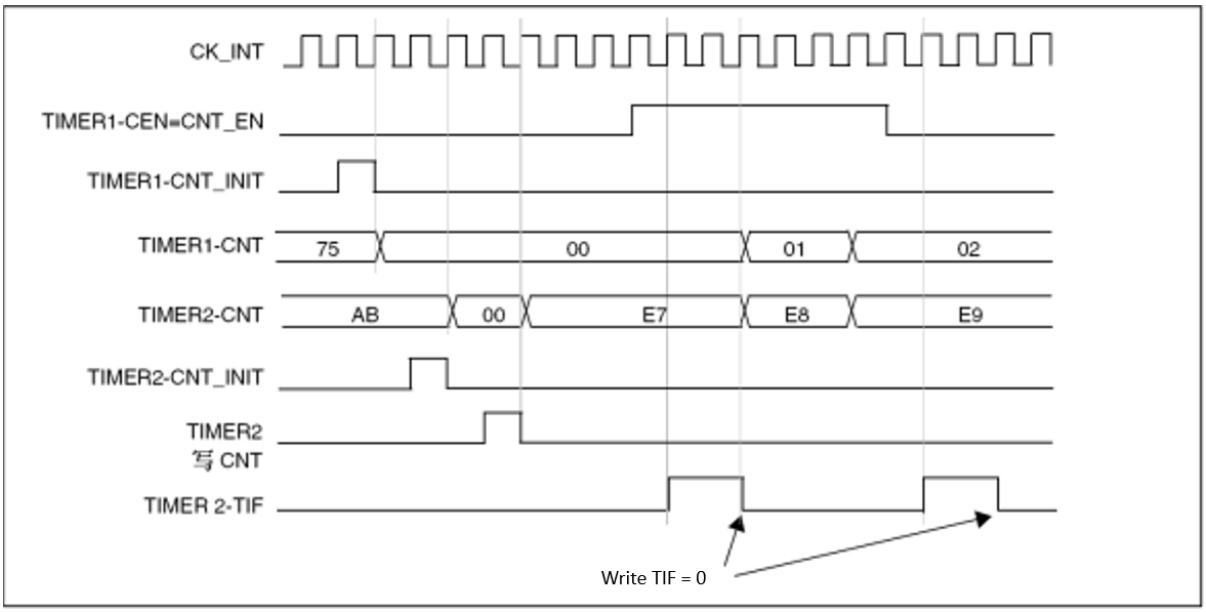


Fig 12-91 Gating Timer 2 with Timer 1 enable

12.8.14.3 Using one timer to start another timer

In this example, the enable of Timer 2 is controlled by the update event of Timer 1. Refer to Figure 12-85 for connections. Once Timer 1 generates an update event, Timer 2 starts counting from its current value (which can be non-zero) on the divided internal clock. Upon receiving the trigger signal, the CEN bit of Timer 2 is automatically set to '1', and the counter starts counting until '0' is written to the CEN bit in the TIM2_CR1 register. Both timers clock frequencies are divided by 3 by the prescaler (fCK_CNT=fCK_INT/3).

- Configure Timer 1 in master mode, sending its update event (UEV) as trigger output (MMS=010 in EPWM_CR2 register).
- Configure the period of Timer 1 (EPWM_ARR register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 in TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS=110 in TIM2_SMCR register).
- Set CEN=1 in EPWM_CR1 register to start Timer 1.

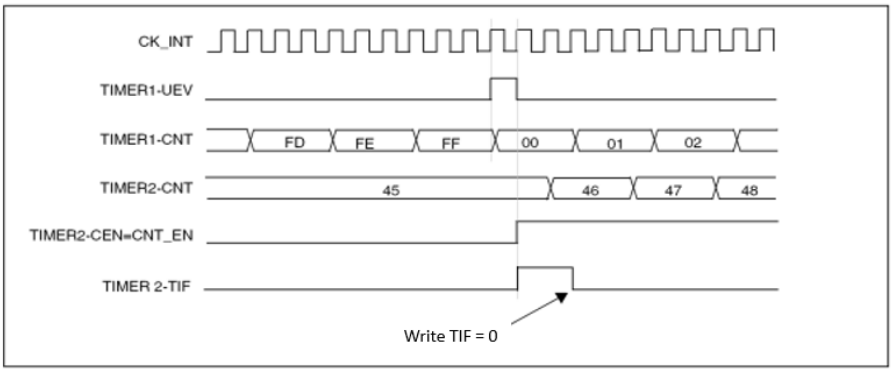


Fig 12-92 Triggering Timer 2 with update of Timer 1

In the previous example, both counters can be initialized before starting counting. Figure 12-89 shows the behavior with the same configuration, using trigger mode instead of gated mode (SMS=110 in TIM2_SMCR register).

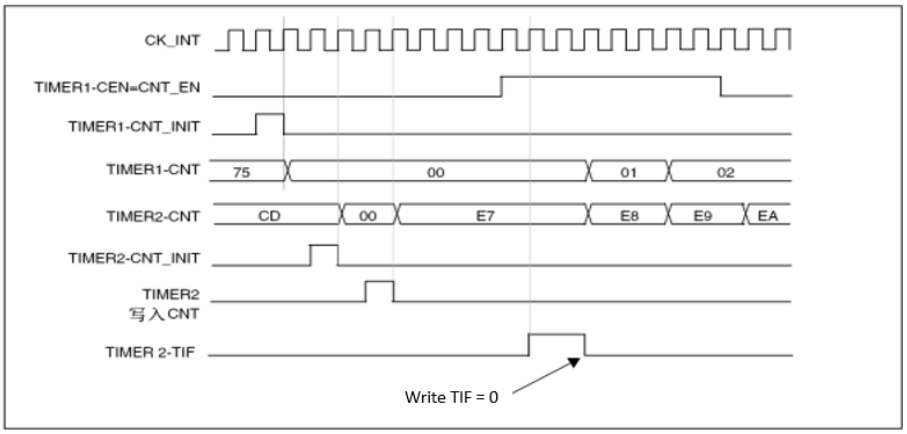


Fig 12-93 Triggering Timer 2 with enable of Timer 1

12.8.14.4 Using one timer as a prescaler for another

This example uses Timer 1 as a prescaler for Timer 2. Refer to Figure 12-85 for connections. Configure as follows:

- Configure Timer 1 in master mode to send its update event (UEV) as trigger output (MMS='010' in EPWM_CR2 register). A periodic signal is output on each counter overflow.
- Configure the period of Timer 1 (EPWM_ARR register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 in TIM2_SMCR register).
- Configure Timer 2 in external clock mode 1 (SMS=111 in TIM2_SMCR register).
- Set CEN=1 in TIM2_CR1 register to start Timer 2.
- Set CEN=1 in EPWM_CR1 register to start Timer 1.

12.8.14.5 Starting 2 timers synchronously from an external trigger

In this example, we enable Timer 1 when a rising edge occurs on TI1 input, and enable Timer 2 at the same time. Refer to Figure 12-85. To ensure counter alignment, Timer 1 must be configured in Master/Slave mode (Slave for TI1, Master for Timer 2):

- Configure Timer 1 in master mode, sending its Enable as trigger output (MMS='001' in EPWM_CR2 register).
- Configure Timer 1 in slave mode, to get input trigger from TI1 (TS='100' in EPWM_SMCR register).
- Configure Timer 1 in trigger mode (SMS='110' in EPWM_SMCR register).
- Configure Timer 1 in Master/Slave mode (MSM='1' in EPWM_SMCR register).
- Configure Timer 2 to get input trigger from Timer 1 (TS=000 in TIM2_SMCR register).
- Configure Timer 2 in trigger mode (SMS='110' in TIM2_SMCR register).

When a rising edge occurs on TI1 of Timer 1, both timers start counting synchronously on the internal clock and both TIF flags are set.

Note: In this example, both timers are initialized before starting (by setting the respective UG bits). Both counters start from 0, but an offset can be inserted between them by writing any of the counter registers (TIM2_CNT). The figure below shows that there is a delay between CNT_EN and CK_PSC of Timer 1 in Master/Slave mode.

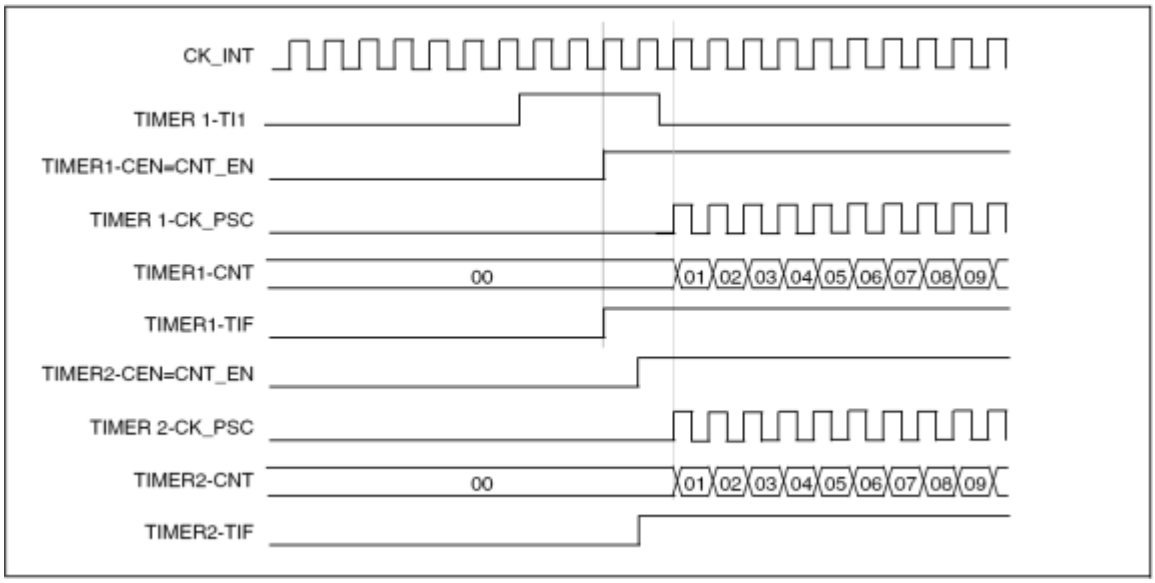


Fig 12-94 Triggering Timer 1 and Timer 2 with Timer 1 TI1 input

12.8.15 Debug mode

When the microcontroller enters debug mode, depending on the DBG_TIM2_STOP bit in the DBG module, the TIM2 counter either continues to work normally or stops.

12.9 TIM2 register table

The peripheral registers can be accessed by words (32-bit).TIM2 base address 0x4000 1000

Address	Name	Description
0x40001000	TIM2_CR1	TIM2 control register 1
0x40001004	TIM2_CR2	TIM2 control register 2
0x40001008	TIM2_SMCR	TIM2 slave mode control register
0x4000100C	TIM2_IER	TIM2 interrupt enable register
0x40001010	TIM2_SR	TIM2 status register
0x40001014	TIM2_EGR	TIM2 event generation register
0x40001018	TIM2_CCMR1	TIM2 capture/compare mode register 1
0x4000101C	TIM2_CCMR2	TIM2 capture/compare mode register 2
0x40001020	TIM2_CCER	TIM2 capture/compare enable register
0x40001024	TIM2_CNT	TIM2 counter register
0x40001028	TIM2_PSC	TIM2 prescaler register
0x4000102C	TIM2_ARR	TIM2 auto-reload register
0x40001030	-	Reserved
0x40001034	TIM2_CCR1	TIM2 capture/compare register 1
0x40001038	TIM2_CCR2	TIM2 capture/compare register 2
0x4000103C	TIM2_CCR3	TIM2 capture/compare register 3
0x40001040	TIM2_CCR4	TIM2 capture/compare register 4

Table 12-6 TIM2 register table

12.10 TIM2 register description

12.10.1 TIM2 Control Register 1(TIM2_CR1)

Bit	Name	R/W	Reset	Description
31:11	Reserved	--	0x0	Always read as 0.
10	ASYMEN	R/W	0x0	Asymmetric counter enable TIM2 asymmetric counting enable in center-aligned mode 0: Symmetric counting enable 1: Asymmetric counting enable
9:8	CKD	R/W	0x0	Clock division These 2 bits define the division ratio between the timer clock (CK_INT) frequency and the sampling clock (tDTS) used by the dead-time generator and the digital filters (ETR, TIx). 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: Reserved, do not use this configuration
7	ARPE	R/W	0x0	Auto-reload preload enable 0: TIM2_ARR register is not buffered 1: TIM2_ARR register is buffered
6:5	CMS	R/W	0x0	Center-aligned mode selection 00: Edge-aligned mode. The counter counts up or down depending on the direction bit (DIR). 01: Center-aligned mode 1. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set only when the counter is counting down. 10: Center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set only when the counter is counting up. 11: Center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIM2_CCMRx register) are set both when the counter is counting up or down. Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1).

4	DIR	R/W	0x0	Direction 0: Counter counts up 1: Counter counts down Note: This bit is read only when the timer is configured in Center-aligned mode or Encoder mode.
3	OPM	R/W	0x0	One pulse mode 0: Counter is not stopped at update event 1: Counter stops counting at the next update event (clearing the CEN bit)
2	URS	R/W	0x0	Update request source This bit is used to select the UEV event source. 0: Any of the following events generate an update interrupt if enabled: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt if enabled.
1	UDIS	R/W	0x0	Update disable This bit enables/disables the UEV event generation. 0: UEV enabled. The Update (UEV) event is generated by one of the following events: - Counter overflow/underflow - Setting the UG bit - Update generation through the slave mode controller Buffered registers are loaded with their preload values. 1: UEV disabled. The Update event is not generated, shadow registers (ARR, PSC, CCRx) keep their value. However the counter and the prescaler are re-initialized if the UG bit is set or if a hardware reset is received from the slave mode controller.
0	CEN	R/W	0x0	Counter enable 0: Counter disabled 1: Counter enabled Note: External clock, Gated mode and Encoder mode work only if the CEN bit has been previously set by software. Trigger mode can set the CEN bit automatically by hardware.。

12.10.2 TIM2 Control Register 2(TIM2_CR2)

Bit	Name	R/W	Reset	Description
31:8	Reserved	--	0x0	Always read as 0.
7	TI1S	R/W	0x0	TI1 selection 0: The TIM2_CH1 pin is connected to TI1 input. 1: The TIM2_CH1, TIM2_CH2 and TIM2_CH3 pins are connected to the TI1 input (XOR combination).
6:4	MMS	R/W	0x0	Master mode selection These 3 bits select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows: 000: Reset - The UG bit from the TIM2_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode), the signal on TRGO is delayed compared to the actual reset. 001: Enable - The Counter Enable signal CNT_EN is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by the logic OR between the CEN control bit and the trigger input when configured in gated mode. When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIM2_SMCR register). 010: Update - The Update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer. 011: Compare Pulse - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred (TRGO). 100: Compare - OC1REF signal is used as trigger output (TRGO). 101: Compare - OC2REF signal is used as trigger output (TRGO). 110: Compare - OC3REF signal is used as trigger output (TRGO). 111: Compare - OC4REF signal is used as trigger output (TRGO).
3	Reserved	--	0x0	Always read as 0.

2	CCUS	R/W	0x0	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit only. 1: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COM bit or when a rising edge occurs on TRGI. Note: This bit acts only on channels that have a complementary output.
1	Reserved	--	0x0	Always read as 0.
0	CCPC	R/W	0x0	Capture/compare preloaded control 0: The CCxE, CCxNE and OCxM bits are not preloaded. 1: The CCxE, CCxNE and OCxM bits are preloaded, after having been written, they are updated only when the COM bit is set. Note: This bit acts only on channels that have a complementary output.

12.10.3 TIM2 Slave Mode Control Register (TIM2_SMCR)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15	ETP	R/W	0x0	External trigger polarity This bit selects whether ETR or inverted ETR is used for trigger operations. 0: ETR is non-inverted, active at high level or rising edge. 1: ETR is inverted, active at low level or falling edge.
14	ECE	R/W	0x0	External clock enable This bit enables External clock mode 2. 0: External clock mode 2 disabled. 1: External clock mode 2 enabled. The counter is clocked by any active edge on the ETRF signal. Note 1: Setting the ECE bit has the same effect as selecting external clock mode 1 with TRGI connected to ETRF (SMS=111 and TS=111). Note 2: It is possible to use the external clock mode 2 with the following slave modes: reset mode, gated mode and trigger mode. Nevertheless, TRGI must not be connected to ETRF in this case (TS bits must not be 111). Note 3: If external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock input is ETRF.

13:12	ETPS [1:0]	R/W	0x0	External trigger prescaler The ETRP signal frequency must be at most 1/4 of TIM2CLK frequency. A prescaler can be used to reduce ETRP frequency when a faster external clock is used. 00: Prescaler off 01: ETRP frequency divided by 2 10: ETRP frequency divided by 4 11: ETRP frequency divided by 8
11:8	ETF[3:0]	R/W	0x0	External trigger filter These bits define the frequency used to sample ETRP signal and the length of the digital filter applied to ETRP. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000: No filter, sampling is done at fDTS 0001: fSAMPLING = fCK_INT, N = 2 0010: fSAMPLING = fCK_INT, N = 4 0011: fSAMPLING = fCK_INT, N = 8 0100: fSAMPLING = fDTS/2, N = 6 0101: fSAMPLING = fDTS/2, N = 8 0110: fSAMPLING = fDTS/4, N = 6 0111: fSAMPLING = fDTS/4, N = 8 1000: fSAMPLING = fDTS/8, N = 6 1001: fSAMPLING = fDTS/8, N = 8 1010: fSAMPLING = fDTS/16, N = 5 1011: fSAMPLING = fDTS/16, N = 6 1100: fSAMPLING = fDTS/16, N = 8 1101: fSAMPLING = fDTS/32, N = 5 1110: fSAMPLING = fDTS/32, N = 6 1111: fSAMPLING = fDTS/32, N = 8
7	MSM	R/W	0x0	Master/slave mode 0: No action 1: The effect of an event on the trigger input (TRGI) is delayed to allow a perfect synchronization between the current timer and its slaves (through TRGO). It is useful if we want to synchronize several timers in a single external event.

6:4	TS[2:0]	R/W	0x0	Trigger selection These 3 bits select the trigger input to be used to synchronize the counter. 000: Internal Trigger 0 (ITR0) 001: Internal Trigger 1 (ITR1) 010: Internal Trigger 2 (ITR2) 011: Internal Trigger 3 (ITR3) 100: TI1 Edge Detector (TI1F_ED) 101: Filtered Timer Input 1 (TI1FP1) 110: Filtered Timer Input 2 (TI2FP2) 111: External Trigger input (ETRF) See Table 12-1 for more details on ITRx. Note: These bits must be changed only when they are not used (e.g. when SMS=000) to avoid wrong edge detections at the transition.
3	Reserved	--	0x0	Always read as 0.

2:0	SMS[2:0]	R/W	0x0	Slave mode selection When external signals are selected, the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input (see Input Control register and Control Register description). 000: Slave mode disabled - if CEN = 1 then the prescaler is clocked directly by the internal clock. 001: Encoder mode 1 - Counter counts up/down on TI2FP2 edge depending on TI1FP1 level. 010: Encoder mode 2 - Counter counts up/down on TI1FP1 edge depending on TI2FP2 level. 011: Encoder mode 3 - Counter counts up/down on both TI1FP1 and TI2FP2 edges depending on the level of the other input. 100: Reset mode - Rising edge of the selected trigger input (TRGI) reinitializes the counter and generates an update of the registers. 101: Gated mode - The counter clock is enabled when the trigger input (TRGI) is high. The counter stops (but is not reset) as soon as the trigger becomes low. Both start and stop of the counter are controlled. 110: Trigger mode - The counter starts at a rising edge of the trigger TRGI (but it is not reset). Only the start of the counter is controlled. 111: External Clock Mode 1 - Rising edges of the selected trigger (TRGI) clock the counter. Note: The gated mode must not be used if TI1F_EN is selected as the trigger input (TS=100). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal.

Slave timer	**ITR0(TS=000)**	**ITR1(TS=001)**	**ITR2(TS=010)**	**ITR3(TS=011)**
TIM2	EPWM_trgo	irq_timer0	irq_timer1	irq_lptimer

Table 12-7 TIM2 Internal trigger connection

Note: If a timer is not present in a specific product, the corresponding trigger signal ITRx does not exist.

12.10.4 TIM2 Interrupt Enable Regsiter (TIM2_IER)

Bit	Name	R/W	Reset	Description
31:14	Reserved	--	0x0	Always read as 0.
13	CCD4IE	R/W	0x0	Capture/Compare 4 down-counting interrupt enable Enable Compare 4 interrupt (Valid when ASYMEN=1 and in down-counting mode) 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
12	UDIE	R/W	0x0	Underflow interrupt enable 0: Underflow interrupt disabled 1: Underflow interrupt enabled
11	OVIE	R/W	0x0	Overflow interrupt enable 0: Overflow interrupt disabled 1: Overflow interrupt enabled
10	CCD3IE	R/W	0x0	Capture/Compare 3 down-counting interrupt enable Enable Compare 3 interrupt (Valid when ASYMEN=1 and in down-counting mode) 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
9	CCD2IE	R/W	0x0	Capture/Compare 2 down-counting interrupt enable Enable Compare 2 interrupt (Valid when ASYMEN=1 and in down-counting mode) 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
8	CCD1IE	R/W	0x0	Capture/Compare 1 down-counting interrupt enable Enable Compare 1 interrupt (Valid when ASYMEN=1 and in down-counting mode) 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
7	Reserved	--	0x0	Always read as 0.
6	TIE	R/W	0x0	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	Reserved	--	0x0	Always read as 0.

4	CC4IE	R/W	0x0	Capture/Compare 4 interrupt enable 0: Capture/Compare 4 interrupt disabled 1: Capture/Compare 4 interrupt enabled
3	CC3IE	R/W	0x0	Capture/Compare 3 interrupt enable 0: Capture/Compare 3 interrupt disabled 1: Capture/Compare 3 interrupt enabled
2	CC2IE	R/W	0x0	Capture/Compare 2 interrupt enable 0: Capture/Compare 2 interrupt disabled 1: Capture/Compare 2 interrupt enabled
1	CC1IE	R/W	0x0	Capture/Compare 1 interrupt enable 0: Capture/Compare 1 interrupt disabled 1: Capture/Compare 1 interrupt enabled
0	UIE	R/W	0x0	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

12.10.5 TIM2 Status Register (TIM2_SR)

Bit	Name	R/W	Reset	Description
31:13	Reserved	--	0x0	Always read as 0.
18	CC4DIF	RC/W0	0x0	Capture/Compare 4 down-counting interrupt flag 1: The TIM2_CNT value matches the TIM2_CCDR4 value in down-counting mode. 0: No match occurred (write 0 to clear).
17	UDIF	RC/W0	0x0	Underflow interrupt flag 1: TIM2_CNT down-counting mode underflow occurred. 0: No underflow occurred.
16	OVIF	RC/W0	0x0	Overflow interrupt flag 1: TIM2_CNT up-counting mode overflow occurred. 0: No overflow occurred.
15	CC3DIF	RC/W0	0x0	Capture/Compare 3 down-counting interrupt flag 1: The TIM2_CNT value matches the TIM2_CCDR3 value in down-counting mode. 0: No match occurred (write 0 to clear).
14	CC2DIF	RC/W0	0x0	Capture/Compare 2 down-counting interrupt flag 1: The TIM2_CNT value matches the TIM2_CCDR2 value in down-counting mode. 0: No match occurred (write 0 to clear).
13	CC1DIF	RC/W0	0x0	Capture/Compare 1 down-counting interrupt flag 1: The TIM2_CNT value matches the TIM2_CCDR1 value in down-counting mode. 0: No match occurred (write 0 to clear).
12	CC4OF	RC/W0	0x0	Capture/Compare 4 overcapture flag See CC1OF description.
11	CC3OF	RC/W0	0x0	Capture/Compare 3 overcapture flag See CC1OF description.
10	CC2OF	RC/W0	0x0	Capture/Compare 2 overcapture flag See CC1OF description.
9	CC1OF	RC/W0	0x0	Capture/Compare 1 overcapture flag This flag is set by hardware only when the corresponding channel is configured in input capture. Write 0 to clear this bit. 0: No overcapture produced. 1: The counter value has been captured in TIM2_CCR1 register while CC1IF flag was already set.

8:7	Reserved	--	0x0	Always read as 0.
6	TIF	RC/W0	0x0	Trigger interrupt flag This bit is set by hardware on a trigger event (active edge detected on TRGI input when the slave mode controller is not in gated mode, or both edges in gated mode). It is cleared by software. 0: No trigger event produced. 1: Trigger interrupt pending.
5	Reserved	--	0x0	Always read as 0.
4	CC4IF	RC/W0	0x0	Capture/Compare 4 interrupt flag Refer to CC1IF description.
3	CC3IF	RC/W0	0x0	Capture/Compare 3 interrupt flag Refer to CC1IF description.
2	CC2IF	RC/W0	0x0	Capture/Compare 2 interrupt flag Refer to CC1IF description.
1	CC1IF	RC/W0	0x0	Capture/Compare 1 interrupt flag If channel CC1 is configured as output: This bit is set by hardware when the counter matches the compare value, with the exception of center-aligned mode (refer to CMS bits in TIM2_CR1 register). It is cleared by software. 0: No match occurred. 1: The content of the counter TIM2_CNT matches the content of the TIM2_CCR1 register. When the contents of TIM2_CCR1 are greater than the contents of TIM2_ARR, the CC1IF bit goes high when the counter overflows (in up-counting or up/down-counting modes) or underflows (in down-counting mode). If channel CC1 is configured as input: This bit is set by hardware when a capture event occurs. It is cleared by software or by reading the TIM2_CCR1 register. 0: No input capture produced. 1: The counter value has been captured in TIM2_CCR1 register (an edge has been detected on IC1 which matches the selected polarity).

0	UIF	RC/W0	0x0	Update interrupt flag This bit is set by hardware when an update event occurs. It is cleared by software. 0: No update event occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: - At overflow or underflow regarding the repetition counter value (update event generated when repetition counter = 0) and if UDIS=0 in the TIM2_CR1 register. - When CNT is re-initialized by software using the UG bit in TIM2_EGR register, if URS=0 and UDIS=0 in the TIM2_CR1 register. - When CNT is re-initialized by a trigger event (refer to Slave mode control register (TIM2_SMCR)), if URS=0 and UDIS=0 in the TIM2_CR1 register.

12.10.6 TIM2 Event Generation Register(TIM2_EGR)

Bit	Name	R/W	Reset	Description
31:7	Reserved	--	0x0	Always read as 0.
6	TG	W	0x0	Trigger generation This bit is set by software in order to generate an event, it is automatically cleared by hardware. 0: No action 1: The TIF flag is set in TIM2_SR register. An interrupt is generated if enabled.
5	保留	--	0x0	始终读为 0。
4	CC4G	W	0x0	Capture/Compare 4 generation Refer to CC1G description.
3	CC3G	W	0x0	Capture/Compare 3 generation Refer to CC1G description.
2	CC2G	W	0x0	Capture/Compare 2 generation Refer to CC1G description.
1	CC1G	W	0x0	Capture/Compare 1 generation This bit is set by software in order to generate a capture/compare event, it is automatically cleared by hardware. 0: No action 1: A capture/compare event is generated on channel CC1: If channel CC1 is configured as output: CC1IF flag is set, corresponding interrupt is sent if enabled. If channel CC1 is configured as input: The current value of the counter is captured in TIM2_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. If the CC1IF flag was already high, the CC1OF flag is set.
0	UG	W	0x0	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 1: Re-initialize the counter and generates an update of the registers. Note that the prescaler counter is cleared too (but the prescaler ratio is not affected). The counter is cleared if the center-aligned mode is selected or if DIR=0 (up-counting), else it takes the auto-reload value (TIM2_ARR) if DIR=1 (down-counting).

12.10.7 TIM2 Capture/Compare Mode Register 1(TIM2_CCMR1)

Output Compare Mode

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15	OC2CE	R/W	0x0	Output Compare 2 clear enable
14:12	OC2M	R/W	0x0	Output Compare 2 mode
11	OC2PE	R/W	0x0	Output Compare 2 preload enable
10	OC2FE	R/W	0x0	Output Compare 2 fast enable
9:8	CC2S [1:0]	R/W	0x0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC2S bit is writable only when the channel is OFF (CC2E = 0 in TIM2_CCER).
7	OC1CE	R/W	0x0	Output Compare 1 clear enable 0: OC1REF is not affected by the ETRF input 1: OC1REF is cleared as soon as a high level is detected on ETRF input

6:4	OC1M	R/W	0x0	Output Compare 1 mode These bits define the behavior of the output reference signal OC1REF from which OC1 and OC1N are derived. OC1REF is active high whereas OC1 and OC1N active level depends on CC1P and CC1NP bits. 000: Frozen - The comparison between the output compare register TIM2_CCR1 and the counter TIM2_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIM2_CNT matches the capture/compare register 1 (TIM2_CCR1). 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIM2_CNT matches the capture/compare register 1 (TIM2_CCR1). 011: Toggle - OC1REF toggles when TIM2_CNT=TIM2_CCR1. 100: Force inactive level - OC1REF is forced low. 101: Force active level - OC1REF is forced high. 110: PWM mode 1 - In upcounting, channel 1 is active as long as TIM2_CNT<TIM2_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF=0) as long as TIM2_CNT>TIM2_CCR1 else active (OC1REF=1). 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIM2_CNT<TIM2_CCR1 else active. In downcounting, channel 1 is active as long as TIM2_CNT>TIM2_CCR1 else inactive. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIM2_BDTR register) and CC1S=00 (the channel is configured in output). Note 2: In PWM mode 1 or 2, the OC1REF level changes only when the result of the comparison changes or when the output compare mode switches from "frozen" mode to "PWM" mode.
3	OC1PE	R/W	0x0	Output Compare 1 preload enable 0: Preload register on TIM2_CCR1 disabled. TIM2_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIM2_CCR1 enabled. Read/Write operations access the preload register. TIM2_CCR1 preload value is loaded in the active register at each update event. Note 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIM2_BDTR register) and CC1S=00 (the channel is configured in output).

				Note 2: The PWM mode can be used without validating the preload register only in one pulse mode (OPM bit set in TIM2_CR1 register). Else the behavior is not guaranteed.
2	OC1FE	R/W	0x0	Output Compare 1 fast enable This bit is used to accelerate the effect of an event on the trigger in input on the CC output. 0: CC1 behaves normally depending on counter and CCR1 values even if the trigger is ON. The minimum delay to activate CC1 output when an edge occurs on the trigger input is 5 clock cycles. 1: An active edge on the trigger input acts like a compare match on CC1 output. Then, OC is set to the compare level independently from the result of the comparison. Delay to sample the trigger input and to activate CC1 output is reduced to 3 clock cycles. OCFE acts only if the channel is configured in PWM1 or PWM2 mode.
1:0	CC1S [1:0]	R/W	0x0	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC1S bit is writable only when the channel is OFF (CC1E = 0 in TIM2_CCER).

