LogicGreen Technologies Co., LTD

Feature overview

- High-performance and low-power 8-bit LGT8XM core
- Advanced RISC architecture
  - 131 instructions, more than 80% of which are executed in a single cycle
  - 32x8 general-purpose work register
  - Execution efficiency of up to 32MIPS can be achieved at 32MHz
  - Internal single-cycle multiplier (8x8)
- **Nonvolatile programs and data storage space**
  - 32Kbytes on-chip programmable FLASH program memory
  - 2Kbytes internal data SRAM
  - Programmable E2PROM analog interface with byte access
  - Brand new program encryption algorithm ensures the security of user code
- **Peripheral controller**
  - Two 8-bit timers with independent prescalers supporting comparison output modes
  - Two 16-bit timers with independent prescalers support input capture and comparison outputs
  - Internal 32KHz calibrated RC oscillator for real-time counter functionality
  - Supports up to nine PWM outputs and three complementary programmable dead zone controls
  - 12-channel 12-bit high-speed analog-to-digital converter (ADC)
    - Selectable internal and external reference voltages
    - Programmable gain (X1/8/16/32) differential amplified input channels
    - Automatic threshold voltage monitoring mode
  - Two analog comparators (ACs) that support expansion from the ADC input channels
  - Internal 1.024V/2.048V/4.096V ±1% calibrated reference
  - An 8-bit programmable DAC can be used to generate a voltage reference
  - Programmable Watchdog Timer (WDT)
  - Programmable synchronous/asynchronous serial interface (USART/SPI)
  - Synchronous peripheral interface (SPI), programmable master/slave mode of operation
  - Two-wire serial interface (TWI), compatible with I2C master-slave mode
  - 16-bit Digital Computing Acceleration Unit (DSC) for direct 16-bit data access
- **Special processor features**
  - SWD two-wire on-chip debug/mass production interface.
  - External interrupt source and I/O level change interrupt support.
  - Built-in power-on reset circuit (POR) and programmable low voltage detection circuit (LVD).
  - Built-in 1% calibrated 32MHz RC oscillator with frequency multiplication output.
  - Built-in 1% calibrated 32KHz RC oscillator.
  - External support 32.768KHz and 400K~32MHz crystal input.
  - 6x high-current push-pull drive IO for high-speed PWM applications.

8-bit LGT8XM

RISC Microcontroller with In-System Programmable FLASH Memory

LGT8F88P
LGT8F168P
LGT8F328P

Data book

Version 1.0.4

English translation by Frankie
Tools: Google & Bing

Application areas:

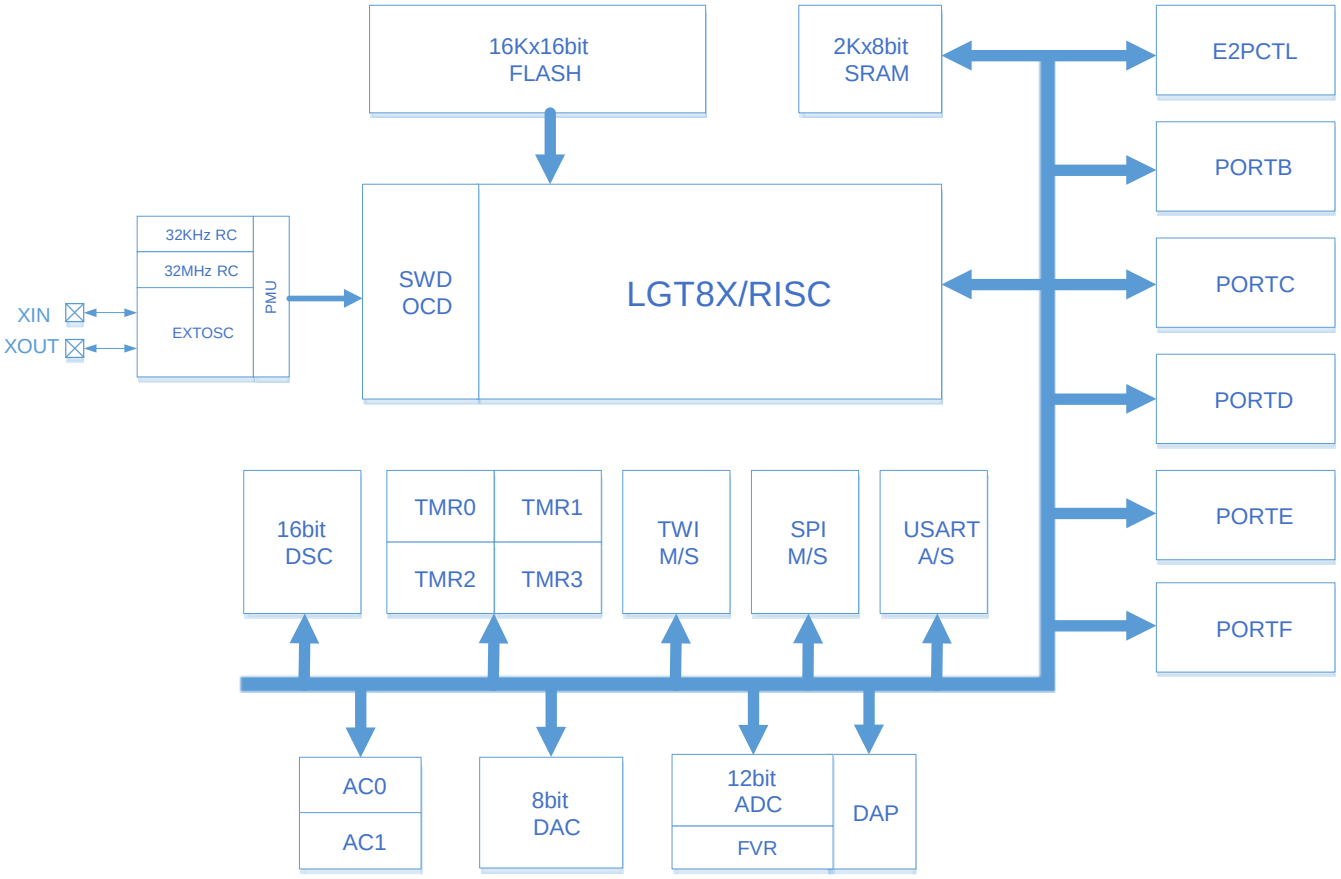
household appliance

Motor drivers

Automation controls

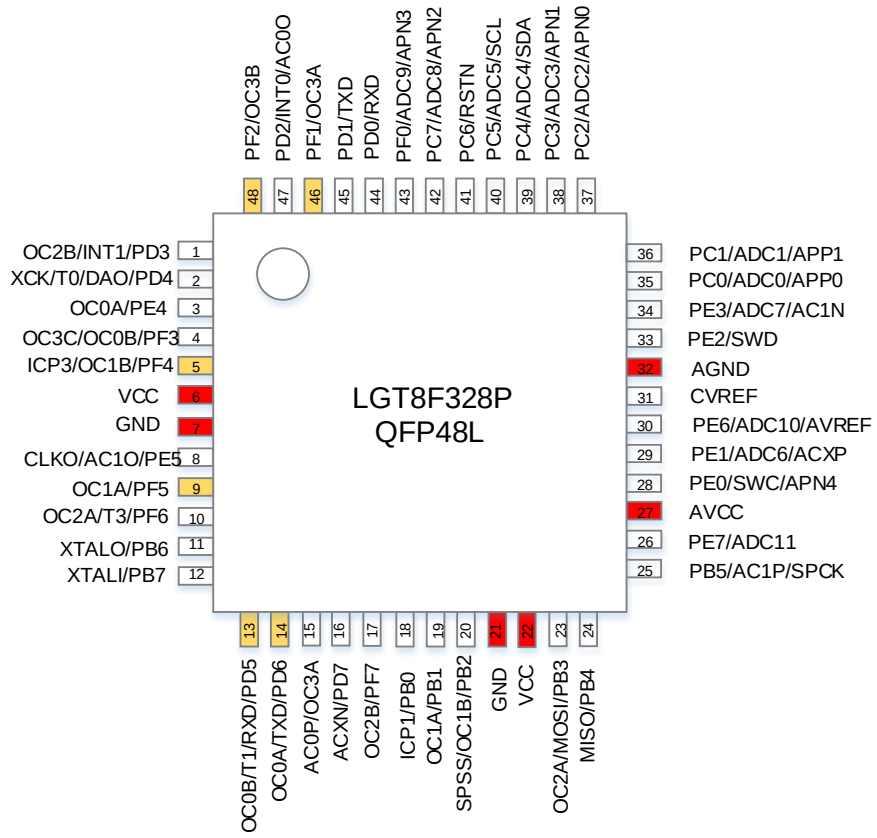
- I/O and package: QFP48/32L, SSOP20L
- Minimum power consumption: 1uA@3.3V
- Working environment:
  - Operating voltage: 1.8V ~ 5.5V
  - Operating frequency: 0 ~ 32MHz
  - Operating temperature: -40C ~ +85C
  - HBM ESD : > 4KV

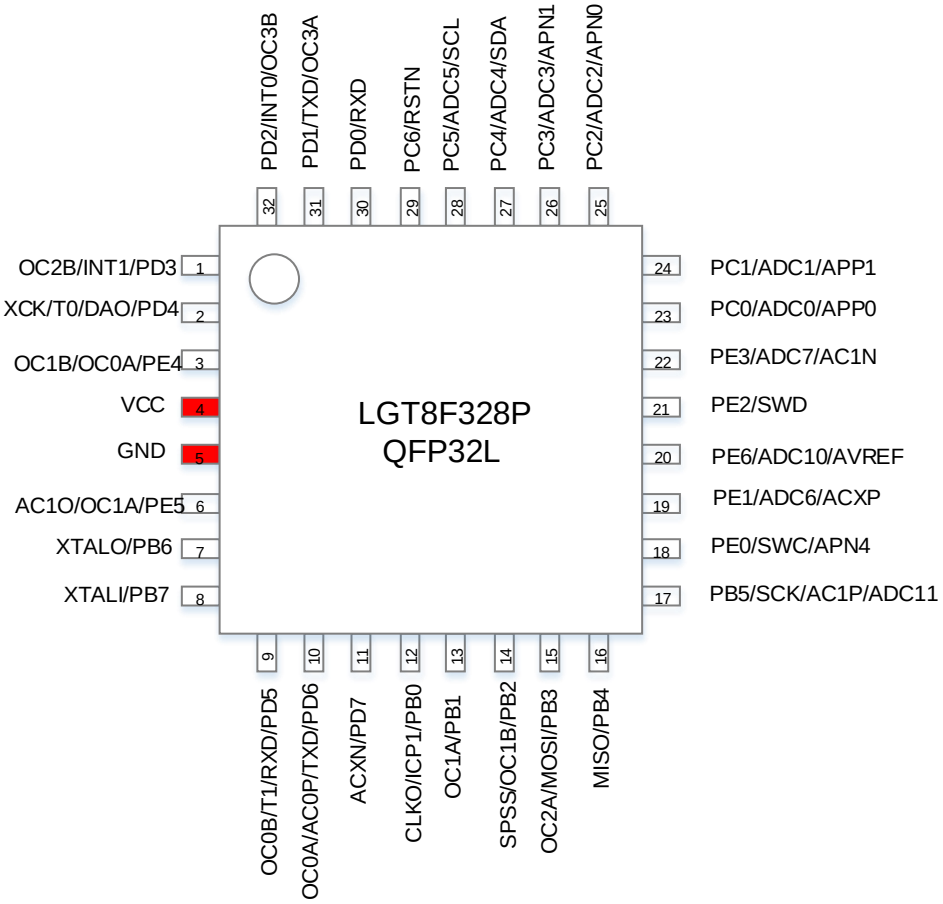
System framework

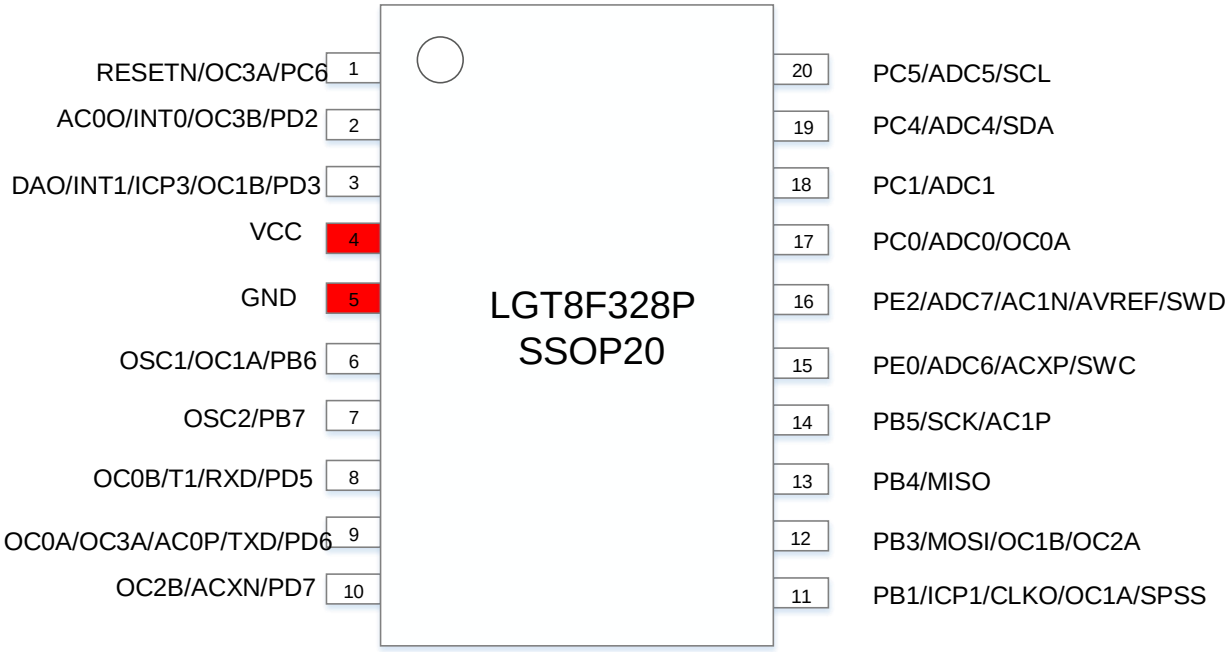


Module name	Module functionality
SWD	Debug module, realize online debugging and ISP function at the same time
LGT8X	8-bit high-performance RISC core
E2PCTL	Flash memory controller
PMU	Power management module (managing transition between system working states)
PORTB/C/D/E/F	Universal programmable I/O ports
DSC	16-bit digital arithmetic acceleration unit
ADC DAP IVREF	8-channel 12-bit analog-to-digital converter Programmable gain differential amplifier 1.024V/2.048V/4.096V internal reference
AC0/1	Analog comparator
TMR0/1/2/3	8/16-bit timer/counter, PWM controller
WDT	Watchdog reset module
SPI M/S	Master-slave SPI controller
TWI M/S	Master-slave two-wire interface controller, compatible with I2C protocol
USART	Synchronous/asynchronous serial transceiver
DAC	8-bit digital-to-analog converter

Package definition






LGT8F328P
SSOP20
1 RESETN/OC3A/PC6
2 AC0O/INT0/OC3B/PD2
3 DAO/INT1/ICP3/OC1B/PD3
4 VCC
5 GND
6 OSC1/OC1A/PB6
7 OSC2/PB7
8 OC0B/T1/RXD/PD5
9 OC0A/OC3A/AC0P/TXD/PD6
10 OC2B/ACXN/PD7
20 PC5/ADC5/SCL
19 PC4/ADC4/SDA
18 PC1/ADC1
17 PC0/ADC0/OC0A
16 PE2/ADC7/AC1N/AVREF/SWD
15 PE0/ADC6/ACXP/SWC
14 PB5/SCK/AC1P
13 PB4/MISO
12 PB3/MOSI/OC1B/OC2A
11 PB1/ICP1/CLKO/OC1A/SPSS

Pin description

In the LGT8FX8P series package, the QFP48L package leads all pins. The other packages are based on the QFP48 by binding multiple internal I/Os to a single pin. Special attention is required when configuring the pin orientation. The following table lists the bindings for the various package pins:

QFP48	QFP32	SSOP20	Function description
01	01	03	PD3/INT1/OC2B*
			PD3: Programmable port D3 INT1: External interrupt input 1 OC2B: Timer 2 compares matching output B
02	02		PD4/DAO/T0/XCK
			PD4: Programmable port D4 DAO: Internal DAC output T0: Timer0 external clock input XCK: USART synchronous transmission clock
03	03	-	PE4/0C0A*
			PE4: Programmable port E4 OC0A: Timer 0 compares matching output A
04	-	-	PF3/OC3C/OC0B*
			PF3: Programmable port F3 OC3C: Timer 3 compares to match output C OC0B: Timer 0 compares match output B
05	03	03	PF4/OC1B*/ICP3
			PF4: Programmable port F4 OC1B: Timer 1 compares matching output B ICP3: Timer 3 captures the input
06	04	04	VCC
07	05	05	GND
08	06	-	PE5/AC1O/CLKO*
			PE5: Programmable port E5 C1O: Analog comparator AC1 output CLKO: System clock output
09		06	PF5/OC1A*
			PF5: Programmable port F5 OC1A: Timer 1 compares matching output A
10	-	-	PF6/T3/OC2A*
			PF6: Programmable port F6 T3: Timer 3 external clock input OC2A: Timer 2 compares matching output A
11	07	06	PB6/XTALO
			PB6: Programmable port B6 XTALO: Crystal IO output port

12	08	07	PB7/XTALI PB7: Programmable Port B7' XTALI: Crystal IO input port
13	09	08	PD5/RXD*/T1/OC0B PD5: Programmable port D5 RXD: USART data reception (optional) T1: Timer 1 external clock input OC0B: Timer 0 compares match output B
14	10	09	PD6/TXD*/OC0A PD6: Programmable port D6 TXD: USART data sending (optional) OC0A: Timer 0 compares matching output A
15			AC0P/0C3A AC0P: Analog comparator 0 positive input OC3A: Timer 3 compares matching output A
16	11	10	PD7/ACXN PD7: Programmable port D7 ACXN: Analog Comparator 0/1 Common Negative Input
17	-		PF7/OC2B PF7: Programmable port F7 OC2B: Timer 2 compares matching output B
18	12	11	PB0/ICP1 PB0: Programmable port B0 ICP1: Timer 1 captures the input
19	13		PB1/OC1A PB1: Programmable port B1 OC1A: Timer 1 compares matching output A
20	14	12	PB2/OC1B/SPSS PB2: Programmable port B2 OC1B: Timer 1 compares matching output B SPSS: SPI slave mode chip selection
21	-	-	GND
22	-	-	VCC
23	15	12	PB3/MOSI/OC2A PB3: Programmable port B3 MOSI: SPI master output/slave input OC2A: Timer 2 compares matching output A
24	16	13	PB4/MISO PB4: Programmable port B4 MISO: SPI master input/slave output
25	17	14	PB5/SPCK/AC1P PB5: Programmable port B5 SPCK: SPI clock signal AC1P: Analog comparator 1 positive input

26	-	-	PE7/ADC11 PE7: Programmable port E7 ADC11: ADC analog input channel 11
27	-	-	AVCC: Internal analog circuit power supply
28	18	15	PE0/SWC/APN4 PE0: Programmable port E0 SWC: SWD debug interface clock APN4: Differential amplifier reverse input channel 4
29	19		PE1/ADC6/ACXP PE1: Programmable port E1 ADC6: ADC analog input channel 6 ACXP: Analog Comparator 0/1 Common Positive Input
30	20	16	PE6/ADC10/AVREF PE6: Programmable port E6 ADC10: ADC analog input channel 10 AVREF: ADC external reference input
31	-	-	CVREF: ADC reference voltage output Only used for external 0.1uF filter capacitors
32	-	-	AGND: Internal analog circuit ground
33	21	16	PE2/SWD PE2: Programmable port E2 SWD: SWD debug interface data cable
34	22		PE3/ADC7/AC1N PE3: Programmable port E3 ADC7: ADC analog input channel 7 AC1N: Analog comparator negative input
35	23	17	PC0/ADC0/APP0 PC0: Programmable port C0 ADC0: ADC analog input channel 0 APP0: Differential amplifier forward input channel 0
36	24	18	PC1/ADC1/APP1 PC1: Programmable port C1 ADC1: ADC analog input channel 1 APP1: Differential amplifier forward input channel 1
37	25	-	PC2/ADC2/APN0 PC2: Programmable port C2 ADC2: ADC analog input channel 2 APN0: Differential amplifier reverse input channel 0
38	26	-	PC3/ADC3/APN1 PC3: Programmable port C3 ADC3: ADC analog input channel 3 APN1: Differential amplifier reverse input channel 1

39	27	19	PC4/ADC4/SDA PC4: Programmable port C4 ADC4: ADC analog input channel 4 SDA: I2C controller data cable
40	28	20	PC5/ADC5/SCL PC5: Programmable port C5 ADC5: ADC analog input channel 5 SCL: I2C controller clock line
41	29	1	PC6/RESETN PC6: Programmable port C6 RESETN: External reset input
42	-	-	PC7/ADC8/APN2 PC7: Programmable port C7 ADC8: ADC analog input channel 8 APN2: Differential amplifier reverse input channel 2
43	-	-	PF0/ADC9/APN3 PF0: Programmable port F0 ADC9: ADC analog input channel 9 APN3: Differential amplifier reverse input channel 3
44	30	-	PD0/RXD PD0: Programmable port D0 RXD: USART data receives input
45	31	-	PD1/TXD PD1: Programmable port D1 TXD: USART data transmission output
46		1	PF1/OC3A PF1: Programmable port F1 OC3A: Timer 3 compares matching output A
47	32	2	PD2/INT0/AC0O PD2: Programmable port D2 INT0: External interrupt input 0 AC0O: Analog comparison 0 output
48			PF2/OC3B PF2: Programmable port F2 OC3B: Timer 3 compares matching output B

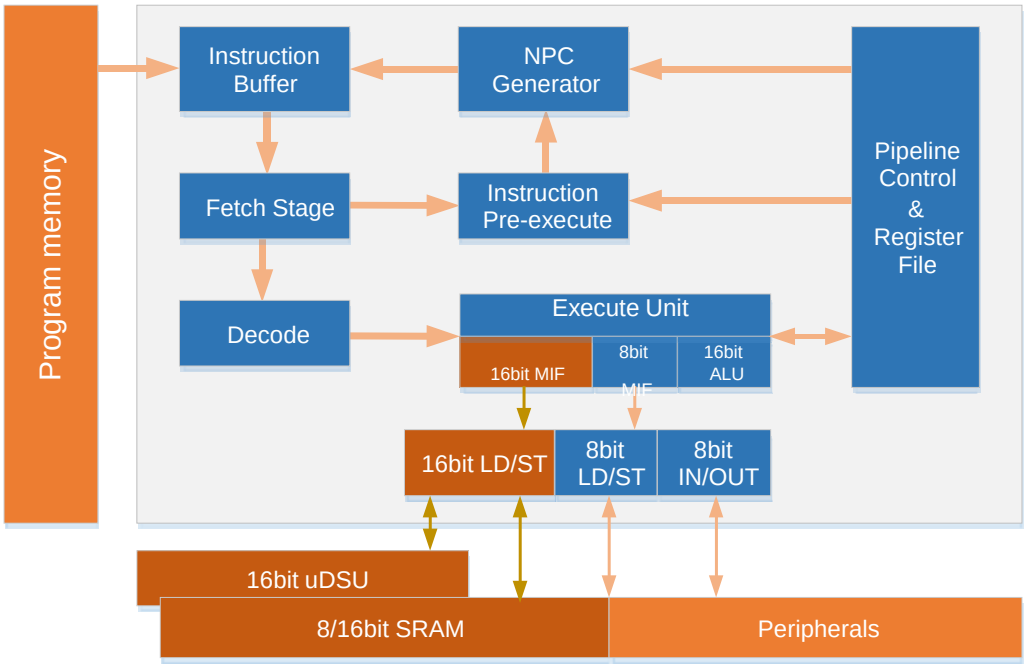
LGT8XM core

- Low power consumption design
- High efficiency RISC architecture
- 16-bit LD/ST extension (uDSU-specific)
- 130 instructions, of which more than 80% are single-cycle
- Built-in over-the-air debugging (OCD) support

Overview

This section describes the LGT8XM core architecture and features. The core is the brain of the MCU and is responsible for ensuring the correct execution of the program, so the core must be able to accurately perform calculations, control peripherals, and handle various interrupts.

Structure of LGT8XM core



To achieve greater efficiency and parallelism, the LGT8XM core uses the Haval architecture – independent data and program buses.

Instructions are executed through an optimized two-stage pipeline, which reduces the number of invalid instructions in the pipeline and reduces access to FLASH program memory, thus reducing the power consumption of core operation. At the same time, the LGT8XM core adds an instruction cache (2 instructions can be cached at the same time) in the pre-execution module of the instruction retrieval cycle, which further reduces the access frequency of the FLASH program memory through the pre-execution module of the instruction retrieval cycle; After extensive testing, LGT8XM can reduce FLASH access by about 50% compared to cores of similar architectures, greatly reducing the operating power consumption of the system.

The LGT8XM core has 32 8-bit high-speed access general-purpose register files that facilitate single-cycle arithmetic logic operations (ALUs). In general, the two operands of the ALU operation come from the common operating register, and the result of the ALU operation is also written to the register file in one cycle.

6 of the 32 pass-through registers are used to form three 16-bit registers that can be used to indirectly address address pointers for accessing external memory as well as the FLASH program space. LGT8XM supports single-cycle 16-bit arithmetic operations, which greatly improves the efficiency of indirect addressing. These three special 16-bit registers in the LGT8XM core are named the X, Y, Z registers and will be described in more detail later.

The ALU supports arithmetic logic operations between registers and between constants and registers, and operations on individual registers can also be performed in the ALU. After the ALU operation is completed, the effect of the operation result on the core state is updated to the status register (SREG).
Program flow control is implemented through conditional and unconditional jumps/calls and can be addressed to all program areas. Most LGT8XM instructions are 16-bit. Each program address space corresponds to a 16-bit or 32-bit LGT8XM instruction.

After the core responds to an interrupt or subroutine call, the return address (PC) is stored on the stack. The stack is allocated in the system's general data SRAM, so the size of the stack is limited only by the size and usage of SRAM in the system. All applications that support interrupt or subroutine calls must first initialize the stack pointer register (SP), which can be accessed through IO space. Data SRAM can be accessed through 5 different addressing modes. The internal memory of the LGT8XM is linearly mapped to a uniform address space. For details, please refer to the Storage section.

The LGT8XM core includes a flexible interrupt controller that can be controlled via a global interrupt enable bit in the status register. All interrupts have an independent interrupt vector. The priority of the interrupt corresponds to the interrupt vector address, and the smaller the interrupt address, the higher the priority of the interrupt.
The I/O space contains 64 register spaces that can be directly addressed by IN/OUT instructions. These registers actually control the core as well as status registers, SPI, and other I/O peripherals. This part of the space can be accessed directly through IN/OUT instructions or through the address they map to the data memory space (0x20 – 0x5F). In addition, the LGT8FX8P also contains extended I/O space, which is mapped to the data storage space 0x60 – 0xFF, which can only be accessed using ST/STS/STD and LD/LDS/LDD instructions.
In order to enhance the computing power of the LGT8XM core, 16-bit LD/ST extensions have been added to the instruction popular line. This 16-bit LD/ST extension works with a 16-digit arithmetic acceleration unit (uDSU) for efficient 16-bit data operations. At the same time, the core also increases the ability to access 16 bits of RAM space. Therefore, 16-bit LD/ST extensions can pass 16-bit data between uDSU, RAM, and operating registers. For details, please refer to the "Number Computing Accelerator" chapter.

Arithmetic Logic Unit (ALU)

LGT8XM contains a 16-bit arithmetic logic operation unit, which can complete 16 arithmetic operations for data in one cycle. The efficient ALU is connected to 32 general-purpose operating registers. It can complete two registers or arithmetic logic operations between registers and immediate numbers in one cycle. There are three types of operations for ALU: arithmetic, logic, and bitwise operations.
At the same time, the ALU section also contains a single-cycle hardware multiplier, which can implement two 8-bit registers directly signed or unsigned operations in one cycle. Please refer to the instruction set section for details.

Status Register (SREG)

The status register mainly stores the result information generated by the execution of the most recent ALU operation. This information is used to control the process of program execution. The status registers are updated after the ALU operation is completely completed, which eliminates the need for separate comparison instructions and enables a more compact and efficient code implementation. The value of the status register is not automatically saved and restored in response to and from interrupts, which requires software to implement.

SREG register definition

SREG system status registers								
address: 0x3F (0x5F)				Default value: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	I	T	H	S	V	N	Z	C
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

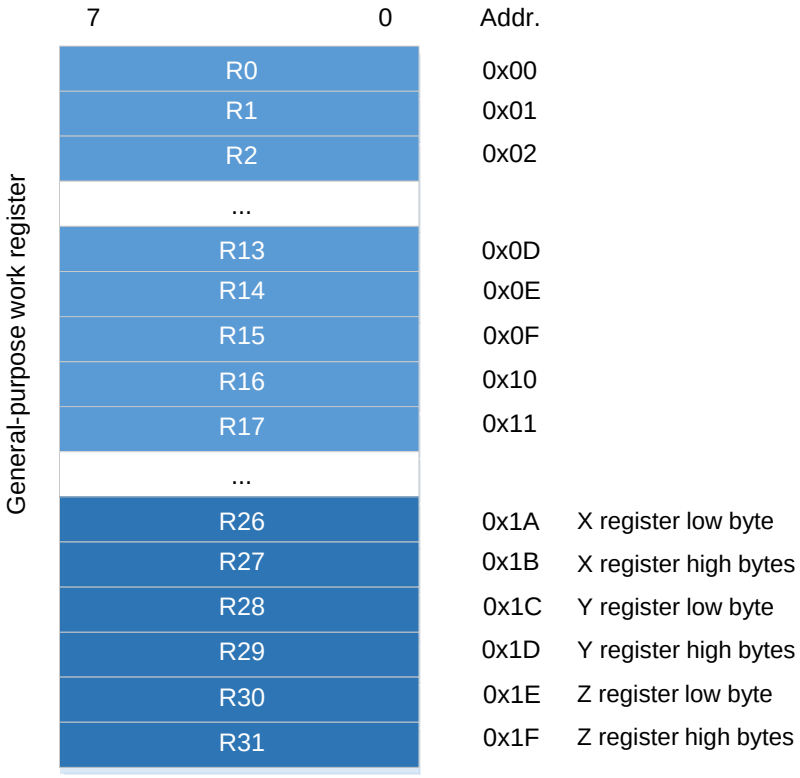
Bit definition		
[0]	C	The carry flag indicates that an arithmetic or logical operation caused the carry to be set, please refer to the instruction description for details.
[1]	Z	The zero flag indicates that the result of arithmetic or logical operations is zero, please refer to the instruction description section
[2]	N	Negative flag, indicating that arithmetic or logical operations produced a negative number, refer to the instruction description section
[3]	V	Overflow flag, indicating that the result of twos complement operation is overflow, please refer to the instruction description section.
[4]	S	The sign bit, equivalent to the XOR result of N and V, please refer to the instruction description section for details
[5]	H	Half-decimal flags, useful in BCD operations, indicate the half-decimals produced by byte operations
[6]	T	Temporary bits, used in bit copy (BLD) and bit memory (BST) instructions, the T bit will be used as a temporary memory bit to temporarily store the value of a bit in a general-purpose register. Please refer to the instruction description section for details.
[7]	I	The global interrupt enable bit, which must be set to 1 to enable the core to respond to interrupt events. The different interrupt sources are controlled by independent control bits. The global interrupt enable bit is the last barrier that controls the entry of interrupt signals into the core. The I bit is automatically cleared by the hardware after the core responds to the interrupt vector and automatically set after the execution of the interrupt return instruction (RETI). The I bit can also be changed using SEI and CLI commands, please refer to the instruction description section.