Input Capture Mode:

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:12	IC2F	R/W	0x0	Input capture 2 filter Refer to IC1F description.
11:10	IC2PSC	R/W	0x0	Input capture 2 prescaler Refer to IC1PSC description.
9:8	CC2S	R/W	0x0	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC2S bit is writable only when the channel is OFF (CC2E = 0 in TIM2_CCER).
7:4	IC1F	R/W	0x0	Input capture 1 filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output: 0000:No filter, sampling is done at fDTS 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

Bit	Name	R/W	Reset	Description
				1101:$f_{SAMPLING}$ = $f_{DTS/32}$， N = 5 1110:$f_{SAMPLING}$ = $f_{DTS/32}$， N = 6 1111:$f_{SAMPLING}$ = $f_{DTS/32}$， N = 8
3:2	IC1PSC	R/W	0x0	Input capture 1 prescaler This bit-field defines the ratio of the prescaler acting on CC1 input (IC1). The prescaler is reset as soon as CC1E=0 (TIM2_CCER register). 00: no prescaler, capture is done at each edge detected on the capture input 01: capture is done once every 2 events 10: capture is done once every 4 events 11: capture is done once every 8 events
1:0	CC1S	R/W	0x0	Capture/Compare 1 Selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC1S bit is writable only when the channel is OFF (CC1E = 0 in TIM2_CCER).

12.10.8 TIM2 Capture/Compare Mode Register 2(TIM2_CCMR2)

Output compare mode:

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15	OC4CE	R/W	0x0	Output Compare 4 clear enable Refer to OC1CE description.
14:12	OC4M [2:0]	R/W	0x0	Output Compare 4 mode Refer to OC1M description.
11	OC4PE	R/W	0x0	Output Compare 4 preload enable Refer to OC1PE description.
10	OC4FE	R/W	0x0	Output Compare 4 fast enable Refer to OC1FE description.
9:8	CC4S [1:0]	R/W	0x0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC4S bit is writable only when the channel is OFF (CC4E = 0 in TIM2_CCER).
7	OC3CE	R/W	0x0	Output Compare 3 clear enable Refer to OC1CE description.
6:4	OC3M [2:0]	R/W	0x0	Output Compare 3 mode Refer to OC1M description.
3	OC3PE	R/W	0x0	Output Compare 3 preload enable Refer to OC1PE description.
2	OC3FE	R/W	0x0	Output Compare 3 fast enable Refer to OC1FE description.

1:0	CC3S [1:0]	R/W	0x0	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC3S bit is writable only when the channel is OFF (CC3E = 0 in TIM2_CCER).

Input capture mode:

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:12	IC4F	R/W	0x0	Input capture 4 filter Refer to IC1F description.
11:10	IC4PSC	R/W	0x0	Input capture 4 prescaler Refer to IC1PSC description.
9:8	CC4S	R/W	0x0	Capture/Compare 4 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC4 channel is configured as output 01: CC4 channel is configured as input, IC4 is mapped on TI4 10: CC4 channel is configured as input, IC4 is mapped on TI3 11: CC4 channel is configured as input, IC4 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC4S bit is writable only when the channel is OFF (CC4E = 0 in TIM2_CCER).
7:4	IC3F	R/W	0x0	Input capture 3 filter Refer to IC1F description.
3:2	IC3PSC	R/W	0x0	Input capture 3 prescaler Refer to IC1PSC description.
1:0	CC3S	R/W	0x0	Capture/Compare 3 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC3 channel is configured as output 01: CC3 channel is configured as input, IC3 is mapped on TI3 10: CC3 channel is configured as input, IC3 is mapped on TI4 11: CC3 channel is configured as input, IC3 is mapped on TRC. This mode is working only if an internal trigger input is selected through TS bit (TIM2_SMCR register) Note: CC3S bit is writable only when the channel is OFF (CC3E = 0 in TIM2_CCER).

12.10.9 TIM2 Capture/Compare Enable Register(TIM2_CCER)

Bit	Name	R/W	Reset	Description
31:14	Reserved	--	0x0	Always read as 0.
13	CC4P	R/W	0x0	Capture/Compare 4 output polarity Refer to CC1P description.
12	CC4E	R/W	0x0	Capture/Compare 4 output enable Refer to CC1E description.
11:10	Reserved	--	0x0	Always read as 0.
9	CC3P	R/W	0x0	Capture/Compare 3 output polarity Refer to CC1P description.
8	CC3E	R/W	0x0	Capture/Compare 3 output enable Refer to CC1E description.
7:6	Reserved	--	0x0	Always read as 0.
5	CC2P	R/W	0x0	Capture/Compare 2 output polarity Refer to CC1P description.
4	CC2E	R/W	0x0	Capture/Compare 2 output enable Refer to CC1E description.
3:2	Reserved	--	0x0	Always read as 0.
1	CC1P	R/W	0x0	Capture/Compare 1 output polarity CC1 channel configured as output: 0: OC1 active high 1: OC1 active low CC1 channel configured as input: This bit selects whether IC1 or inverted IC1 is used for trigger or capture operations. 0: non-inverted: capture is done on a rising edge of IC1. When used as external trigger, IC1 is non-inverted. 1: inverted: capture is done on a falling edge of IC1. When used as external trigger, IC1 is inverted. Note: This bit cannot be modified as long as LOCK level 2 or 3 has been programmed (LOCK bits in TIM2_BDTR register).

0	CC1E	R/W	0x0	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 output level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. 1: On - OC1 signal is output on the corresponding output pin. Its output level depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits. CC1 channel configured as input: This bit determines if a capture of the counter value can occur in the TIM2_CCR1 register. 0: Capture disabled 1: Capture enabled

CCxE bit	OCx output state
0	Output Disabled (OCx=0, OCx_EN=0)
1	OCx=OCxREF + Polarity, OCx_EN=1

Table 12-8 Output control bits for standard OCx channels

Note: The state of the external I/O pin connected to the standard OCx channel depends on the OCx channel state and the GPIO and AFIO registers.

12.10.10 TIM2 Counter Register(TIM2_CNT)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CNT	R/W	0x0	Counter value

12.10.11 TIM2 Prescaler Register(TIM2_PSC)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	PSC	R/W	0x0	Prescaler value The counter clock frequency (CK_CNT) is equal to fCK_PSC / (PSC[15:0] + 1). PSC contains the value to be loaded in the active prescaler register at each update event (including when the counter is cleared through UG bit of TIM_EGR register or through trigger controller when configured in "reset mode").

12.10.12 TIM2 Auto-Reload Register(TIM2_ARR)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	ARR	R/W	0x0	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to the section concerning ARR update and behavior for more details. The counter is blocked if the auto-reload value is null.

12.10.13 TIM2 Capture/Compare Register 1(TIM2_CCR1)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CCR1	R/W	0x0	Capture/Compare 1 value If the CC1 channel is configured as output: CCR1 contains the value to be loaded in the active capture/compare 1 register (preload value). If the preload feature is not selected in the TIM2_CCMR1 register (OC1PE bit), the written value is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 1 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM2_CNT and generates the output signal on OC1. If the CC1 channel is configured as input: CCR1 contains the counter value transferred by the last input capture 1 event (IC1).

12.10.14 TIM2 Capture/Compare Register 2(TIM2_CCR2)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CCR2	R/W	0x0	Capture/Compare 2 value If the CC2 channel is configured as output: CCR2 contains the value to be loaded in the active capture/compare 2 register (preload value). If the preload feature is not selected in the TIM2_CCMR1 register (OC2PE bit), the written value is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 2 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM2_CNT and generates the output signal on OC2. If the CC2 channel is configured as input: CCR2 contains the counter value transferred by the last input capture 2 event (IC2).

12.10.15 TIM2 Capture/Compare Register 3(TIM2_CCR3)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CCR	R/W	0x0	Capture/Compare 3 value If the CC3 channel is configured as output: CCR3 contains the value to be loaded in the active capture/compare 3 register (preload value). If the preload feature is not selected in the TIM2_CCMR2 register (OC3PE bit), the written value is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 3 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM2_CNT and generates the output signal on OC3. If the CC3 channel is configured as input: CCR3 contains the counter value transferred by the last input capture 3 event (IC3).

12.10.16 Capture/Compare Register 4(TIM2_CCR4)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:0	CCR4	R/W	0x0	Capture/Compare 4 value If the CC4 channel is configured as output: CCR4 contains the value to be loaded in the active capture/compare 4 register (preload value). If the preload feature is not selected in the TIM2_CCMR2 register (OC4PE bit), the written value is transferred immediately to the active register. Otherwise the preload value is transferred to the active capture/compare 4 register only when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIM2_CNT and generates the output signal on OC4. If the CC4 channel is configured as input: CCR4 contains the counter value transferred by the last input capture 4 event (IC4).

12.10.17 TIM2 Capture/Compare Down Register 1 (TIM2_CCDR1)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCDR1	R/W	0x0	TIM2 capture/compare down value (Channel 1 value) (Valid when asymmetric counting is enabled in TIM2 center-aligned mode)

12.10.18 TIM2 Capture/Compare Down Register 2 (TIM2_CCDR2)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCDR2	R/W	0x0	TIM2 capture/compare down value (Channel 2 value) (Valid when asymmetric counting is enabled in TIM2 center-aligned mode)

12.10.19 TIM2 Capture/Compare Down Register 3 (TIM2_CCDR3)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCDR3	R/W	0x0	TIM2 capture/compare down value (Channel 3 value) (Valid when asymmetric counting is enabled in TIM2 center-aligned mode)

12.10.20 TIM2 Capture/Compare Down Register 4 (TIM2_CCDR4)

Bit	Name	R/W	Reset	Description
31:20	Reserved	--	0x0	Always read as 0.
19:0	CCDR4	R/W	0x0	TIM2 capture/compare down value (Channel 4 value) (Valid when asymmetric counting is enabled in TIM2 center-aligned mode)

13. Watchdog (WDG)

13.1 Functions

13.1.1. WDG clock source

The watchdog timer (WDG) operates on the on-chip OSC_32K clock and is a 32-bit decrementing counter. The counter is initially loaded into the reload setting register when the WDG starts, and then decrements each cycle driven by the clock.

13.1.2. Start WDG

After power-on reset, the WDG module is disabled by default. Application software must set the WDGCR_INTE bit to enable WDG operation. Setting this bit to 1 also means the system will reset the WDG counter to zero.

13.1.3. WDG interruption

After the WDG counter starts, a WDG interrupt will be triggered when the count value first reaches zero. Once a WDG interrupt is triggered, the system will be reset after another WDG timeout period. Application software can write its own non-maskable interrupt service routine to handle this WDG interrupt event. If a WDG interrupt occurs, it generally means that the application software has not performed the watchdog clearing operation for too long, which can be used for reliability diagnosis during the software debugging phase. Released applications cannot execute the watchdog clearing instruction in their non-maskable interrupt service routines; otherwise, the watchdog will never trigger a system reset, losing its system monitoring function.

13.1.4. WDG Reset

After the WDG counter reaches zero once, it is reloaded with its initial value and continues to decrement. When the count value reaches zero a second time and the WDGCR_RSTE bit is set to 1, a system hardware reset, i.e., a watchdog reset, will be immediately triggered.

13.1.5. WDG Timer Setting

The setting of the watchdog reload register WDG_LOAD directly determines the WDG's timing interval. The formula for calculating the WDG_LOAD reload value, based on the application's desired watchdog timing interval, is as follows:

$WDG\_LOAD = T_{WDG} * f_{CLK} - 1$

In:

T_{WDG} is the desired WDG timing time, in milliseconds (ms).

f_{CLK} is the WDG clock frequency, in kHz, 32kHz.

For example, if the application software wants to set a watchdog timer of approximately 20ms, with an internal 32kHz clock, then according to the above formula, fCLK = 32, and with guaranteed precision, WDGLOAD should be assigned the value 639.

To clear the WDG, the application software must write any number to the WDG_INTCLR register once within the set WDG timer range during normal operation to reload and reset the WDG counter. Under normal circumstances, a WDG interrupt should not occur, and a WDG reset should absolutely not occur.

13.2 Register table

Address	Name	Description
0x4000_4000	WDG_LOAD	WDG Count reload register
0x4000_4004	WDG_VALUE	WDG Count value register
0x4000_4008	WDG_CR	WDG Control Register
0x4000_400C	WDG_INTCLR	WDG Interrupt Clear Register
0x4000_4010	WDG_RIS	WDG Raw interrupt flag register
0x4000_4014	WDG_MIS	WDG Masking interrupt flag register
0x4000_4400	WDG_LOCK	WDG Lock control register

Table 13-1 WDG register table

13.3 Register description

13.3.1. WDG count reload register (WDG_LOAD)

Bit	Name	R/W	Reset	Description
31:0	WDG_LOAD	R/W	0xFFFFFFFF	This 32-bit register is used to set the initial reload value of the decrementing counter after the WDG starts or is reset. It determines the WDG's overflow reset time.

13.3.2. WDG Count value register (WDG_VALUE)

Bit	Name	R/W	Reset	Description
31:0	WDG_VALUE	R	0xFFFFFFFF	This 32-bit wide register holds the current count value of the WDG when it is operating. The WDG_LOAD setting is loaded into this register when the WDG is started or after it is cleared, and then the count is decremented under the drive of the WDG clock. The watchdog value register indicates the current count value of the decrementing counter.

13.3.3. WDG Control Register (WDG_CR)

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2	DBGE	R/W	0x0	WDG Debug enable control bit 0: WDG is suspended when debugging is paused 1: WDG continues working when debugging is paused
1	RSTE	R/W	0x0	WDG Reset enable control bit 0: WDG reset disabled. 1: WDG reset enabled.
0	INTE	RW	0x0	INTE WDG Interrupt enable control bit 0: WDG module disabled, no interrupt response. 1: WDG module enabled, and WDG interrupt (NMI) enabled simultaneously.

13.3.4. WDG Interrupt Clear Register (WDG_INTCLR)

This register is write-only. Writing any data to it will reload the value of WDG_LOAD into the WDG_VALUE register, restarting the decrementing count and resetting the WDG interrupt flag (if it is already set to 1). After configuring and starting the WDG module, the application software must perform a write operation to this register before the WDG count reaches zero to reset the WDG.

13.3.5. WDG Raw interrupt flag register (WDG_RIS)

This register is read-only. Its least significant bit, RIF, is the original flag for the WDG counter to return to zero interrupt. Whenever the WDG counter returns to zero, this bit is set to 1 regardless of the WDG_CR_INTE setting. Writing to the WDG_INTCLR register will clear RIF.

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	RIF	R	0x0	WDG Raw interrupt flag bit 0: WDG has not been zeroed 1: WDG has been zeroed (but whether it responds to interrupts depends on the WDG_CR.INTE setting). Write WDG_INTCLR to clear it to 0.

13.3.6. WDG Masking interrupt flag register (WDG_MIS)

This register is read-only. Its least significant bit, MIF, represents the state of the WDG count reset interrupt original flag RIF after being masked by INTE. If both RIF and INTE are 1, then MIF is 1, triggering a WDG interrupt; otherwise, it is 0, and no WDG interrupt is generated. Writing to the WDG_INTCLR register clears RIF, and also clears MIF at the same time.

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	MIF	R	0x0	WDG masking interrupt flag information bit 0: WDG masking interrupt flag is invalid (possibly due to no actual interrupt, or WDG_CR.INTE not being enabled). 1: WDG masking interrupt flag is valid; write WDG_INTCLR to clear it to 0.

13.3.7. WDG Lock control register (WDG_LOCK)

This register provides a mechanism for locking the WDG system, preventing accidental modification of the WDG module's configuration during system operation due to software design flaws or program crashes. Once the WDG module is locked, all its registers cannot be modified by software.

Writing the special value "0x1ACCE551" to the WDGLOCK register unlocks the WDG module, allowing software to freely configure related registers. Writing any other value to this register locks the WDG module, preventing any configurable registers within the module from being modified.

When reading this register, only the least significant bit is meaningful, indicating the WDG's locked state..

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	LOCK	R	0x0	WDG Module locked status bit 0: The WDG module is unlocked; related registers within the module can be modified. 1: The WDG module is locked; all registers within the module cannot be modified.

13.4 User Operations

There is synchronization logic between the two clock domains. When the watchdog timer loads and the control register is updated by program operation, the new value in the WDG_CLK clock domain will take effect within two WDG_CLK cycles. When the watchdog timer counts in WDG_CLK, the synchronization logic first locks the counter value on WDG_CLK, and then synchronizes with the system CLK so that the CPU can read the watchdog value register. Clearing the watchdog interrupt register also requires two WDG_CLK cycles and two PCLK clock synchronizations, which must be handled carefully by the software.

14. ADC

14.1 Overview

The chip contains a 12-bit Analog-to-Digital Converter (ADC). This ADC has 16 channels, allowing the ADC to measure 12 external input pin voltages, as well as other internal voltage channels.

Requirement: The sampling interval must be greater than 12 times the clk main frequency time.

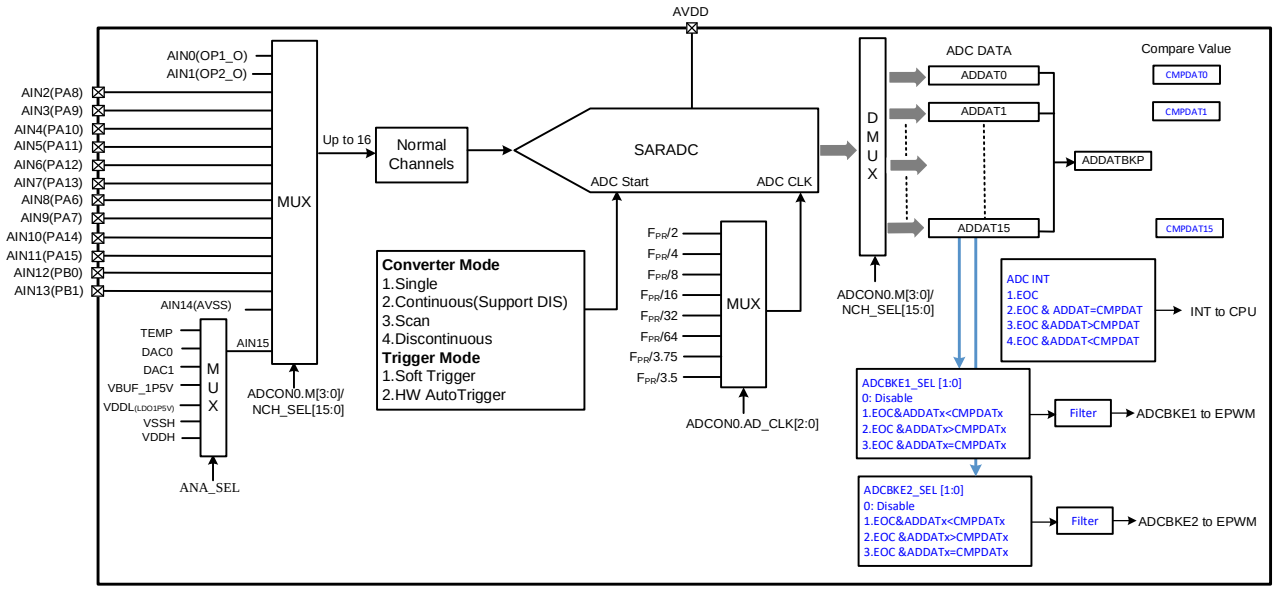


Fig. 14-1 ADC Block Diagram

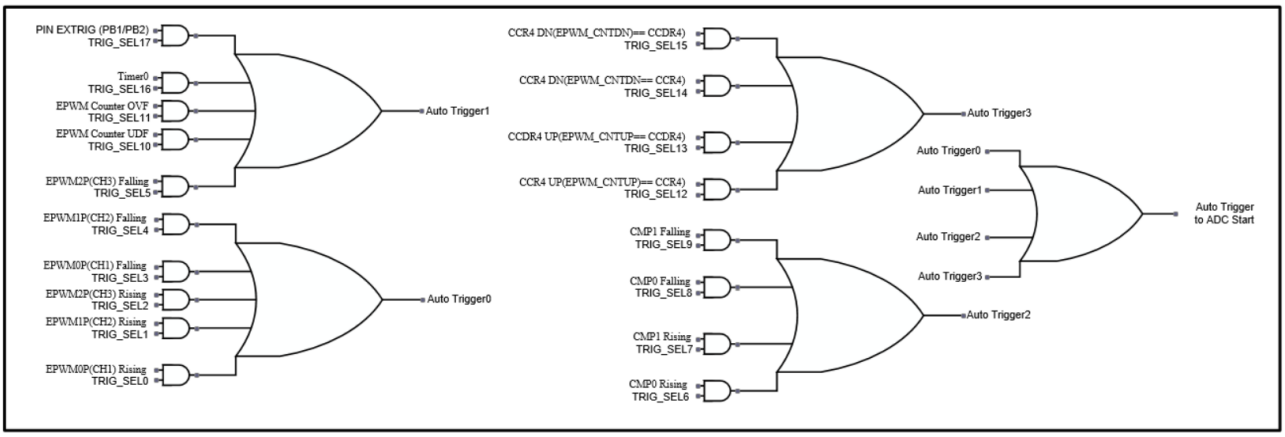


Fig. 14-2 ADC Auto Trigger Block Diagram

14.2 Features

- 12-bit analog-to-digital conversion resolution
- - -40~105 degrees Celsius
- Absolute accuracy: 12-bit configuration
- Max 16MHz main frequency input, 1Msps sampling rate. (Min 937.5KHz main frequency input)
- Up to 12 single-ended external analog signal inputs
- Provides Single Mode, Continuous Mode, Scan Mode, and Discontinuous Mode
- **Auto Trigger Mode (Timer0, EPWM, CMP, External Input PIN)**
- Internal channels include:
    - VDD
    - GND
    - TEMP,DAC0/1,LDO 1.5V
- Input voltage range: 0~Vdd

14.3 Conversion Timing

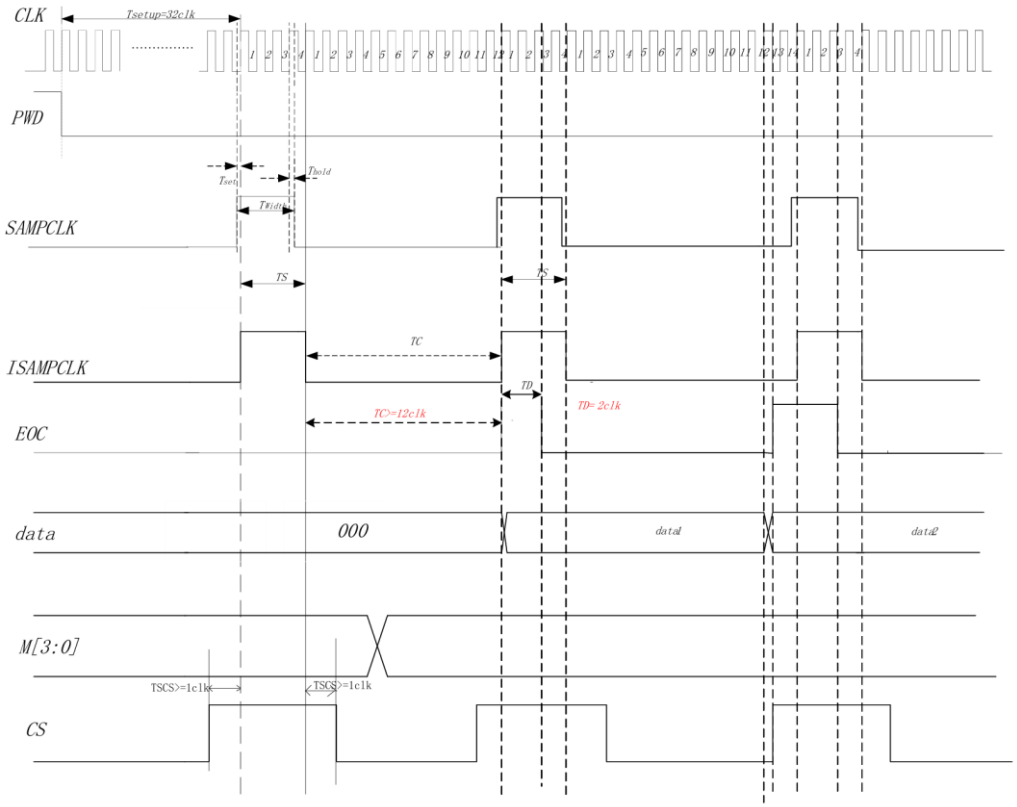


Fig. 14-3 ADC Sample Timing

Note: To achieve 12-bit conversion at a 1MSPS sampling rate, the mode without waiting for EOC must be used; otherwise, the efficiency requirement cannot be met

14.4 Functional Description

Conversion mode settings are shown in the following table:

ADC_CON0.CONTINUE	ADC_CON0.ENCONT	ADC_CHSEL.DISCEN	Functional Description
0	X	X	**Single Mode** (ADC_CON0.M[3:0] selects the channel)
1	1	X	**Continuous Mode** (ADC_CHSEL[15:0] selects channels) (Triggers once, converts continuously/infinitely)
1	0	0	**Scan Mode** (ADC_CHSEL[15:0] selects channels) (Number of conversions is configurable)
1	0	1	**Discontinuous Mode** (ADC_CHSEL[15:0] selects channels) (Triggers conversions for set channels multiple times. When all set channels are completed, it resets/loops)

Table 14-1 Conversion mode table

Single Mode

In Single Mode, the ADC performs only one conversion after startup, and can convert on all 16 ADC channels. This mode can be started by setting the ADC_CON0.START bit or by external trigger through setting ADC_TRGSEL.TRIG_SELx[2:0] or ADC_CHSEL.TRIG_SELx[2:0]. Once the channel is selected by ADCCON0.M[3:0] and the ADC is triggered, the conversion completes, the ADCCON0.START bit is automatically cleared, and the conversion result is stored in the ADDAT0~15 registers.

- Example: If M[3:0] channel is selected, set ADC_CON0.START or use external trigger start (TRIG_SELx[2:0])

- Conversion data is stored in the 16-bit ADC_DATx registers
- DONE (End of Conversion) flag is set
- If INT_EN=0x1 is set, an interrupt is generated.

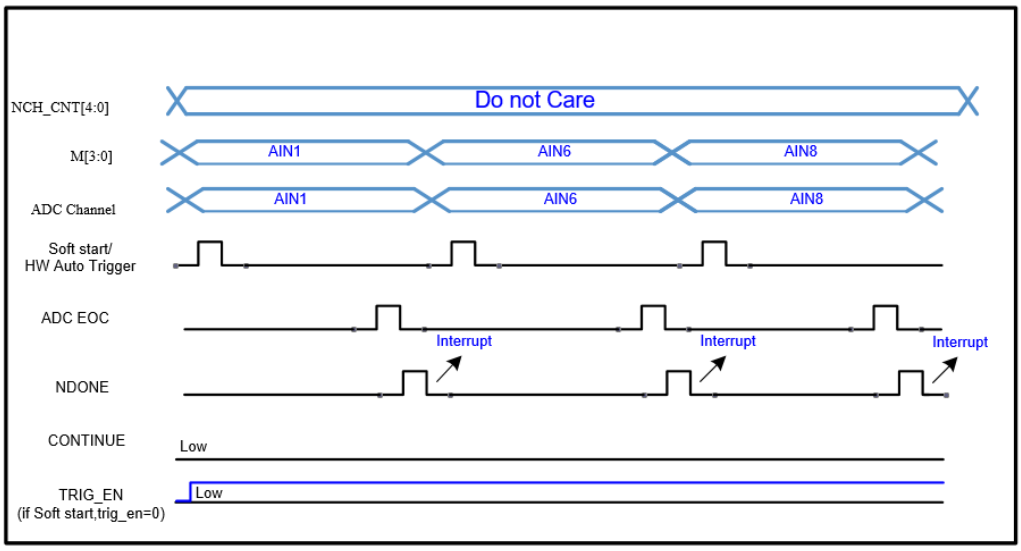


Fig. 14-4 Single Mode Timing

14.4.1 Continuous Mode

In Continuous Mode, another conversion starts immediately after the previous ADC conversion finishes. This mode can be started by external trigger or by setting the ADCCON0.START bit. In this case, ADCCON0.CONTINUE and ADCCON0.ENCONT bits must be 1.

- Continuous conversion loops infinitely, so the NDONE (End of Conversion) flag will remain 0 (This mode does not support interrupts)

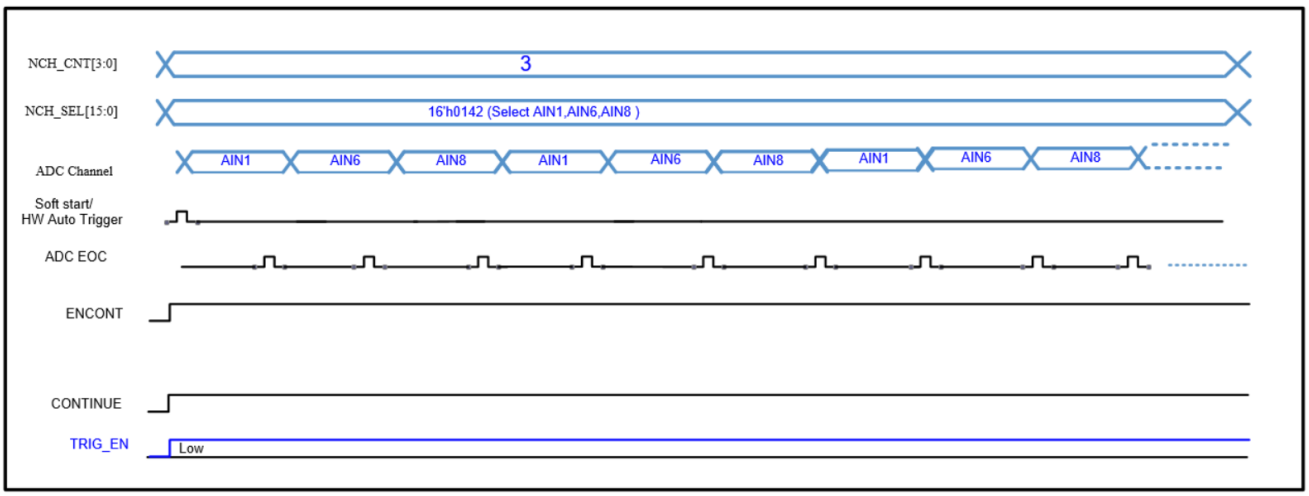


Fig. 14-5 Continuous Mode Unlimited Timing

14.4.2 Scan Mode

If a channel is converted: (Based on ADC_CHSEL.CH_SEL channel selection, ADC_CON0.CONTINUE=1, ADC_CON0.ENCONT=0, ADC_CHSEL.DISCEN=0)

- CH_SEL[15:0]=0x0142 (AIN1, 6, 8; Lower channels have higher conversion priority)

ADC_CHSEL.CH_CNT=0x2 (3 conversions)

- First Trigger: Convert AIN1, AIN6, AIN8. Conversion data is stored in 16-bit ADDAT1, 6, 8 registers
- DONE (End of Conversion) flag is set
- If INT_EN=0x1 is set, an interrupt is generated.

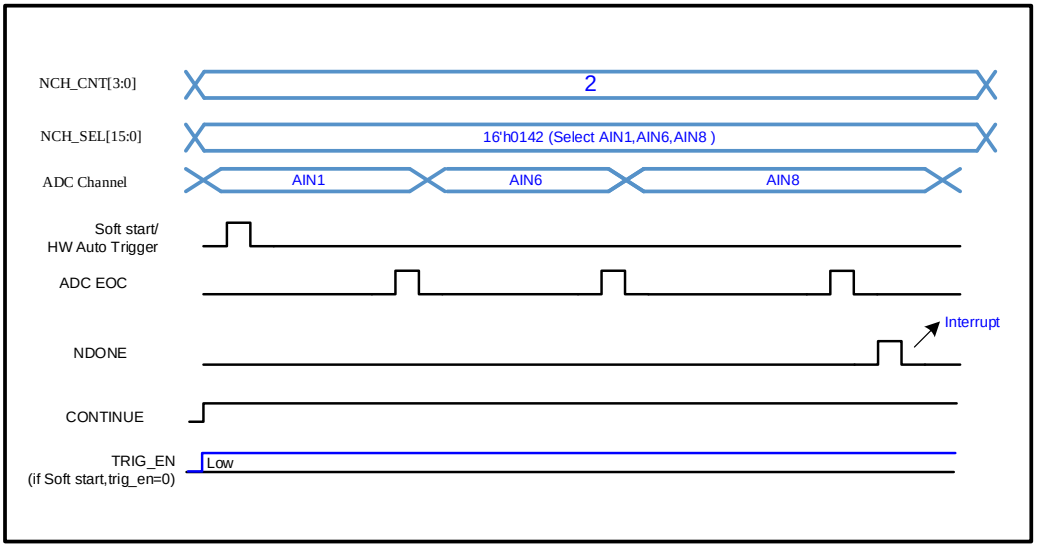


Fig. 14-6 Scan Mode Channel Conversion Timing

- When conversion count CH_CNT > quantity of channels selected by CH_SEL, it will repeat conversion of channels selected by CH_SEL until CH_CNT conversions are completed
- CH_SEL[15:0]=0x0101 (AIN0, 8; Lower channels have higher conversion priority) ADC_CHSEL.CH_CNT=3 (4 conversions)
- First Trigger: Repeatedly convert AIN0, AIN8, AIN0, AIN8. Conversion data is stored in 16-bit ADDAT0, 8 registers
- DONE (End of Conversion) flag is set
- If INT_EN=0x1 is set, an interrupt is generated.

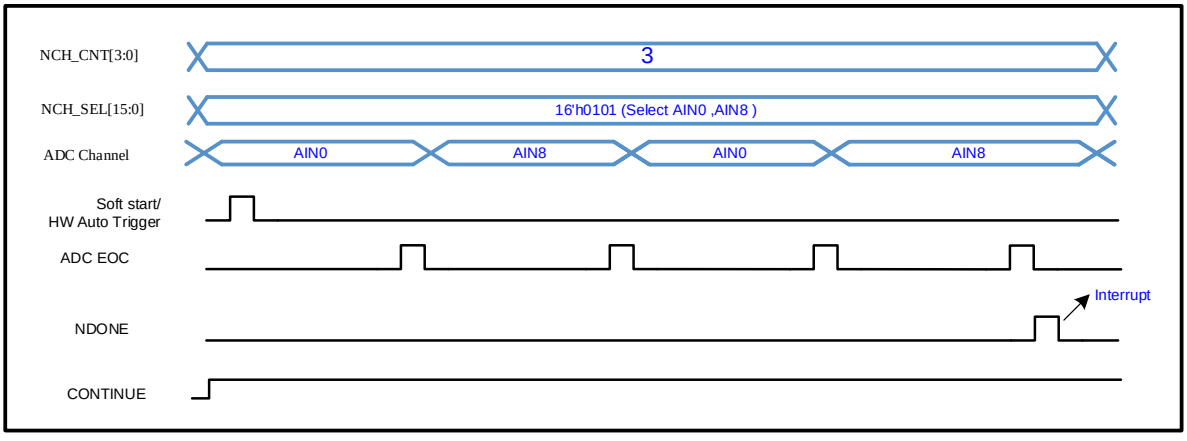


Fig. 14-7 Scan Mode Channel Conversion Timing 2

14.4.3 Discontinuous Mode

(1) When ADC_CHSEL.DISCEN =1, ADC_CON0.CONTINUE=1,ADC_CHSEL.DISCNUM[2:0]=3,
ADC_CHSEL.CH_CNT[3:0]=11,

CH_SEL[15:0]=0xDEB3 (AIN0,1,4,5,7,9,10,11,12,14,15 Lower channels have higher conversion priority)

Continuous conversion is divided into 4 segments to complete (CH_CN[3:0] / DISCNUM[2:0] = 3 with a remainder of 2. Total triggers required for all channel conversions: 3 + 1 = 4)

- ADC_CHSEL.DISC_INTSEL = 0 (Interrupt occurs after all channel conversions are completed)
  ADC_CON0.ENCONT = 0

- First Trigger: Convert AIN0, AIN1, AIN4 (Conversion data is stored in 16-bit ADC_DAT0, 1, 4 registers)
- Second Trigger: Convert AIN5, AIN7, AIN9 (Conversion data is stored in 16-bit ADC_DAT5, 7, 9 registers)
- Third Trigger: Convert AIN10, AIN11, AIN12 (Conversion data is stored in 16-bit ADC_DAT10, 11, 12 registers)
- Fourth Trigger: Convert AIN14, AIN15 (Conversion data is stored in 16-bit ADC_DAT14, 15 registers)
- DONE (End of Conversion) flag is set
  → If INT_EN = 0x1 is set, an interrupt is generated.
- Fifth Trigger: Return to convert AIN0, AIN1, AIN4

- DISC_INTSEL = 1 (Interrupt occurs after every trigger conversion completion)

- First Trigger: Convert AIN0, AIN1, AIN4 (Conversion data is stored in 16-bit ADC_DAT0, 1, 4 registers)
  → If INT_EN = 0x1 is set, an interrupt is generated.
- Second Trigger: Convert AIN5, AIN7, AIN9 (Conversion data is stored in 16-bit ADC_DAT5, 7, 9 registers)
  → If INT_EN = 0x1 is set, an interrupt is generated.
- Third Trigger: Convert AIN10, AIN11, AIN12 (Conversion data is stored in 16-bit ADC_DAT10, 11, 12 registers)
  → If INT_EN = 0x1 is set, an interrupt is generated.
- Fourth Trigger: Convert AIN14, AIN15 (Conversion data is stored in 16-bit ADC_DAT14, 15 registers)
- DONE (End of Conversion) flag is set
  → If INT_EN = 0x1 is set, an interrupt is generated.
- Fifth Trigger: Return to convert AIN0, AIN1, AIN4
  → If INT_EN = 0x1 is set, an interrupt is generated.

(2) DISCEN =1 , CONTINUE=1, DISCNUM[2:0]=1, CH_CN[3:0]=2
CH_SEL[15:0]=16'b0000_0000_0000_0001 (Only Select AIN0)
Continuous conversion is divided into 2 segments to complete (CH_CN[3:0] / DISCNUM[2:0] = 2)

- DISC_INTSEL = 0 (Interrupt occurs after all channel conversions are completed)
  - First Trigger: Convert AIN0 (First trigger conversion data is stored in 16-bit ADC_DAT0 register)
  - Second Trigger: Convert AIN0 (Second trigger conversion data is stored in 16-bit ADC_DAT0 register, and the first trigger value ADC_DAT0 is backed up in the ADC_BAKDAT register)

  DONE (End of Conversion) flag is set

  → If INT_EN = 0x1 is set, an interrupt is generated.
- DISC_INTSEL = 1 (Interrupt occurs after every trigger conversion completion)
  - First Trigger: Convert AIN0 (First trigger conversion data is stored in 16-bit ADC_DAT0 register)

    → If INT_EN = 0x1 is set, an interrupt is generated.
  - Second Trigger: Convert AIN0 (Second trigger conversion data is stored in 16-bit ADC_DAT0 register, and the first trigger value ADC_DAT0 is backed up in the ADC_BAKDAT register)

    → If INT_EN = 0x1 is set, an interrupt is generated.

    DONE (End of Conversion) flag is set

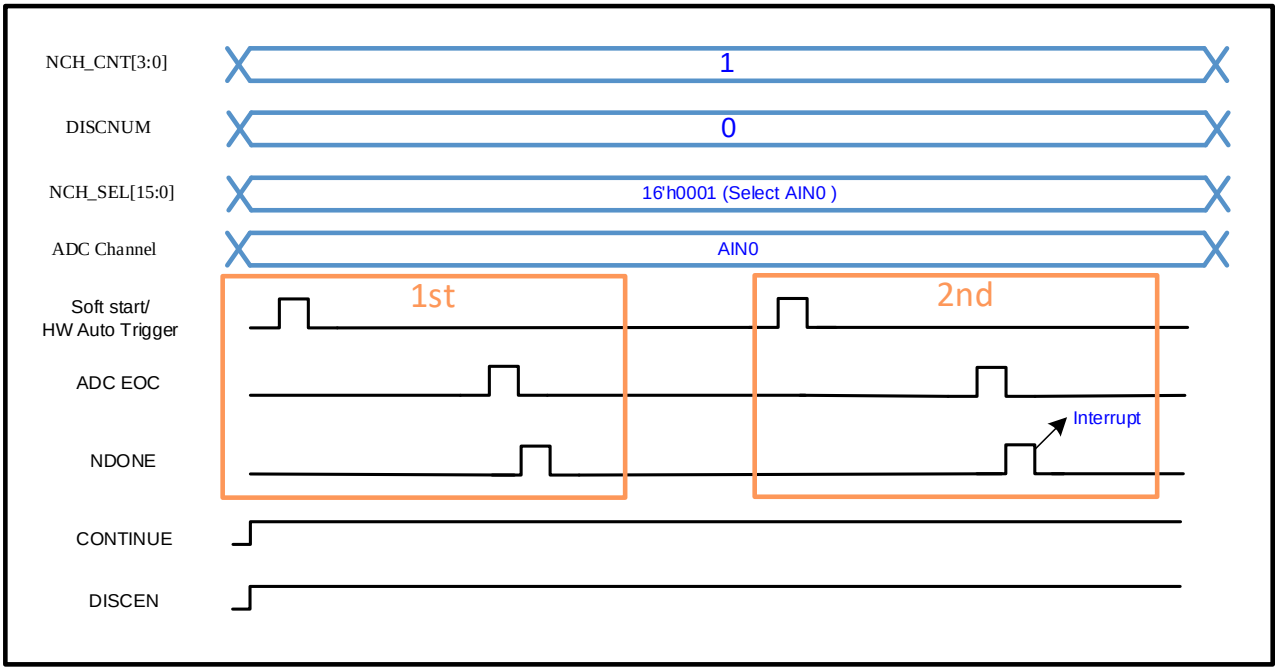


Fig. 14-8 Discontinuous Mode Conversion Same Channel Timing (INT_EN=0x1, DISC_INTSEL=0)

14.4.4 Comparison and Interrupts

The ADC interrupt is enabled by ADC_CON0.INT_EN. When INT_EN = 0x1, the ADC interrupt is enabled when ADC_STAT.DONE is set to 1.

In Continuous Conversion Mode, Scan Mode, or Discontinuous Mode, the DONE flag is set only after the last group of conversions is completed. Therefore, the ADC interrupt is generated only after the completion of the last group of conversions.

The interrupts generated by comparison results (INT_EN = 0x2, 0x4, 0x8) occur at the same time as when INT_EN = 0x1. When the DONE flag is 1, if the comparison result in ADC_STAT.comp_result matches the condition set by INT_EN, an ADC interrupt is generated.

1. Read the data result register (ADC_DATx). This does not include ADC_BAKDAT.
2. Write 1 to ADC_STAT.INTCLR.

If the DONE flag is not cleared, it will be automatically cleared upon the next START trigger.

(1) If the DONE flag is not cleared, it will be automatically cleared upon the next START trigger.

When the threshold (COMP_TH) is set to 0x555 for all channels, and AIN1, AIN6, and AIN8 are converted respectively, the results of interrupt generation and timing for three triggers under different INT_EN settings are shown in the table and figure below. The orange boxes represent the timing and data for the three trigger conversions:

Single Mode	1st NDONE	2nd NDONE	3rd NDONE
Comp_result	01 (TH<DATA)	11 (TH=DATA)	10 (TH>DATA)
INT_EN=0x1 (Interrupt generated for all DONE)	Interrupt Generated	Interrupt Generated	Interrupt Generated
INT_EN=0x2 (Interrupt generated on DONE when COMP_TH < ADC_DATA)	Interrupt Generated	No Interrupt Generated	No Interrupt Generated
INT_EN=0x4 (Interrupt generated on DONE when COMP_TH > ADC_DATA)	No Interrupt Generated	No Interrupt Generated	Interrupt Generated
INT_EN=0x8 (Interrupt generated on DONE when COMP_TH = ADC_DATA)	No Interrupt Generated	Interrupt Generated	No Interrupt Generated

Table 14-2 Single Conversion Mode Interrupt Generation Table

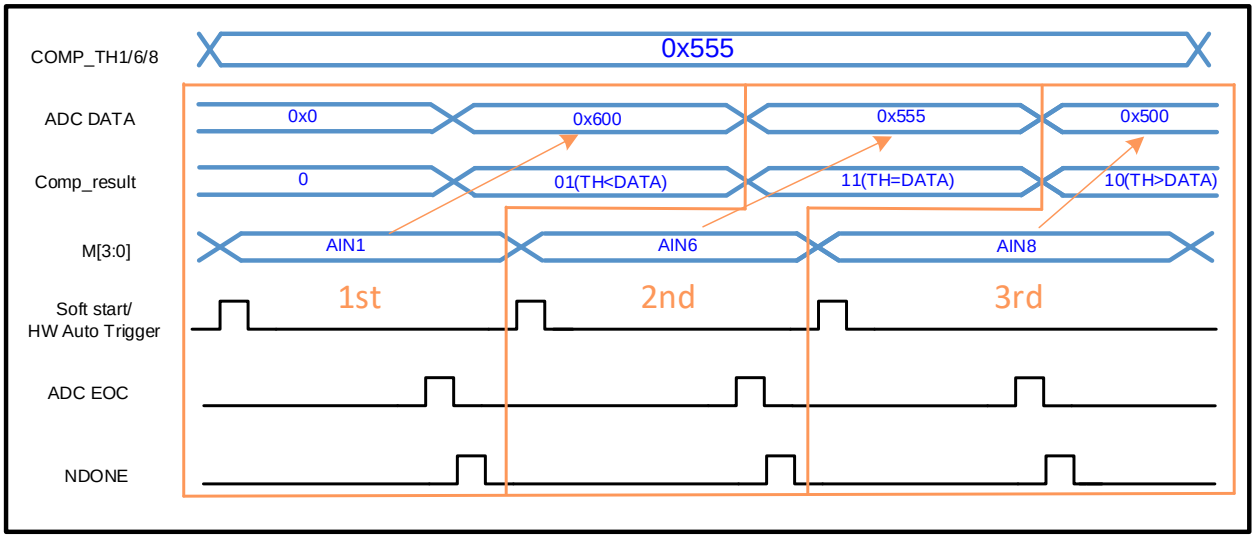


Fig. 14-9 Single Mode Interrupt Generation Timing

(2) In Continuous Mode, Scan Mode, or Discontinuous Mode, since the DONE flag is enabled only after the last group of channel conversions is completed, the channels converted during the process cannot generate interrupts.

In Discontinuous Mode, with CH_CNT = 3, DISCNUM = 1, and INT_EN = 0x1, two START generations convert two channels respectively. If DISC_INTSEL = 0 is set, an interrupt is generated only after the last conversion is completed. However, if a comparison result interrupt is set (INT_EN = 0x8, 0x4, 0x2), regardless of the DISC_INTSEL setting, an interrupt is generated after every conversion completion based on the comparison result. When the threshold (COMP_TH) is set to 0x555 for all channels, and AIN0 and AIN8 are converted respectively, the results of interrupt generation and timing for two triggers under different INT_EN settings are shown in the table and figure below. Note that the conversion of AIN0 does not generate DONE, so it does not generate an interrupt:

Discontinuous Mode	DISC_INSEL	1st DONE	2nd DONE
Comp_result		10 (TH>DATA)	01 (TH<DATA)
INT_EN=0x1	0 (Interrupt generated when CH_CNT conversion completes)	No Interrupt Generated	Interrupt Generated
	1 (Interrupt generated for all DONE)	Interrupt Generated	Interrupt Generated
INT_EN=0x2 (Interrupt generated on DONE when COMP_TH < ADC_DATA)	X	No Interrupt Generated	Interrupt Generated
INT_EN=0x4 (Interrupt generated on DONE when COMP_TH > ADC_DATA)	X	Interrupt Generated	No Interrupt Generated
INT_EN=0x8 (Interrupt generated on DONE when COMP_TH = ADC_DATA)	X	No Interrupt Generated	No Interrupt Generated

Table 14-3 Discontinuous Mode Interrupt Generation Table

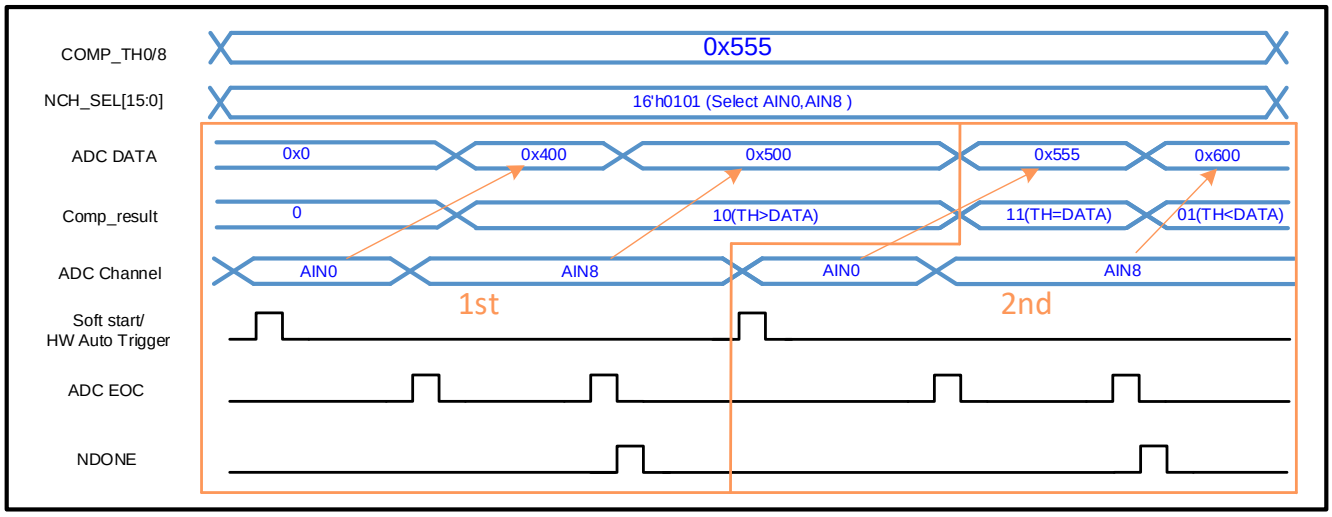


Fig. 14-10 Discontinuous Mode Interrupt Generation Timing

14.5 User Operation

As shown in the timing diagram above, regardless of Single Mode or Continuous Mode, after enabling the ADC IP (setting ADC_PWD in the ADC_CON0 register to 0), a software delay of 32 * adc_clk is required to wait for the ADC internal reference source to stabilize before starting the ADC conversion.

The ADC has four conversion modes:

14.5.1 Single Mode

1)Software writes 0 to ADC_CON0.CONTINUE to select Single Mode.

2)Software configures the ADC by writing to the control bits in the ADC_CON0 register (M[3:0] channel selection, CLK).

3)Software writes 1 to the START bit in the ADC_CON0 register to start a conversion. Upon completion, the ADC generates an A/D conversion completion flag (ADC_STAT.DONE is set to 1). After the software reads the ADC_DAT register, the DONE bit in the ADC_STAT register is automatically cleared to 0.

14.5.2 Continuous Mode

1） Software configures the ADC by writing to the control bits (CLK) in the ADC_CON0 register.

2） Channel selection is configured via CH_SEL[15:0] in the ADC_CHSEL register.

3)Software writes 1 to CONTINUE and 1 to ENCONT in the ADC_CON0 register to select Continuous Mode.

4)After selecting Continuous Mode, software can use the START bit in the ADC_CON0 register for software triggering or use hardware triggering to start continuous channel conversion.

5)The ADC will continuously convert the channels selected by CH_SEL. It will not stop conversion and set DONE to 1 until ENCONT or CONTINUE is disabled.

Note 1: Since Continuous Mode never stops conversion, DONE remains 0, and interrupt signal generation is not supported.

Note 2: ADC 1MSPS Sampling Rate Description To achieve a 1MSPS sampling rate at 12-bit resolution, the mode that does not wait for EOC feedback must be used (ADC_STAT.EOC_CHECK_DIS = 0x1). Otherwise, 1MSPS cannot be achieved.

When using the mode that does not wait for EOC feedback, Continuous Mode sampling must be enabled along with the no-wait EOC feedback mode. In this case, initiating a single start is sufficient.

14.5.3 Scan Mode

1） Software configures the ADC by writing to the CLK bits in the ADC_CON0 register.

2） Channel selection is configured via CH_SEL[15:0] in the ADC_CHSEL register, and the number of conversions is set by CH_CNT.

3)Software writes 1 to CONTINUE, 0 to ENCONT, and 0 to DISCEN in the ADC_CON0 register to enable Scan Mode.

4)After selecting Scan Mode, software can use the START bit in the ADC_CON0 register for software triggering or use hardware triggering to start continuous channel conversion.

5)Conversion stops and DONE is set to 1 after CH_CNT conversions are completed.

Note 1: When CH_CNT > the number of channels selected by CH_SEL, the channels selected by CH_SEL are converted repeatedly until CH_CNT conversions are completed. Refer to Section 14.4.3.

Note 2: When CH_CNT < the number of channels selected by CH_SEL, only CH_CNT conversions are performed, and the channels selected by CH_SEL will not be fully converted. The next trigger will start conversion from the highest priority channel again and will not resume from the previous channel.

14.5.4 Discontinuous Mode

1） Software configures the ADC by writing to the CLK bits in the ADC_CON0 register.

2） Channel selection is configured via CH_SEL[15:0] in the ADC_CHSEL register. The total number of conversions is set by CH_CNT, and the number of conversions per trigger is set by DISCNUM.

3)Software writes 1 to CONTINUE, 0 to ENCONT, and 1 to DISCEN in the ADC_CON0 register to enable Discontinuous Mode.

4)After selecting Discontinuous Mode, software can use the START bit in the ADC_CON0 register for software triggering or use hardware triggering to start continuous channel conversion.

5)Each trigger performs DISCNUM conversions and sets DONE to 1. After completing CH_CNT conversions in total, conversion stops and DONE is set to 1.

14.6 Register table

Address	Name	Description
0x4000_4800	ADC_CON0	ADC Control Register 0
0x4000_4804	ADC_STAT	ADC Status Register
0x4000_4808	ADC_DAT0	ADC DATA Register 0
0x4000_480C	ADC_DAT1	ADC DATA Register 1
0x4000_4810	ADC_DAT2	ADC DATA Register 2
0x4000_4814	ADC_DAT3	ADC DATA Register 3
0x4000_4818	ADC_DAT4	ADC DATA Register 4
0x4000_481C	ADC_DAT5	ADC DATA Register 5
0x4000_4820	ADC_DAT6	ADC DATA Register 6
0x4000_4824	ADC_DAT7	ADC DATA Register 7
0x4000_4828	ADC_DAT8	ADC DATA Register 8
0x4000_482C	ADC_DAT9	ADC DATA Register 9
0x4000_4830	ADC_DAT10	ADC DATA Register 10
0x4000_4834	ADC_DAT11	ADC DATA Register 11
0x4000_4838	ADC_DAT12	ADC DATA Register 12
0x4000_483C	ADC_DAT13	ADC DATA Register 13
0x4000_4840	ADC_DAT14	ADC DATA Register 14
0x4000_4844	ADC_DAT15	ADC DATA Register 15
0x4000_4848	ADC_CHSEL	ADC Channel Selection Register
0x4000_484C	ADC_TRGSEL	ADC Trigger Selection Register
0x4000_4850	ADC_BKSEL	ADC Brake Source Selection Register
0x4000_4854	ADC_BAKDAT	ADC Backup Data Register

Table 14-4 ADC register table

14.7 Register Description

14.7.1 ADC Control Register 0(ADC_CON0)

Bit	Name	R/W	Reset	Description
31	RST	RW	0x1	ADC Analog Module Reset Signal (PWD must be 0) 1: Reset ADC 0: Normal Mode
30	PWD	RW	0x1	ADC Analog Module Power Down Signal 1: Power down 0: The ADC module will start operating after a delay of 32 adc clocks
29	BAKEN	RW	0x1	ADC Backup Enable When ADC channel conversion is completed, the previous ADC conversion value of the current channel is backed up. 0: Disable 1: Enable backup
28:23	Reserved	--	0x0	Always read as 0.
22:20	TZO	RW	0x4	Zero Offset Change Trim Value Corrects the ADC output data. 0x0: -4 LSB 0x1: -3 LSB 0x2: -2 LSB 0x3: -1 LSB 0x4: Default (+0 LSB) 0x5: +1 LSB 0x6: +2 LSB 0x7: +3 LSB
19:16	INT_EN	RW	0x0	ADC Module Interrupt Output Enable (Refer to Table 14-2/14-3) 0x0: Disable interrupt 0x1: Enable ADC channel conversion completion interrupt (DONE)

				0x2: Enable interrupt when the threshold is less than the conversion result upon ADC channel conversion completion 0x4: Enable interrupt when the threshold is greater than the conversion result upon ADC channel conversion completion 0x8: Enable interrupt when the threshold is equal to the conversion result upon ADC channel conversion completion
15	ENCONT	RW	0x0	Continuous Conversion Mode (CONTINUE must be 1) (Refer to Table 14-1) 0: Conversion is performed based on the number of times set by ADC_CHSEL.CH_CNT[3:0] 1: Infinite Continuous Conversion Mode
14	Reserved	R/W	0x0	--
13	ALIGN	RW	0x0	ADC_DATx.DATA Read Alignment Selection 0: ADC_DATx.DATA read result is right-aligned 1: ADC_DATx.DATA read result is left-aligned Note: For example, if data is 0x555, the left-aligned read of data_reg is 0x5550, and the right-aligned read is 0x0555.
12	CONTINUE	RW	0x0	ADC Conversion Continuous Mode Enable (Refer to Table 14-1) 0: Single Mode Channel specified by M[3:0] 1: Continuous Mode / Discontinuous Mode / Scan Mode Mode selected by ENCONT and DISCEN Channel conversion based on ADC_CHSEL.CH_SEL selection Note: Write (0 to 1 or 1 to 0): Updates only after conversion is completed.
11	TRIG_EN	RW	0x0	Trigger Signal ADC Conversion Mode Enable 0: Disable hardware trigger START sampling (Software trigger) 1: Enable hardware trigger START sampling (Software/Hardware trigger)

10:8	CLK	RW	0x1	ADC Converter Operating Clock Selection (937.5KHz <= AD_CLK <= 16MHz) 0x0: PCLK/2 0x1: PCLK/4 0x2: PCLK/8 0x3: PCLK/16 0x4: PCLK/32 0x5: PCLK/64 0x6: PCLK/3.75 (16M@PCLK = 60Mhz) 0x7: PCLK/3.5 (17M@PCLK = 60Mhz) Default is divide by 4. Clock switching requires 4 ADC clock cycles to complete. Therefore, after switching the clock, wait for 4 ADC clock cycles before issuing a START request.
7	START	RW	0x0	ADC Conversion Channel Conversion Enable 0: Disable ADC conversion 1: Enable ADC conversion. AutoTrigger (ADC_TRGSEL) will also enable START. When conversion is not yet finished (ADC_STAT.DONE has not occurred), all enable attempts are invalid. Note: Requirement (START is valid only 32 adc clocks after PWD is set to 0).
6	Reserved	--	0x0	Always read as 0.
5	EN	RW	0x0	ADC Controller Operation Enable (Excluding registers) 0: ADC Controller Disabled 1: ADC Controller Enabled
4	Reserved	--	0x0	Always read as 0.
3:0	M	RW	0x0	Channel Selection in Single Mode Input channel selection

				When (CONTINUE = 0): 0x0: AIN0 0x1: AIN1 0x2: AIN2 0x3: AIN3 0x4: AIN4 0x5: AIN5 0x6: AIN6 0x7: AIN7 0x8: AIN8 0x9: AIN9 0xA: AIN10 0x: AIN11 0xC: AIN12 0xD: AIN13 0xE: AIN14 0xF: AIN15 [1*] When (CONTINUE = 1 and DONE has not occurred during conversion): Read: Current channel being converted (according to ADC_CHSEL.CH_SEL) Write: Unchanged, channel selection for Single Mode, does not affect CH_SEL Note: [1*] Must be used in conjunction with VBUF_CR.ANA_SEL and REZ_CR.ANA2ADC_EN. SYSCLK must be selected as 30Mhz, CLK set to 0x5, and ADC_STAT.TS_SET set to 18 to achieve a sampling time of 40μs.

14.7.2 ADC Status Regsiter(ADC_STAT)

Supports configuring the ADC sampling data of PGA0 and PGA1 channels to be output to different output registers.

Bit	Name	R/W	Reset	Description
31:28	START_CNT	R	0x0	Sent START Statistics Record Counts the number of START signals sent.
27:24	EOC_CNT	R	0x0	Received EOC Statistics Record Counts the number of EOC signals received.
23:21	DLY_SET	RW	0x0	ISAMPLCLK Low Level Hold Time (Unit: pclk, refer to TC in Figure 14-3) 0x0: Delay 12 adc clk (Used for 1MSPS sampling rate with 16M adc clk) 0x1: Delay 14 adc clk 0x2: Delay 18 adc clk 0x3: Delay 24 adc clk 0x4: Delay 26 adc clk
20:16	TS_SET	RW	0x3	ADC Sampling Time (ISAMPLCLK High Level Hold Time, Unit: pclk, refer to TS in Figure 14-3) TS = TS_SET + 1 clk. When TS_SET <= 3, TS is 4 clk. TS maximum is 32 clk. Note: ADC sampling interval is TS + TC. The default is 12 clk + 4 clk = 16 clk. When adc clk is 16M, the ADC sampling rate is 1Msps.
15:7	Reserved	R	0x0	Reserved
6	EOC_CHECK_DIS	RW	0x1	EOC Check Disable 0: Check EOC feedback before issuing the next request in Continuous Mode. 1: Do not check EOC feedback before issuing the next request in Continuous Mode. Note: Requirement: Continuous Mode must be valid. When EOC_CHECK_DIS is valid in Continuous Mode, enable ADC sampling once to start. To disable, cancel both EOC_CHECK_DIS and Continuous Mode.