General-purpose work register

The general-purpose operating registers are optimized according to the LGT8XM instruction set architecture. In order to achieve the efficiency and flexibility required for core execution, the LGT8XM's internal general-purpose operating registers support the following access modes:

- An 8-bit read and an 8-bit write operation
- Two 8-bit reads and one 8-bit write operation
- Two 8-bit reads and one 16-bit write operation
- A 16-bit read and a 16-bit write operation

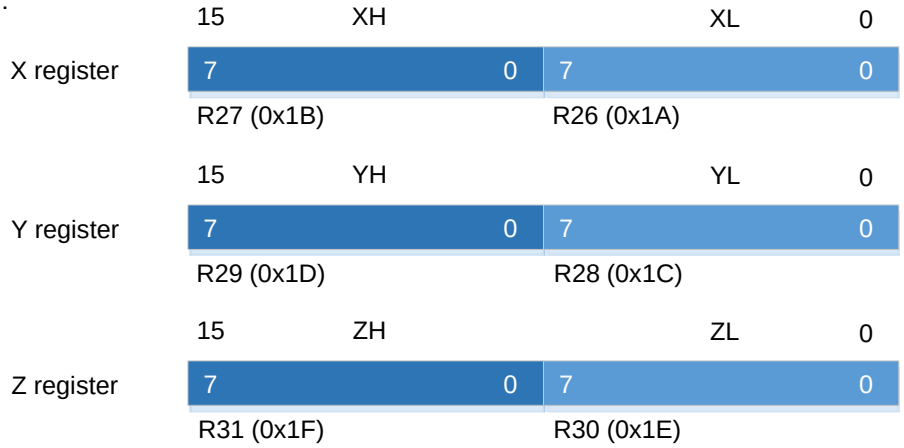
LGT8XM general-purpose operating register



Most of the instructions have direct access to all general-purpose operating registers, and most of them are single-cycle instructions.
As shown in the figure above, each register corresponds to the address of a data storage space, and these general-purpose working registers are mapped to the data storage space. As soon as they don't really exist in SRAM, but this unified mapping of the storage organization gives them a lot of flexibility to access them. X/Y/Z registers can be indexed as pointers to any general-purpose register.

X/Y/Z registers

Register R26... R31 can be combined in pairs to form three 16-bit registers. These three 16-bit registers are mainly used for address pointers for indirect addressing access, and the X/Y/Z registers are structured as follows:



In different addressing modes, these registers are used as fixed offset, auto-increment, and auto-decrement address pointers, please refer to the instruction description section for details.

Stack pointer

The stack is used to store temporary data, local variables, and return addresses for interrupts and subroutine calls. It is important to note that the stack is designed to grow from high addresses to low addresses. The stack pointer register (SP) always points to the top of the stack. The stack pointer points to the physical space where the data SRAM is located, which holds the stack space necessary for subroutines or interrupt calls. The PUSH instruction will decrement the stack pointer.
The position of the stack in SRAM must be set correctly by the software before the subroutine executes or interrupts are enabled. Typically, initialize the stack pointer to the highest address of SRAM. The stack pointer must be set to the high-bit SRAM start address. For addresses of SRAM mapping in the system data storing, refer to the System Data storing section.

Instructions related to stack pointers

Instruction	Stack pointer	description
PUSH	increased by 1	Data is pushed onto the stack
CALL ICALL RCALL	Increased by 2	The return address of an interrupt or subroutine call is pushed onto the stack
POP	Decreased by 1	Data is taken out of the stack
RET RETI	Decreased by 2	The return address of an interrupt or subroutine call is taken off the stack.

The stack pointer consists of two 8-bit registers allocated in the I/O space. The actual length of the stack pointer is relative to the system implementation. In some chip implementations of the LGT8XM architecture, the data space is so small that SPL alone can satisfy the addressing needs, in which case the SPH register will not be present.

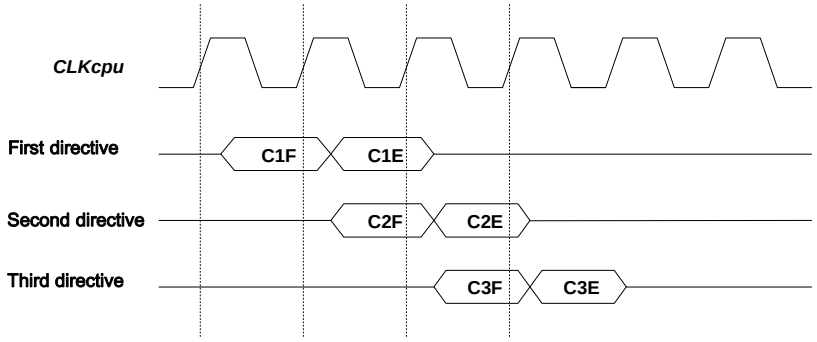
SPH/SPL stack pointer register definition

SPH/SPL Stack pointer registers		
SPH: 0x3E (0x5E) SPL: 0x3D (0x5D)		Default value: RAMEND
SP	SP[15:0]	
R/W	R/W	
Bit definition		
[7:0]	SPL	Stack pointer LSB
[15:8]	SPH	Stack pointer MSB

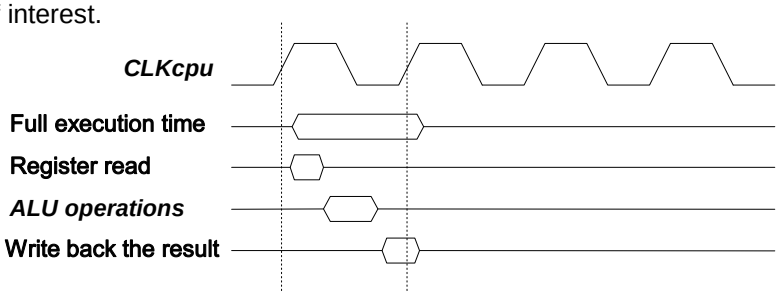
Instruction execution timing

This section describes general timing concepts for instruction execution. The LGT8XM core is driven by a core clock (CLKcpu) that comes directly from the clock source selection circuit with the system.
The following figure shows the instruction pipeline execution timing based on the Haval architecture and the concept of fast access register files. This is a physical guarantee that allows the core to achieve 1MIPS/MHz execution efficiency.

As can be seen from the figure below, the second instruction is read out at the same time during the execution of the first instruction. When the second instruction enters the execution



During the line, a third instruction is read out at the same time. In this way, there is no need to spend additional cycles for reading instructions during the entire execution period, and from the pipeline point of view, the efficiency of executing one instruction every Monday is achieved.
The figure below shows the access timing of the general-purpose operating registers, where an ALU operation uses two registers as operands during a cycle in which the ALU execution result is written to the register of interest.



Reset and interrupt handling

LGT8XM supports multiple interrupt sources. These interrupts and reset vectors correspond to a separate program vector entry in program space. In general, all interrupts are controlled by individual control bits. The core can only respond to this interrupt when this control bit is set and the core's global interrupt enable bit is enabled.

The lowest program space is reserved by default as the reset and interrupt vector regions. For a complete list of interrupts supported by LGT8FX8P, please refer to the Interrupts section. This list also determines the priority of different interrupts. The lower the vector address, the higher the corresponding interrupt priority. Reset (RESET) has the highest priority, followed by INT0 – External interrupt request 0.
The start address of the interrupt vector table (except the reset vector) can be redefined to the beginning of any 256-byte alignment and needs to be implemented by the IVSEL bit in the MCU control register (MCUCR) as well as the IVBASE vector base address register.

When the core responds to an interrupt, the global interrupt enable flag of I is automatically cleared by the hardware. The user can implement interrupt nesting by enabling the I-bit. This way, any subsequent interrupts interrupt the current interrupt service program. The I bit is automatically asserted after executing an interrupt return instruction (RETI) so that it can respond normally to subsequent interrupts.

There is a basic type of interrupt. The first type is triggered by an event, and the interrupt flag bit is set after an interrupt event occurs. For this interrupt, after the core responds to the interrupt request, the current PC value is directly replaced with the actual interrupt vector address, the corresponding interrupt service subroutine is executed, and the hardware automatically clears the interrupt flag bit. The interrupt flag bit can also be cleared by writing 1 to the position of the interrupt flag bit. If the interrupt enable bit is cleared when an interrupt occurs, the interrupt flag bit is still set to log the interrupt event. When the interrupt is enabled, the logged interrupt event is immediately responded. Similarly, if the interrupt occurs, the global interrupt enable bit (SERG. i) is cleared, the corresponding interrupt flag bit is also set to log interrupt events, etc

After the global interrupt enable bit is set, these logged interrupts will be executed in order of priority.

The second type of interrupt is when the interrupt condition is always in response to the interrupt condition. This interrupt does not require an interrupt flag bit. If the interrupt condition disappears before the interrupt is enabled, the interrupt will not get a response.

When the LGT8XM core exits the interrupt service subroutine, the execution flow returns to the main program. One or more instructions are executed in the main program before responding to other waiting interrupt requests.

It is important to note that the System Status Register (SREG) is not automatically saved after entering interrupt service and is not automatically recovered after returning from interrupt service. It must be handled by the software.

When an interrupt is disabled using the CLI directive, the interrupt is immediately disabled. Interrupts that occur after CLI instructions will not get a response. Even interrupts that occur at the same time as CLI instruction execution will not be responded to. The following example shows how to use the CLI to avoid interrupting the write timing of EEPROM:

Interrupt response time

The LGT8XM core is optimized for interrupt response so that any interrupt must respond within 4 system clock cycles. After 4 system clock cycles, the interrupt service subroutine enters the execution cycle. During these 4 clocks, the PC value before the interrupt is pushed into the stack, and the system execution process jumps to the interrupt vector corresponding to the interrupt service. If the interrupt occurs during a multi-cycle instruction execution, the kernel guarantees that the current instruction ends correctly. If the interrupt occurs in the system sleep state, the interrupt response requires an additional 4 clock cycles. This increases the synchronization cycle of the clock cycle used to wake up the operation from the selected sleep mode. For more information about sleep modes, refer to the chapter on power management.

Returning from the interrupt service subroutine takes 2 clock cycles. During these 2 clock cycles, the PC resumes from the stack, the stack pointer is incremented by 2, and the global interrupt control bit is automatically enabled.

Storage unit

overview

This section describes the different memory units inside the LGT8FX8P series. The LGT8XM architecture supports two main types of internal memory space, namely data storage space and program storage space. LGT8FX8P also contains data FLASH, and the data storage function of EEPROM interface can be realized through the internal controller. In addition, the LGT8FX8P system also contains a special memory unit to store the system configuration information and the global device number (GUID) of the chip.

LGT8FX8P series chips include LGT8F88P/168P/328P four different models; The four models of peripherals and packages are fully compatible, the difference is FLASH program storage space and internal data SRAM, the following table clearly describes the different storage space configurations of LGT8FX8P series chips:

DEVICE	FLASH	SRAM	E2PROM	Interrupt vector
LGT8F88P	8KB	1KB	2KB	1 command word
LGT8F168P	16KB	1KB	4KB	2 command words
LGT8F328P	32KB	2KB	(Configurable to share with FLASH with 0K/1K/2K/4K/8K)	2 command words

LGT8F328P does not have a separate FLASH space for emulating the E2PROM interface; The storage space used to simulate E2PROM is shared with the program FLASH, and the user can choose the appropriate configuration according to the application requirements.
Due to the unique implementation of the analog E2PROM interface, the system requires twice the program FLASH space to simulate the E2PROM storage space, for example, for LGT8F328P, if the user configures 1KB of E2PROM space, 2KB bytes of program space will be reserved, leaving 30KB of FLASH space for storing programs.

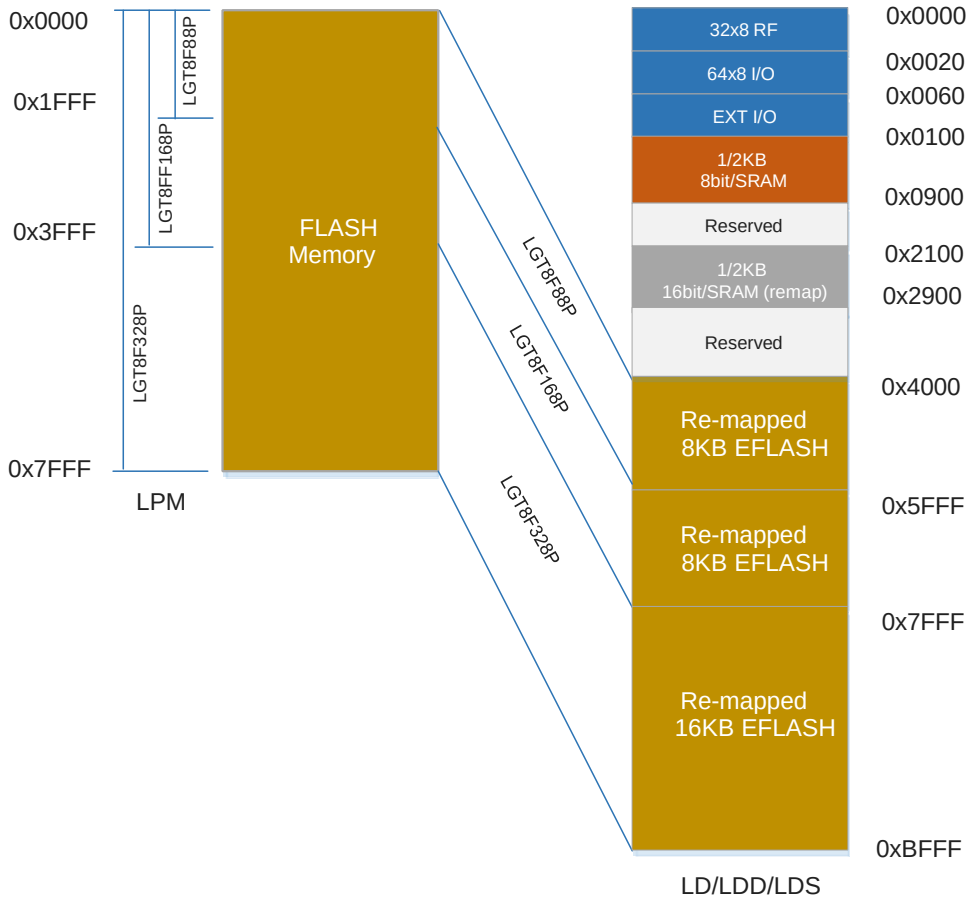
LGT8F328P program FLASH shares configuration table with E2PROM:

DEVICE	FLASH	E2PROM
LGT8F328P	32KB	0KB
	30KB	1KB
	28KB	2KB
	24KB	4KB
	16KB	8KB

System programmable FLASH program memory unit

The LGT8FX8P series of microcontrollers includes 8K/16K/32K bytes of on-chip online programmable FLASH program memory units.
The program FLASH can guarantee at least 100,000 erasing cycles. LGT8FX8P integrates a FLASH interface controller, which can realize the self-upgrade function of in-system programming (ISP) and program. for specific implementation details, please refer to the description of the flash interface controller section of this chapter.

The program space can also be accessed (read) directly through LPM instructions, which enables application-related constant lookup tables. At the same time, the FLASH program space is also mapped to the system data storage space, and users can also use LD/LDD/LDS to achieve access to the FLASH space. The program space is mapped to the address range where the datastore 0x4000 begins.
As shown in the following figure:



SRAM data storage unit

The LGT8FX8P family of microcontrollers is a relatively complex microcontroller that supports many different types of peripherals whose controllers are allocated over 64 I/O register spaces. It can be accessed directly through IN/OUT commands. The control registers of other peripherals are allocated in the 0x60 ~ 0xFF region, and because this part of the space is mapped to the data storage space, it can only be accessed through ST/STS/STD and LD/LDS/LDD instructions.

The LGT8FX8P's system data storage space starts from address 0 and maps the general purpose working register file, I/O space, extended I/O space, and internal data SRAM space. The first 32 bytes of addresses correspond to the 32 general-purpose operating registers of the LGT8XM core. The next 64 addresses are standard I/O spaces that can be accessed directly via IN/OUT commands.
Then 160 addresses are extended I/O space, followed by up to 2K bytes of data SRAM. This part of the space from the beginning of the 0x4000 to the end of the 0xBFFF maps the FLASH program storage unit.
The 1K/2K bytes SRAM within the system are mapped to two spaces. This space from the beginning of 0x0100 to the end of 0x0900 is read and written by the kernel in a width of 8 bits. From the beginning of 0x2100 to the end of 0x2900, this area is 16 bits wide access space. The system RAM is mapped to the high-bit address at the beginning of the 0x2100, which is mainly used to work with the uDSU module to achieve efficient 16-bit data storage. When programming, you can switch to 16-bit access mode by adding the ordinary 8-bit addressing variable address to the 0x2000 offset.

The system supports 5 different addressing modes that can cover the entire data space: direct access, indirect access with offset, indirect access, indirect access with decrementing addresses before access, and indirect access with incremented addresses after access. Address pointers for general-purpose operating registers R26 through R31 for indirect access. Indirect access can address the entire data storage space. Indirect access with offset addresses can address 63 nearby address spaces with Y/Z registers as base addresses.
When using the register indirect access mode that supports address autoincrement/decrement, address registers X/Y/Z are automatically decremented/incremented by hardware before/after access occurs. Please refer to the instruction set description section for details.