5:4	COMP_RESULT	R	0x0	Comparison Result 0x0: Cleared after software reads the data result register (ADC_DATx). 0x1: Threshold is smaller than the ADC sampling result. 0x2: Threshold is larger than the ADC sampling result. 0x3: Threshold is equal to the ADC sampling result.
3:2	Reserved	--	0x0	Always read as 0.
1	INT_CLR	R/W	0x0	Interrupt Clear Bit Write 1 to clear the ADC interrupt flag (DONE). 1: Clear interrupt flag (DONE). INT_CLR is cleared to 0 after clearing. 0: No action.
0	DONE	R	0x0	ADC Channel Conversion Completion Flag 1: Set to 1 when conversion completes and new data is available. Triggers an interrupt if INT_EN is enabled. 0: Cleared when software reads the data result register (any ADC_DATx), or writes 1 to INT_CLR, or enables the next conversion.

14.7.3 ADC Channel Selection Register(ADC_CHSEL)

Bit	Name	R/W	Reset	Description
31:30	EXTRIG_SEL	R/W	0x0	PIN Trigger Selection 0x0: PB1 Rising 0x1: PB1 Falling 0x2: PB2 Rising 0x3: PB2 Falling
29:28	TRIG_SEL17	R/W	0x0	External PIN Trigger Enable 0x3: Reserved 0x2: Reserved 0x1: Enable external PIN EXTRIG (PB1/PB2) trigger conversion sampling 0x0: Disable
27:26	TRIG_SEL16	R/W	0x0	TIM0 Overflow Trigger Enable 0x3: Reserved 0x2: Reserved 0x1: Enable TIM0 overflow signal trigger conversion sampling 0x0: Disable Note: TIM0 timing trigger can select software and EPWM P0, 1, 2, 3 rising/falling.
25	DISCEN	R/W	0x0	TIM0 Overflow Trigger Enable 0x3: Reserved 0x2: Reserved 0x1: Enable TIM0 overflow signal trigger conversion sampling 0x0: Disable Note: TIM0 timing trigger can select software and EPWM P0, 1, 2, 3 rising/falling.
24:22	DISCNUM[2:0]	R/W	0x0	Discontinuous Mode Conversion Count (DISCEN and ADC_CON0.CONTINUE must be 1) Software defines the number of regular channel conversions after each START trigger in Discontinuous Mode via these bits.

				0x0: 1 conversion 0x1: 2 conversions ... 0x7: 8 conversions
21	DISC_INTSEL	R/W	0x0	Discontinuous Mode Interrupt Rule Selection (Valid when DISCEN = 1 and ADC_CON0.INT_EN[0] = 1) 0: Interrupt generated after all channel conversions are completed 1: Interrupt generated after one channel conversion is completed
20	Reserved	--	0x0	Always read as 0.
19:16	CH_CNT[3:0]	R/W	0x0	Scan Mode / Discontinuous Mode Conversion Channel Count 0x0: Convert 1 channel (Based on ADC_CHSEL[15:0]) 0x1: Convert 2 channels (Based on ADC_CHSEL[15:0]) 0x2: Convert 3 channels (Based on ADC_CHSEL[15:0]) ... 0xE: Convert 15 channels continuously (Based on ADC_CHSEL[15:0]) 0xF: Convert 16 channels continuously (Based on ADC_CHSEL[15:0]) Note: Lower channels have higher conversion priority: AIN0 (ADC_CHSEL[0] = 1: Convert, 0: Do not convert) -> AIN1 (ADC_CHSEL[1] = 1: Convert, 0: Do not convert) -> AIN2 (ADC_CHSEL[2] = 1: Convert, 0: Do not convert) ... -> AIN13 (ADC_CHSEL[13] = 1: Convert, 0: Do not convert) -> AIN14 (ADC_CHSEL[14] = 1: Convert, 0: Do not convert) -> AIN15 (ADC_CHSEL[15] = 1: Convert, 0: Do not convert)
15	CH_SEL[15]	R/W	0x0	CH_SEL[15] 1: Select AIN15 for conversion 0: No selection
14	CH_SEL[14]	R/W	0x0	CH_SEL[14] 1: Select AIN14 for conversion

				0: No selection
13	CH_SEL[13]	R/W	0x0	CH_SEL[13] 1: Select AIN13 for conversion 0: No selection
12	CH_SEL[12]	R/W	0x0	CH_SEL[12] 1: Select AIN12 for conversion 0: No selection
11	CH_SEL[11]	R/W	0x0	CH_SEL[11] 1: Select AIN11 for conversion 0: No selection
10	CH_SEL[10]	R/W	0x0	CH_SEL[10] 1: Select AIN10 for conversion 0: No selection
9	CH_SEL[9]	R/W	0x0	CH_SEL[9] 1: Select AIN9 for conversion 0: No selection
8	CH_SEL[8]	R/W	0x0	CH_SEL[8] 1: Select AIN8 for conversion 0: No selection
7	CH_SEL[7]	R/W	0x0	CH_SEL[7] 1: Select AIN7 for conversion 0: No selection
6	CH_SEL[6]	R/W	0x0	CH_SEL[6] 1: Select AIN6 for conversion 0: No selection
5	CH_SEL[5]	R/W	0x0	CH_SEL[5] 1: Select AIN5 for conversion 0: No selection
4	CH_SEL[4]	R/W	0x0	CH_SEL[4] 1: Select AIN4 for conversion 0: No selection

3	CH_SEL[3]	R/W	0x0	CH_SEL[3] 1: Select AIN3 for conversion 0: No selection
2	CH_SEL[2]	R/W	0x0	CH_SEL[2] 1: Select AIN2 for conversion 0: No selection
1	CH_SEL[1]	R/W	0x0	CH_SEL[1] 1: Select AIN1 for conversion 0: No selection
0	CH_SEL[0]	R/W	0x0	CH_SEL[0] 1: Select AIN0 for conversion 0: No selection

14.7.4 ADC Trigger Source Selection Register(ADC_TRGSEL)

Bit	Name	R/W	Reset	Description
31:30	TRIG_SEL15	R/W	0x0	TRIG_SEL15 0x3: Reserved 0x2: Reserved 0x1: Enable CCDR4 DN (EPWM Down-counting == CCDR4) trigger conversion sampling 0x0: Disable
29:28	TRIG_SEL14	R/W	0x0	TRIG_SEL14 0x3: Reserved 0x2: Reserved 0x1: Enable CCR4 DN (EPWM Down-counting == CCR4) trigger conversion sampling 0x0: Disable
27:26	TRIG_SEL13	R/W	0x0	TRIG_SEL13 0x3: Reserved 0x2: Reserved 0x1: Enable CCDR4 UP (EPWM Up-counting == CCDR4) trigger conversion sampling 0x0: Disable
25:24	TRIG_SEL12	R/W	0x0	TRIG_SEL12 0x3: Reserved 0x2: Reserved 0x1: Enable CCR4 UP (EPWM Up-counting == CCR4) trigger conversion sampling 0x0: Disable
23:22	TRIG_SEL11	R/W	0x0	TRIG_SEL11 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM Counter OVF (Overflow) trigger conversion sampling

				0x0: Disable
21:20	TRIG_SEL10	R/W	0x0	TRIG_SEL10 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM Counter UDF (Underflow) trigger conversion sampling 0x0: Disable
19:18	TRIG_SEL9	R/W	0x0	TRIG_SEL9 0x3: Reserved 0x2: Reserved 0x1: Enable CMP1 Falling trigger conversion sampling 0x0: Disable
17:16	TRIG_SEL8	R/W	0x0	TRIG_SEL8 0x3: Reserved 0x2: Reserved 0x1: Enable CMP1 Rising trigger conversion sampling 0x0: Disable
15:14	TRIG_SEL7	R/W	0x0	TRIG_SEL7 0x3: Reserved 0x2: Reserved 0x1: Enable CMP0 Falling trigger conversion sampling 0x0: Disable
13:12	TRIG_SEL6	R/W	0x0	TRIG_SEL6 0x3: Reserved 0x2: Reserved 0x1: Enable CMP0 Rising trigger conversion sampling 0x0: Disable
11:10	TRIG_SEL5	R/W	0x0	TRIG_SEL5 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM3P (CH3) Falling trigger conversion sampling

				0x0: Disable
9:8	TRIG_SEL4	R/W	0x0	TRIG_SEL4 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM2P (CH2) Falling trigger conversion sampling 0x0: Disable
7:6	TRIG_SEL3	R/W	0x0	TRIG_SEL3 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM1P (CH1) Falling trigger conversion sampling 0x0: Disable
5:4	TRIG_SEL2	R/W	0x0	TRIG_SEL2 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM3P (CH3) Rising trigger conversion sampling 0x0: Disable
3:2	TRIG_SEL1	R/W	0x0	TRIG_SEL1 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM2P (CH2) Rising trigger conversion sampling 0x0: Disable
1:0	TRIG_SEL0	R/W	0x0	TRIG_SEL0 0x3: Reserved 0x2: Reserved 0x1: Enable EPWM1P (CH1) Rising trigger conversion sampling 0x0: Disable

14.7.5 ADC DATA Register 0(ADC_DAT0)

After ADC conversion is completed, the 12-bit digital value is stored in the ADC_DAT register, available for application software to read. After initiating an ADC conversion, it is necessary to wait for the conversion process to finish before reading the correct result. This can be implemented via software polling (waiting for the ADC_STAT.DONE bit to be 1) or via interrupt response (requires enabling the system ADC interrupt and providing an interrupt service routine). After software reads the data in ADC_DAT or writes 1 to ADC_STAT.INT_CLR, the ADC_STAT.DONE bit is automatically cleared.

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PGA0_O (AIN0) (Refer to M/CH_SEL parameters)

14.7.6 ADC DATA Register 1(ADC_DAT1)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PGA1_O (AIN1) (Refer to M/CH_SEL parameters)

14.7.7 ADC DATA Register 2(ADC_DAT2)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA8 (AIN2) (Refer to M/CH_SEL parameters)

14.7.8 ADC DATA Register 3(ADC_DAT3)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA9 (AIN3) (Refer to M/CH_SEL parameters)

14.7.9 ADC DATA Register 4(ADC_DAT4)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA10 (AIN4) (Refer to M/CH_SEL parameters)

14.7.10 ADC DATA Register 5(ADC_DAT5)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA11 (AIN5) (Refer to M/CH_SEL parameters)

14.7.11 ADC DATA Register 6(ADC_DAT6)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA12 (AIN6) (Refer to M/CH_SEL parameters)

14.7.12 ADC DATA Register 7(ADC_DAT7)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA13 (AIN7) (Refer to M/CH_SEL parameters)

14.7.13 ADC DATA Register 8(ADC_DAT8)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA6 (AIN8) (Refer to M/CH_SEL parameters)

14.7.14 ADC DATA Register 9(ADC_DAT9)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA7 (AIN9) (Refer to M/CH_SEL parameters)

14.7.15 ADC DATA Register 10(ADC_DAT10)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA14 (AIN10) (Refer to M/CH_SEL parameters)

14.7.16 ADC DATA Register 11(ADC_DAT11)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PA15 (AIN11) (Refer to M/CH_SEL parameters)

14.7.17 ADC DATA Register 12(ADC_DAT12)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PB0 (AIN12) (Refer to M/CH_SEL parameters)

14.7.18 ADC DATA Register 13(ADC_DAT13)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) PB1 (AIN13) (Refer to M/CH_SEL parameters)

14.7.19 ADC DATA Register 14(ADC_DAT14)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	0x0	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x0	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) AVSS (AIN14) (Refer to M/CH_SEL parameters)

14.7.20 ADC DATA Register 15(ADC_DAT15)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	CMPTH	RW	00	CMPTH: Comparison Threshold Setting When ADC result comparison is enabled, the channel's DATA is compared with CMPTH, and the comparison result is stored in the ADC_STAT.COMP_RESULT register.
15:0	DATA	R	0x	DATA: 12-bit ADC Conversion Value (Default is right-aligned format. The conversion result can be left-aligned by configuring the ADC_CON0.ALIGN bit.) ANASEL (AIN15) (Refer to AMISC_VBUF_CR.ANASEL)

14.7.21 ADC Brake Selection Register(ADC_BKSEL)

Bit	Name	R/W	Reset	Description
31:16	Reserved	--	0x0	Always read as 0.
15:14	ADCBKE2_EN	R/W	0x0	ADC Brake Source 2 Enable Selection 0x0: ADC Brake Source 2 Brake Disabled and Reset ADCBKE2 Filter count 0x1: ADC Brake Source 2 Enable Brake when Threshold < Conversion Result 0x2: ADC Brake Source 2 Enable Brake when Threshold > Conversion Result 0x3: ADC Brake Source 2 Enable Brake when Threshold == Conversion Result The comparison is based on the channel selected by ADCBKE2_CH. Every conversion result increments the ADCBKE2 Filter value.
13:12	ADCBKE2_FLT	RW	0x0	ADC Brake Source 2 Signal Filter (ADCBKE2 Filter) 0x0: Brake triggers when 1 ADC conversion result matches (selected by ADCBKE2_EN) 0x1: Brake triggers when 2 consecutive ADC conversion results match (selected by ADCBKE2_EN) 0x2: Brake triggers when 4 consecutive ADC conversion results match (selected by ADCBKE2_EN) 0x3: Brake triggers when 7 consecutive ADC conversion results match (selected by ADCBKE2_EN) The filtering is based on the comparison result of the channel selected by ADCBKE2_CH. Note: ADCBKE2_FLT can only be written when ADCBKE2_EN = 0x0.
11:8	ADCBKE2_CH	R/W	0x0	ADC Brake Source 2 Channel Selection 0x0: Select Channel 0 as Brake Source 2 0x1: Select Channel 1 as Brake Source 2

				0x2: Select Channel 2 as Brake Source 2 ... 0xF: Select Channel 15 as Brake Source 2 Note: ADCBKE2_CH can only be written when ADCBKE2_EN = 0x0.
7:6	ADCBKE1_EN	R/W	0x0	ADC Brake Source 1 Enable Selection 0x0: ADC Brake Source 1 Brake Disabled and Reset ADCBKE1 Filter count 0x1: ADC Brake Source 1 Enable Brake when Threshold < Conversion Result 0x2: ADC Brake Source 1 Enable Brake when Threshold > Conversion Result 0x3: ADC Brake Source 1 Enable Brake when Threshold == Conversion Result The comparison is based on the channel selected by ADCBKE1_CH. Every conversion result increments the ADCBKE1 Filter value.
5:4	ADCBKE1_FLT	RW	0x0	ADC Brake Source 1 Signal Filter (ADCBKE1 Filter) 0x0: Brake triggers when 1 ADC conversion result matches (selected by ADCBKE1_EN) 0x1: Brake triggers when 2 consecutive ADC conversion results match (selected by ADCBKE1_EN) 0x2: Brake triggers when 4 consecutive ADC conversion results match (selected by ADCBKE1_EN) 0x3: Brake triggers when 7 consecutive ADC conversion results match (selected by ADCBKE1_EN) The filtering is based on the comparison result of the channel selected by ADCBKE1_CH. Note: ADCBKE1_FLT can only be written when ADCBKE1_EN = 0x0.
3:0	ADCBKE1_CH	R/W	0x0	ADC Brake Source 1 Channel Selection 0x0: Select Channel 0 as Brake Source 1

				0x1: Select Channel 1 as Brake Source 1 0x2: Select Channel 2 as Brake Source 1 ... 0xF: Select Channel 15 as Brake Source 1 Note: ADCBKE1_CH can only be written when ADCBKE1_EN = 0x0.

14.7.22 ADC Backup Data Register(ADC_BAKDAT)

Bit	Name	R/W	Reset	Description
31:28	Reserved	--	0x0	Always read as 0.
27:16	-	R	0x0	
15:0	ADC_BAKDAT	R	0x0	ADC_BAKDAT: (Set) When ADC_BKP_EN = 1: When channel conversion is completed, the value from the previous conversion of this channel is simultaneously backed up to this register.

15. Analog Miscellaneous(AMISC)

15.1. Overview

PEC930 contains several analog modules that require user configuration. Associated miscellaneous controls are also included in this chapter for convenient software management. These miscellaneous controls are implemented via registers, allowing users to manage various analog functions by configuring the corresponding registers. More independent analog modules, such as ADCs, have their own dedicated controllers and are not described in this chapter.

15.2. LVD/LVR Function Overview

When the LVD (Low Voltage Detect) module is enabled, the chip continuously monitors the supply voltage **(**VDD**)**. Users can configure the LVD filter clock parameters and check the status through the LVD output bit. If VDD falls below the set LVD threshold and the LVD interrupt is enabled, a system interrupt will be triggered.

When the LVR (Low Voltage Reset) module is enabled, the chip also continuously monitors VDD**.** If VDD drops below the set LVR threshold, the system will trigger a reset, and the chip will re-enter the initialization phase.

For detailed configuration, refer to the AMISC_LVD_LVR_CR register.

15.3. PGA Function Overview

The PEC930 chip includes three basic operational amplifier modules and two programmable gain amplifiers (gain range:1× to 16×). With a small number of external components, these modules can be used to implement basic signal amplification and signal processing functions.

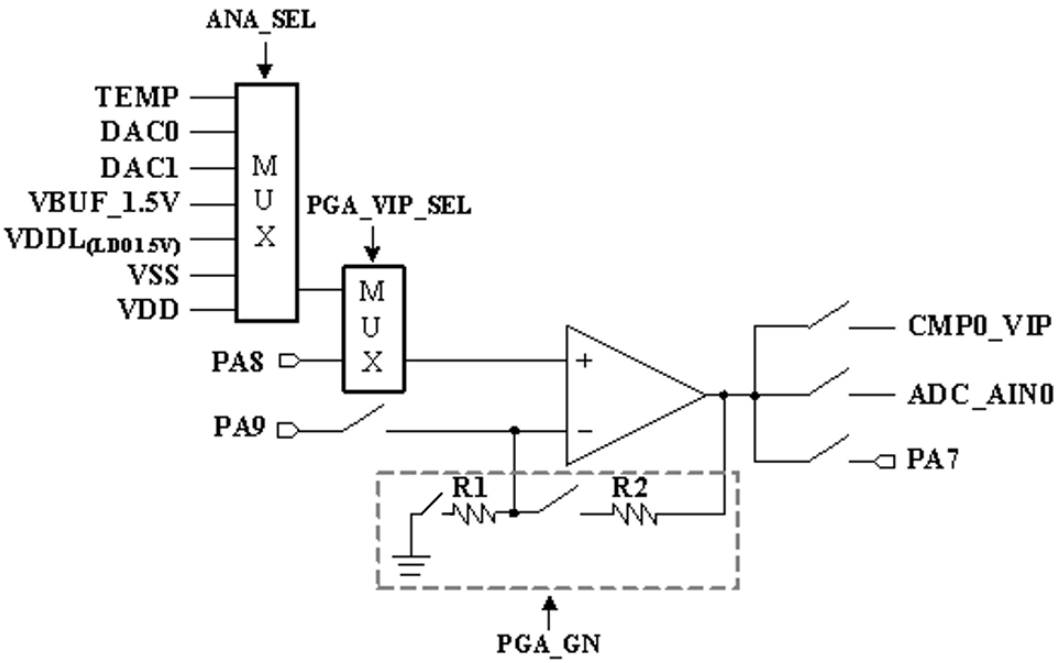


Fig. 15-1 PGA0 Block Diagram

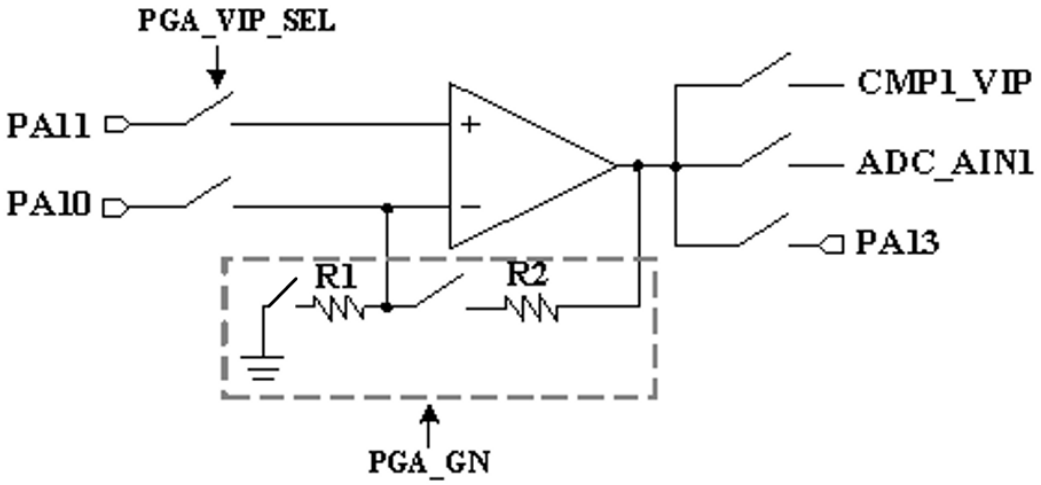


Fig. 15-2 PGA1 Block Diagram

- PGA0/1 use the internal programmable-gain configuration of the chip.

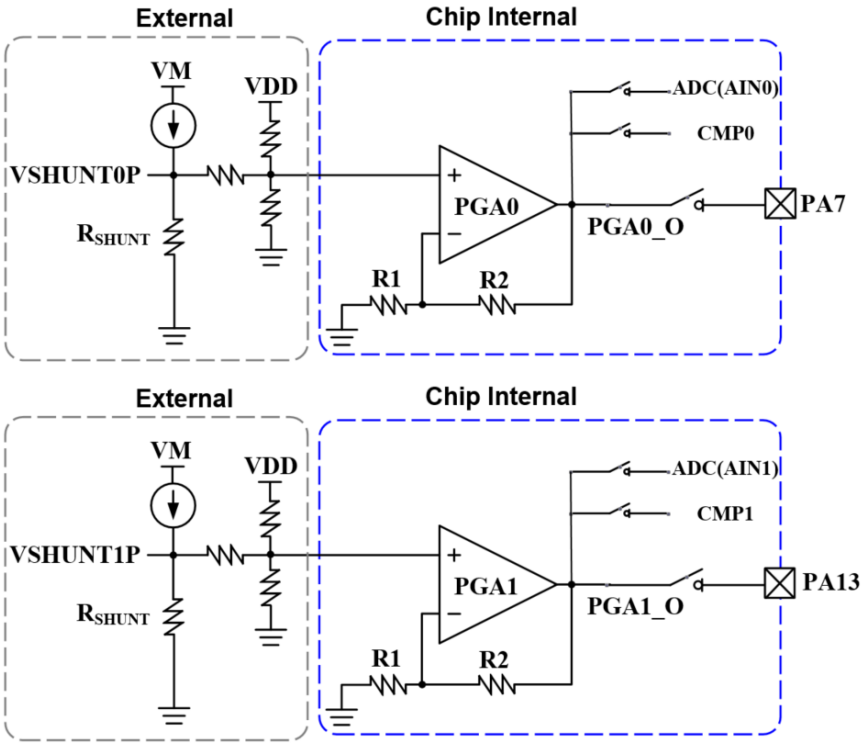


Fig. 15-3 PGA Internal Gain Block Diagram

- PGA0/1 use external resistor gain.

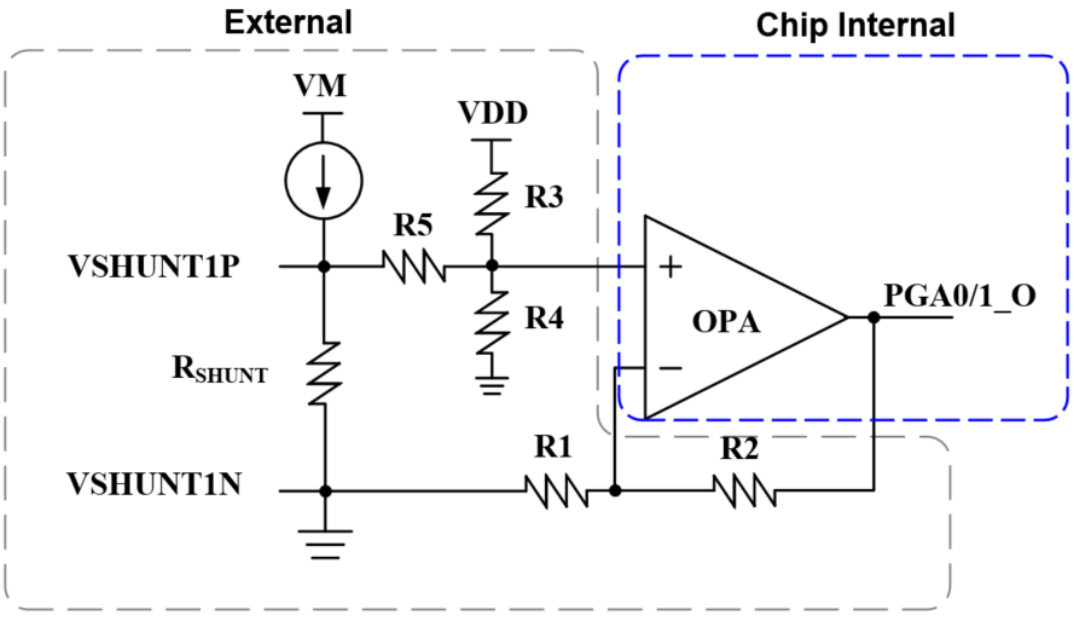


Fig. 15-4 PGA External Gain Block Diagram

15.4. Register table

Address	Name	Description
0x4000_5800	AMISC_LVD_LVR_CR	LVD/LVR control register
0x4000_5804	AMISC_VBUF_CR	VBUF control register
0x4000_5810	AMISC_DAC_CR	DAC control register
0x4000_5820	AMISC_HSI_CR	HIRC control register
0x4000_5824	AMISC_LSI_CR	LIRC control register
0x4000_5830	AMISC_ADC_AIN_CR	ADC analog input control register
0x4000_5860	HWTRIM_LDO_TRIM	LDO trim read register
0x4000_5864	HWTRIM_VBUF_TRIM	VBUF trim read register
0x4000_5868	HWTRIM_HSI_TRIM	HIRC trim read register
0x4000_586C	HWTRIM_LSI_TRIM	LIRC trim read register
0x4000_5870	HWTRIM_MISC_CFG	Miscellaneous configuration read register
0x4000_5880	OPAMP0_PGA_CR	OPAMP0(PGA0) control register
0x4000_5888	OPAMP1_PGA_CR	OPAMP1(PGA1) control register

Table 15-1 AMISC register table

15.5. Register description

15.5.1 LVD/LVR control register (AMISC_LVD_LVR_CR)

Bit	Name	R/W	Reset	Description
31:16	-	R	0x0	Reserved
15:13	LVR_SEL	R/W	0x0	LVR voltage threshold selection 0x0: 2.0 V 0x1: 2.4 V 0x2: 2.7 V 0x3: 3.0 V 0x4: 3.7 V Others: Reserved
12	-	R	0x0	Reserved
11	LVD_STATE	R	0x1	Output status of VDD corresponding to selected LVD_SEL: 0: VDD below selected LVD threshold 1: VDD above selected LVD threshold
10	TEMP_EN	R/W	0x1	Temperature sensor enable: 0: Disabled 1: Enabled
9	LVD_INT_EN	R/W	0x0	LVD interrupt enable: 0: LVD interrupt disabled 1: LVD interrupt enabled
8	LVR_EN	R/W	0x0	LVR enable control: 0: LVR disabled 1: LVR enabled (AMISC_VBUF_CR.VBUF_EN must be enabled)

7:5	LVD_SEL	R/W	0x1	LVD voltage threshold selection 0x0: 2.0V 0x1: 2.2V 0x2: 2.4V 0x3: 2.7V 0x4: 3.0V 0x5: 3.7V 0x6: 4.0V 0x7: 4.3V
4	-	R/W	0x0	Reserved
3:2	LVD_F_SEL	RW	0x0	LVD Filter duration selection 0x0: 3*HCLK 0x1: 3*(HCLK/2) 0x2: 3*(HCLK/4) 0x3: 3*(HCLK/8)
1	LDO_LP_EN	R/W	0x0	LDO low power module enable: 0: Disabled 1: Enabled
0	LVD_EN	R/W	0x0	LVD Enable Control: 0: LVD disabled 1: LVD enabled (AMISC_VBUF_CR.VBUF_EN must be enabled)

15.5.2 VBUF control register (AMISC_VBUF_CR)

Bit	Name	R/W	Reset	Description
31:13	-	R	0x0	Reserved
12:6	ANA_SEL	R/W	0x0	Analog Signal Selection: 0x0: Reserved 0x1: TEMP (AMISC_LVD_LVR_CR.TEMP_EN must be enabled) 0x2: DAC0 (AMISC_DAC_CR.DAC0_EN must be enabled) 0x4: DAC1 (AMISC_DAC_CR.DAC1_EN must be enabled) 0x8: VBUF 1.5V (AMISC_VBUF_CR.VBUF_EN must be enabled) 0x10: VDDL 0x20: VSS 0x40: VDD Others: Reserved Usage: Must be used with any of the ANA2PGA_EN, ANA2IO_EN, or AMISC_ADC_AIN_CR.ANA2ADC_EN registers
5	ANA2PGA_EN	R/W	0x0	Analog Signal Output to OPAMP0(PGA0): 0: Disabled 1: Enabled, with ANA_SEL selecting the signal connected to OPAMP0(PGA0) positive input (Refer to OPAMP0_PGA_CR.PGA_VIP_SEL register)
4	ANA2IO_EN	R/W	0x0	Analog Signal Output to IO(PA8): 0: Disabled 1: Enabled, with ANA_SEL selecting the output signal
3:1	-	R/W	0x0	Reserved
0	VBUF_EN	R/W	0x0	VBUF Enable Control 0: Disabled 1: Enabled

15.5.3 DAC control register (AMISC_DAC_CR)

Bit	Name	R/W	Reset	Description
31:24	-	R	0x0	Reserved
23:22	-	R/W	0x0	Reserved
21	DAC1_EN	R/W	0x0	DAC1 Enable Control 0: Disabled 1: Enabled
20	DAC0_EN	R/W	0x0	DAC0 Enable Control 0: Disabled 1: Enabled
19:10	DAC1B	R/W	0x200	DAC1 bit input
9:0	DAC0B	R/W	0x200	DAC0 bit input

Note: When entering DeepSleep mode, the DAC must be disabled: set DAC1/0_EN = 0x0 and DAC1/0B = 0x0.

15.5.4 HIRC control register (AMISC_HSI_CR)

Bit	Name	R/W	Reset	Description
31:24	HSI_EN	R/W	0x01	HIRC Enable Control (write: 0x80，read: 0x00) HIRC Disabled (write: 0x01，read: 0x01) HIRC Enabled Writing other values keeps the state unchanged. Note: Be very careful when disabling the HIRC clock!
23:16	-	R	0x0	Reserved
15:0	LDO_SEL	R/W	0x0101	HIRC power source selection (write: 0x8080，read: 0x0000)**:** Select Shared Power with LDO (Due to package bonding, this option is recommended) (write: 0x8001，read: 0x0001)**:** Disable (write: 0x0180，read: 0x0100)**:** Turn off HIRC Dedicated LDO Power (write: 0x0101，read: 0x0101): Select HIRC Dedicated LDO Power Writing other values keeps the state unchanged. Note: Be very careful when selecting the HIRC power source!

15.5.5 LIRC control register (AMISC_LSI_CR)

Bit	Name	R/W	Reset	Description
31:24	LSI_EN	R/W	0x1	LIRC Enable Control (write:0x80，read: 0x00)LIRC Disabled (write:0x01，read: 0x01)LIRC Enabled Writing other values keeps the state unchanged. Note: Be very careful when disabling the LIRC clock!
23:0	-	R	0x0	Reserved

15.5.6 ADC analog input control register (AMISC_ADC_AIN_CR)

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2:1	-	R/W	0x3	Reserved
0	ANA2ADC_EN	R/W	0x1	Analog Signal Output to ADC; 0: Disabled 1: Enabled, with AMISC_VBUF_CR.ANA_SEL selecting the signal connected to ADC AIN15 (refer to ADC_CON0.M register)

15.5.7 LDO trim read register (HWTRIM_LDO_TRIM)

Bit	Name	R/W	Reset	Description
31:12	-	R	0x0	Reserved
11:10	LDO_TUNE	R	0x2	LDO high-temp leakage current compensation 0x0: Strength 0x1: weak 0x2: Middle 0x3: Off
9:5	LDO_LP_TRIM	R	0x10	LDO lower power module trim 0x0: [vmax] 0x1F: [vmin]
4:0	LDO_TRIM	R	0x10	LDO trim 0x0: [vmax] 0x1F: [vmin]

15.5.8 VBUF trim read register(HWTRIM_VBUF_TRIM)

Bit	Name	R/W	Reset	Description
31:7	-	R	0x0	Reserved
6:0	VBUF_TRIM	R	0x20	VBUF 1.5V voltage trim 0x0: Reserved 0x1: [vmin] 0x3F: [vmax] Note: bit 6 don’t care

15.5.9 HIRC trim read register (HWTRIM_HSI_TRIM)

Bit	Name	R/W	Reset	Description
31:23	-	R	0x0	Reserved
22:18	-	R	0x10	Reserved
17:16	HSI_TC	R	0x2	HIRC temperature compensation trim 0x0: [fmax] 0x3: [fmin]
15:9	HSI_FSEL_CFG	R	0x40	HIRC frequency high trim 0x0: [fmin] 0x70: [fmax]
8:0	HSI_D_CFG	R	0x0	HIRC frequency low trim 0x0: [fmin] 0x1FF: [fmax]

15.5.10 LIRC trim read register (HWTRIM_LSI_TRIM)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	LSI_TRIM	R	0xB3	LIRC frequency trim 0x0: [fmin] 0xFF: [fmax]

15.5.11 Miscellaneous configuration read register (HWTRIM_MISC_CFG)

Bit	Name	R/W	Reset	Description
31:2	-	R	0x0	Reserved
1	-	R	0x0	Reserved
0	EXT_nRST_EN	R	0x1	External reset enable 1: Enabled 0: Disabled

15.5.12 OPAMPn (PGAn) control register (OPAMPn_PGA_CR, n=0,1)

Bit	Name	R/W	Reset	Description
31:12	-	R	0x0	Reserved
11	PGA_VIP_SEL	R/W	0x0	OPAMP(PGA) P terminal selection: Input to PGA0 0: internal signal(refer to the usage of the AMISC_VBUF_CR.ANA2PGA_EN and AMISC_VBUF_CR.ANA_SEL registers) 1: IO(PA8) Input to PGA1 0: floating 1: IO(PA11)
10	PGA_VIN_SEL	R/W	0x0	OPAMP(PGA) N terminal selection: 0: use internal gain circuit 1: IO(PA9) input to PGA0, IO(PA10) input to PGA1
9:8	-	R	0x0	Reserved
7:2	PGA_GAIN	R/W	0x0	OPAMP(PGA) gain selection: 0x0: COMP module or external PGA module 0x1: X1 0x6: X2 0xA: X3 0xE: X4 0x12: X5 0x16: X6

				0x1A: X7 0x1E: X8 0x22: X9 0x26: X10 0x2A: X11 0x2E: X12 0x32: X13 0x36: X14 0x3A: X15 0x3E: X16 Others: Revered
1	PGA_IO_EN	R/W	0x0	OPAMP(PGA) output to IO enable control: 0: Disabled 1: Enabled, PGA0 output to IO(PA[7]), PGA1 output to IO(PA[13])
0	PGA_EN	R/W	0x0	OPAMP(PGA) enable signal 0: Disabled 1: Enabled

16. Cyclic Redundancy Check Calculation Unit (CRC)

16.1 Overview

Cyclic Redundancy Check (CRC) is an error-detecting code. This module performs CRC calculations on arbitrary byte data based on a fixed polynomial. In practical applications, CRC is primarily used for: data transmission integrity verification, stored data consistency verification, and communication error detection.

The CRC application workflow is as follows:

Data Sender (calculate and append CRC) → Transmission → Data Receiver (recalculate and verify CRC)

16.2 Function Description

This module is implemented in accordance with the ISO/IEC 13239 standard and supports the following generator polynomials:

(1) CRC-16: $x^{16} + x^{12} + x^5 + 1$ (corresponding polynomial: 0x1021)

(2) CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (corresponding polynomial: 0x04C11DB7)

Initial value: Fixed at 0xFFFFFFFF

Functions are divided into: CRC encoding and CRC verification

Supports three data input widths: 8-bit (byte), 16-bit (half-word), 32-bit (word)

16.2.1 CRC Encoding Mode

This mode is used to calculate the CRC value for raw data. The resulting value is typically appended to the end of the raw data packet to form a complete transmission frame with a checksum mechanism.

Operation Steps:

(1) Mode Configuration: Configure the CRC_CR.POLYSEL register to select the polynomial.

(2) Initialization: Write the initial value 0xFFFFFFFF to the CRC_DOUT register.

(3) Data Input: Sequentially write the raw data into the CRC_DIN register.

- Input Width: Supports 8 / 16 / 32-bit write widths.
- Sequence Requirement: Data must be written sequentially in its original order.

(4) Result Reading: Once the calculation is complete, read the final CRC value from the CRC_DOUT register.

16.2.2 CRC Verification Mode

This mode is used to verify the integrity of the received data. By performing a division calculation on the complete data (original data + CRC), the system determines whether the remainder is zero.

Operation Steps:

(1) Mode Configuration: Configure the CRC_CR.POLYSEL register to select the polynomial.

(2) Initialization: Write the initial value 0xFFFFFFFF to the CRC_DOUT register.

(3) Data Input: Sequentially write the complete data (original data + CRC) into the CRC_DIN register.

- Input Width: Supports 8 / 16 / 32-bit write widths.
- Sequence Requirement: Data must be written in the original order; the CRC value must be written in Little-Endian format (write the low-order byte first, followed by the high-order byte).

(4) Result Judgment: Read the state of the CRC_CR.VERF bit. Determine whether the verification passed based on this result.

16.3 Register table

Address	Name	Description
0x4001_E000	CRC_CR	CRC control register
0x4001_E004	CRC_DIN	CRC data input register
0x4001_E008	CRC_DOUT	CRC result output register

Table 16-1 CRC register table

16.4 Register description

16.4.1 CRC control register (CRC_CR)

Bit	Name	R/W	Reset	Description
31:2	-	R	0x0	Reserved
1	VERF	R	0x0	CRC verification result flag : Indicates the integrity check status of the current data packet. In verification mode, this flag bit must only be read after all raw data and the 16/32-bit CRC checksum have been fully written to the CRC data input register (CRC_DIN). 0: Verification Failed 1: Verification Passed
0	POLYSEL	R/W	0x0	CRC generator polynomial selection： Selects the CRC mathematical model for the module operation. 1: CRC-16 → $x^{16} + x^{12} + x^5 + 1$(0x1021) 0: CRC-32 → $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$(0x04C11DB7)

16.4.2 CRC data input register (CRC_DIN)

Bit	Name	R/W	Reset	Description
31:0	DIN	R/W	0x0	CRC data input register: Used to input data for CRC calculation. Supports 8-bit, 16-bit, and 32-bit widths. The hardware automatically processes the input stream based on the bit-width of the write instruction.

16.4.3 CRC result output register (CRC_DOUT)

Bit	Name	R/W	Reset	Description
31:0	DOUT	R/W	0x0	CRC result output register: Stores intermediate values and final results of the calculation; also functions as a reset trigger. Read Operation: Returns the Bitwise NOT of the current internal register per standard protocols. This design ensures the read result aligns with the final CRC checksum required by the protocol. Write Operation: Writing 0xFFFFFFFF to this register resets the CRC calculation to its initial state.

17. DSP Hardware Acceleration Module(DSP)

17.1 Overview

The DSP hardware acceleration module includes a 32-bit signed divider and a 32-bit unsigned square-root unit. Users can access and configure these functional units through the APB bus by writing input data, triggering the operation, and reading back the corresponding results, thereby accelerating DSP-related computation tasks.

Each computational unit requires a certain number of clock cycles to complete one operation (refer to the Functional Description section for detailed cycle counts).

The 32-bit signed divider and the 32-bit unsigned square-root unit share the input configuration registers and result registers. Users must select the desired operation mode before starting a computation. After an operation is triggered, the next operation cannot be started until the current computation is completed. During this time, writing to the data source configuration registers has no effect (is ignored).

17.2 Functional Description

Before starting an operation, the user must configure the desired computation mode by writing the MODE field in the DSP_CR register.The same operation mode does not need to be reconfigured repeatedly. Modifying this field is ignored while the corresponding computation unit is in progress (refer to the operation flow described in the following sections).

When writing input data, the user must write the second Data Source Configuration Register (e.g., DSP_SDAT2) last to ensure proper operation triggering.

17.2.1 32-bit Signed Division

The 32-bit signed division operation is expressed as follows:

$$(\mathbf{signed})\ \mathbf{X}[\mathbf{31}:\mathbf{0}] = \frac{(\mathbf{signed})\ \mathbf{A}[\mathbf{31}:\mathbf{0}] - (\mathbf{signed})\ \mathbf{Y}[\mathbf{31}:\mathbf{0}]}{(\mathbf{signed})\ \mathbf{B}[\mathbf{31}:\mathbf{0}]}$$

- A input valid range : $-2^{31} \sim 2^{31}-1$
- B input valid range : $-2^{31} \sim 2^{31}-1$
- X output valid range : $-2^{31} \sim 2^{31}-1$
- Y output valid range : $-2^{31} \sim 2^{31}-1$

The divider uses a multi-cycle algorithm, requiring 18 clock cycles for computation.

Software operation steps are as follows:

1. Write DSP_SDAT1 to configure the Numerator A[31:0] (1 write instruction cycle).
2. Write DSP_SDAT2 to configure the Denominator B[31:0] and trigger the operation (1 write instruction cycle).
3. Wait 18 clock cycles, then read the DONE bit in the DSP_SR register to confirm completion (1 read instruction cycle).
4. Read DSP_RSLT1 and DSP_RSLT2 to obtain the Quotient (X[31:0]) and remainder (Y[31:0]) (2 read instruction cycles total).

The control flow diagram of the division operation is shown below:

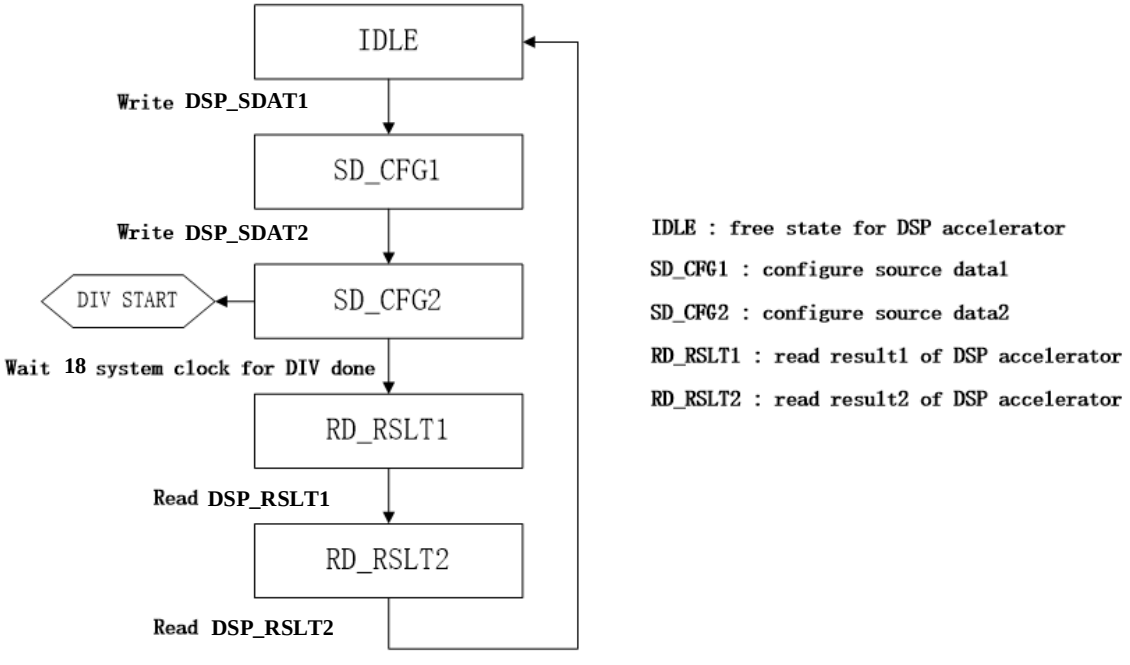


Fig. 17-1 Division operation control flowchart

17.2.2 32-bit Unsigned Square Root

The 32-bit unsigned square root operation is expressed as follows:

$$(\mathbf{unsigned})\ \mathbf{X[16:0]} = \sqrt{(\mathbf{unsigned})\ \mathbf{A[31:0]}}$$

- A input valid range : $0 \sim 2^{32} - 1$
- X output valid range : $0 \sim 2^{16} - 1$

The square root uses a multi-cycle algorithm, requiring 18 clock cycles to complete.

Software operation steps are as follows:

1. Write DSP_SDAT1 register to configure the input A[31:0] (1 write instruction cycle).
2. Write DSP_SDAT2 register with 0 to initiate the square root operation (1 write instruction cycle).
3. Wait 18 clock cycles, then read the DONE bit of DSP_SR register to check whether it is 1 (1 read instruction cycle).
4. Read DSP_RSLT1 register to obtain the square root result X[16:0] (1 read instruction cycle).

The state control flow of the square root operation is shown as follows:

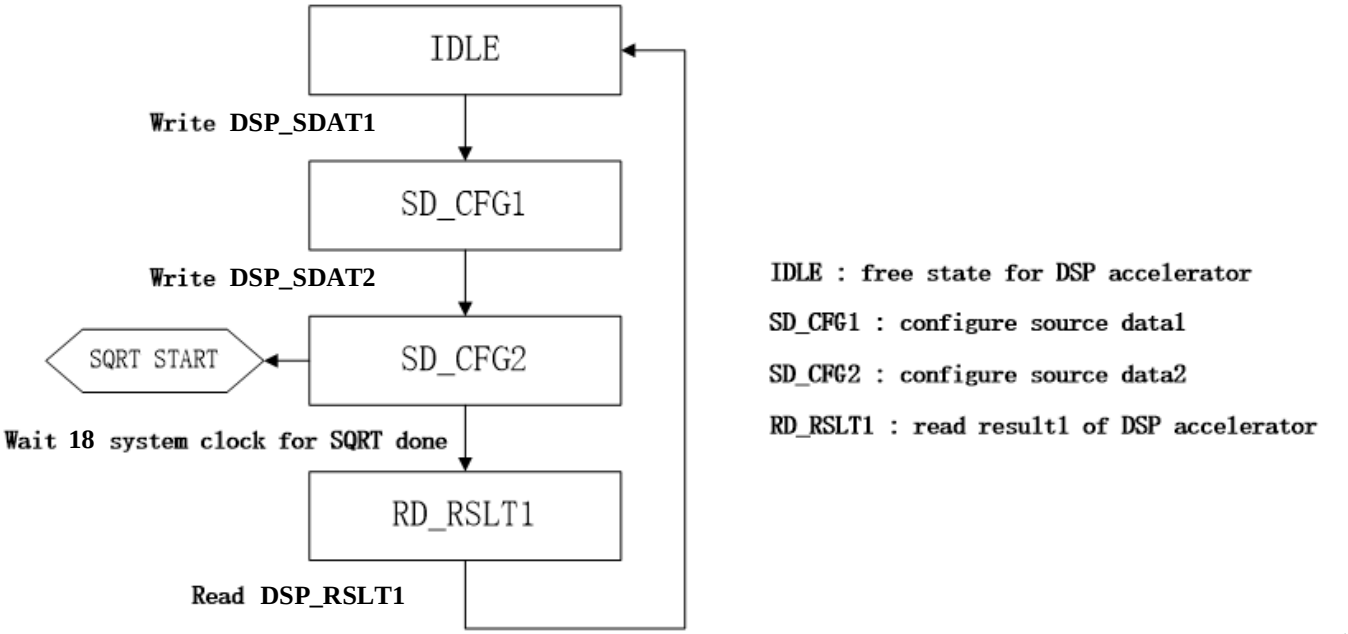


Fig. 17-2 Square root operation control flowchart

17.3 Operating Timing

17.3.1. 32-bit Signed Division

The operating timing of 32-bit signed division is shown in the figure below:

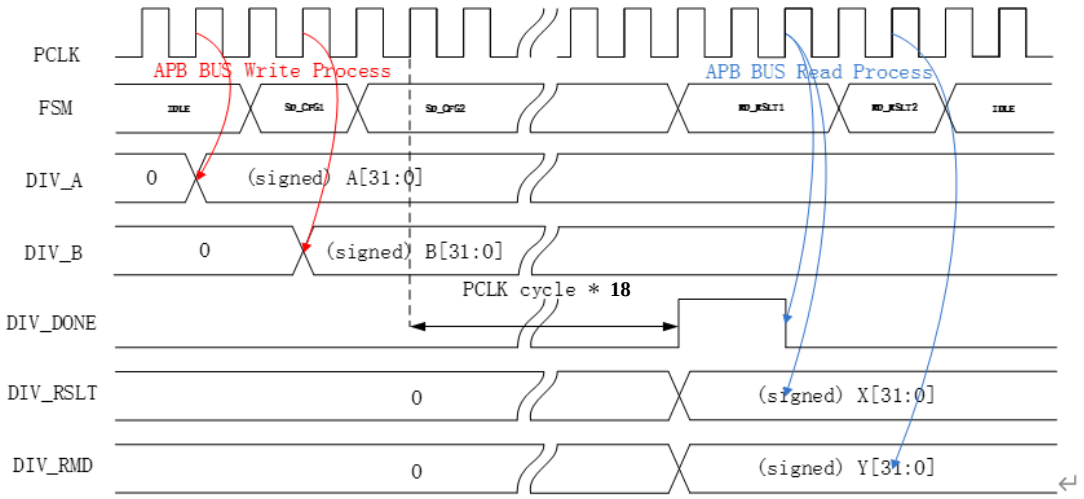


Fig. 17-3 32-bit Signed Division Operational Timing

17.3.2. 32-bit Unsigned Square Root

The operating timing of 32-bit unsigned square root is shown in the figure below:

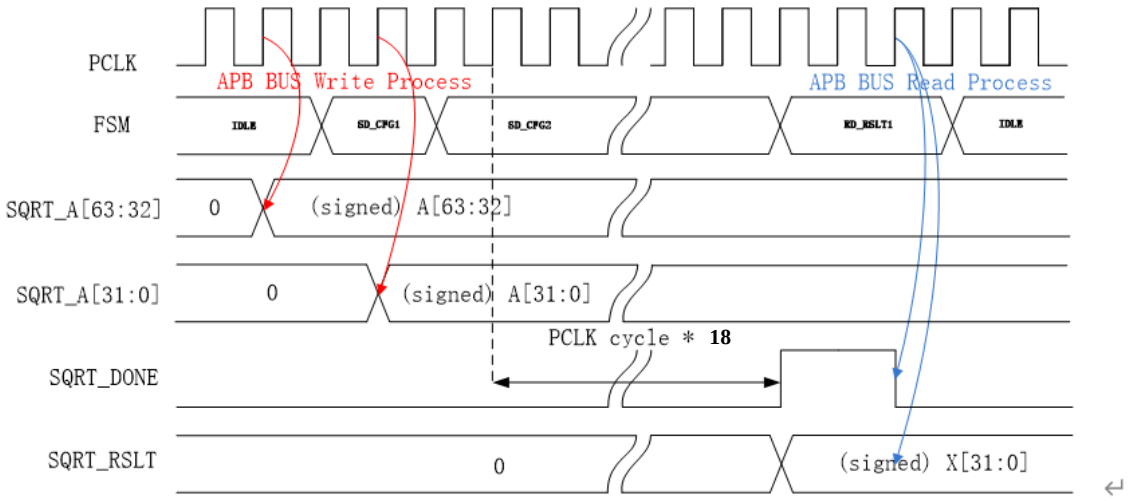


Fig. 17-4 32-bit Unsigned Division Operational Timing

17.4 Register table

Address	Name	Description
0x4000_8000	DSP_CR	DSP hardware accel control register
0x4000_8004	DSP_SR	DSP hardware accel status register
0x4000_8008	DSP_SDAT1	DSP hardware accel source data1 register
0x4000_8010	DSP_SDAT2	DSP hardware accel source data2 register
0x4000_8018	DSP_RSLT1	DSP hardware accel result1 register
0x4000_8020	DSP_RSLT2	DSP hardware accel result2 register

Table 17-1 DSP register table

17.5 Register description

17.5.1 DSP hardware accel control register(DSP_CR)

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2:0	MODE	R/W	0x0	Operation Mode Selection (Must be configured before starting a new operation mode and must not be modified during an ongoing operation) 0x1 : 32-bit Signed Division Mode 0x4 : 32-bit Unsigned Square Root Mode Others : Reserved

17.5.2 DSP hardware accel status register(DSP_SR)

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	DONE	R	0x0	32-bit Signed Division/32-bit Unsigned Square Root Operation Completion Flag 0: Operation idle 1: Operation completed (Cleared after reading the DSP_ACCEL_RSLT1 register or the DSP_ACCEL_RSLT2 register)

17.5.3 DSP hardware accel source data1 register(DSP_SDAT1)

Bit	Name	R/W	Reset	Description
31:0	SDAT1	R/W	0x0	Data Source 1 Configuration for Different Operations - 32-bit division operation: input A[31:0] (32-bit signed value) - 32-bit square root operation: input A[31:0] (32-bit unsigned value)

17.5.4 DSP hardware accel source data2 register(DSP_SDAT2)

Bit	Name	R/W	Reset	Description
31:0	SDAT2	R/W	0x0	Data Source 2 Configuration for Different Operations - 32-bit division operation: input B[31:0] (32-bit signed value) - 32-bit square root operation: write 0 (32-bit unsigned value)

17.5.5 DSP hardware accel result1 register(DSP_RSLT1)

Bit	Name	R/W	Reset	Description
31:0	RSLT1	R	0x0	Result 1 for Different Operations 32-bit division operation: quotient output X[31:0] 32-bit square root operation: result output X[31:0]

17.5.6 DSP hardware accel result2 register(DSP_RSLT2)

Bit	Name	R/W	Reset	Description
31:0	RSLT2	R	0x0	Result 2 for Different Operation Modes 32-bit division operation: remainder output Y[31:0] 32-bit square root operation: invalid / reserved

18. Comparator (COMP)

18.1 Overview

Comparator modules compare two analog input signals. The output state is determined by the polarity relationship between the positive and negative input terminals.

The comparator output state is read through the corresponding register bits. Depending on the application requirements, the comparison result can be used to generate internal interrupts or routed to an external device pin (e.g., a port's digital output).

The comparator output is processed by a digital filtering module to suppress glitches that may occur during signal transition states. This filtering ensures the reliability of both internal interrupt generation and external device pin output routing.

18.2 Block Diagram

The following figure shows the internal block diagram of the comparator module:
The usage of VC0A_O and VC1A_O refers to the COMP1_0A_SEL bits in the SYSCFG_TIM2_CON_SEL and SYSCFG_EPWM_CON_SEL register.

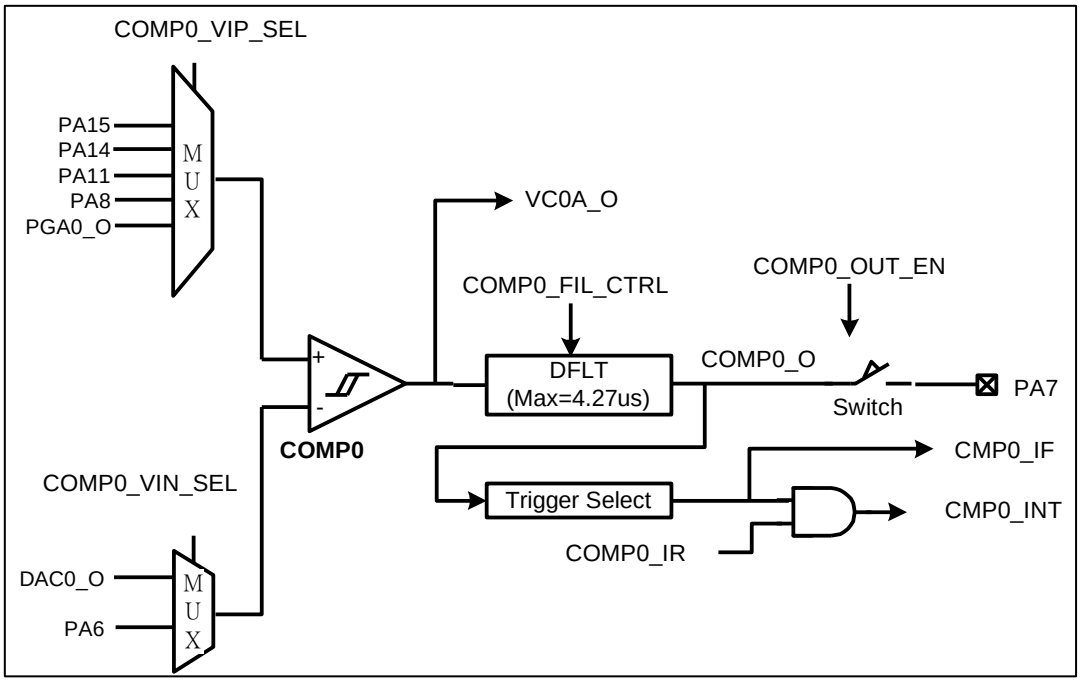


Fig. 18-1 COMP0 block diagram

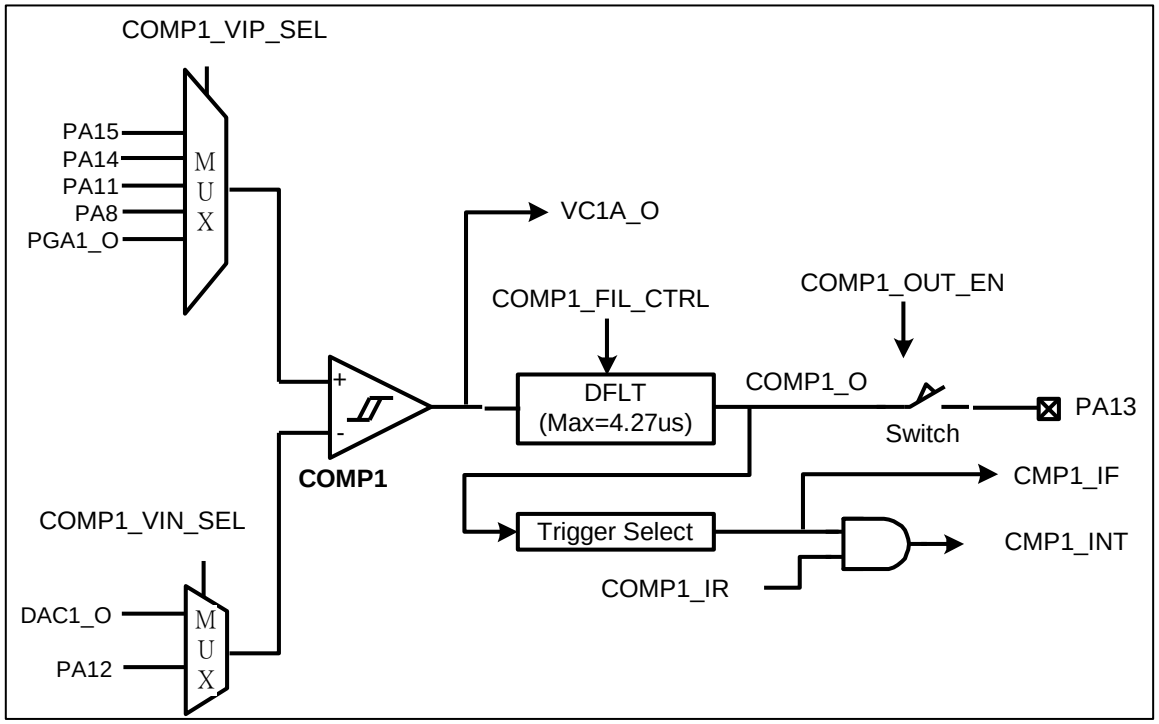


Fig. 18-2 COMP1 block diagram

18.3 Functional Description

18.3.1. Comparator Control

When the comparator's positive input ($V+$) is greater than the negative input ($V-$), the comparator output is High; otherwise, the output is Low.

Both positive and negative inputs are selectable. The positive input can be connected to either an I/O pin or the PGA output, controlled by the COMP_VIPSEL register. The negative input can be connected to either an I/O pin or the DAC output, controlled by the COMP_CTRL.VIN_SEL bit. Refer to the associated register description for detailed configuration.

When the comparator is disabled, its output is held Low. When enabled, the comparator output is valid following the initialization delay and the comparator propagation delay.

The initialization delay is dependent on the selection of the negative input:

- When the negative input is an I/O voltage, the maximum initialization delay is 17.1 µs.
- When the negative input is the DAC output, the maximum initialization delay is 34.2 µs.

Note:
The initialization delay is required each time the user updates the DAC control register AMISC_DAC_CR (see Section 15 – Analog Miscellaneous) to ensure the DAC output voltage stabilizes.

The default initialization delay is 60 system clock cycles (based on a 60 MHz system clock). If the system clock frequency changes, the initialization delay count must be adjusted. The delay is configured through the COMP_INITCNT register, where INIT_DELAY specifies the number of system clock cycles.

Following the initialization delay, the digital filter module is enabled. The filter suppresses glitches in the comparator output caused by input voltage crossings, leading to a valid and stable comparator result after the propagation delay. The digital filter introduces an additional delay that is added to the propagation delay.

The specific comparator control timing is shown below:

(1) The negative input is an I/O voltage.