The 16-bit registers X/Y/Z and the associated automatic addressing modes (increment, decrement) also play a very important role in 16-bit extended mode. The 16-bit extended mode can use the increment/decrement mode of LD/ST to achieve automatic increment and decrement addressing with variables. This mode is very effective when performing operations on arrays. For specific implementation, please refer to the section "Number Computing Accelerator (uDSU)".

General-purpose I/O registers

The I/O space of the LGT8FX8P has three general-purpose I/O registers, GPIOR2/1/0, which can be accessed using IN/OUT instructions to store user-defined data.

Peripheral register space

For a detailed definition of the I/O space, refer to the "Register Overview" section of the LGT8FX8P data sheet.

LGT8FX8P all peripherals are allocated to I/O space. All I/O space addresses can be accessed by LD/LDS/LDDD and ST/STS/STD instructions. The data accessed is passed through 32 general-purpose work registers. The I/O registers between 0x00 ~ 0x1F can be accessed via bit-addressing instructions SBI and CBI. In these registers, the value of a bit can be detected using SBIS and SBIC instructions to control the execution of the program. Please refer to the instruction set description section for details.
When using the IN/OUT instruction to access the I/O registers, the address between 0x00 ~ 0x3F must be addressed. When using LD or ST instructions to access the I/O space, it must be accessed through the mapped address of the I/O space in the system data memory unified map space (plus the offset of the 0x20). Other peripheral registers (0x60 ~ 0xFF) allocated in the extended I/O space can only be accessed using ST/STS/STD and LD/LDS/LDD instructions.

For compatibility with future devices, reserved bits must be written to 0 at write operations. Write operations cannot be performed on reserved I/O space.

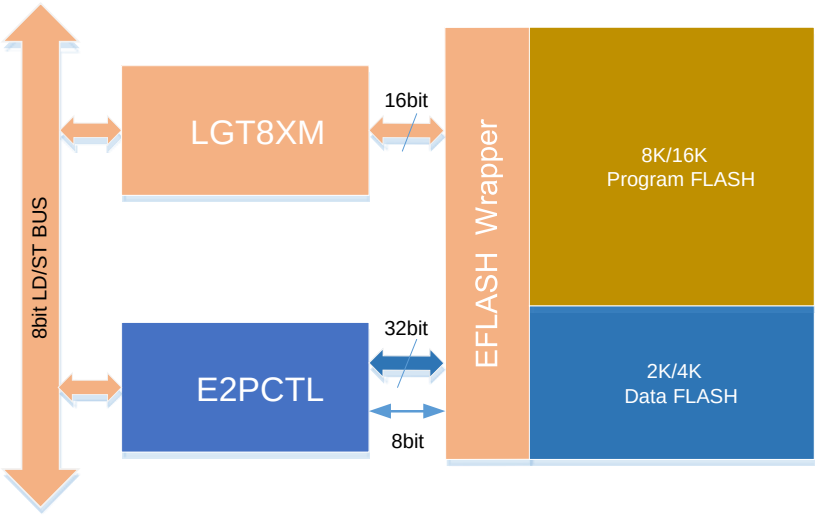
Some registers include status flags that need to be written 1 to clear zero. It is important to note that CBI and SBI instructions only support specific bits, so CBI/SBI can only operate on registers that contain these status flags. In addition, CBI/SBI instructions can only operate in registers in the address range 0x00 to 0x1F.

FLASH controller (E2PCTL)

LGT8FX8P integrates a flexible and reliable EFLASH read and write controller, which can use the existing data FLASH storage space in the system to realize the storage space of byte read and write access, and realize storage applications similar to E2PROM; E2PROM interface simulation adopts the erasing equalization algorithm, which can increase the use cycle of data FLASH by about 1 times, and can ensure more than 100,000 erasing cycles.

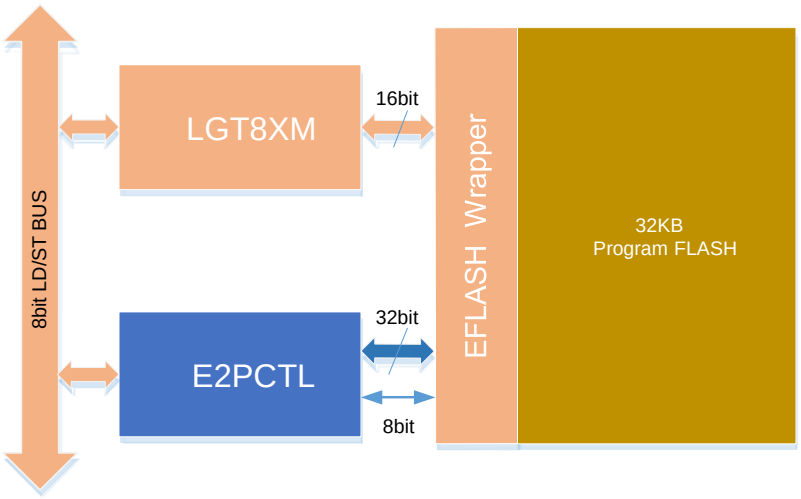
The E2PCTL controller also realizes the online erasing operation of the FLASH program space, and the function of automatically upgrading the firmware online can be realized through the software. Access to the FLASH program space through the FLASH controller, only page erasure (1024 bytes) and 32-bit width read and write access are supported.

LGT8F88D/168D E2PCTL controller block diagram



E2PCTL simulates that when the E2PROM function accesses the data FLASH space, it can support 8-bit and 32-bit read and write widths. When accessing the program FLASH space, page erasure and 32-bit data reading and writing are supported. Since the minimum memory cell of the LGT8FX8P internal FLASH is 32 bits, 32-bit access is recommended, especially for write operations. The read and write operations of 32-bit access are not only efficient, but also help protect the erase and write life of the FLASH memory unit.

LGT8F328P E2PCTL controller block diagram

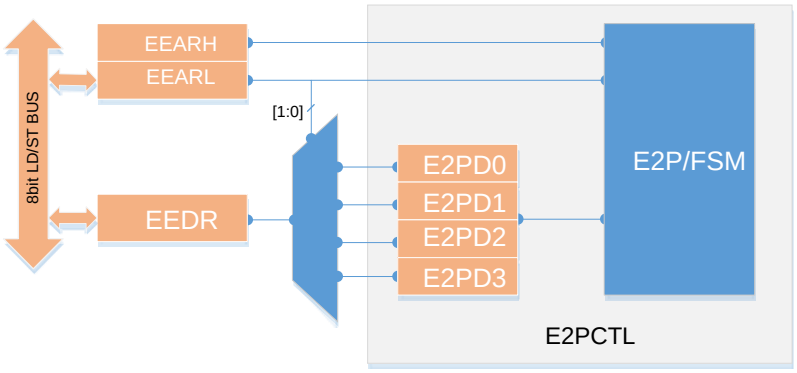


LGT8F328P has no extra data FLASH inside. as a result, the lgt8xm core shares an internal 32K byte of flash storage with E2PCTL. Users can divide the 32K byte FLASH space into program space and data space according to their needs. By configuring the E2PCTL controller, you can set the space size for emulating the E2PROM. E2PCTL uses a page switching mode to emulate E2PROM logic, and the algorithm is measured in pages (1K bytes). Therefore, simulating 1K bytes of e2prom space requires 2K bytes of flash space, and so on, to achieve 4k-byte e2prom, it takes 8K bytes of flash space. For specific implementation, please refer to the description of the E2PCTL algorithm implementation.

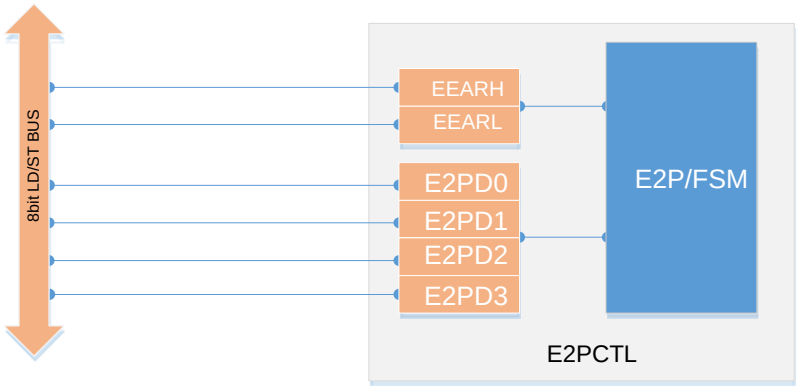
E2PCTL data register

The E2PCTL controller has a 4-byte data cache (E2PD0~3), and this 4-byte cache constitutes a 32-bit data interface that ultimately accesses the FLASH space.

When the E2PCTL controller works in byte-read-write mode, EEDR is used as an interface for reading and writing byte data, E2PCTL loads the data into the correct data cache with the address information of EEARL [1:0], and completes the other three bytes of data according to the data of the current FLASH destination address, and finally updates the combined complete 32-bit data to FLASH.
When E2PCTL operates in 32-bit read and write mode, the EEDR register can still be used as a common data interface to address the internal data cache through EEARL [1:0] to achieve reading and writing a complete 32-bit data.
In addition, you can directly use the data cache to map the registers of IO space for direct access (E0~3).
Data access diagram of E2PCTL operating in 8-bit byte read and write mode:



Data access diagram of E2PCTL operating in 32-bit word read and write mode:



Byte mode is used for byte read and write mode that is backward compatible with LGT8FX8D. LGT8FX8P's built-in FLASH is 32-bit interface width, using 32-bit read/write mode will bring great benefits to read and write efficiency and FLASH erase and write life, so it is recommended to use 32-bit read/write mode.

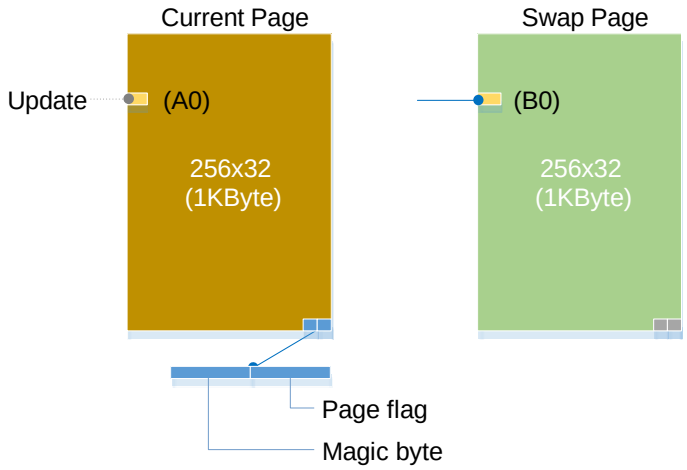
E2PCTL simulates the E2PROM interface algorithm

The flash memory must be erased before it can be written, and the erase operation is on a page-by-page basis. LGT8FX8P has built-in FLASH memory, and the size of one page is 1K bytes. Therefore, in order to update a byte of data in the page, it is also necessary to first erase the data of the entire page, then update the destination address data, and restore the data of other bytes in the page at the same time, the whole operation is not only time-consuming, but also brings the risk of accidental data loss due to power supply.

E2PCTL uses a page-switching algorithm to simulate E2PROM. The page switching algorithm mode can ensure that when performing page erase operations, the original data will not be lost due to unexpected conditions such as power failure. At the same time, the switching algorithm uses two pages spaces to exchange each other alternately, which also increases the service life of the simulated E2PROM space.

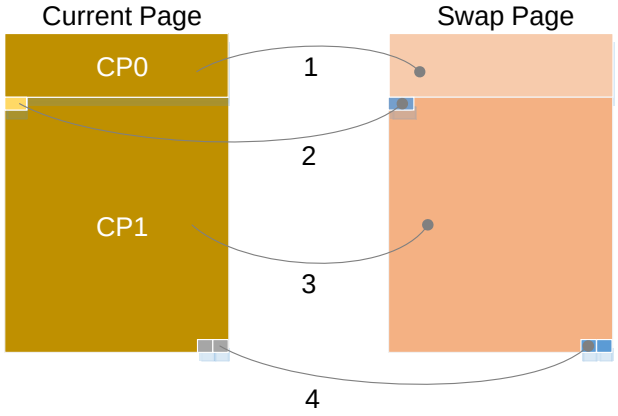
In terms of efficiency, the E2PCTL controller implements a continuous data update mode, which reduces the repetitive erasing process caused by repeated data updates.
In terms of implementation, E2PCTL manages each page separately and occupies the last 2 bytes of a page as page state information. Therefore, when using E2PROM simulation space larger than 1K, users need to pay attention to the special handling of address spanning 1K space. Because the last 2 bytes of each 1K space are reserved for E2PCTL, users cannot read and write these 2 bytes normally.

The following figure shows a schematic diagram of E2PCTL's page-switched algorithm:

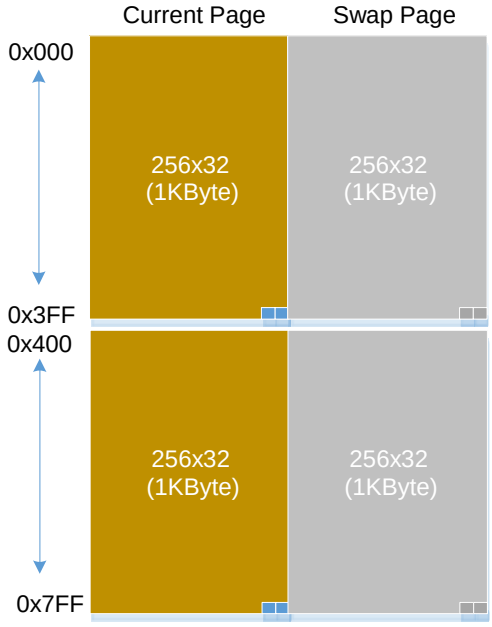


As shown, E2PCTL internally uses 2 pages to simulate the E2PROM space of one page size. One of the two pages is marked as the current page and the other is the swap page. E2PCTL uses the last 2 bytes of the page to store page information. When we need to update a certain byte in the page, such as the A0 byte in the image above. First, instead of erasing the current page, we erase the swap page. Then the current page is divided into 3 parts for operation. First the data before A0, we call this part of the space CP0, and then the data after A0, which is CP1. E2PCTL will copy the data corresponding to CP0 to the corresponding address of the exchange page according to the user configuration, and then write the data to be updated to the address corresponding to the exchange page (B0), and finally copy the data of CP1 to the exchange page.
After completing the above operations, the data has been exchanged, but the page state has not been updated. Therefore, if a power failure or other exception occurs before this, the update operation is not completed, and the previous data will not be destroyed, ensuring the integrity of the data. If all goes well, E2PCTL will write the updated page status to the page information of the previous exchange page at the end of the CP1 exchange data, and realize the face-to-face page replacement. After that, the swap page becomes the current page.

The E2PCTL page swapping process is shown in the following figure (1->2->3->4):



When the E2PROM simulation space configured by the system is greater than 1K, E2PCTL still implements the simulation algorithm of E2PROM space in the smallest unit of pages. For example, if the user configures a 2K E2PROM area, the E2PCTL will actually occupy 4 pages (4K) of space. Two pages are set to implement E2PROM space that simulates one page size.



It should be noted that the user-configured 2K bytes of E2PROM space is not contiguous, because the last 2 bytes of each page will be used to store page state information.

E2PCTL continuous programming mode

Since updating through E2PCTL will cause page swapping, the swap page will be erased during the page swap process, which is not only time-consuming, but also increases the loss of FLASH life. Therefore, E2PCTL adds a continuous write mode. In continuous write mode, the user can continuously update the E2PROM area, and only at the end of the consecutive address, the page swap operation will occur, which is more effective for applications that require continuous updating of a whole block of data.

SWM bit enable of the ECCR control register ECCR in continuous programming mode. After continuous mode is enabled, subsequent write operations will directly write data to the address corresponding to the swap page, and in SWM mode, write operations will not perform CP0/1 area data copy operations. Before writing the last byte, the software disables contiguous mode via SWM, then performs the write, after which E2PCTL performs a full CP0/1 copy operation and updates the page state information.

E2PCTL read and write the FLASH program space

Through the E2PCTL controller, read and write access to the FLASH space of the program can be realized. Unlike analog E2PROM, access to the program's FLASH space via E2PCTL requires complete software control. The steps are as follows:

1. Erase the target page, before updating the data, you need to erase the target page first, and the page address is given through the EEAR register. For the erasure command control of the FLASH page, please refer to the definition of EECR register;
2. The FLASH space must be written in 32-bit minimum. Set data through E2PD0~3;
3. The destination address is given by the EEAR register, and the address EEAR[1:0] will be ignored;

Through the E2PCTL read and write program FLASH space, the online program update (IAP) function can be realized, which is very useful in some applications that need to update application data in the field and need to provide product custom updates.