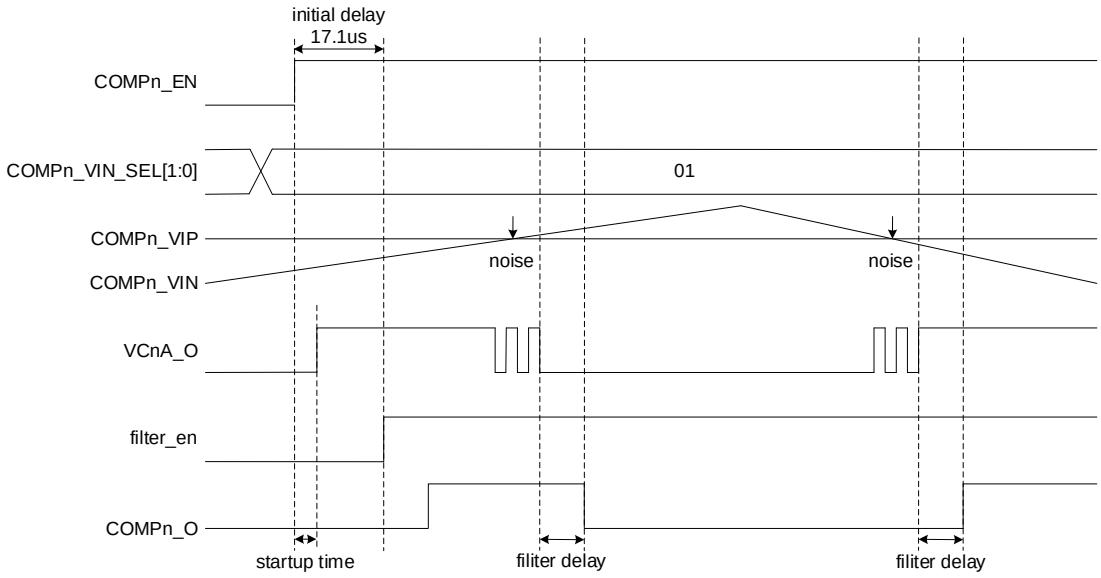


Fig. 18-3 Comparator Control Timing for Negative Input from an I/O Voltage

(2) The negative input is the DAC output.

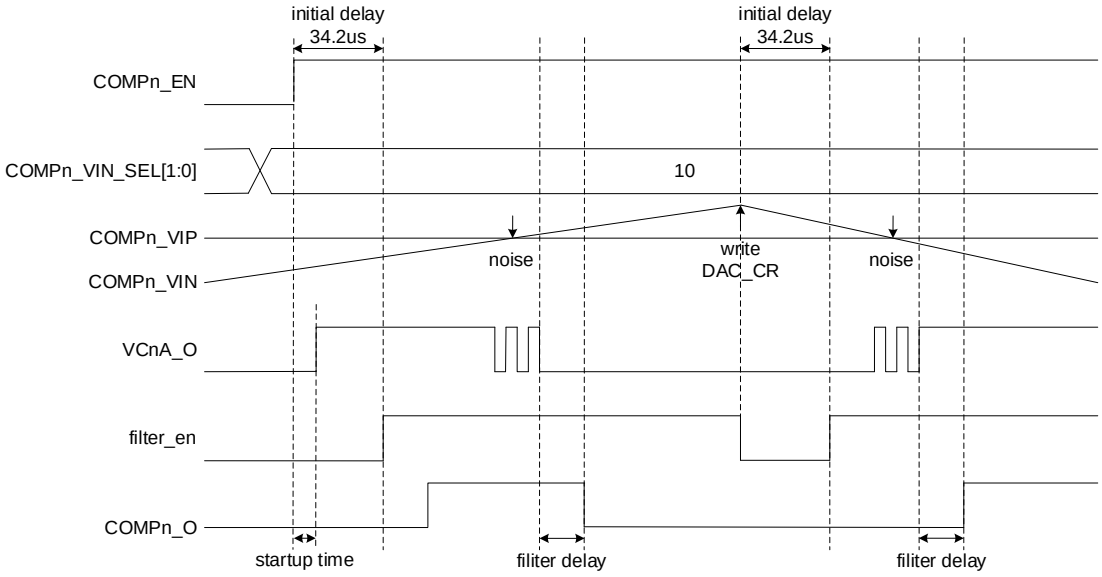


Fig. 18-4 Comparator Control Timing for Negative Input from the DAC Output

The signal filter_en serves as the hardware enable control for the digital filter module. It is automatically managed by hardware to disable the digital filter during the initialization phase.

When the filter_en signal is low, the digital filter does not process the comparator's output and holds its previous digital output state. This state retention prevents unintended interrupt triggers during the initialization delay period.

The output polarity of the comparator can be modified by configuring the POL_SEL bit in the COMP_CTRL register. If the POL_SEL bit is set to 1, the comparator output polarity is inverted; otherwise, the original output polarity is retained.

18.3.2. Digital Filtering

Comparator outputs are prone to glitches when the input signals at the positive and negative terminals cross each other. To eliminate indeterminate states generated during such crossings, the comparator result is routed to the digital filter module.

The user can configure the number of filter sampling cycles through the FIL_CTRL bits (Check if it's multiple bits) in the COMP_CTRL register. This allows filtering of glitches of different widths, with a maximum filterable glitch width of 256 PCLK cycles. For detailed configuration, refer to the description of the FIL_CTRL bits in the COMP_CTRL register.

The principle of digital filtering is as follows: within a unit period, the comparator output is sampled a number of times specified by the FIL_CTRL setting. If all consecutive sampled points have the same logic level, the filter outputs the current level; otherwise, the filter output retains its previous state.

To simplify this process, the filter output state is divided into two levels: High and Low (default is Low).

When the filter output state is Low, the Logic AND of all sampled points is evaluated. If the result is 1 (indicating all sampled points were High), the filter output toggles, and the output state changes to High. If the result is 0, the output remains Low.

When the filter output state is High, the Logic OR of all sampled points is evaluated. If the result is 0 (indicating all sampled points were Low), the filter output toggles, and the output state changes to Low. If the result is 1, the output remains High.

The filtering procedure of the digital filter circuit is illustrated as follows:

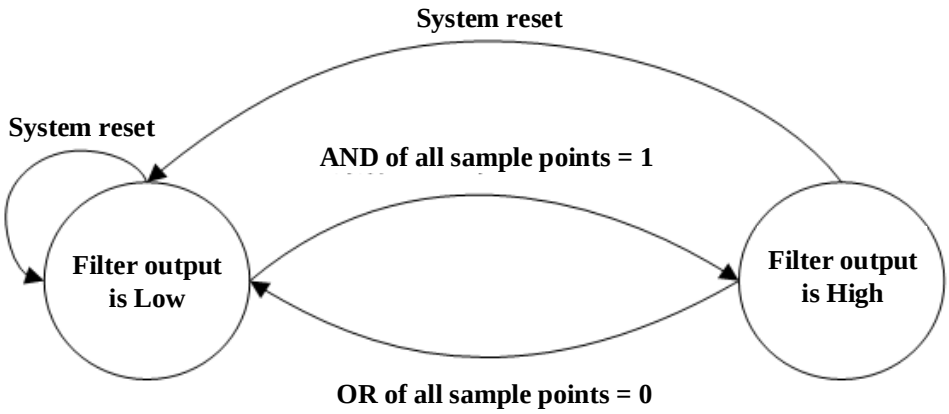


Fig. 18-5 Digital Filter Operation Flow

The FIL_CTRL bits default to no filter sampling. Since the digital filter introduces a delay, setting a large FIL_CTRL value may reduce the comparator's response speed and cause it to fail to respond to rapidly changing input signals. Therefore, the user must configure the FIL_CTRL value appropriately according to the specific application scenario.

18.3.3. Interrupt Generation

After the comparator output is processed by the digital filter, it can generate corresponding interrupts based on application requirements.

Three interrupt trigger modes are provided, based on signal edge changes: rising-edge triggered, falling-edge triggered, and edge-triggered (triggered on both rising and falling edges). The user can enable these interrupts through the COMP_IR interrupt enable register:

- Setting the RIE bit to 1 triggers an interrupt when the comparator output transitions from Low to High (rising edge).
- Setting the FIE bit to 1 triggers an interrupt when the comparator output transitions from High to Low (falling edge).
- Setting both RIE and FIE bits to 1 triggers an interrupt on both rising and falling edges.

When interrupts are triggered on both rising and falling edges, the user can check the edge type of the currently triggered interrupt via the COMP_IF interrupt flag register. For more details, refer to the description of the COMP_IF register.

18.4 Register table

Address	Name	Description
0x4000_8800	COMP0_CTRL	Comparator 0 control register
0x4000_8804	COMP0_VIPSEL	Comparator 0 P terminal input selection register
0x4000_8808	COMP0_IR	Comparator 0 Interrupt enable register
0x4000_880C	COMP0_IF	Comparator 0 interrupt flag register
0x4000_8810	COMP0_INITCNT	Comparator 0 set initial delay register
0x4000_9800	COMP1_CTRL	Comparator 1 control register
0x4000_9804	COMP1_VIPSEL	Comparator 1 P terminal input selection register
0x4000_9808	COMP1_IR	Comparator 1 Interrupt enable register
0x4000_980C	COMP1_IF	Comparator 1 interrupt flag register
0x4000_9810	COMP1_INITCNT	Comparator 1 set initial delay register

Table 18-1 COMP register table

18.5 Register description

18.5.1 Comparator n control register(COMPn_CTRL, n=0,1)

Bit	Name	R/W	Reset	Description
31:13	-	R	0x0	Reserved
12	POL_SEL	R/W	0x0	Comparator output polarity selection 0: Normal polarity 1: Inverted polarity
11	OUT_EN	R/W	0x0	Comparator Output enable 0: Disabled 1: Enabled (COMP0: PA7, COMP1: PA13)
10:9	-	R	0x0	Reserved
8	HYS_EN	R/W	0x0	Comparator hysteresis enable 0: Disabled 1: Enabled
7:4	FIL_CTRL	R/W	0x0	Digital filter sampling selection 0x0: Bypass (no filter) 0x1: 2 samples 0x2: 4 samples 0x3: 8 samples 0x4: 16 samples 0x5: 32 samples 0x6: 64 samples 0x7: 128 samples

				0x8~0xF: 256 samples
3:2	VIN_SEL	R/W	0x0	Comparator n terminal input selection 0x1: IO (COMP0: PA6, COMP1: PA12) 0x2: DAC(COMP0: DAC0, COMP1: DAC1) Others: reserved
1	COUT	R	0x0	Comparator output result 0: Low level 1: High level
0	EN	R/W	0x0	Comparator module enable 0: Disabled 1: Enabled

18.5.2 Comparator n P terminal input selection register(COMPn_VIPSEL, n=0,1)

Bit	Name	R/W	Reset	Description
31:6	-	R	0x0	Reserved
4:0	VIP_SEL	R/W	0x10	Comparator p terminal input selection 0x10: IO3 (PA15) 0x08: IO2 (PA14) 0x04: IO1 (PA11) 0x02: IO0 (PA8) 0x01: PGA(COMP0: PGA0, COMP1: PGA1) Others: reserved

18.5.3 Comparator n interrupt enable register(COMPn_IR, n=0,1)

Bit	Name	R/W	Reset	Description
31:4	-	R	0x0	Reserved
3:2	-	R/W	0x0	Reserved
1	RIE	R/W	0x0	Comparator output rising edge interrupt enable Control 0: Rising edge interrupt disabled. 1: Rising edge interrupt enabled.
0	FIE	R/W	0x0	Comparator output falling edge interrupt enable Control 0: Falling edge interrupt disabled. 1: Falling edge interrupt enabled.

18.5.4 Comparator n interrupt flag register(COMPn_IF, n=0,1)

Bit	Name	R/W	Reset	Description
31:2	-	R	0x0	Reserved
1	RIF	R/W1c	0x0	Comparator output rising edge interrupt flag 0: No rising edge interrupt has occurred. 1: A rising edge interrupt has occurred. Write 1 to this bit in software to clear it.
0	FIF	R/W1c	0x0	Comparator output falling edge interrupt flag 0: No falling edge interrupt has occurred. 1: A falling edge interrupt has occurred. Write 1 to this bit in software to clear it.

18.5.5 Comparator n set initial delay register(COMPn_INITCNT, n=0,1)

Bit	Name	R/W	Reset	Description
31:10	-	R	0x0	Reserved
9:0	INIT_DELAY	R/W	0x3C	Comparator set initial delay. (default: 60 system clock cycles)

19. Universal asynchronous receiver transmitter (UART)

19.1. UART overview

The Universal Asynchronous Receiver/Transmitter (UART) provides a flexible serial data exchange interface. The UART offers a very wide range of baud rates using a fractional baud rate generator. The main features of the UART module include:

- Programable baud rate:9600bps、19200bps、28800bps、38400bps、57600bps、 115.2Kbps.
- Programable data length:8 data bits, 7 data bits + 1 parity bit, 8 data bits + 1 parity bit, 9 data bits
- Support hardware odd or even check
- Programble stop bit:0.5, 1, 1.5 or 2bits.
- Embedded transmit & receive fifo buffers with 9-bit data width and depth of 8 under
- Full-duplex asynchronous operation

19.2. UART features

The UART protocol can be configured by setting the UART_CR register. Serial transmission becomes active immediately when data is written to UART_TXB (a write-only UART_DAT register). Due to the transmit FIFO buffer, software may continuously write multiple data bytes until the buffer is full. Received data and the corresponding parity bit are stored in UART_RXB (a read-only UART_DAT register). UARTn_RXB is implemented as an 8-level FIFO buffer. Regardless of whether the software immediately reads the received data, the UART module continues receiving subsequent bytes until the FIFO buffer becomes full. An overflow error occurs when the FIFO buffer is full and additional data is received. When an overflow occurs, the overflow status flag OVERR is set to ‘1’, and the oldest data in the FIFO is discarded. The UART_ST register is also updated accordingly to reflect the reception of new data. A loopback communication mode is provided. In full-duplex mode, both devices must operate at the same baud rate. Transmit data is output on the TXD pin, whereas received data is captured on the RXD pin.

- Serial data transmission and reception can only proceed after the baud-rate generator control bit (RUN) is set to ‘1’. If the RUN bit is cleared to ‘0’, the transmit/receive process is terminated immediately, and the TXD pin will remain idle (logical high for normal polarity). Unless UART is idle, the RUN bit should always remain set to ‘1’.
- UART will be unpredictable if UART_CR.MODE is set to a reserved state.
- In 9-bit data mode, or in 8-bit data plus receive wake-up mode, the parity error interrupt must be masked to prevent false parity error detection, because no parity bit is present in the data frame in these two configuration modes.

19.3. UART frame format

The length of the data frame can be set as 8 bits or 9 bits.

Two types of 8-bit data frame:

- 8 data bits (UARTn_CR.MODE = 0x1)
- 7 data bits plus one parity bit (UART_CR.MODE = 0x3)

Three types of 9-bit data frame:

- 9 data bits (UARTn_CR.MODE = 0x4)
- 8 data bits plus one parity bit (UARTn_CR.MODE = 0x7)
- 8 data bits plus one wake-up bit (UARTn_CR.MODE = 0x5)

The generation of the parity bit can be either odd or even, depending on the setting of UART_CR.PAR. When set to odd parity, if the number of “1s” is even, the parity bit is set to 1; in the case of even parity, the parity bit is set to 0. During reception, the parity bit will be stored to the receive buffer UART_RXB along with the data bits.

When UART is in mute mode, the received data will store to UART_RXB and triggers a receive interrupt request only if the 9th bit of the data frame is 1. If the 9th bit is 0, no interrupt request is generated, and no data is stored in UART_RXB.

When the master needs to send a sequence of data to a specific slave, it first transmits an 8-bit address plus a 9th bit set to 1, forming a 9-bit addressing frame. All slaves can receive this special addressing frame (because its 9th bit is 1) and compare it with their configured address. If the address matches, the slave immediately switch to 9-bit data mode and continues to receive the following data frames (whose 9th bit is 0). Slaves with non-matching addresses do not respond to subsequent data frames, avoiding unnecessary interrupt handling and improving system efficiency. UART can only switch mode during idle state. To ensure that the UART can properly send and receive data after switching mode, the transmit and receive FIFO buffer should be reset.

19.4. Buad rate generation

The UART_CR.RUN bit enables the baud rate generator. The baud-rate control register UARTn_BR is divided into two fields: The BR field, which is 16 bits wide and serves as the system clock divider, and the SR field, which is 2 bits wide and determines the sampling rate of each symbol. The software configures the baud rate of UART communication by setting the value of UARTn_BR, using the calculation formula shown below:

$$BPS_{UART} = f_{SYS} / (16/(2^{SR})*BR) \text{ or } BR = f_{SYS} / (16/(2^{SR})*BPS_{UART})$$

Which:

- BPS_{UART} is the desired baud rate, used for both reception and transmission.
- f_{SYS} is the system clock frequency.
- BR is the value configured in the BR field of the UARTn_BR register.
- SR is the value configured in the SR field of the UARTn_BR register.

The commonly used baud-rate and their corresponding errors (with SR = 0x0) under different system clock frequencies are listed below for quick reference.

	f_{CLK} = 64MHz		f_{CLK} = 60MHz		f_{CLK} = 56MHz	
Buad rate	UART_BR	error	UART_BR	error	UART_BR	error
115200	0x0022	-0.21%	0x0020	0.04%	0x001E	0.10%
76800	0x0034	-0.16%	0x0030	0.06%	0x002D	0.12%
57600	0x0045	-0.64%	0x0041	0.05%	0x003C	0.11%
38400	0x0068	-0.16%	0x0061	0.06%	0x005B	0.12%
19200	0x00D0	-0.16%	0x00C3	0.06%	0x00B6	0.12%
9600	0x01A0	-0.16%	0x0186	0.06%	0x016C	0.12%
4800	0x0341	-0.04%	0x030D	0.06%	0x02D9	0.12%
200	0x4E20	0%	0x493E	0.06%	0x445C	0.12%

Table 19-1 commonly used baud-rate table

If a non-standard baud rate is required, the UART_BR value can be calculated by using the formula above. Baud rate error should be under 1%; otherwise, reliable communication cannot be ensured.

19.5. UART transmitter

If a data is written to UART_TXB(UART_DAT) register when UART_CR.RUN is 1. Serial transmission will be initiated. The transmitted data frame consists of three parts:

- One start bit
- 8 or 9 data bits, the LSB is transmitted first.
- One optional parity bit
- 0.5, 1, 1,5 or 2 stop bit

The transmit queue has an 8-level FIFO buffer. While data is being transmitted, more data can be continuously written to UART_TXB until the buffer is full. After the last bit of the final data frame is transmitted, the transmitter idle status flag TXE is set to 1, indicating that transmission has been completed. The polarity of transmission can be configured. In normal polarity, the TXD pin remains high when no data is being transmitted; the start bit is low, the stop bit is high, and data bits are represented as 0 = low and 1 = high. In inverted polarity, all signal levels are the opposite of normal polarity.

Data transmission typically follows the steps below, taking TXE polling as an example

1. Configure work mode (CR.MODE), baud rate (BR), parity check (CR.PAR), stop bit length (CR.STOPB) and polarity (CR.TXPOL).
2. Enable baud rate generator (CR.RUN)
3. Write data to FIFO (UART_DAT).
4. Check that ST.TXE is 0, then enable inturrpt (IE.TXEE)
5. Wait for ST.TXE to trigger an interrupt, indicating that data transmission is complete, then disable the interrupt (IE.TXEE).

19.6. UART receiver

When the RUN bit and RXEN is set to 1, UART module is in receive mode. Reception of a data frame is initiated when a start bit is detected on the RXD pin (falling edge for normal polarity). Each data bit is sampled 16 times and is determined by using a majority-vote method, preventing occasional noise from causing incorrect data reception. After the final stop bit is received, the UART module stores the data into the receive buffer queue UART_RXB (i.e., UARTn_DAT) and sets the queue non-empty flag RXNE to 1. If the receive queue is already full, flag RXF is set to 1. However, the parity error flag PERR and frame error flag FERR are not updated at this time (the receive data error flags are updated when the data is read from the receive queue).

If the RXEN bit is cleared during the reception of a data frame, the current frame being received can still be fully received and all status flags will be updated. However, subsequent data frames will not be received. In mute mode, only data frames with the 9th bit set to 1 can be received and stored in the receive buffer queue with status flags updated; data frames with the 9th bit set to 0 are discarded.

Data reception typically follows the steps below, taking RXNE polling as an example.

1. Configure work mode (CR.MODE), baud rate (BR), parity check (CR.PAR), stop bit length (CR.STOPB) and polarity (CR.TXPOL).

2. Enable baud rate generator (CR.RUN) and timeout function (TO).

3. Enable interrupt (IE.RXNEE)

4. An interrupt is triggered when ST.RXNE is set, indicating data reception is complete. The (IE.RXNEE) interrupt should then be disabled.

19.7. UART error check

To ensure reliable serial communication, the UART module provides communication error detection mechanisms and reflects different error conditions through status bits in the UART_ST status register.

When the UART_RXB queue overflows, the UART_ST.OVERR flag is set immediately.

When data is read from the receive queue, the corresponding parity and frame error information is reflected in the UART_ST register. If a parity error occurs, the UART_ST.PERR bit is set to 1; if a frame reception error occurs (i.e., a logic 1 is not detected at the stop bit), the UART_ST.FERR bit is set to 1.

All error flag bits can trigger a UART error interrupt through a logical “OR” operation if their corresponding interrupt enable bits are set. The software must individually check and handle each error condition.

19.8. UART interrupt

The status register UART_ST and the interrupt enable register UART_IE are used to control interrupt responses. Relevant flag bits in the status register indicate the type of interrupt that has occurred, while the corresponding bits in the interrupt enable register determine whether the type of interrupt is enabled. Multiple enabled interrupt flags are logically ORed to generate a single unified interrupt request.

The method for clearing interrupt flags differs depending on the specific flag. Transmit-related interrupt flags TXE and TXHE are automatically cleared by hardware based on the amount of data written into the transmit buffer queue. Receive-related interrupt flags RXF are automatically cleared by hardware when the receive buffer queue is is not full. RXNE is automatically cleared by hardware only after the receive buffer queue is empty. Error interrupt flags PERR, FERR, and OVERR can only be cleared by writing “1” to the corresponding bit in the UART_ST status register.

The conditions for generating interrupt flags and their descripttions during can be summarized as follows. There are three available interrupt flags for data transmission and reception.

Data transmit flag:

- TXE: This flag is set to 1 when the last data in the transmit queue has been transferred to the shift register, emptying the queue. However, the transmit operation may still be in the process. When the UART is idle and no data is being transmitted, the queue is guaranteed to be empty; therefore, if this interrupt is enabled by software, the interrupt service routine will be entered immediately.

- TXHE: This flag is set to 1 when a data entry in the transmit queue has been transferred to the shift register and the available space in the queue is equal to or greater than half of its total length. At this point, the transmit buffer queue is half-empty.

- TXEND: This flag is set to 1 after the last bit of the final data has been transmitted. At this point, the transmit buffer queue is guaranteed to be completely empty.

Data receive flag:

- RXNE: This flag is set to 1 after a received data frame is transferred from the receive shifter into the UART_RXB receive buffer queue, indicating that the receive buffer queue is not empty (i.e., it contains at least one data frame).

- RXHF: This flag is set to 1 when a received data frame is transferred from the receive shifter into the UART_RXB receive buffer queue and the queue becomes more than half full**,** indicating that the receive buffer queue is over half occupied but not yet full**.**

- RXF:When a received data frame is transferred from the receive shifter into the UART_RXB receive buffer queue and the queue becomes full, this flag is set to 1, indicating that the receive buffer queue is completely full. If the buffer is already full, the next received data will cause a overflow error.

For both sets of flags used in normal and inverted-polarity reception, if their corresponding interrupt-enable bits are set to 1, they are logically ORed to trigger a receive interrupt. The software must then check each flag individually, determine the cause, and handle it accordingly.

UART interrupt:

- Transmission interrupt: If any of the TXEND, TXHE, or TXE bits in the UARTn_STm status register is set to 1 and the corresponding bits in the UARTn_IEm interrupt enable register are also set to 1, an interrupt request is triggered. Within the interrupt service routine, the application software must check both the relevant status and control bits to determine which bit(s) triggered the interrupt.

- Reception interrupt: If any of the RXF, RXHF, or RXNE bits in the UARTn_STm status register is set to 1 and the corresponding bits in the UARTn_IEm interrupt enable register are also set to 1, an interrupt request is triggered. Within the interrupt service routine, the application software must check both the relevant status and control bits to determine which bit(s) triggered the interrupt.

- Error interrupt: If any of the TOIDLE, TONE, OVERR, FERR or PERR bits in the UARTn_STm status register is set to 1 and the corresponding bits in the UARTn_IEm interrupt enable register are also set to 1, an interrupt request is triggered. Within the interrupt service routine, the application software must check both the relevant status and control bits to determine which bit(s) triggered the interrupt.

19.9. Register table

Address	Name	Description
0x4000_2000	UART_DAT	UART transmit / receive data register
0x4000_2004	UART_CR	UART control register
0x4000_2008	UART_BR	UART baud rate register
0x4000_200C	UART_IE	UART interrupt enable register
0x4000_2010	UART_ST	UART status register
0x4000_2014	UART_GT	UART guard time register
0x4000_2018	UART_TO	UART time-out control register
0x4000_201C	UART_TXFR	UART transmit FIFO reset register
0x4000_2020	UART_RXFR	UART receive FIFO reset register

Table 19-2 UART register table

19.10. Register description

19.10.1 UART transmit / receive data register (UART_DAT)

UART_DAT is the FIFO buffer queue for both transmit and receive data, with a depth of 8.

The transmit data frame can be either 8 or 9 bits. The contents are defined as shown in the following table.

Bit	Name	R/W	Reset	Description
31:9	-	R	0x0	Reserved
8	TB8	W	0x0	Can serve as a data bit, a parity bit, a wake-up bit, or be fixed at 0, depending on the CR.MODE setting in the control register.
7	TB7	W	0x0	Can serve as a data bit or a parity bit, depending on the CR.MODE setting in the control register.
6:0	TD6-TD0	W	0x0	7 Data bits

When receiving data, the lower 10 bits value are valid. The contents are defined in the following table.

Bit	Name	R/W	Reset	Description
31:10	-	R	0x0	Reserved
9	FE	R	0x0	Frame error flag Is only valid in 9-bit frame mode.
8	RB8	R	0x0	Can serve as a data bit, a parity bit, a wake-up bit, or be fixed at 0, depending on the CR.MODE setting in the control register.
7	RB7	R	0x0	Can serve as a data bit or a parity bit, depending on the CR.MODE setting in the control register.
6:0	RD6-RD0	R	0x0	7 Data bits

19.10.2 UART control register (UART_CR)

Bit	Name	R/W	Reset	Description
31:18	-	R	0x0	Reserved
17	TXPOL	R/W	0x0	TX data polarity inverse control 0: Idle state = high, start bit = low; data bit 1 = high, data bit 0 = low 1: Idle state = low, start bit = high; data bit 1 = low, data bit 0 = high
16	RXPOL	R/W	0x0	RX data polarity inverse control 0: Idle state = high, start bit = low; data bit 1 = high, data bit 0 = low 1: Idle state = low, start bit = high; data bit 1 = low, data bit 0 = high
15:9	-	R	0x0	Reserved
8	RXEN	R/W	0x0	RX enable 0: Disable data reception 1: Enalbe data reception
7	RUN	R/W	0x0	Buad rate generator enable When the baud-rate generator is disabled, the TXD pin remains in idle state and data reception is disabled. 0: Buad rate generator disable 1: Buad rate generator enable

6	LPB	R/W	0x0	Loop-back mode 0: Loop-back mode disable 1: Loop-back mode enable
5	PAR	R/W	0x0	Parity check 0: Even parity (the parity bit is set to 1 when the number of 1s in the data bits is odd) 1: Odd parity (the parity bit is set to 1 when the number of 1s in the data bits is even)
4:3	STOPB	R/W	0x0	Length of stop bit 00: 0.5 stop bits 01: 1 stop bit 10: 1.5 stop bits 11: 2 stop bits
2:0	MODE	R/W	0x0	Mode select 000: Reserved 001: 8 data bits 010: Reserved 011: 7 data bits + 1 parity bit 100: 9 data bits 101: 8 data bits + 1 wakup bit 110: Reserved 111: 8 data bits + 1 parity bit

19.10.3 UART baud rate register (UART_BR)

Bit	Name	R/W	Reset	Description
31:18	-	R	0x0	Reserved
15:0	BR	R/W	0x0001	Baud rate Please refer to section “UART Baud Rate generation”

19.10.4 UART interrupt enable register (UART_IE)

Bit	Name	R/W	Reset	Description
31:12	-	R	0x0	Reserved
11	TXFE	R/W	0x0	TX FIFO full interrupt enable 0: Interrupt disable 1: Interrupt enable
10	TXEND	R/W	0x0	TX end interrupt enable 0: Interrupt disable 1: Interrupt enable Since the corresponding flag has a reset value of 1, it should be enabled after data transmission begins and disabled within the interrupt handler.
9	RXFE	RW	0x0	RX FIFO full interrupt enable 0: Interrupt disable 1: Interrupt enable
8	RXHFE	RW	0x0	RX FIFO half-full interrupt enable 0: Interrupt disable 1: Interrupt enable
7	TOIDLEE	RW	0x0	TX idle time-out interrupt enable 0: Interrupt disable 1: Interrupt enable Since the corresponding flag has a reset value of 1, it should be enabled after data transmission begins and disabled within the interrupt handler.

6	TONEE	R/W	0x0	Receive buffer empty timeout interrupt enable 0: Interrupt disable 1: Interrupt enable
5	OVERRE	R/W	0x0	RX overflow interrupt enable 0: Interrupt disable 1: Interrupt enable
4	FERRE	R/W	0x0	Frame error interrupt enable 0: Interrupt disable 1: Interrupt enable
3	PERRE	R/W	0x0	Parity error interrupt enable 0: Interrupt disable 1: Interrupt enable
2	TXHEE	R/W	0x0	TX FIFO half-empty interrupt enable 0: Interrupt disable 1: Interrupt enable Since the corresponding flag has a reset value of 1, it should be enabled after data transmission begins and disabled within the interrupt handler.
1	TXEE	R/W	0x0	TX FIFO empty interrupt enable 0: Interrupt disable 1: Interrupt enable Since the corresponding flag has a reset value of 1, it should be enabled after data transmission begins and disabled within the interrupt handler.
0	RXNEE	R/W	0x0	RX FIFO not empty interrupt enable 0: Interrupt disable 1: Interrupt enable

19.10.5 UART status register (UART_ST)

Bit	Name	R/W	Reset	Description
31:12	-	R	0x0	Reserved
11	TXF	R	0x0	TX FIFO full
10	TXEND	R	0x1	TX end 0: Data transmission in progress 1: Data transmission end
9	RXF	R	0x0	RX FIFO full
8	RXHF	R	0x0	RX FIFO half-full
7	TOIDLE	R	0x1	TX idle timeout When the timeout period configured in the UART_TO register expires while the receive buffer queue is still empty, this bit is set to 1. 0:No timeout 1: Receive buffer queue empty timeout
6	TONE	R	0x0	Timeout RX fifo not empty When the timeout period configured in the UART_TO register expires while the receive buffer queue has not been cleared, this bit is set to 1. 0:No timeout 1: Receive buffer queue not empty timeout
5	OVERR	R/W1c	0x0	RX FIFO overflow flag 1: Overflow occur
4	FERR	R/W1c	0x0	Frame error flag 1:Frame error occur
3	PERR	R/W1c	0x0	Parity check error flag 1: Parity check error occur
2	TXHE	R	0x1	TX FIFO half-empty
1	TXE	R	0x1	TX FIFO empty
0	RXNE	R	0x0	RX FIFO not empty

19.10.6 UART guard time register(UARTn_GT)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	GT	R/W	0x0	The module reloads the interval time value configured in this register when the transmit buffer queue becomes empty.

19.10.7 UART time-out control register(UART_TO)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	TO	RW	0x0	If UARTn starts receiving data, the UART_ST.TONE and UART_ST.TOIDLE bits will be cleared to 0, and will not be updated again until the programmed timeout period expires.Timeout counting is performed when UART is enabled and the receiver is in the idle state. Each read of the UART_DAT register or a rewrite of the UART_TO register will restart the timeout counter. If UART is not enabled, the values of UART_ST.TONE and UART_ST.TOIDLE are updated immediately according to the status of the receive buffer.

19.10.8 UART TX FIFO reset register (UART_TXFR)

Bit	Name	R/W	Reset	Description
31:0	TXFR	W	-	After a write operation to this register, regardless of the value written, the UARTn_DAT transmit buffer queue is reset.

19.10.9 UART RX FIFO reset register (UART_RXFR)

Bit	Name	R/W	Reset	Description
31:0	RXFR	W	-	After a write operation to this register, regardless of the value written, the UARTn_DAT transmit buffer queue is reset.

20. Serial peripheral interface (SPI)

20.1. Overview

The SPI serial communication standard was developed by Motorola. It uses a 3-wire connection to enable data communication between a microcontroller and peripheral devices, or between microcontrollers. Through chip-select control of the slave devices, an SPI bus with a master-slave architecture can be formed. The SPI communication pins are described as follows:

- SCK: The serial clock is generated only by the SPI master.
- MISO: Master In / Slave Out data.
- MOSI: Chip select, each slave device has its own dedicated chip-select signal.

SPI communication is full-duplex, the MSB is transmitted and received first. Data reception occurs simultaneously with data transmission. Depending on how the clock is provided, SPI operates in either master or slave mode. The SPI master generates the clock and controls the start and end of communication. The SPI slave receives the clock externally (from the SPI master) and responds passively to the master. A typical SPI communication system is shown in Figure 20-1.

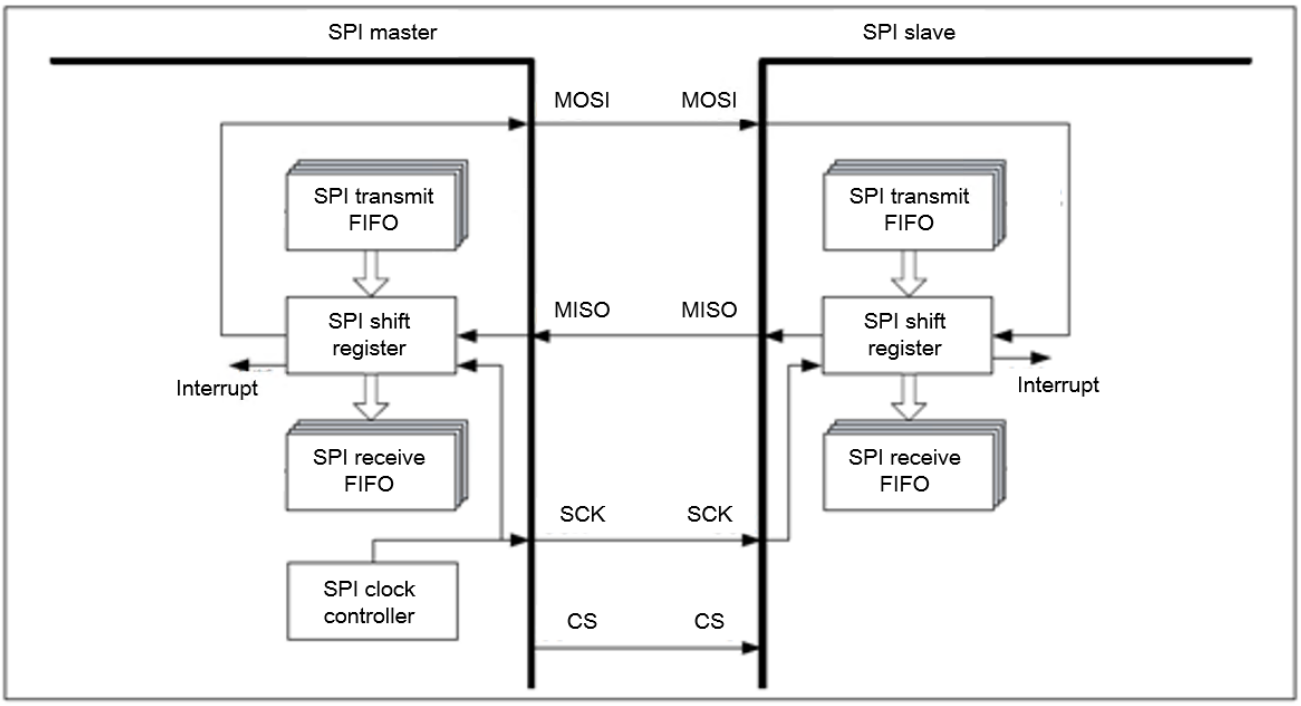


Fig. 20-1 SPI block diagram

PEC930 SPI features:

- The transfer word size is configurable with 2 bits, supporting 8, 16, 24, or 32-bit data sizes.
- Bit order can be LSB or MSB.
- The maximum master transmission rate is 10 Mbps (system clock 60 MHz).
- Supports four slave chip-select outputs in master mode
- 8-entry × 4-byte-depth transmit / receive FIFO
- The master SPI transfer rate can be programmed by applying an internal clock scaler.
- Programmable interrupt generation

20.2. SPI operating modes

20.2.1 SPI master mode

SPI can only initiate data transfer in master mode. In master mode, the SPI controls data transfer between itself and any connected slave device. The 10th bit in the SPI_CR .0 register can be configured to drive the CS pin, allowing selection of slave device for communication.

When both the spi_enable bit in the SPI_ENABLE register and the mode bit (bit 0) in the SPI_CR register are set high, the SPI enters master mode. In addition to this automatic peripheral chip-select, manual peripheral chip-select mode can be enabled through the MCS bit in the SPI_CR register.

20.2.2 SPI slave mode

In slave mode, the SPI module receives data from master and can simultaneously transmit data. In this mode, the slave clock input comes from the master's clock. The clock polarity and phase can be configured through registers; for detailed configuration, refer to the clock configuration section. When the mode bit (bit 0) in the SPI_CR register is low and the spi_enable bit (bit 0) in the SPI_ENABLE register is set high, the SPI operates in slave mode. When the SPI operates in slave mode, if the CS pin is high (inactive), communication is not initiated. Once CS is pulled low, it must remain low (active) throughout the access period. If this signal goes high during data transmission, the bit1_mdf (modefail) bit in the SPI_SR register is set to indicate a mode failure. At the same time, the current transmission is aborted, and updates to the FIFO are inhibited. When the SPI is re-enabled (by setting bit0_spi_enable in the SPI_ENABLE register to 1), data transfers interrupted due to the mode failure will be resumed.

To allow the SPI module to synchronize with the master after being enabled, the module can detect the boundaries of SPI transfer words in slave mode. Synchronization can be achieved when any of the following conditions is detected:

1. If the CS pin is high (inactive), the slave controller will treat the first active edge of SCK after CS is pulled low as the start of transfer word. The SPI master and slave will then synchronize from this point until the spi_enable register is reconfigured.

2. When the SPI slave is enabled, the slave controller begins counting the duration of SCK in its inactive state (measured in SPI_REF_CLOCK cycles). When this count matches the value set in the IDLE COUNT register, the slave controller considers the SPI master and slave to be synchronized.

Note: SPI_REF_CLOCK is pclk

20.2.3 Clock configuration

The SPI clock phase (CPHA) and clock polarity (CPOL) are configurable. The clock phase can be set by writing to the CPHA bit (bit 2) in the SPI_CR register, and the clock polarity can be set by writing to the CPOL bit (bit 3) in the SPI_CR register. Configuring the clock phase (CPHA) in master mode has the following effects:

- When CPHA = 0, the master SPI automatically sets the chip-select signal to the inactive state and maintains it for at least one SPI_REF_CLOCK cycle. The length of this delay can be configured via the DELAY register. In this case, a delay of at least three SPI_REF_CLOCK (or external clock cycles) is maintained between the last bit of the current word and the first bit of the next word. The delay duration can be configured using the DELAY register.

- When CPHA = 1, the chip-select signal is no longer set to inactive between word transfers. In this mode, the minimum delay between the last bit of the current transfer word and the first bit of the next word is one SPI_REF_CLOCK cycle (or external clock cycle). The delay duration can also be configured via the DELAY register.

Different combinations of clock polarity and phase produce multiple data transfer formats, depending on the settings of the CPOL (clock polarity) and CPHA (clock phase) bits in the SPI module control register. The SPI supports four distinct operating modes: Mode 00, Mode 01, Mode 10, and Mode 11.

20.2.3.1. SPI mode 00

When the SPI is set to Mode 00, the SPI clock remains low in the idle state. After the master software pulls the slave chip-select signal (CS) low, the first bit of data from the slave is ready. Once the master writes data into the SPI data transmit register, the first bit of data to the slave immediately appears on the MOSI pin. Subsequently, on each rising edge of the clock, the master and slave simultaneously sample the data bit sent by the other side and load it into their shift registers, while on the following falling edge, each outputs the next data bit. After a SPI data frame (4–32 bits) is transmitted, the received data is transferred from the shift register to the SPI data receive register, and the SPI clock returns to the low state. The master can continue writing data to the transmit register for continuous transmission, or pull the slave chip-select signal high to end the SPI communication.

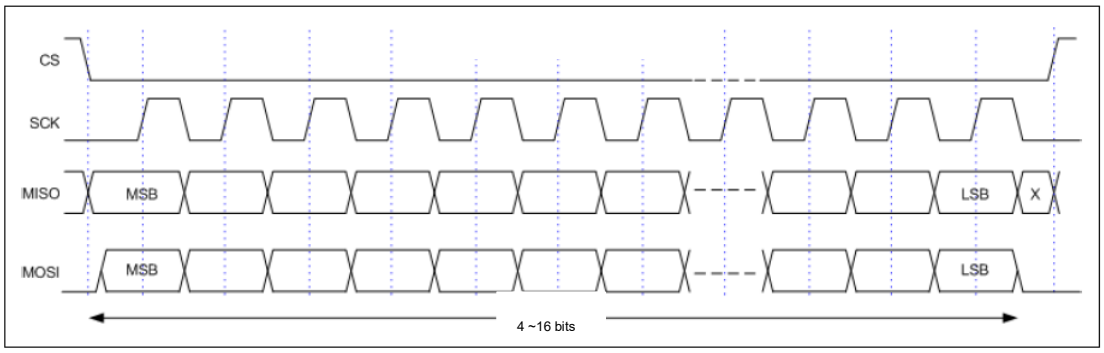


Fig. 20-2 SPI mode 00 timing diagram(CPOL = 0, CPHA = 0)

20.2.3.2. SPI mode 01

When the SPI is set to Mode 01, the SPI clock remains low in the idle state. After the master pulls the slave chip-select signal (CS) low and writes data into the SPI data transmit register, a SPI data frame transfer is initiated. On each rising edge of the clock, the master and slave each output the next data bit, and on the subsequent falling edge, they sample the data received from the other side and load it into their shift registers. After a SPI data frame (4–32 bits) is transmitted, the received data is transferred from the shift register to the SPI data receive register, and the SPI clock returns to the low state. The master can continue writing data to the transmit register for continuous transmission, or pull the slave chip-select signal high to end the SPI communication.

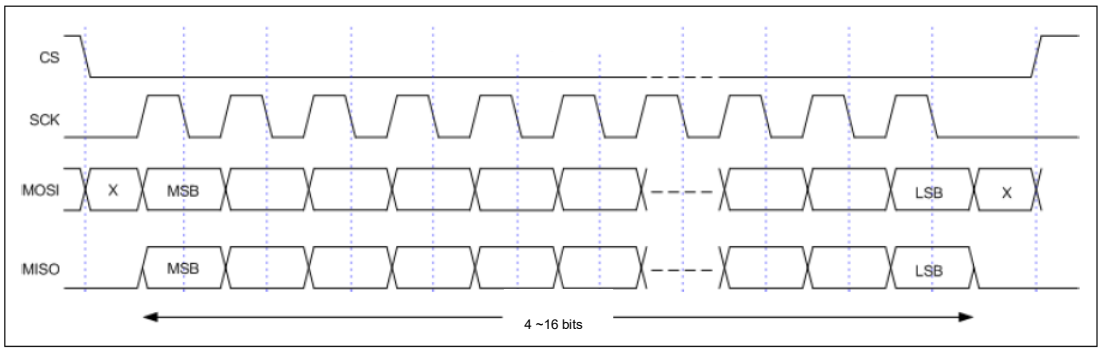


Fig. 20-3 SPI mode 01 timing diagram(CPOL = 0, CPHA = 1)

20.2.3.3. SPI mode 10

When the SPI is set to Mode 10, the SPI clock remains high in the idle state. After the master pulls the slave chip-select signal (CS) low, the first bit of data from the slave is ready on the master's MISO pin. Once the master writes data into the SPI data transmit register, the first bit of data to the slave immediately appears on the MOSI pin. Subsequently, on each falling edge of the clock, the master and slave simultaneously sample the data bit sent by the other side and load it into their shift registers, while on the following rising edge, each outputs the next data bit. After a SPI data frame (4–32 bits) is transmitted, the received data is transferred from the shift register to the SPI data receive register, and the SPI clock returns to the high state. The master can continue writing data to the transmit register for continuous transmission, or pull the slave chip-select signal high to end the SPI communication.

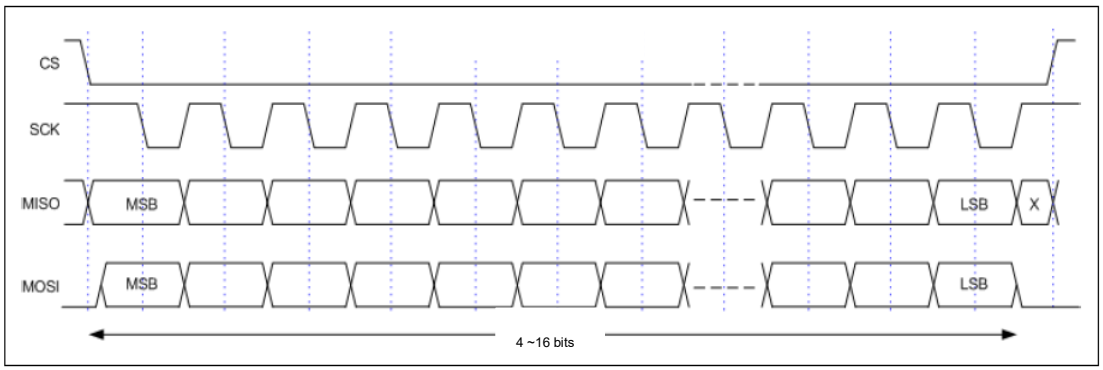


Fig. 20-4 SPI mode 01 timing diagram(CPOL = 1, CPHA = 0)

20.2.3.4. SPI 模式 11

When the SPI is set to Mode 11, the SPI clock remains high in the idle state. After the master pulls the slave chip-select signal (CS) low and writes data into the SPI data transmit register, a SPI data frame transfer is initiated. On each falling edge of the clock, the master and slave each output the next data bit, and on the subsequent rising edge, they sample the data received from the other side and load it into their shift registers. After a SPI data frame (4–32 bits) is transmitted, the received data is transferred from the shift register to the SPI data receive register, and the SPI clock returns to the high state. The master can continue writing data to the transmit register for continuous transmission, or pull the slave chip-select signal high to end the SPI communication.

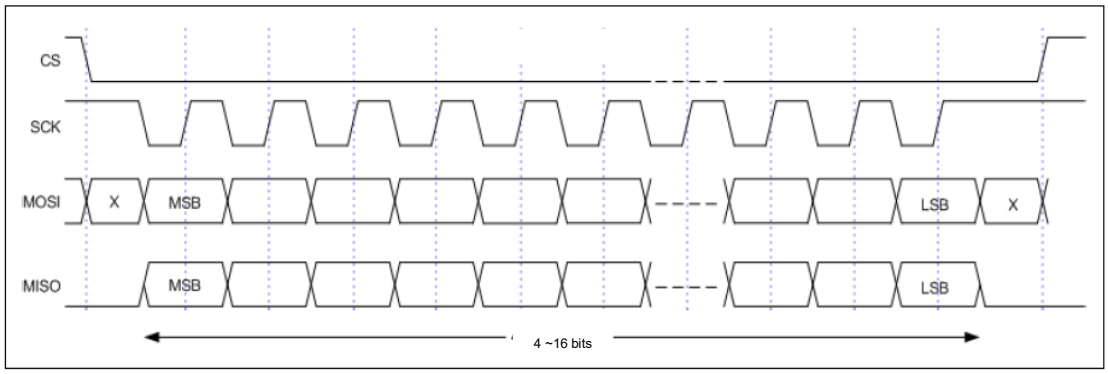


Fig. 20-5 SPI mode 11 timing diagram(CPOL = 1, CPHA = 1)

20.3. SPI baud rate

When the chip is configured as an SPI master, the communication clock is generated and controlled internally and sent to the slave. The SPI baud rate is calculated as follows:

BPSSPI= SPI_REF_CLK/ SPI_BD

Where:

- BPSSPI is the desired SPI communication baud rate, used for both transmission and reception. - SPI_REF_CLK is the SPI reference clock frequency.
- SPI_BD is the baud-rate divisor setting, which can be 2, 4, 8, 16, 32, 64, 128, or 256.

When the chip is configured as an SPI slave, the SPI communication clock is provided by an external source.

20.4. SPI Clock

The SPI master clock domain is SPI_REF_CLOCK, and all read or write operations of the shift registers are governed by this clock. In addition, the SPI state machine and most control logic are synchronized with this clock. All status registers are synchronized with PCLK. This SPI module also supports an alternative timing scheme, the operation of which depends on whether it is configured in master or slave mode, as described below:

1. Master Clock Generation

In master mode, the SPI clock output pin must remain synchronized with the SPI_REF_CLOCK domain. In addition, the frequency of SCK can be further reduced by setting the BD bits [5:3] in the *SPI_CR* register.

2. Slave Clock Generation

In slave mode, SCK must be used for sampling received data and for transmitting slave data. This clock is only valid in slave mode and is driven by an external master. Its maximum frequency must not exceed one-fourth of the SPI_REF_CLOCK. To read data from the RX FIFO and sample received data, SCK is synchronized with SPI_REF_CLOCK. And its control follows the rules below:

1. When the SPI module is disabled, SCK remains low at all times.

2. When the SPI is enabled but the slave synchronization start condition is not met, SCK is set low.

20.5. SPI FIFOs

The transmit and receive FIFOs each have a depth of 8. The status of the FIFOs can be obtained by reading the STATUS register. SPI module interrupts can be triggered based on the FIFO status. If the RX FIFO is full and a write operation is attempted, the RX FIFO overflow flag is set. This flag remains active until a read operation from the RX FIFO occurs and the corresponding bit in the STATUS register is cleared by writing 1. When the RX FIFO is full, any additional data cannot be written. To avoid data loss, read operations must be performed before the RX FIFO reaches full capacity. In slave mode, a full RX FIFO will result in incoming data being lost.

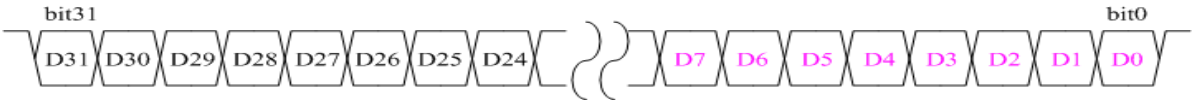
20.6. SPI data format

Each SPI data frame in the PEC930 module can be programmed to have a length of 4–32 bits. When the module is idle, the SCK clock signal remains in the inactive state (the inactive level is determined by the SPO bit). Additionally, when there is data in the receive buffer, the duration of the idle state is used to generate a non-empty timeout status for the receive queue. Each data frame supports both LSB-first and MSB-first transmission configurations.

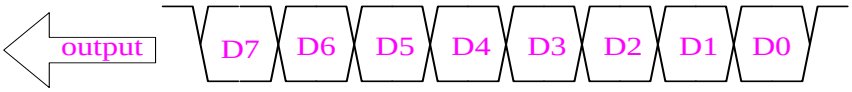
20.7. SPI data transmit

20.7.1. 8 bit data frame

TXFIFO DATA

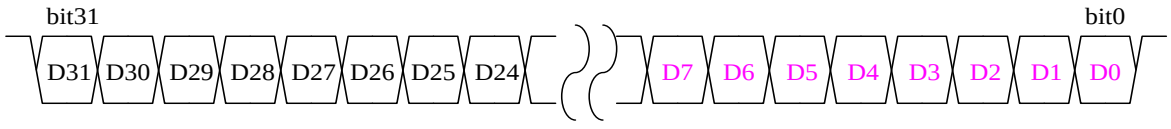


SPI transmits the MSB first



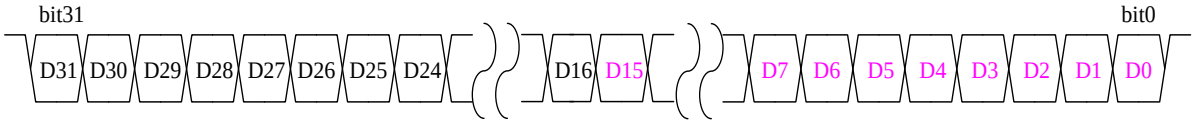
SPI transmits the LSB first

RXFIFO DATA

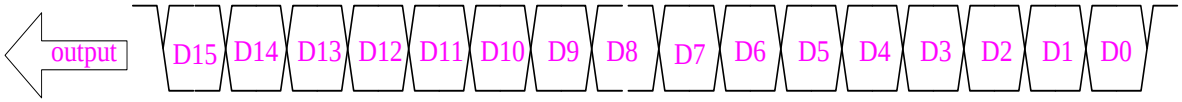


20.7.2. 16 bit data frame

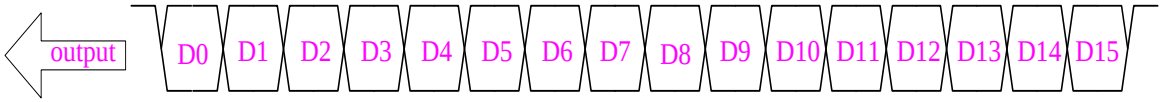
TXFIFO DATA



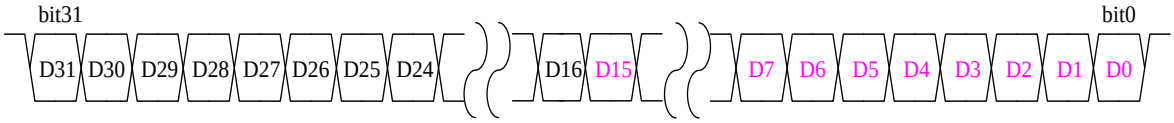
SPI transmits the MSB first



SPI transmits the LSB first

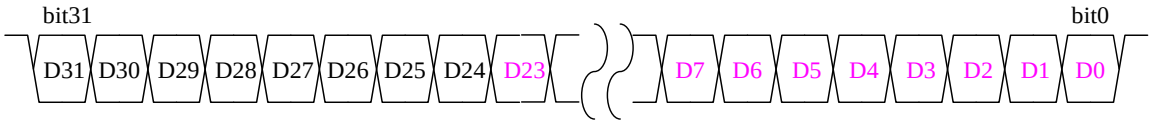


RXFIFO DATA

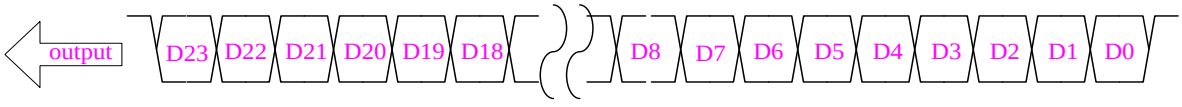


20.7.3. 24 bit data frame

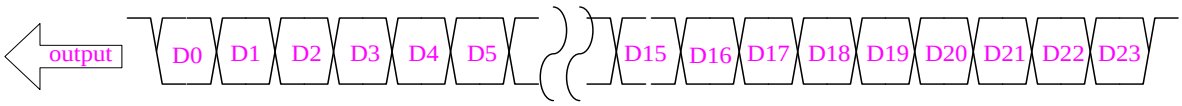
TXFIFO DATA



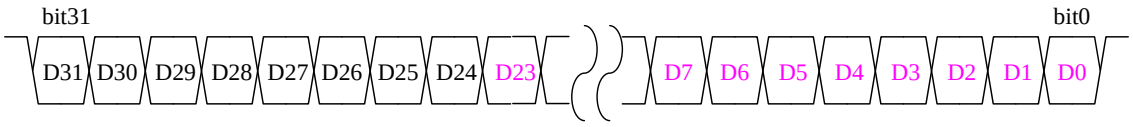
SPI transmits the MSB first



SPI transmits the LSB first

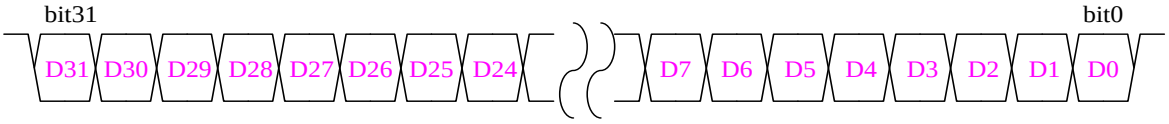


RXFIFO DATA

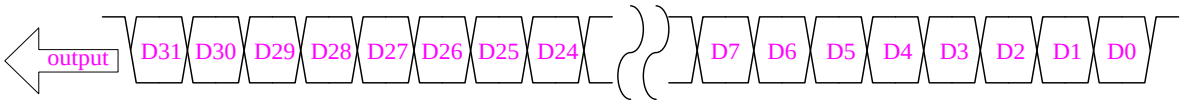


20.7.4. 32 bit data frame

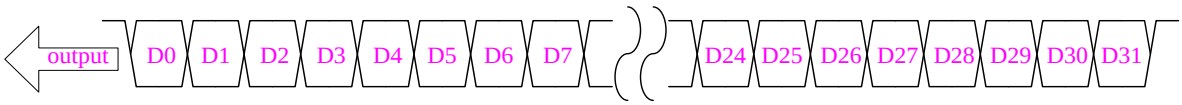
TXFIFO DATA



SPI transmits the MSB first



SPI transmits the LSB first



RXFIFO DATA

20.8. SPI interrupt

The PEC930 SPI module has four interrupt sources that can generate interrupt requests, corresponding to different states of the transmit and receive FIFOs:

- **Receive queue overflow: I**f the receive queue is already full with 8 data entries, receiving a 9th data entry causes the FIFO to overflow. The earliest received data is pushed out and discarded, and the receive overflow flag is set.
- **Receive queue non-empty timeout: I**f one or more data entries are present in the receive queue but the software does not read them within a certain period (64 system clock cycles), the non-empty timeout flag is set.
- **Receive queue full:**When the receive queue is full, the receive full flag is set.
- **Transmit queue full:**When the transmit queue is full the transmit full flag is set.

The four status flags described above are synchronously reflected in the status register SPI_SR according to the actual state of the SPI transmit and receive FIFOs. However, they do not directly generate interrupt requests. The interrupt enable control register SPI_INTEN determines which interrupt sources are enabled and can generate interrupt requests.. All enabled interrupt flags ultimately trigger a single interrupt request to the CPU. The user software must query each interrupt flag individually in the interrupt service routine to identify the specific interrupt source and handle it accordingly.

20.9. Register table

Address	Name	Description
0x4000_3800	SPI_CR	SPI control register
0x4000_3804	SPI_SR	SPI status register
0x4000_3808	SPI0_INTEN	SPI interrupt enable register
0x4000_380C	SPI0_INTDIS	SPI interrupt disable register
0x4000_3810	SPI0_INTMASK	SPI interrupt mask register
0x4000_3814	SPI0_ENABLE	SPI enable register
0x4000_3818	-	Reserved
0x4000_381C	SPI0_TX	SPI transmit data register
0x4000_3820	SPI0_RX	SPI receive data register
0x4000_3824	SPI0_IDLECNT	SPI idle count register
0x4000_3828	SPI0_TXTH	SPI transmit FIFO threshold register
0x4000_382C	SPI0_RXTH	SPI receive FIFO threshold register

Table 20-1 SPI register table

20.10. Register discription

20.10.1. SPI control register (SPI_CR)

Bit	Name	R/W	Reset	Description
31:19	-	R	0x0	Reserved
18	-	R/W	0x0	Reserved
17	MODEF	R/W	0x1	ModeFail enable 0: ModeFail disable 1: ModeFail enable
16	MSTC	R/W	0x0	Manual start command Writing “0” to this bit has no effect. Always read as 0. When MSE = 1, writing 1 to this bit initiates the data transmission function as long as the TX FIFO is not empty.
15	MSE	R/W	0x0	Manual start enable When this bit is enabled while the manual start command is not enabled, multiple-byte write operations to the FIFO will be performed to prepare for transmission. In manual mode, if the manual chip-select signal is driven low (active state), the chip-select signal will only become active when a data word is written. When this bit is set to 0, manual mode is disabled, and data transmission will start immediately after data is written into the FIFO.
14	MCS	R/W	0x0	Manual chip select When this bit is enabled, the chip-select bus is always driven by the programmed peripheral select code.