E2PCTL interface operation flow

The E2PCTL controller is mainly realized through 4 registers, namely E2PCTL control status registers EECR and ECCR; Data register EEDR (E2PD0~E2PD3) and address register EEAR (eearl/eearh).
The ECCR register is used to set the working state of E2PCTL, and most of the state needs to be set before E2PCTL work, which is generally implemented during system initialization. The SWM bit in the ECCR register is used to enable continuous write mode, and this control bit needs to be set during the implementation of continuous write operations.
The EECR register is used to control the type of select operation and to select operation instructions, such as setting read and erase commands.
The EEDR register is used for the 8-bit byte mode interface, and E2PD0~3 is used for the read and write operation of 32-bit mode; The EEAR register is used to set the destination address for read, write, and also to set the page address for page erase operations. The page address is aligned in page units, and the size of a page is 1K bytes, and it should be noted that the address specified by EEAR is a byte address.

Access FLASH program space via the E2PCTL interface:

Through the E2PCTL interface, it can read and write and erase the FLASH program space. Reading and writing to flash space only supports 32-bit access width. The erase operation is in page bits, 1K bytes (256x32) per page size.
Before writing the FLASH program space, first erase the page where the destination address is located. E2PCTL write FLASH program space does not support continuous mode, users need to complete write operations sequentially. The following is the process of erasing the flash program space :

1. Program FLASH page erasing operation

- ❑ Set EEAR [14:0] as the target page address to be erased, and the size of one page of the program FLASH is 1K bytes, so EEAR [14:10] will be used as the page address, and EEAR [9:0] will be set to 0
- ❑ Set EEPM[3:0] = 1X01, where EEPM[2] can be set to 0 or 1
- ❑ Set EEMPE = 1 and EEPE = 0
- ❑ In four cycles, set EEPE = 1 to start the program FLASH erasing process

2. Program FLASH programming operation

- ❑ Write E2PD0~3 and prepare 32-bit programming data
- ❑ Set EEAR as the destination address, where the address is 4-byte aligned
- ❑ Set EEPM[3:0] = 1X10, where EEPM[2] can be set to 0 or 1
- ❑ Set EEMPE = 1 and EEPE = 0
- ❑ In four cycles, set EEPE = 1 to start the FLASH programming process

Access the E2PROM simulation space via the E2PCTL interface:

The E2PCTL controller logically accesses the data FLASH space by simulating the E2PROM interface. The analog E2PROM supports 8-bit, 16-bit, and 32-bit data width read and write access. The 8-bit byte mode has better compatibility with the E2PROM interface.

32-bit mode is conducive to improving storage efficiency and FLASH life, so 32-bit read/write mode is the recommended read-write mode. The E2PROM analog interface supports continuous read/write mode, which is recommended for data applications that require multiple sequential addresses to be updated at once.

For LGT8F88P/168P, data flash is a separate storage space. There is no need to configure and enable the FLASH data space through ECCR registers. LGT8F328P does not have an independent data flash space, and data flash shares 32k bytes of flash space with program flash. it is necessary to enable the data flash partitioning function through the ECCR register, and configure the size of the data flash through the ECS [1:0] bit of the ECCR register. After the configuration takes effect, other usage methods are the same as LGT8F88P/168P.

When the FLASH controller implements the E2PROM interface, the logic of automatically erasing data FLASH when necessary has been implemented internally, so the EPROM erase command is optional, and this command is only used when the user needs to perform the erase separately.

The EECR register controls the erase/write timing of FLASH, including the program FLASH and E2PROM. The specific type of operation needs to be set via the EEPME and EEPM [3:0] of the EECR register. The read operation of E2PROM is relatively simple, after setting the destination address and mode, the EERE bit is written to read the 32-bit data corresponding to the target address into the FLASH controller, and the user can read the bytes of interest through the EDR register. The flash controller does not realize the read operation of the program flash space, and the user can easily use lpm or ld/ldd/lds instructions to read at the address of the data unified mapping space through the program flash.

1. 8-bit mode, programming E2PROM

- ❑ Set the destination address to the EEARH/L register
- ❑ Set new data to the EEDR register
- ❑ Set EEPM[3:1] = 000, EEPM[0] can be set to 0 or 1
- ❑ Set EEMPE = 1 and EEPE = 0
- ❑ Within four cycles, set EEPE = 1 When the setup is complete, the FLASH controller will initiate the programming operation, during which the CPU will remain at the current instruction address and will not continue to run until the operation is completed. During programming, if data FLASH needs to be erased, the FLASH controller will automatically initiate the erasing process.

2. 32-bit mode, programming E2PROM

- ❑ Through E2PD0~3, prepare 32-bit data
- ❑ Set the destination address to the EEARH/L register. Note that this is a byte-aligned address, and the flash controller uses EEAR[15:2] as the address to access flash.
- ❑ Set EEPM[3:1] = 010, EEPM[0] can be set to 0 or 1
- ❑ Set EEMPE = 1 and EEPE = 0
- ❑ Over four periods, set EEPE = 1

3. 8-bit mode, read E2PROM

- ❑ Set the destination address to the EEARH/L register
- ❑ Set EEPM[3:1] = 000
- ❑ Set EERE = 1 to start the E2PROM read operation
- ❑ Wait 2 cycles (perform two NOP operations)
- ❑ The data corresponding to the destination address is updated to the EEDR register

4. 32-bit mode, read E2PROM

Set EEARH/L as the destination address, and the address is 4-byte aligned
Set EEPM[3:1] = 010 to enable 32-bit interface mode
Set EERE = 1 to initiate E2PROM read
Wait for 2 system clock cycles (execute two NOP instructions)

E2PCTL access emulates the E2PROM space and supports continuous programming mode, which is very efficient for applications that need to update one block at a time, and also helps to improve the lifetime of FLASH. Continuous programming mode only supports 32-bit wide data programming operations.
Continuous access mode is enabled via the SWM bit of the ECCR register. After SWM is enabled, the next operations that simulate the E2PROM space via E2PCTL are in continuous programming mode. In continuous programming mode, the E2PCTL controller automatically handles page feeds based on the data in the destination address. However, if page change occurs during continuous programming mode, the controller will not automatically exchange data in the CP0/1 region during continuous programming, nor will it update the page information.
When continuous programming to the last operation, turn off the continuous programming mode by clearing the SWM bit, and then start the last programming operation in non-SWM mode, after the end of programming, E2PCTL will automatically copy the data of the CP0/1 region to the exchange page, and update the information of the exchange page to make it the current valid page, so as to complete the entire continuous programming operation.

5. Continuous programming mode operation process:

1. Configure the size of the data flash via ECCR and enable the SWM bit
2. E2PROM area is simulated using 32-bit mode programming
3. If it is not the last operation, go back to step 2 to continue programming the next data
4. If the last programming is reached, first disable the continuous programming mode via SWM, and then complete the last programming using the operating procedure of step 2

E2PCTL efficient FLASH data management

In addition to implementing a continuous programming mode, the E2PCTL controller can also be independently controlled by the CP0/1 bit of the ECCR register to copy the page exchange process data. The CP0/1 of the ECCR register is used to control the exchange of data for the CP0/1 region of the current page during page swapping. If the CP0/1 bit is cleared, the data of the corresponding region in the current page will not be exchanged during the page swapping process. This section provides an efficient management method that will take advantage of this feature.
In the flash data update process, the most time-consuming operation occurs during the swap page erasing process. As a result, we can address a data management approach that minimizes page erase times, improving both programming efficiency and lifetime loss.
Here we provide a reference algorithm for block-based data management applications:

1. Assume that the user data is just a complete block of data, an integer multiple of the block size of 4 bytes;
2. Each data update will update a complete block of data
3. In addition to storing user data, block information also needs to store a block management information

Under the above three conditions, we can make full use of the continuous programming mode and automatic page switching mechanism of E2PCTL to achieve an efficient FLASH data management method.
Since the data updated each time is a block of data of the same size, and each data structure holds the address information pointing to the next piece of data, we can program FLASH in address order each time we update the data, without the need to do CP0/1 data copying. At the same time, since the data is updated to an erased area every time, page erasure does not occur.
When the last piece of data is written, its structure information points to the next piece of data area back to the start address of the page. After a data write operation, E2PCTL initiates a page erase process and updates the currently active page.

Protection measures for flash operation

If the VCC voltage is low, the FLASH erase operation may be wrong because the voltage is too low.
FLASH/data erase and write errors at low pressure can be caused by two reasons. First of all, normal FLASH erase operation requires a minimum working voltage, below this voltage, the operation will fail and cause data errors. The second reason is that the core runs at a certain frequency, which also requires a minimum voltage requirement, and when it is lower than this voltage, it will cause the instruction execution error, resulting in the operation of FLASH error.
You can avoid similar problems in the following simple ways:
When the supply voltage is low, let the system enter a reset state. This can be achieved by configuring an internal low-voltage detection circuit (VDT). If the VDT detects that the current operating voltage is below the set threshold, the VDT outputs a reset signal. If the threshold value of the VDT does not meet the needs of the application, consider adding an external reset circuit.

Register description

FLASH Address registers EEARH/EEARL

EEARH/EEARL		
EEARH: 0x22 (0x42) EEARL: 0x21 (0x41)		Default value: 0x0000
bits	EEAR[15:0]	
R/W	R/W	
Bit definition		
[7:0]	EEARL	EFLASH/E2PROM access address LSB.
[14:8]	EEARH	EFLASH/E2PROM access address 7 MSB
[15]	-	Reserved

When using the E2PCTL controller to access the FLASH area of the program, EEAR [14:2] is used to access the entire program space aligned in 4 bytes. EEAR[1:0] is only used when accessing the data register EEDR. Please refer to the description of EEDR data registers below. The E2PCTL controller supports 8/16/32-bit modes, and in either mode, EEAR here is byte-aligned addressing.

FLASH data register - EEDR/E2PD0

EEDR/E2PD0 – FLASH/E2PROM Data Register 0		
EEDR/E2PD0: 0x20 (0x40)		Default value: 0x00
bits	EEDR[7:0]	
R/W	R/W	
Bit definition		
[7:0]	EEDR E2PD0	E2PCTL data register In 16/32-bit mode, used to access the lowest bytes

FLASH data register - E2PD1

E2PD1 – E2PCTL data register 1	
E2PD1: 0x5A	Default value: 0x00
bits	E2PD1[7:0]

R/W		R/W
Bit definition		
[7:0]	E2PD1	The upper 8 bits used to store 16-bit data in 16-bit mode In 32-bit mode, it is used to store the upper 8 bits of the lower 16 bits of data

FLASH data register - E2PD2

E2PD2 – FLASH Data Register 2		
E2PD2: 0x57		Default value: 0x00
Bits	E2PD2[7:0]	
R/W	R/W	
Bit definition		
[7:0]	E2PD2	In 32-bit mode, it is used to store the lower 8 bits of data in the upper 16 bits

FLASH data register - E2PD3

E2PD3 – FLASH Data Register 3		
E2PD3: 0x5C		Default value: 0x00
Bits	E2PD3[7:0]	
R/W	R/W	
Bit definition		
[7:0]	E2PD3	In 32-bit mode, it is used to store the upper 8 bits of data in the upper 16 bits

FLASH mode control register - ECCR

ECCR – FLASH/E2PROM configuration register								
ECCR: 0x36 (0x56)					Default value: 0x0C			
bits	WEN	EEN	ERN	SWM	CP1	CP0	ECS1	ECS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	1	1	0	0

Bit definition		
[7]	WEN	ECCR write enable control Before modifying the ECCR, the WEN must be written 1 and then the contents of the ECCR register must be updated within 6 system cycles
[6]	EEN	E2PROM enabled and only valid for LGT8F328P 1: Enabling E2PROM simulation will reserve some space from 32KFLASH 0: Disable E2PROM emulation, 32KFLASH is all used for program space
[5]	ERN	Write 1 will reset the E2PCTL controller
[4]	SWM	Continuous write mode for emulating E2PROM controller operation
[3]	CP1	Page Swap CP1 region enable control
[2]	CP0	Page Switching CP0 Region Enable Control
[1:0]	ECS[1:0]	E2PROM space configuration 00: 1KB E2PROM, 30KB FLASH 01: 2KB E2PROM, 28KB FLASH

		10: 4KB E2PROM, 24KB FLASH 11: 8KB E2PROM, 16KB FLASH

FLASH Access Control Register - EECR

EECR – FLASH/E2PROM 控制寄存器								
EECR: 0x1F (0x3F)				Default value: 0x00				
bits	EEPM3	EEPM2	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Bit definition

[7:4]	EEPM[3:0]	EFLASH/EPROM Access mode control bits				
		[3]	[2]	[1]	[0]	Mode description
		0	0	0	x	8-bit mode read/write E2PROM (default)
		0	0	1	x	16-bit mode read/write E2PROM
		0	1	0	x	32-bit mode read/write E2PROM
		1	x	0	0	E2PROM Erase (optional)
		1	x	0	1	Program flash erase (page erase)
		1	x	1	0	Program FLASH programming
		1	x	1	1	Reset the FLASH/E2PROM controller
[3]	EERIE	FLASH/E2PROM Ready Interrupt Enable Control. Write 1 enabled, write 0 prohibited. When EEPE is automatically zeroed by the hardware, the E2PROM ready interrupt is valid. This interrupt will not be generated during EPROM operation				
[2]	EEMPE	FLASH/E2PROM programming operation enable control bit EEMPE is used to control whether EEPE is valid, when EEMPE is set to 1 and EEPE is 0 at the same time, in the next four cycles, setting EEPE to 1 will start the programming operation. Otherwise, the programming operation is invalid. After four cycles, EEMPE is automatically zeroed				
[1]	EEPE	FLASH/E2PROM programming operates the enable bit				
[0]	EERE	The E2PROM reads the enable bit, and the data will be valid after two system cycles				

General-purpose I/O register - GPIOR2

GPIOR2 – General-purpose I/O register 2		
GPIOR2: 0x2B (0x4B)	Default value: 0x00	
Bits	GPIOR2[7:0]	
R/W	R/W	
Initial value	0x00	
Bit definition		
[7:0]	GPIOR2	General-purpose I/O registers2 for storing user-defined data

General-purpose I/O register - GPIOR1

GPIOR1 – General-purpose I/O register 1		
GPIOR1: 0x2A (0x4A)		Default value: 0x00
Bits	GPIOR1[7:0]	
R/W	R/W	
Initial value	0x00	
Bit definition		
[7:0]	GPIOR1	General-purpose I/O registers1 for storing user-defined data

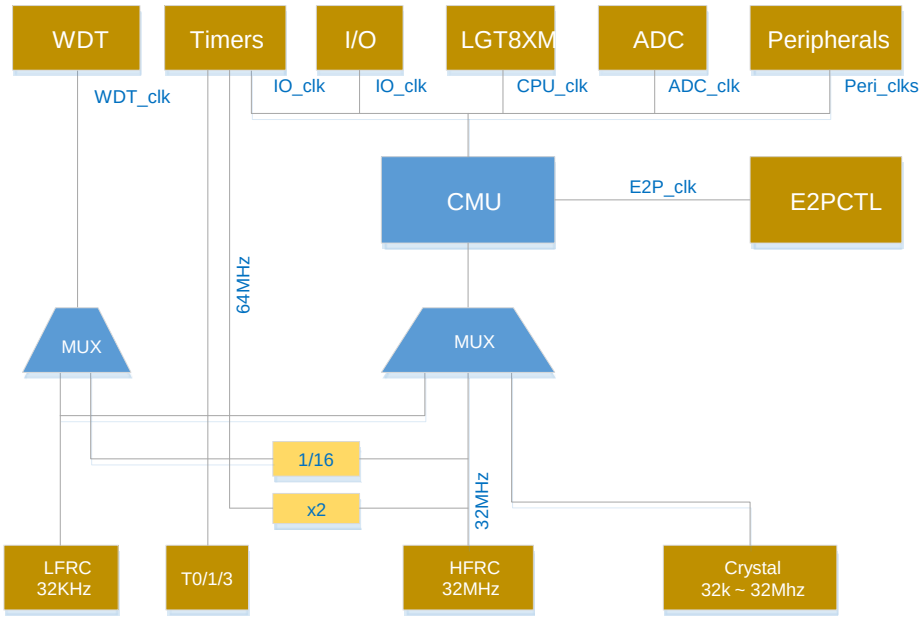
General-purpose I/O register - GPIOR0

GPIOR0 – General-purpose I/O register 0		
GPIOR0: 0x1E (0x3E)		Default value: 0x00
Bits	GPIOR0[7:0]	
R/W	R/W	
Initial value	0x00	
Bit definition		
[7:0]	GPIOR0	General-purpose I/O register 0 to store user-defined data

System clock and configuration

System clock distribution

LGT8FX8P supports a variety of clock inputs. The system can operate from three main clock sources, namely an internal 32KHz calibrated RC oscillator, an internal 32MHz calibrated RC oscillator, and an external 400KHz ~ 32MHz crystal input.
The following figure shows the distribution of LGT8FX8P clock system, CMU is the center of the entire clock management, responsible for the frequency division of the system clock, generating independent clocks for different modules and controlling the clock, etc. In general applications, it is not necessary to work all clocks at the same time, in order to reduce system power consumption, system power management according to different sleep modes, turn off the unused module clock.
For details on operation, see the Power Management section.



CPU_clk

Used to drive the LGT8XM core and SRAM operation. For example, drive general-purpose operating registers, status registers, etc.
Once the CPU clock stops, the kernel will not continue to execute instructions and perform calculations. After the system executes the SLEEP command into sleep mode, the kernel clock will be turned off.

Peri_clk

Used to drive most peripheral modules, such as timing/counter, SPI, USART, etc. The IO clock is also used to drive an external interrupt module. When the peripheral clock is stopped due to sleep, some peripheral parts that can be used to wake up the system operate in separate clock or asynchronous mode. For example, TWI's address recognition function can wake up most of the sleep mode, and the address recognition part at this time works in asynchronous mode.

E2P_clk

E2P_clk clock is used to generate the FLASH interface access timing. E2P_clk generate timing for accessing the E2PCTL access FLASH interface. E2P_clk fixed divide-32 (1MHz) from the internal 32MHz HFRC oscillator. If the user needs to use the E2PCTL module to read and write the internal program FLASH or data FLASH space, the internal 32MHz oscillator needs to be enabled in advance.

Asy_clk

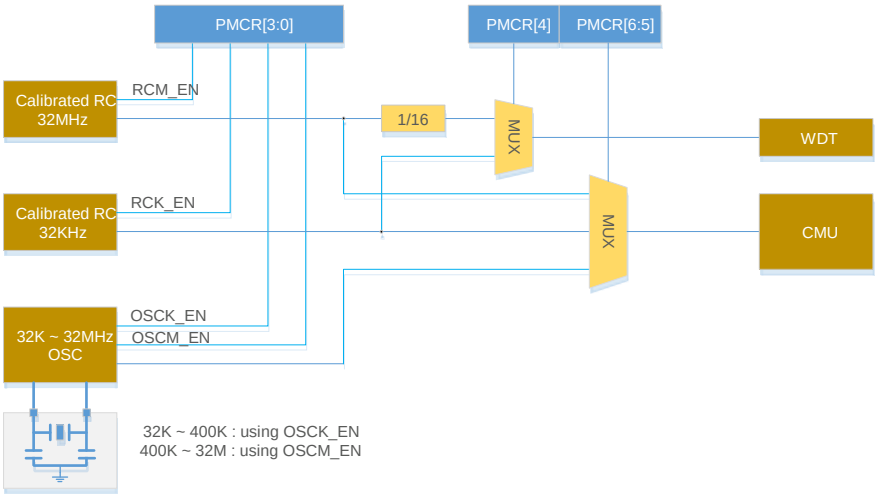
Asynchronous timer clock. The timer/counter can be driven directly using an external clock or crystal (32.768K). This independent clock mode allows the timer to remain running while the system is processing sleep mode.

WDT_clk

Internal watchdog timer clock source, configurable to select the internal 32KHz LFRC oscillator, or divide-by-16 (2MHz) from the internal 32MHz HFRC. After powering up the system, the watchdog default clock source is a 32KHz LFRC oscillator.

Clock source selection

LGT8FX8P supports four clock source inputs, and users can enable and control the clock source and complete the switching of the master clock through the PMCR register. The following is the control structure diagram of PMCR:



LGT8FX8P internal OSC oscillator can work in high frequency and low frequency modes, users need to control the internal OSC oscillator according to the actual size of the external crystal oscillator to work in the correct mode. Similarly, the internal RC oscillator is also divided into two types: high frequency and low frequency. The lowest 4 bits of the PMCR register are used to control these four clock sources. The control relationship is as follows:

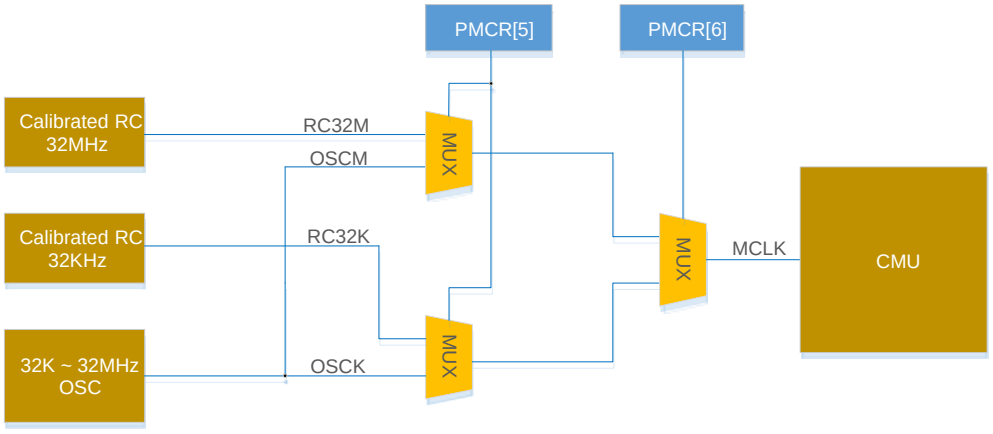
PMCR	Corresponds to the clock source
PMCR[0]	32MHz RC enable control, 1 enabled, 0 off
PMCR[1]	32KHz RC enable control, 1 enabled, 0 off
PMCR[2]	400K ~ 32MHz OSC mode enabled, 1 enabled, 0 off
PMCR[3]	32K ~ 400K OSC mode enabled, 1 enabled, 0 off

After the LGT8FX8P system is powered on, 32MHz RC is used as the system clock source by default, and the core operates at 1/8 of the clock source (4MHz). The user can change the default configuration by setting the PMCR register and the system prescaler register (CLKPR).

If the user needs to change the configuration of the main clock source, you need to ensure that the clock source after switching is in a stable working state before switching the clock. Therefore, before switching the master clock source, the required clock source needs to be enabled by PMCR [3:0], and wait until the clock stabilizes before switching.

When the user switches the master clock to the external crystal oscillator, although the user enables the external crystal oscillator, it cannot be ruled out that the crystal oscillator cannot start due to configuration errors or crystal oscillator failure. If you switch to an external crystal at this point, the system will stop working after switching. Therefore, from the perspective of system reliability, it is recommended to turn on the watchdog timer to avoid such problems from the perspective of software design.

After the clock source is enabled and waiting for stabilization, the master clock can be switched via PMCR [6:5]. Among them, PMCR [5] is used to select internal RC oscillators and external crystal oscillators, and PMCR [6] is used to select high-speed clock sources and low-speed clock sources.



Master clock source selection：

PMCR[6]	PMCR[5]	Master clock source
0	0	Internal 32MHz RC Oscillator (System Default)
0	1	External 400K ~ 32MHz high-speed crystal oscillator
1	0	Internal 32KHz RC oscillator
1	1	External 32K ~ 400KHz low speed crystal oscillator

The clock source controls the timing

To protect the PMCR register from accidental modifications, modifications to the PMCR registers require strict installation of the specified timing.

The highest bit of the PMCR register (PMCR[7]) is used for sequencing. Before modifying other PMCR bits, the user must first set PMCR[7] to 1 and change the values of the other PMCR registers within 6 cycles after the 1 operation. After 6 cycles, direct modifications to the PMCR will become invalid.

The following takes switching to an external high-speed crystal as an example to list the recommended operation steps:

(1) Enable the clock source

- ❑ Set PMCR[7] = 1
- ❑ Over six cycles, set PMCR[2] = 1 to enable external high-speed mode external crystals
- ❑ Wait for the external crystal oscillator to stabilize (the waiting time varies from the crystal oscillator to the different crystal oscillators, generally the US level can wait)

(2) Switch the master clock source

- ❑ Set PMCR[7] = 1
- ❑ Set PMCR[6:5] = 01 within six cycles, and the system automatically switches the working clock to an external crystal oscillator
- ❑ Perform several NOP operations to improve stability (optional operation)

[Note]: In the above operation of switching the master clock, to ensure that the current system clock works normally, the previous internal RC oscillator can be turned off after switching to the external crystal oscillator.

System clock prescaled control

The LGT8FX8P has a system clock prescaler inside, which can be controlled via the clock prescaler register (CLKPR).
This feature can be used to reduce system power consumption when the system does not require very high processing power. The prescale-off settings are valid for all supported clock sources in the system. Clock predivision can affect the core execution clock and therefore synchronize peripherals.
When switching between different clock prescaled settings, the system clock prescales ensure that there are no glitches during the switching, and there is no intermediate state of excessive high frequencies. The crossover is performed immediately, and when the register change takes effect, the system clock switches to a new crossover clock after up to 2~3 current system clock cycles.

To avoid misoperation of the clock division registers, modifications to CLKPR must also follow a special timing process:

- ❑ Set the clock prescale-off change enable bit (CLKPCE) to 1 and the CLKPR other so bit to 0
- ❑ Within four cycles, the required values are written to CLKPS and CLKPCE is written to 0

Before changing the clock predivision register, the interrupts must be disabled to ensure that the write sequence can be performed correctly.
For a specific definition of the master clock predivision register CLKPR, refer to the Register Description section of this chapter.

Internal RC oscillator calibration

The LGT8FX8P contains two calibrated RC oscillators, both calibrated to within ±1% accuracy. that
The medium 32MHz RC is used by default for the system operating clock.
Before the LGT8FX8P was shipped, both the internal 32MHz HFRC and 32KHz LFRC were calibrated and the calibration values were written to the system configuration information area. During system power saving, these calibration values are read into internal registers and implemented through registers
The RC frequency is now recalibrated.
The calibration register is located in the IO address space and can be read and written by the user program. For applications with special frequency requirements, the frequency output of the internal oscillator can be adjusted by modifying the calibration register. Modifying the calibration register does not change the factory configuration letter
In the event of a system power-on or a user-initiated configuration bit reload, the calibration register will be restored to factory settings.

Register definitions

32MHz HFRC Oscillator Calibration Register - RCMCAL

RCMCAL – 32MHz HFRC calibration register		
RCMCAL: 0x66	Default value: Factory configuration	
Bits	RCCAL[7:0]	
R/W	R/W	
Bit definition		
[7:0]	RCCAL	When the system is powered on, the value of the register is replaced by the RC calibration value in the system configuration information.

32KHz RC Oscillator Calibration Register - RCKCAL

RCKCAL – 32MHz RC calibration register		
RCKCAL: 0x67	Default value: Factory setting	
Bits	RCKCAL[7:0]	
R/W	R/W	
Bit definition		
[7:0]	RCKCAL	Writing the calibration value to the RCKCAL register completes the calibration of the 32KHz RC oscillator

Clock source management register - PMCR

PMCR – Clock Source Management Register							
PMCR: 0xF2			Default value: 0x03				
Bits	PMCE	CLKFS/CLKSS	WCLKS	OSCKEN	OSCMEN	RCKEN	RCMEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition							

[0]	RCMEN	Internal 32MHz RC oscillator enable control, 1 enabled, 0 disabled
[1]	RCKEN	Internal 32KHz RC oscillator enable control, 1 enabled, 0 disabled
[2]	OSCMEN	External high-frequency crystal enable control, 1 enabled, 0 disabled
[3]	OSCKEN	External low-frequency crystal oscillator enable control, 1 enabled, 0 disabled
[4]	WCLKS	WDT clock source selection, 0 – Select divide-by-16 of the internal 32MHz HFRC oscillator 1 – Internal 32KHz LFRC oscillator
[5]	CLKSS	Master clock source selection control, select the clock source type, refer to the clock source selection section
[6]	CLKFS	For master clock source frequency control, select the clock frequency type, please refer to the clock source selection section
[7]	PMCE	The PMCR register changes the enable control bit. Before changing the PMCR other position, this bit must first be set, and then the values of the other bits must be set within four cycles.

Master clock prescaler register - CLKPR

CLKPR – Master clock prescaler register								
CLKPR: 0x61				Default value: 0x03				
Bits	WCE	CKOEN1	CKOEN0	-	PS3	PS2	PS1	PS0
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Bit definition								

[3:0]	CLKPS	Clock prescale-off select bits				
		PS3	PS2	PS1	PS0	Prescale factor
		0	0	0	0	1
		0	0	0	1	2
		0	0	1	0	4
		0	0	1	1	8 (default)

		0	1	0	0	16
		0	1	0	1	32
		0	1	1	0	64
		0	1	1	1	128
		1	0	0	0	256
		Other values				Reserved
[4]	-	Reserved				
[5]	CKOEN0	Sets whether the system clock is output on the PB0 pin				
[6]	CKOEN1	Sets whether the system clock is output on the PE5 pin				
[7]	WCE	Clock prescaler changes clock control Before changing the other bits of the CLKPR register, the CKWEN must be set individually to 1, and then the other bits must be set over the next four system cycles. At the end of four cycles, CKWEN automatically clears zero.				

Power management

overview

Sleep mode reduces system power consumption by turning off the system clock and clock module. The LGT8FX8P offers a very flexible sleep mode and module controller, allowing users to achieve the most ideal low-power configuration depending on the application.

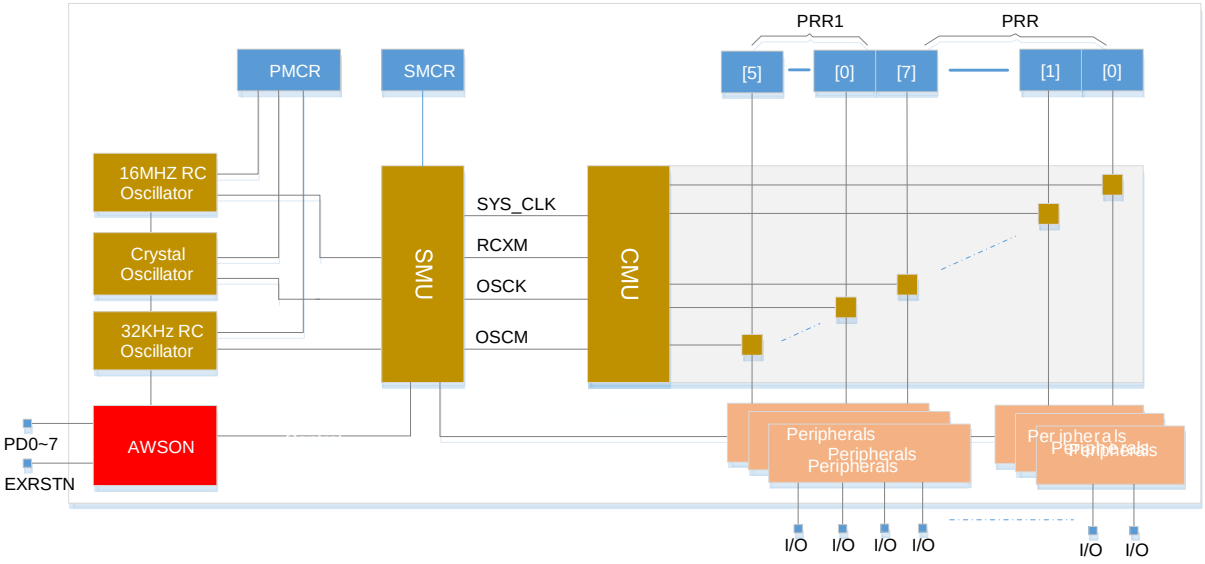
LGT8FX8P does not automatically turn off analog function blocks such as ADCs, DACs, comparators (AC), low voltage reset modules (LVD), etc. when entering sleep mode, and the software needs to turn off unwanted analog functions before entering sleep according to application requirements, and restore the correct state after the system wakes up.