<table>
<tr><td>13:11</td><td>-</td><td>R/W</td><td>0x0</td><td>Reserved</td></tr>
<tr><td>10</td><td>CS</td><td>R/W</td><td>0x0</td><td>Chip select
0: Chip selected
1:No chip selected</td></tr>
<tr><td>9</td><td>-</td><td>R/W</td><td>0x0</td><td>Reserved</td></tr>
<tr><td>8</td><td>LSB</td><td>R/W</td><td>0x0</td><td>Bit order select
0: MSB is transmitted first
1: LSB is transmitted first</td></tr>
<tr><td>7:6</td><td>TXWDSZ</td><td>R/W</td><td>0x0</td><td>Transfer word size.
The word size must either match the FIFO width or be a value that divides the FIFO width evenly. If an evenly divisible word size is selected, multi-word transfer mode becomes available.”
00:8 bits datasize
01:16 bits datasize
10:24 bits datasize
11: 32 bits datasize</td></tr>
<tr><td>5:3</td><td>BD</td><td>R/W</td><td>0x0</td><td>Baud-rate divider.
The SPI baud rate is derived from SPI_REFERENCE_CLOCK / BD.
<table>
<tr><td>BD[2:0]</td><td>Discription</td></tr>
<tr><td>3’b000</td><td>÷ 2</td></tr>
<tr><td>3’b001</td><td>÷ 4</td></tr>
<tr><td>3’b010</td><td>÷ 8</td></tr>
<tr><td>3’b011</td><td>÷ 16</td></tr>
</table></td></tr>
</table>

				3’b100	÷ 32
				3’b101	÷ 64
				3’b110	÷128
				3’b111	÷256
2	CPHA	R/W	0x0	Clock Phase Select bit 0: The SPI clock remains active between transfer words (chip-select is not de-asserted). 1: The SPI clock remains inactive between transfer words (chip-select is de-asserted).	
1	CPOL	R/W	0x0	Clock polarity select bit 0: The clock is low in the idle state (between two transmissions 1: The clock is high in the idle state (between two transmissions)	
0	MODE	R/W	0x0	Mode select 0: Slave mode 1: Master mode	

Note: The table below describes the relationship between **Datasize** (in bits) and the FIFO word width (**FF_W**).

FF_W	Datasize	Words Transferred/FIFO Location
32	32	1
32	16	2 (First transfer = FIFO[31:16])
32	8	4 (First transfer = FIFO[31:24])

20.10.2. SPI status register (SPI_SR)

The SPI_SR register is a read-only register with a valid width of 7 bits. When the corresponding event occurs and the interrupt is enabled, the value in this register will change. Any bit set high in this register indicates that the corresponding event has triggered an interrupt signal output. By default, bits 0, 1, and 6 of this register can be cleared by writing 1.

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7	BUSY	R	0x0	SPI Busy flag 0: SPI idle 1: SPI busy
6	TXUFL	R/W1c	0x0	TX FIFO under flow flag 0:No underflow occur 1:An underflow occurs when the system attempts to transmit data while the FIFO is empty
5	RXFUL	R	0x0	RX FIFO full flag 0:FIFO not full 1:FIFO full
4	RXNEP	R	0x0	RX FIFO not empty flag 0:FIFO is below the set threshold 1:FIFO is greater than or equal to the set threshold
3	TXFUL	R	0x0	TX FIFO full flag 0: FIFO not full 1: FIFO full

2	TXNFUL	R	0x1	TX FIFO not full flag 0:FIFO is greater than or equal to the set threshold 1:FIFO is below the set threshold
1	MDF	R/W1c	0x0	Mode fail flag Indicates that the voltage on n_ss_in does not match the current SPI mode. When this bit is set to 1, it means that n_ss_in went high (inactive) during a transfer word in slave mode. Setting this status bit to 1 will disable the SPI module. (Note: Mode Fail is only supported in slave mode!) 0:No mode fail occurred 1: Mode fail occured
0	RECVOV	R/W1c	0x0	Receive overflow flag This bit is set to 1 if there is an attempt to write to the RX FIFO when it is already full. If a new write to the RX FIFO occurs while this bit is already high, the flag remains set to 1. 0:No overflow occured 1: Overflow occured

20.10.3. SPI interrupt enable register (SPI_INTEN)

Bit	Name	R/W	Reset	Description
31:7	-	R	0x0	Reserved
6:0	INTEN	WO	0x0	Enables the generation of interrupts for the corresponding events. Each bit controls the enable function for one interrupt-generating event, which corresponds one-to-one with the events in the status register. Writing “0” to this bit has no effect. 1: Enable interrupt

20.10.4. SPI interrupt disable register (SPI_INTDIS)

Bit	Name	R/W	Reset	Description
31:7	-	R	0x0	Reserved
6:0	INTEN	WO	0x0	Disables the generation of interrupts for the corresponding events. Each bit can disable one interrupt-generating event, which corresponds one-to-one with the events in the status register. Writing “0” to this bit has no effect. 1: Disable interrupt

20.10.5. SPI interrupt mask register (SPI_INTMASK)

Bit	Name	R/W	Reset	Description
31:7	-	R	0x0	Reserved
6:0	INTEN	RO	0x0	0: The interrupt generation mechanism for the event corresponding to the same bit in the status register is disable 1: The interrupt generation mechanism for the event corresponding to the same bit in the status register is enabled.

20.10.6. SPI enable register (SPI_ENABLE)

Bit	Name	R/W	Reset	Description
31:1	-	R	0x0	Reserved
0	SPI_EN	R/W	0x0	SPI enable bit When the SPI module is disabled during a transfer, the SPI module will shut down after completing the current word. In disable mode, all SPI output enables are disabled, and all SPI module pins are set to input mode. 0: SPI disable 1: SPI enable

20.10.7. SPI transmit FIFO register (SPI_TX)

Bit	Name	R/W	Reset	Description
31:0	TXFIFO	WO	0x0	Transmit data register.

20.10.8. SPI receive FIFO register (SPI_RX)

Bit	Name	R/W	Reset	Description
31:0	RXFIFO	RO	0x0	Receive data register.

20.10.9. SPI idle count register (SPI_IDLECNT)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	IDLECNT	R/W	0xFF	In slave mode, the SPI can detect the start of a word transfer only when the serial clock input (SCLK_IN) from the master remains at a stable IDLE level and the number of SPI reference clock cycles is equal to or greater than the preset SPIn_IDLECNT value.When the slave device is not selected by the chip select signal, it remains in the idle state. During this period, the counter counts the duration of the slave idle state based on the SPI reference clock.

20.10.10. SPI transmit FIFO threshold register (SPI_TXTH)

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2:0	TXTHOLD	R/W	0x1	The number of usable bits in this register matches the depth of the TX FIFO. This register defines the threshold for generating the "not full" interrupt for the TX FIFO.

20.10.11. SPI receive FIFO threshold register (SPI_RXTH)

Bit	Name	R/W	Reset	Description
31:3	-	R	0x0	Reserved
2:0	RXTHOLD	RW	0x1	The number of usable bits in this register matches the depth of the RX FIFO. This register defines the threshold for generating the "not empty" interrupt for the RX FIFO.

21. I2C

21.1. I2C overview

I2C is a serial communication bus frequently used in embedded system design. It operates over a two-wire interface consisting of SCL (serial clock) and SDA (serial data), and enables bidirectional data communication among multiple interconnected devices in a master-slave architecture. Detailed descriptions of the I2C protocol can be downloaded from NXP's official website.

The PEC930 on-chip I2C module supports both master and slave operating modes. Its key features include:

- Includes a parallel-to-serial I2C protocol converter
- Provides bus arbitration mechanism, supporting multi-master operation
- Supports 7-bit slave addressing mode
- Supports general call addressing
- Provides transmit and receive status indication
- Provides byte-transfer-complete status indication
- Supports communication error detection
- Supports Standard-mode (100 kbps) and Fast-mode (400 kbps)

As an I2C master, the module provides the following features:

- Generates the bus communication clock
- Performs bus arbitration
- 7-bit addressing of slave devices and control of data transfer direction
- Generation of START, repeated START, and STOP conditions
- Detection of communication errors

As an I2C slave, the module provides the following features:

- Detection of START, repeated START, and STOP conditions
- Supports programmable 7-bit address matching
- Monitors START and STOP conditions during transfers
- Detection of communication errors

Before the I2C module starts operation, the software must first configure the clock divider coefficients CR2/CR1/CR0 in the I2C_CTLSET register according to the system clock frequency, to generate the desired SCL communication clock. Then, the EN bit in the I2C_CTLSET register must be set to enable the I2C module. During communication, the status information is reflected in the I2C_STAT status register. If the module is used as an I2C slave, the device address must also be configured in the I2C_ADDR register.

Before enabling the I2C module, the SDA and SCL pins must be configured for their alternate I2C functions. The GPIO function of these pins must be disabled so that the pins can operate as SDA and SCL. Refer to the GPIO module documentation for details.

Once the I2C module is enabled, the SDA and SCL lines are configured as open drain. The chip supports an internal pull-up option (see the Analog chapter for related registers). Alternatively, external pull-up resistors may be used to pull both lines to a high level. The resistor value depends on the operating voltage, communication speed, and bus loading, and typically ranges from 5 kΩ to 10 kΩ**.** Before transmitting data, the sender drives SCL low**.** After software writes one byte to the I2C_DATA register, the SDA and SCL lines begin outputting data bits and clock pulses for serial transmission. When receiving data, the receiver holds SCL low until the received byte is read from the I2C_DATA register. If the I2C module is disabled, the SDA and SCL pins can be used as general-purpose I/O.”

21.2. I2C protocol

The I2C bus protocol requires that all devices connected to the bus use open-drain outputs for the SDA and SCL lines. In other words, any device may pull the SDA or SCL line low, but no device is allowed to drive the line high. The high level on the bus can only be achieved through hardware pull-up resistors.

A typical I2C bus communication consists of four parts:

- Transmission of the START condition, indicating the beginning of a communication
- Transmission of the slave address
- Data transfer
- Transmission of the STOP condition, indicating the end of the communication

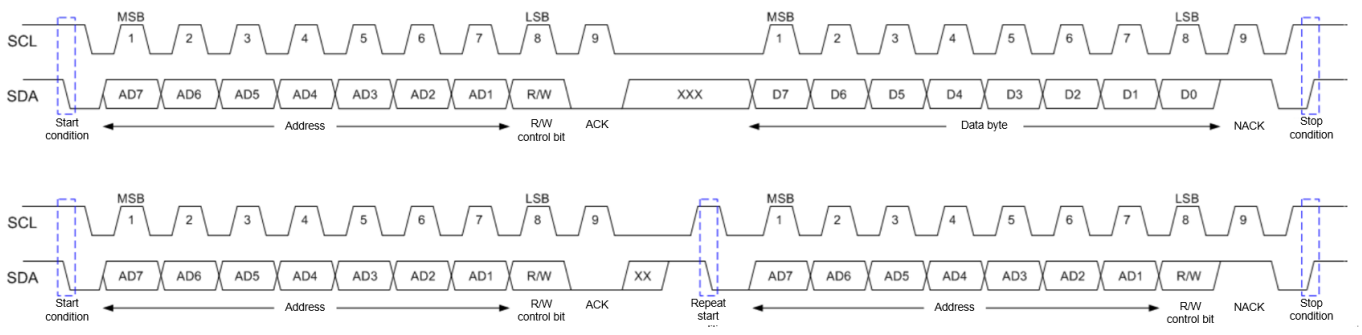


Fig. 21-1 I2C communication timing diagram

21.2.1. Start condition

When the bus is idle (both SCL and SDA lines are at logic high), the master can initiate communication by sending a START condition. The START condition is defined as a high-to-low transition on the SDA line while SCL remains high. This signal indicates the beginning of a new data transfer (which may consist of multiple bytes) and causes all slave devices to exit the idle state.

21.2.2. Slave address protocol

The first byte transmitted after the START condition is the slave address sent by the master. The upper 7 bits represent the slave address, while the least significant bit (R/W) determines the data transfer direction on the bus:

1 = read transfer, slave sends data to master

0 = write transfer, master sends data to slave

When the address sent by the master matches a slave's address, the slave pulls the SDA line low during the 9th clock cycle (see Figure 16-1) to acknowledge the transfer.
There cannot be two slaves with the same address on the I2C bus. If the I2C module is configured as a master, it should not send an address identical to its own slave address. An I2C device cannot simultaneously act as both master and slave; however, if arbitration is lost during addressing, the device will revert to slave mode, operate correctly, and can be addressed by another master.

21.2.3. Data transfer

Once the slave has been successfully addressed, data can be transmitted byte by byte in the direction specified by the R/W bit sent by the master. All transfers following the address byte are considered data transfers, even if they include the slave's subaddress. Each data byte is 8 bits long. Data can only change while SCL is low; when SCL is high, SDA must remain stable. Each clock pulse of SCL transmits one data bit, with the most significant bit (MSB) sent first, as shown in Fig. 16-1. After each data byte, there is an acknowledgment (ACK) bit—the 9th bit—sent by the slave (during a write) or the master (during a read). The ACK is indicated by pulling the SDA line low during the 9th clock pulse. Thus, transmitting a complete data byte requires 9 clock pulses. If the receiving side does not acknowledge during the 9th clock cycle, it must leave the SDA line high. The master interprets this NACK (no acknowledgment) as an unsuccessful data transfer. Conversely, if the master does not acknowledge a received data byte, the slave interprets this as the end of the data transfer and releases the SDA line. In both cases, the data transfer is terminated, and the master can perform one of the following actions:

- Send a stop condition to release the bus
- Send a repeated start condition to initiate a new transfer

21.2.4. Stop condition

The master can terminate communication by sending a stop condition to release the bus. However, the master can also issue a repeated start condition and initiate a new transfer without first sending a stop condition; this is called a repeated start. A stop condition is defined as a transition of the SDA line from low to high while SCL is at a logic high level (see Fig. 13-1). Even if a slave has sent an acknowledgment, the master can transmit a stop condition, and the slave must release the bus accordingly.

21.2.5. Repeated start condition

As shown in Fig. 13-1, a repeated start condition is a signal sent without first generating a stop condition to terminate communication. The master uses this signal to communicate with another slave, or with the same slave in a different mode (transmit/receive), without releasing the bus.

21.2.6. Bus arbitration

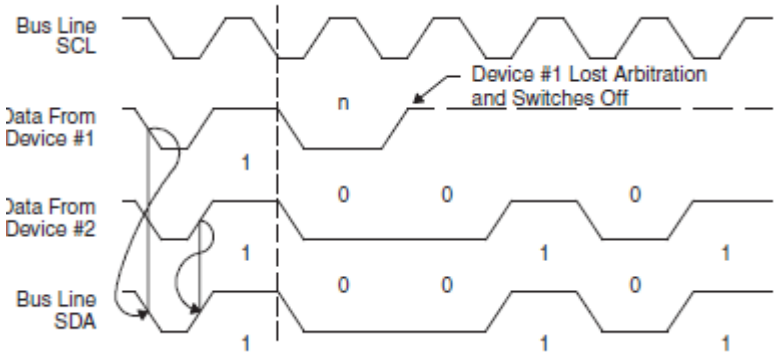


Fig. 21-2 I2C arbitration process diagram between two masters

The I2C bus is a multi-master bus, allowing more than one master to be connected. If two or more masters attempt to control the bus simultaneously, the bus clock is determined through a clock synchronization process: The low period of the bus clock equals the longest low phase among the competing clocks, and the high period equals the shortest high phase. The relative priority of competing masters is determined by data arbitration. If one master outputs a logic 1 while another outputs a logic 0, the master outputting 1 loses arbitration. The losing master immediately switches to slave (receiver) mode and stops driving the SDA line. This transition from master to slave mode does not generate a stop condition. At the same time, a status bit is set by hardware to indicate that arbitration has been lost.

21.2.7. Clock synchronization

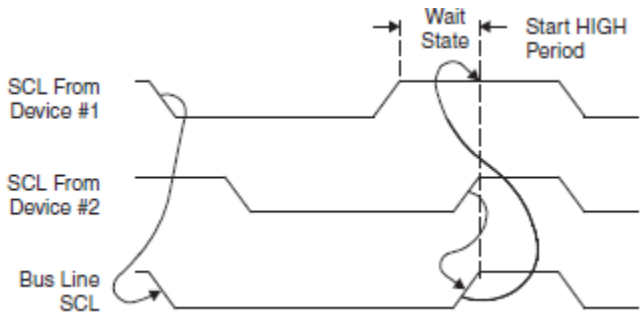


Fig. 21-3 I2C SCL waveform diagram during bus arbitration

During arbitration, the clocks from different masters need to be synchronized. Because the SCL line is a wired logic line, the voltage transitions on SCL are affected by all devices connected to the bus. As shown in Figure 16-3, during arbitration, once a master enters the low phase of its clock, it holds the SCL line low until the clock reaches high. However, if another master's clock is still low, the first master's transition from low to high does not change the SCL line state. Therefore, the low period of the synchronized SCL is determined by the master with the longest low phase. Masters with shorter low phases enter a wait state during this period (see Figure 21-3). Once all masters have completed their low phases, the synchronized SCL line is released and pulled high. At this point, the master clocks and the SCL line are aligned, and all masters start counting their high periods. The first master to complete its high period then pulls the SCL line low again. Thus, in the synchronized SCL clock, the low period is determined by the master with the longest low phase among all masters, and the high period is determined by the master with the shortest high phase.

21.2.8. Handshake

The clock synchronization mechanism can be used as a handshake in data transmission. After completing the transfer of one byte (9 bits), the slave can hold the SCL line low. In this case, the slave effectively pauses the bus clock, forcing the master's clock into a wait state until the slave releases the SCL line.

21.2.9. Clock Stretching

The clock synchronization mechanism can be used by the slave to slow down the bit rate of transmission. After the master has pulled the SCL line low, the slave can continue to hold SCL low for a certain period before releasing it. If the slave's low period is longer than the master's, the low period of the SCL bus signal will be extended.

21.3. I2C master mode

After initialization, the I2C module defaults to slave mode. When the bus is idle (I2C_STAT = 0xF8), if software sets I2C_CTLCLR_STA = 1, the module enters master mode and generates a START condition on the bus. Once the START condition is generated, I2C_STAT updates to 0x08. If interrupts are enabled, an interrupt request is triggered. At this moment, the master holds the SCL line low until software writes the first data byte into the I2C_DATA register to begin transmission.

21.3.1. I2C master mode addressing

In I2C master mode, the first byte sent after the start condition must be the slave address. The module supports 7-bit addressing.

7-bit addressing:

- The master sends the address byte (7-bit address + 1-bit R/W control).
- Wait for the byte to be transmitted or for an interrupt response.
- If the slave acknowledges, I2C_STAT = 0x18; if not acknowledged, I2C_STAT = 0x20.
  After the address is sent and the slave correctly acknowledges, the master holds the SCL line low, and then proceeds to either data transmission or data reception.

21.3.2. I2C master mode data transfer

After the master sends the slave address (with the R/W bit set to 0), it can continue writing data into the I2C_DATA register for transmission. Before the data is written into I2C_DATA, the master keeps the SCL line held low. After each byte is transmitted and acknowledged by the slave, I2C_STAT = 0x28. If the slave does not acknowledge, I2C_STAT = 0x30, and the module must send a stop condition to terminate the I2C communication. Once the last byte is sent, the master issues a stop condition, and the module immediately returns to slave mode.

21.3.3. I2C master mode data receive

After the master sends the slave address (with the R/W bit set to 1), it begins reading data from the slave. After each byte is read, the master automatically generates an ACK or NACK depending on the ACK bit setting (I2C_CTLSET_AA = 1 for ACK, I2C_CTLSET_AA = 0 for NACK). Software can check the I2C_STAT status or wait for an interrupt response. During this process, the master keeps the SCL line held low until the received data is read from the I2C_DATA register. Before reading the last byte of the transfer, the master should clear the ACK bit (I2C_CTLSET_AA = 0) to ensure that a NACK is sent after reading the final byte. Once the last byte is read, the master sends a stop condition, and the module immediately returns to slave mode.

21.3.4. I2C master mode error detection

When the I2C module is operating in master mode, if a bus communication error is detected, the I2C_STAT status register is set to 0x00, and the module releases control of the bus.

21.4. I2C slave mode

After initialization, the I2C module defaults to slave mode, waiting for a start condition on the bus. It then receives the address sent by the master and compares it with its own configured address for a match.

21.4.1. I2C slave mode addressing

I2C slave mode addressing uses a 7-bit address:

- The slave compares the high 7 bits of the received address byte with the high 7 bits of its own address set in the I2C_ARD0 register. The lowest bit of the received byte is the read/write control bit and is not used for address matching.
- If the 7-bit address matches, the slave responds with an ACK bit; otherwise, it sends a NACK bit and waits for the next start condition.
- After a successful address match, the slave holds the SCL line low until the software reads the I2C_SR0 register, after which SCL is released.

21.4.2. I2C slave mode data receive

After the address matching process (with the read / write control bit = 0), the I2C slave enters data reception mode. For each received byte the slave sends back an ACK/NACK depending on the ACK bit setting. If interrupts are enabled, an interrupt request is generated. The slave holds the SCL line low until the software reads the received data from the I2C_DR register.

21.4.3. I2C slave mode data transfer

After the address matching process (with the read / write control bit set to 1), the slave enters the data transmit state and holds the SCL line low. Once the slave writes one byte into the I2C_DATA register, the SCL line is released and data transmission begins. After receiving an acknowledge from the counterpart, the I2C_STAT status is updated, and an interrupt request is generated if interrupts are enabled.

21.4.4. I2C slave mode STOP signal

After the transmission of the final byte, if the master issues a STOP condition, the slave immediately terminates the current communication session and generates an interrupt request if interrupts are enabled.

21.4.5. I2C slave mode error message

In slave mode, if the I2C module detects a bus communication error, the I2C_STAT status register will be cleared to 0x00, and the module will release control of the bus.

21.5. I2C clock configuration

The I2C communication clock speed depends on the system clock frequency and the clock divider.

The divider is selected by the CR2/CR1/CR0 bits in the **I2C_CTLSET** register.

The clock frequency is calculated as follows:

fSCL = fSYS / DIV

Where:

- fSCL is the desired SCL clock frequency
- fSYS is the system clock frequency
- DIV is the system clock division factor, selected according to Table 22-1

With the system clock frequency known, the application software selects an appropriate divider to configure the I2C communication clock frequency.

The maximum I2C clock frequency must not exceed 1Mbps.

CR2	CR1	CR0	DIV
0	0	0	256
0	0	1	244
0	1	0	192
0	1	1	160
1	0	0	960
1	0	1	120
1	1	0	60
1	1	1	Reserved

Table 21-1 System Clock Prescaler for Generating the I2C Clock Frequency

21.6. I2C Status Information and Interrupt Response

The I2C module has multiple events that can generate the same interrupt flag (I2C_CTLSET_SI).

These events are indicated by the status information in bits [7:3] of the I2C_STAT, as described in Table 21-2.

I2C_STAT[7:3]	I2C bus status	I2C_CTLSET				Next Action of the I2C Module
		STA	STO	SI	AA	
00000	I2C errors in master mode or when addressed as a slave	0	1	0	X	I2C bus released
00001	Master mode: START condition transmitted	X	0	0	X	Send slave address + write control bit, receive ACK
00010	Master mode: Repeated START condition transmitted	X	0	0	X	Same as sending START condition (00001) Send slave address + read control bit, master enters receive mode
00011	Master mode: Slave address + Write bit transmitted, ACK received	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Send data byte and receive ACK Send repeated START Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared
00100	Master mode: Slave address + Write bit transmitted, ACK not received	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Send data byte and receive ACK Send repeated START Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared
00101	Master mode: Data byte transmitted, ACK received	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Send data byte and receive ACK Send repeated START Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared
00110	Master mode: Data byte transmitted, ACK not received	0 1 0 1	0 0 1 1	0 0 0 0	X X X X	Send data byte and receive ACK Send repeated START Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared

00111	Bus arbitration lost in master mode while sending address or data	0 1	0 0	0 0	X X	I2C bus released Send START condition when I2C bus is idle
01000	Master mode: sent slave address + read control bit, received ACK	0 0	0 0	0 0	0 1	Receive data, no ACK Receive data, ACK sent
01001	Sent slave address + read control bit, no ACK received	1 0 1	0 1 1	0 0 0	X X X	Send Repeated START condition Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared
01010	Data byte reception completed, ACK sent	0 0	0 0	0 0	0 1	Receive data, no ACK Receive data, ACK sent
01011	Data byte reception completed, NACK sent	1 0 1	0 1 1	0 0 0	X X X	Send Repeated START condition Send STOP condition, STO flag set After STOP, immediately send START, STO flag cleared
01100	Slave mode: received address + write control bit, ACK sent	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK
01101	Master mode bus arbitration lost, received address + write control bit in slave mode, ACK sent	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK
01110	Received broadcast address (0x00), ACK sent	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK
01111	Master mode bus arbitration lost, received broadcast address in slave mode, ACK sent	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK

10000	Slave mode addressed, received a data byte, ACK sent	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK
10001	Slave mode addressed, received a data byte, NACK sent	0 0 1 1	0 0 0 0	0 0 0 0	0 1 0 1	Module switches to non-addressed slave mode, no longer recognizes address or broadcast addressing Module switches to non-addressed slave mode, can recognize address and broadcast addressing Module switches to non-addressed slave mode, no longer recognizes address or broadcast addressing; will send START condition when bus is idle Module switches to non-addressed slave mode, can recognize address and broadcast addressing; will send START condition when bus is idle
10010	Receive a data byte under broadcast addressing and respond with ACK	X X	0 0	0 0	0 1	Receive data, respond with NACK Receive data, respond with ACK
10011	Receive a data byte under broadcast addressing and respond with NACK	0 0 1 1	0 0 0 0	0 0 0 0	0 1 0 1	Module switches to non-addressed slave mode, no longer recognizes address or broadcast addressing Module switches to non-addressed slave mode, can recognize address and broadcast addressing Module switches to non-addressed slave mode, no longer recognizes address or broadcast addressing; will send START condition when bus is idle Module switches to non-addressed slave mode, can recognize address and broadcast addressing; will send START condition when bus is idle
10100	In slave mode, receive	0	0	0	0	Module switches to non-addressed slave

	a STOP condition or a Repeated START condition	0	0	0	1	mode, no longer recognizes address or broadcast addressing
		1	0	0	0	Module switches to non-addressed slave mode, can recognize address and broadcast addressing
		1	0	0	1	Module switches to non-addressed slave mode, no longer recognizes address or broadcast addressing; will send START condition when bus is idle
						Module switches to non-addressed slave mode, can recognize address and broadcast addressing; will send START condition when bus is idle
10101	In slave mode, reception of address + read control bit completed, respond with ACK	X	0	0	0	Send last byte, receive ACK response
		X	0	0	1	Send a byte, receive ACK response
10110	In master mode, bus arbitration lost; as slave, received address + read control bit, respond with ACK	X	0	0	0	Transmit the last byte, receive ACK
		X	0	0	1	Transmit a byte, receive ACK
10111	In slave mode, transmission of a data byte completed, received ACK	X	0	0	0	Transmit the last byte, receive ACK
		X	0	0	1	Transmit a byte, receive ACK
11000	In slave mode, transmission of a data byte completed, no ACK received	0	0	0	0	Module switches to non-addressed slave mode, no longer recognizes address or broadcast address
		0	0	0	1	Module switches to non-addressed slave mode, can recognize address and broadcast address
		1	0	0	0	Module switches to non-addressed slave mode, no longer recognizes address or broadcast address; when the bus is idle,
		1	0	0	1	

						a START condition will be sent Module switches to non-addressed slave mode, can recognize address and broadcast address; when the bus is idle, a START condition will be sent
11001	In slave mode, transmission of the last data byte completed, received ACK	0 0 1 1	0 0 0 0	0 0 0 0	0 1 0 1	Module switches to non-addressed slave mode, no longer recognizes address or broadcast address Module switches to non-addressed slave mode, can recognize address and broadcast address Module switches to non-addressed slave mode, no longer recognizes address or broadcast address; when the bus is idle, a START condition will be sent Module switches to non-addressed slave mode, can recognize address and broadcast address; when the bus is idle, a START condition will be sent
11111	Bus idle	0	0	0	0	-

Table 21-2 I2C communication status information

21.7. Register table

Address	Name	Description
0x4000_3000	I2C_CTLSET	I2C control set register
0x4000_3004	I2C_STAT	I2C status register
0x4000_3008	I2C_DATA	I2C data register
0x4000_300C	I2C_ADDR	I2C address register
0x4000_3018	I2C_CTLCLR	I2C control clear register

Table 21-3 I2C register table

21.8. Register description

21.8.1. I2C control set register (I2C_CTLSET)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7	CR2	R/W	0x0	Writing “0” to this bit has no effect. Set together with CR1/CR0 to configure the I2C master mode clock frequency, as described in Table 22-1.
6	EN	R/W	0x0	I2C enable bit Writing “0” to this bit has no effect. 0: I2C disable 1: I2C enable
5	STA	R/W	0x0	I2C start bit 0: STA set 0 1: STA set 1
4	STO	R/W	0x0	I2C stop bit 0: STO set 0 1: STO set 1
3	SI	R/W	0x0	I2C interrupt flag Writing “0” to this bit has no effect. 0:No interrupt occur 1: Interrupt occur
2	AA	R/W	0x0	Ack control bit Writing “0” to this bit has no effect. 0:Ack not send 1: Send an ACK after receiving an address or data byte Write operation
1	CR1	R/W	0x0	Writing “0” to this bit has no effect. Set together with CR2/CR0 to configure the I2C master mode clock frequency, as described in Table 22-1.
0	CR0	R/W	0x0	Writing “0” to this bit has no effect. Set together with CR2/CR1 to configure the I2C master mode clock frequency, as described in Table 22-1.

21.8.2. I2C status register (I2C_STAT)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:3	STAT	R	0x1F	I2C bus status There are a total of 27 possible states, 26 of which can generate an interrupt request, as described in Table 22-2.
2:0	-	R	0x0	Reserved

21.8.3. I2C data register (I2C_DATA)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7:0	DATA	R/W	0x0	Stores the data to be sent or received on the bus. Transmission**:**After the software writes 8-bit data, the sending process is immediately initiated. Reception: The first automatically received byte is the lower byte of the address. The software must read the received data from the I2C_DR register one by one to sequentially receive the subsequent data bytes.

21.8.4. I2C address register (I2C_ADDR)

Bit	Name	R/W	Reset	Description
31:24	filterdelay_enable	R	0x0	Filter delay enable bit Bit [19:16] is only valid after writing 8'hAC to filterdelay_enable
23:20	-	R	0x0	Reserved
19:16	filterdelay	R/W	0x0	Number of clock cycles for input filter delay
15:8	-	R	0x0	Reserved
7:1	ADDR	R/W	0x0	Slave address
0	GC	RW	0x0	Broadcast addressing enable bit 0: Disable broadcast addressing 1: Enable broadcast addressing

21.8.5. I2C control clear register (I2C_CTLCLR)

Bit	Name	R/W	Reset	Description
31:8	-	R	0x0	Reserved
7	CR2_CLR	R/W	0x0	Writing "0" to this bit has no effect. 1: CR2 clear
6	EN_CLR	R/W	0x0	Writing "0" to this bit has no effect. 1: EN clear
5	STA_CLR	R/W	0x0	Writing "0" to this bit has no effect. 1: STA clear
4	STO_CLR	RW	0x0	Writing "0" to this bit has no effect. 1: STO clear
3	SI_CLR	RW	0x0	Writing "0" to this bit has no effect. 1: SI clear
2	AA_CLR	RW	0x0	Writing "0" to this bit has no effect. 1:AA clear
1	CR1_CLR	RW	0x0	Writing "0" to this bit has no effect. 1: CR1 clear
0	CR0_CLR	RW	0x0	Writing "0" to this bit has no effect. 1: CR0 clear