The LGT8FX8P supports several sleep modes, including ADC-specific noise cancellation mode to eliminate interference from the digital portion of the ADC power supply during ADC conversion. In addition, the others are power control modes, which are divided into five types:

Sleep mode	Function description
Idle Mode (IDLE)	By simply turning off the core clock, other peripheral modules operate normally, and all valid interrupt sources can wake up the kernel
Power saving mode (Save)	Like DPS0 mode, Save mode is compatible with LGT8FX8D
Power-down mode (DPS0)	As with Save mode, supported wake sources include: ❑ All pin level changes ❑ Watchdog wakes up regularly ❑ TMR2 wake-up in asynchronous mode
Power-down mode (DPS1)	Turn off all internal and external oscillators, supported wake-up sources include: □ External level changes in all pins □ External interrupt 0/1 □ Watchdog timer working on 32K LFRC
Power-down mode (DPS2)	Power off the core, lowest power mode, supported wake sources: □ External reset □ PORTD pin level change □ LPRC timed wake-up (128ms/256ms/512ms/1s) It is important to note that the process of waking up from DPS2 is the same as power-on reset

LGT8FX8P supports deep sleep DPS2, in this mode, the internal LDO of the system is power-down, the core registers, all peripheral controllers, and SRAM are power-down state, and the data in it will not be maintained. The FLASH memory cell will also be powered-down, so DPS2 mode can achieve the minimum power consumption of the system. Power-down mode can be woken up via a level change in the Port D (PORTD) pin, or a 5-level timed wake-up can be selected. The DPS2 timer for wake-up is only suitable for timed wake-up applications with low accuracy because it does not support calibration and has an accuracy of about 15%.
When the system wakes up from DPS2 mode, the LDO will be turned on first, and the process is the same as the power-up process. The chip performs a complete power-on reset startup process, loads configuration information, and then runs the program from the address pointed to by the reset vector.
In modes other than DPS2, the internal power is not turned off, and all register information as well as RAM data is not lost during sleep. After waking up, the kernel resumes execution from the last instruction before sleeping.

System power management diagram:



As shown in the figure above, the LGT8FX8P mainly controls the power consumption of the entire system through the sleep mode controller (SMU) and the clock management unit (CMU). From the level of power saving, we can divide the power consumption into 4 levels: The first stage is to control the module working clock through the PRR register, and save the dynamic power consumption of the system operation by turning off the clock that is not using the module. In general, the power savings at this level are not significant.

The second stage is to switch the main clock source to the low-frequency clock and turn off the unused clock source module and other analog modules, which basically results in a very significant system operating power consumption and sleep power consumption.

The third level is to enter the system into power-down mode (DPS1), in DPS1 mode LGT8FX8P can obtain polar standby power consumption, after waking up from power-down mode, the software can read the state before reset through the MCUSR register.

The fourth level is power-down mode (DPS2), which powers down the core for minimal system power consumption. Because the kernel is powered off, all data information will be lost in this mode. Immediately after wake-up, a power-on reset process is performed, and the system restarts running from the reset vector.

AWSON Power Management

Compared to the LGT8FX8D, the power-down mode DPS2 is a new power consumption mode. DPS2 mode is used in applications where sleep power consumption is critical. After entering DPS2 mode, the system keeps only one static module (AWSON) in the working state, and all other circuits are in a completely powered down state.

The AWSON module is dedicated to sleep and wake-up control in DPS2 mode, and the AWSON module mainly consists of IO wake-up control logic and a low-power LPRC. The software can control AWSON through IOCWK registers as well as DPS2R registers.

The IOCWK register is used to control the wake-up function of PD0~7 level change. The DPS2R register is used to control the DPS2 mode as well as the functional mode of the LPRC. For more information, refer to the Register Definitions section at the end of this section.

Before using DPS2 mode, the software sets the IOCWK to enable the required wake-up IO, or enables LPRC through the DPS2R register and configures the timed wake-up cycle, and then enables DPS2 mode through the DPS2EN bit of the DPS2R register. After the setting is completed, the software needs to set the DPS2 sleep mode through the SMCR register, and then execute the SLEEP instruction to enter sleep.

Sleep mode vs. wake source

LGT8FX8P supports 5 sleep modes, and users can choose the appropriate sleep mode according to application requirements. The SMCR register contains the sleep mode control setting, and after executing the SLEEP instruction, the kernel enters sleep mode. For optimal sleep power, it is recommended to turn off all unused clocks and analog modules before the core enters sleep mode. However, it should be noted that the generation of some wake-up sources requires a working clock, and if such wake-up sources are required, please keep the relevant clock source in working condition.
Sleep mode and wake-up mode:

Sleep mode	Valid clock				Wake source							
	Core clock	Peripheral clock	ADC clock	Asynchronous clock	Pin level changes	External interrupt 0/1	TWI address matching	TMR2 interrupt	ADC conversion complete	Watchdog timeout	Peripheral interrupts	PD level changes
Idle Mode (IDLE)		X	X	X	X	X	X	X	X	X	X	X
ADC noise suppression			X	X	X	X	X	X	X	X		X
Power Save Mode (SAVE)				X	X	X	X	X		X		X
Power-down mode (DPS0) (With RC32K)				X	X	X		X		X		X
Power-down mode (DPS1) (Without RC32K)				X	X	X		X				X
Power-down mode (DPS2) (Without LDO)												X

If the above five sleep modes need to be entered, the SE bit in the SMCR must be set to 1 to enable sleep mode control. Then execute a SLEEP command. SM0/1/2 in SMCR is used to select different sleep modes. Please refer to the description below for specific information.
In sleep mode of the MCU, if the wake-up source is active, the MCU will be woken up after 4 cycles to continue executing instructions. If the interrupt remains valid, the interrupt also responds immediately to the interrupt service subroutine. If a system reset occurs in SLEEP mode, the MCU will also be woken up and executed from the reset vector.
When the MCU is in Power/Off mode, the system can wake up by an external interrupt INT0/1, and the MCU will resume execution from the position before sleep after wake-up.

Idle Mode (IDLE)

When SM2... 0 is set to 000, after executing the SLEEP command, the MCU enters the IDLE mode, which will turn off the core working clock, and other peripherals can work normally.
The IDLE mode can be woken up by external interrupts as well as internal interrupts, etc. If you do not need to use the comparator and ADC as a wake-up source, it is recommended to turn it off.
IDLE mode does not result in significant power consumption reduction because it only turns off the clock on which the core runs. In IDLE mode, the kernel will also stop executing and fetching instructions, so the power consumption of the internal program FLASH can be reduced.
However, IDLE mode has a more flexible wake-up method, and users can obtain better operating power consumption by reducing the system master clock and turning off unwanted modules.

ADC noise suppression mode

When SM2... 0 is set to 001, and after executing the SLEEP instruction, the MCU enters ADC noise suppression mode. In this mode, the core and most peripherals will stop operating, and the ADC, external interrupts, TWI address matching, WDT, and timer/counter 2 operating in asynchronous clock mode will all work.
The ADC noise constant mode is mainly used to provide a good operating environment for ADC conversion. Reduce high-frequency interference from digital modules to analog conversion. After entering this mode, the ADC will automatically initiate the sampling conversion, and after the converted data is saved to the ADC data register, the ADC conversion end interrupt wakes the MCU from the ADC noise mode.

Power saving mode (Save)

When SM2... 0 is set to 010, and after executing the SLEEP command, the MCU enters Save mode. In this mode, the system turns off the working clocks of all modules. This mode can only be woken up by asynchronous mode because the working clock of all modules is turned off, and wake-up signals in this mode can be generated by external interrupts, TWI address matching, and WDT operating in independent clock source mode.
This mode shuts down all modules except the master clock source. For optimal operating power consumption, it is recommended to switch the system master clock to an internal 32K RC or an external 32KHz low-frequency crystal before entering this mode, and then turn off the clock source and analog module that are not in use.

Power-down mode DPS0

When SM [2:0] is set to 110, after executing the SLEEP command, the MCU will enter DPS0 mode. After entering DPS0, all clock sources except the internal 32KHz RC are turned off. This mode can be woken up by an external interrupt INT0/1; If the interrupt function of the WDT is enabled, timed wake-up can also be implemented through the WDT.

Power-down mode DPS1

When SM [2:0] is set to 011, after executing the SLEEP command, the MCU will enter DPS1 mode. When DPS1 is entered, all clock sources in the system are turned off. This mode can use the level change of IO and the watchdog wakes up.

Power-down mode DPS2

Set SM [2:0] to 111 and enable the AWSON module through DPS2EN of the DPSR2 register, and enter DPS2 mode after executing the SLEEP instruction. After entering DPS2 mode, the system powers off the kernel. So registers and RAM data will be lost. The wake-up process from DSP2 is the same as the power-on reset process.
In DPS2 mode, register information is lost due to the core voltage being turned off, so the control state of the port will all return to the input state, and the output drive and pull-up control of all IOs will also be turned off.

FLASH power control and fast wake-up

When the system is in SLEEP mode, the kernel will not continue to execute instructions, at which point you can choose to power down FLASH for lower standby power consumption. This function can be achieved through FPDEN bit control of the MCUCR register;
In power-down mode, the system can use external interrupt or WDT wake-up, in order to filter out possible interference from the external signal, the internal wake-up circuit contains a configurable filter circuit, the user can choose the appropriate filter width according to the user. The configuration of the filter circuit can be implemented by the FWKPEN of the MCUCR register.

MCUCR [FWKPEN] filter width control:

FWKPEN	Filter width
0	260us (default)
1	32us

Register description

Sleep mode control register - SMCR

SMCR – Sleep Mode Control Register						
SMCR: 0x33(0x53)			Default value: 0x00			
Bits			SM2	SM1	SM0	SE
R/W	-		R/W	R/W	R/W	R/W
Bit definition						
[0]	SE	Sleep mode enables the control bit, after setting it to 1, execute the SLEEP instruction, and the kernel will enter sleep mode. The SE bit protects the system from accidentally entering sleep mode. After waking up, it is recommended to clear the SE bit immediately.				
[3:1]	SM	Sleep mode selection				
		SM2	SM1	SM0	Mode description	
		0	0	0	IDLE mode	
		0	0	1	ADC Noise suppression mode	
		0	1	0	Save mode	
		0	1	1	DPS1 mode	
		1	1	0	DPS0 mode	
		1	1	1	DPS2 mode	
		Others			Reserved	
[7:4]	-	Reserved				

Power-saving control register - PRR

PRR – Power-saving control register								
PRR: 0x64					Default value: 0x00			
PRR	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUART0	PRADC
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Bit definition								
[0]	PRADC	Set to 1 to turn off the ADC controller clock						
[1]	PRUART0	Set to 1 to turn off the clock for the USART0 module						
[2]	PRSPI	Set to 1 to turn off the clock of the SPI module						
[3]	PRTIM1	Set to 1 to turn off the clock for timer/counter 1						
-	-	Reserved						
[5]	PRTIM0	Set to 1 to turn off the clock for timer/counter 0						
[6]	PRTIM2	Set to 1 to turn off the clock for timer/counter 2						
[7]	PRTWI	Set to 1 to turn off the clock of the TWI module						

Power-saving control register - PRR1

PRR1 – Power-saving control register 1								
PRR1: 0x65					Default value: 0x00			
PRR1			PRWDT	-	PRTIM3	PREFL	PRPCI	-
R/W			R/W	-	R/W	R/W	R/W	-
Bit definition								
[0]	-	Reserved						
[1]	PRPCI	Set to 1 to turn off external pin changes and external interrupt module clocks						
[2]	PREFL	Set to 1 to turn off the FLASH controller interface clock						
[3]	PRTIM3	Set to 1 to turn off the clock of the TMR3 controller						
[4]	-	Reserved						
[5]	PRWDT	Set to 1 to turn off the WDT counter clock						
[7:6]	-	Reserved						

MCU control register - MCUCR

MCUCR – MCU control register								
MCUCR: 0x35(0x55)				Default value: 0x00				
MCUCR	FWKEN	FPDEN	EXRFD	PUD	IRLD	IFAIL	IVSEL	WCE
R/W	R/W	R/W	R/W	R/W	W/O	R/O	R/W	R/W
Bit definition								
[0]	WCE	The MCUCR update enable bit, which needs to be set before updating the MCUCR, and then complete the update of the MCUCR register within 6 cycles						
[1]	IVSEL	The interrupt vector selects bits, after this position 1, the interrupt vector address will be mapped to the new address based on the value of the IVBASE register						
[2]	IFAIL	System configuration bit load failure flag bit, 0 = Configuration information validation passed 1 = Configuration information failed to load						
[3]	IRLD	Writing 1 will reload the system configuration information						
[4]	PUD	Global pull-up disable bits 0 = Global pull-up control enabled 1 = Turn off pull-up resistors for all IOs						
[5]	EXRFD	External reset filter disable bits 0 = (190us) digital filter with external reset enabled 1 = Digital filter circuit with external reset disabled						
[6]	FPDEN	Flash Power/down enable control 0: FLASH remains powered on after the system sleeps 1: FLASH power off after system SLEEP						
[7]	FWKEN	Fast wake-up mode enables control and is only valid for Power/Off mode 0:260us filter delay 1:32us filter delay						

PD bank level change wake-up control register - IOCWK

IOCWK – PD bank level change wake-up control register								
IOCWK: 0xAE					Default value: 0x00			
Bits	IOCD7	IOCD6	IOCD5	IOCD4	IOCD3	IOCD2	IOCD1	IOCD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	IOCWK	Set the bit corresponding to 1 to enable the wake-up function of the pin level change of the PD group IO						

DPS2 Mode Control Register - DPS2R

DPS2R – DPS2 mode control register								
DPS2R: 0xAF					Default value: 0x00			
Bits	-	-	-	-	DPS2E	LPRCE	TOS1	TOS0
R/W	-	-	-	-	R/W	R/W	R/W	R/W
Bit definition								
[1:0]	TOS	LPRC timed wake-up settings: 00 = 128ms 01 = 256ms 10 = 512ms 11 = 1s						
[2]	LPRCE	LPRC enable control 0 = LPRC timer disabled 1 = LPRC timer enabled						
[3]	DPS2E	DPS2 mode enables control bits 0 = DPS2 mode disabled 1 = DPS2 mode enabled						
[7:4]	-	Reserved						

System control and reset

Overview

After the system is reset, all I/O registers are set to their initial values, and the program executes from the reset vector. The interrupt vector address of the LGT8FX8P must be jumped to the reset handler with an RJMP – Relative Jump command. If the program does not use interrupts, without enabling the interrupt source, the interrupt vector will not be used, and the interrupt vector area can be used to store the user's program code.

As soon as the reset is valid, all I/O ports enter their initial state. Most I/Os are initialized to the input state and turn off the internal pull-up resistors. I/O with analog input function is also initialized to digital I/O function.

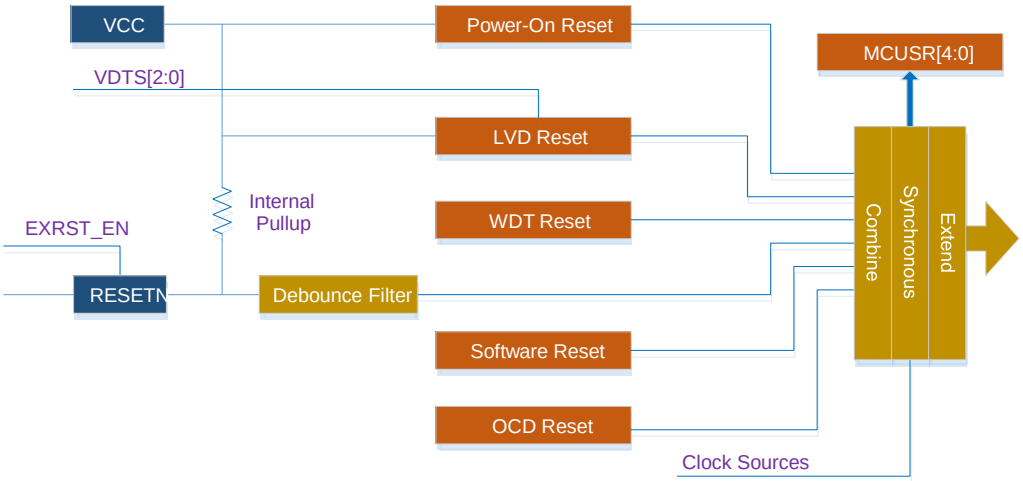
When the reset becomes invalid, the timing counter inside the LGT8FX8P starts to start to widen the reset. Broadening the width of the reset signal is used to ensure that the power supply and clock modules in the system enter a stable state.

Reset sources

The LGT8FX8P supports a total of six reset sources:

- **Power-on reset**: When the system's working voltage is low voltage and the reset threshold of the internal POR module, the power-on reset is effective.
- **External reset**: An external reset is valid for a low-level pulse of a certain width on the external reset pin of the chip.
- **Watchdog reset**: After enabling the watchdog module, if the watchdog timer expires, the system will reset.
- **Low voltage reset**: LGT8FX8P has a low voltage detection module (LVD) inside, and the MCU will also be reset when the system operating power supply is below the reset threshold set by LVD.
- **Software reset**: LGT8FX8P has a dedicated software-triggered reset register through which the user can reset the MCU at any time.
- **OCD reset**: OCD reset is issued by a debugger module to directly reset the MCU core.

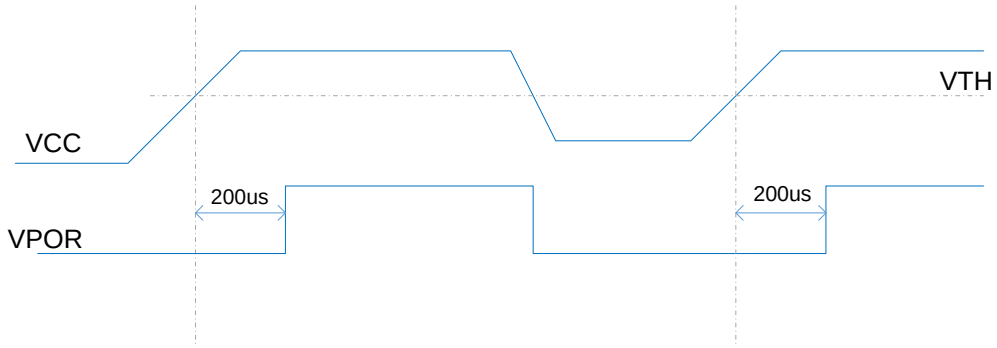
Reset system structure diagram:



Power-on reset

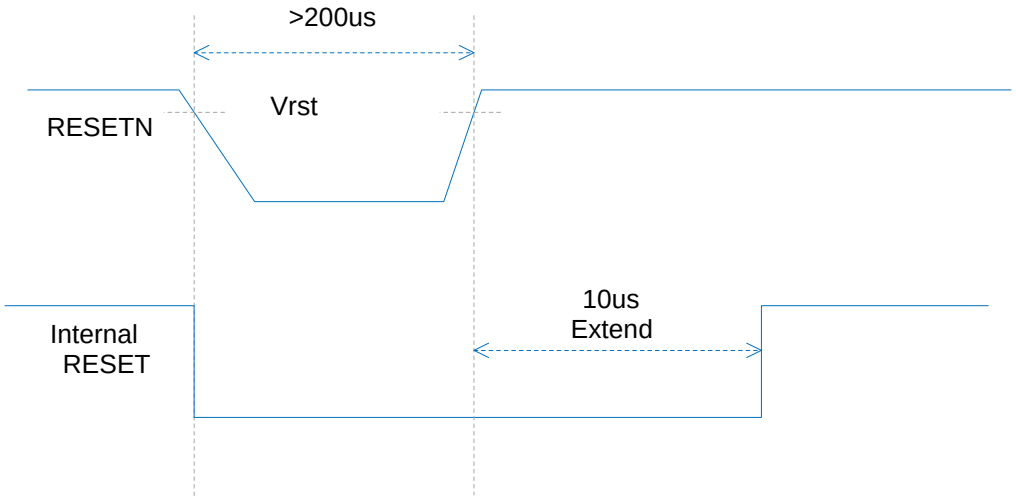
The power-on reset signal is generated by internal voltage sensing circuitry. When the system power supply (VCC) falls below the sense threshold, the power-on reset signal is valid. For the detection threshold value of power-on reset, please refer to the Electrical Parameters section.

The power-on reset circuit ensures that the chip is in a reset state during power-up, and the chip can start operation from a known stable state after power-up. The power-on reset signal will also be broadened by the counter inside the chip to ensure that various analog modules inside after power-up, such as RC oscillators, can enter a stable working state.



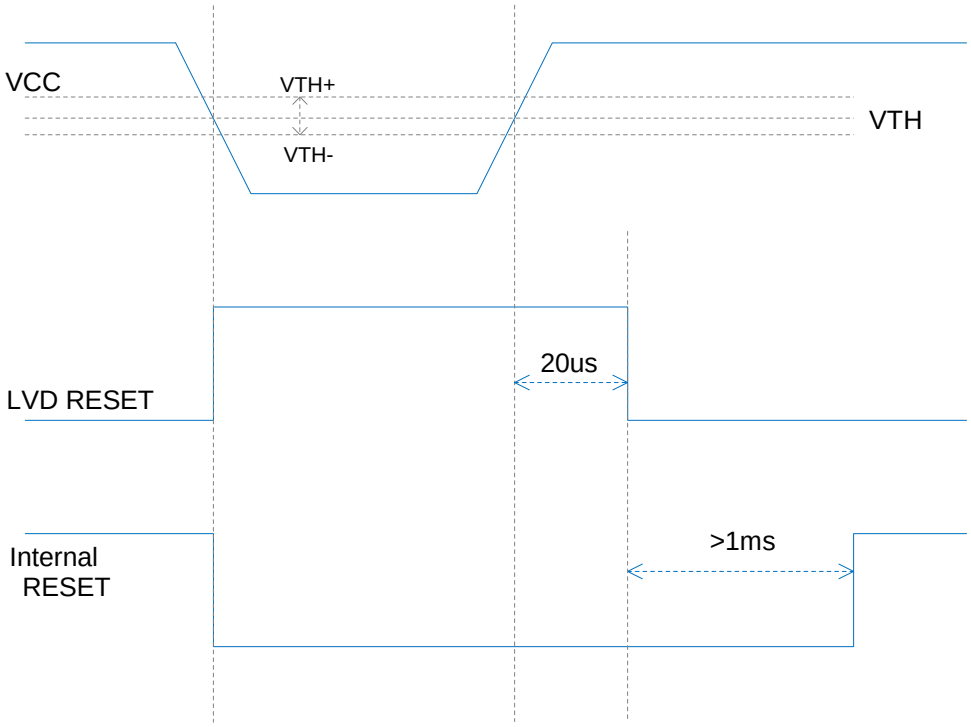
External reset

Apply a low level on the external reset pin (RSTN) and the external reset is effective immediately. The width of the low level should be greater than the width of a minimum reset pulse. An external reset is an asynchronous reset that resets the chip even when the chip is not clocked. The LGT8FX8P's external reset pin can also be used as a general-purpose I/O. After the chip is powered on, it defaults to an external reset function. The external reset function of this pin can be turned off by the user through register configuration, allowing it to be used as normal I/O. For details, please refer to the description section of IOCR registers.



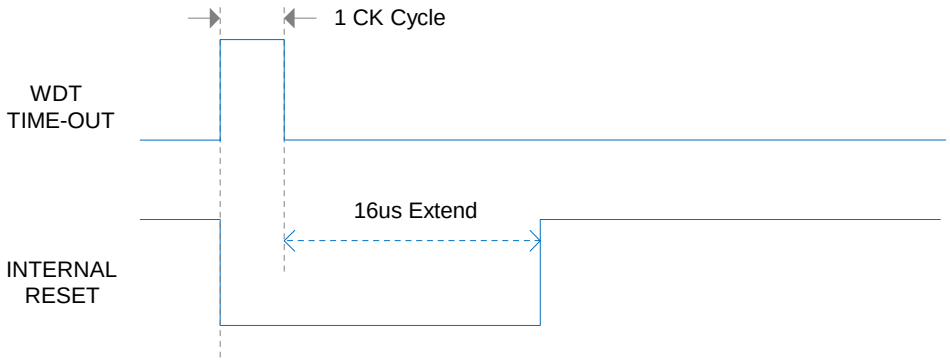
Low voltage detection (LVD) reset

The LGT8FX8P contains a programmable low-voltage detection (LVD) circuit. LVD also detects the voltage change of VCC, but unlike power-on reset, LVD can choose the threshold value of the detection voltage. The user can choose between different voltage thresholds by operating the VDTCR register directly. LVD's voltage detection circuit has a hysteresis of ±10mV~±50mV to filter out the jitter of VCC voltage. When LVD is enabled, LVD reset will be effective immediately if the voltage of VCC drops to the set reset threshold. When VCC increases above the reset threshold, the internal reset expansion circuit activates and continues to widen the reset for at least 1 millisecond.



The watchdog resets

When the watchdog timer overflows, if the watchdog system reset function is enabled, a periodic system reset signal will be generated immediately. The watchdog reset signal is also broadened by the internal delay counter. For more information on the operation of the watchdog controller, see the detailed description section below.



Software reset, OCD reset

The software reset is trggered by the user by manipulating the sixth bit of the VDTCR register, and the timing of the software reset is exactly similar to the watchdog reset. The reset signal is internally spread by 16us.

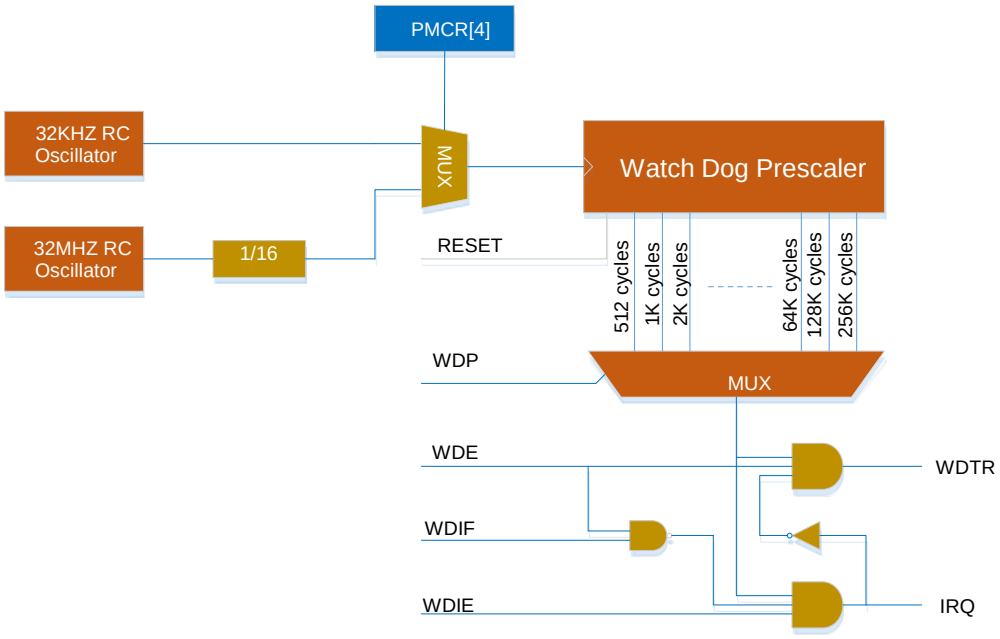
OCD reset is generated by the debugger unit inside the chip, OCD reset is generally controlled by the debugger, and the user software cannot trigger OCD reset.

Watchdog timer

- Clock selectable internal 32KHz RC or internal 32MHz RC divide-by-16 (2MHz)
- Support interrupt mode, reset mode and reset interrupt mode
- The timer timeout can be up to 8 seconds

The LGT8FX8P contains an enhanced watchdog timer (WDT) module. The WDT timer can operate with either an internal 32KHz RC oscillator or a divide-by-16 of an internal 32MHz RC oscillator. When the WDT counter is overflowed, it can output an interrupt or a system reset signal. In normal use, software is required to execute a WDR – watchdog timer reset command to restart the counter before overflowing. If the system does not execute WDR instructions even after execution, WDT will generate an interrupt or system reset.

The structure diagram of the watchdog timer is shown in the following figure:



In interrupt mode, an interrupt request signal is generated after the WDT overflows. This interrupt can be used as a wake-up signal for sleep mode, or as a general system timer. For example, you can use this interrupt to limit the execution time of an operation and terminate the current task in an overflow. In system reset mode, the WDT generates a system reset signal immediately after the counter overflows. The most typical use is to prevent the system from freezing or running away.
The third mode, the reset interrupt mode, combines both interrupt and reset functions. First, the system will respond to the WDT interrupt function, exit the WDT interrupt reset procedure, and immediately switch to reset mode. This function can support the storage of some critical parameter information before reset.

To prevent the WDT from being accidentally disabled, the operation to shut down the WDT must be performed in a strictly defined timing. The following code describes how to turn off the watchdog timer. The following example assumes that interrupts have been disabled so that the entire operational process is not interrupted.

Sample code for watchdog enable and shutdown operations:

Assembly code

```
WDT_OFF:
   ; Turn off global interrupt
   CLI
   ; Reset watchdog timer
   WDR
   ; Clear WDRF in MCUSR
   IN r16, MCUSR
   ANDI r16, ~(1 << WDRF)
   OUT MCUSR, r16
   ; Write logical one to WDCE and WDE
   ; Keep old Prescaler setting to prevent unintentional time-out
   LDS r16, WDTCSR
   ORI r16, (1 << WDCE) | (1 << WDE)
   STS WDTCSR, r16
   ; Turn off WDT
   LDI r16, (0 << WDE)
   STS WDTCSR, r16
   ; Turn on global interrupt
   SEI
   RET
```

C code

```
void WDT_OFF(void)
{
   __disable_interrupt();
   __watchdog_reset();
   /* Clear WDRF in MCUSR */
   MCUSR &= ~(1 << WDRF);
   /* Write logical one to WDCE and WDE */
   /* Keep old Prescaler setting to prevent unintentional time-out */
   WDTCSR |= (1 << WDCE) | (1 << WDE);
   /* Turn off WDT */
   WDTCSR = 0x00;
   __enable_interrupt();
}
```

[Tips for use]

If WDT is accidentally enabled, such as a program running away, the chip will be reset, but WDT is still enabled. If WDT is not handled in the user code, this will cause a loop reset. To avoid this, it is recommended that the user software clear the watchdog reset flag bit (WDRF) and the WDE control bit in the initialization program.

The following code describes how to change the watchdog timer timeout value.

Assembly code

```
WDT_TOV_Change:
   ; Turn off global interrupt
   CLI
   ; Reset watchdog timer
   WDR
   ; Start timed sequence
   LDS r16, WDTCSR
   ORI r16, (1 << WDCE) | (1 << WDE)
   STS WDTCSR, r16
   ; -- Got for cycles to set the new value from here --
   ; Set new time-out value = 64k cycles
   LDI r16, (1 << WDE) | (1 << WDP2) | (1 << WDP0)
   STS WDTCSR, r16
   ; -- Finished setting new value, used 2 cycles -
   ; Turn on global interrupt
   SEI
   RET
```

C code

```
void WDT_TOV_Change(void)
{
   __disable_interrupt();
   __watchdog_reset();
   /* Start timed sequence */
   WDTCSR |= (1 << WDCE) | (1 << WDE);
   /* Set new time-out value = 64K cycles */
   WDTCSR |= (1 << WDE) | (1 << WDP2) | (1 << WDP0);
   __enable_interrupt();
}
```

Instructions for use】

Before changing the WDP configuration bits, it is recommended to reset the watchdog timer. Because changing the WDP bit to a relatively small timeout period is likely to cause the watchdog to reset timeout.

Register definitions

Low Voltage Detection (LVD) Control Register - VDTCR

VDTCR – LVD control register								
VDTCR: 0x62				Default value: 0x00				
Bits	WCE	SWR	-	VDTS2	VDTS1	VDTS0	VDREN	VDTEN
R/W	R/W	W/R	-	R/W	R/W	R/W	R/W	R/W

Bit definition		
[0]	VDTEN	Low voltage detection module enable control, 1 enabled, 0 disabled
[1]	VDREN	Low voltage reset function enable control, 1 enabled, 0 disabled
[4:2]	VDTS	Low pressure detection threshold configuration bit ✓ 000 = 1.8V ✓ 001 = 2.2V ✓ 010 = 2.5V ✓ 011 = 2.9V ✓ 100 = 3.2V ✓ 101 = 3.6V ✓ 110 = 4.0V ✓ 111 = 4.4V
[5]	-	Reserved
[6]	SWR	Soft reset enable bit, which clears zero will result in a software reset
[7]	WCE	VDTCR Value Change Enable Bit Before changing the value of the VDTCR register, the user must first write this bit to 1, and change the value of the other VDTCR bits for the next 6 clock cycles. After four cycles, the WCE automatically clears zero, and the update operation on the VDTCR register is invalid.

IO function multiplexing register - PMX2

PMX2 – IO function multiplexing register								
PMX2: 0xF0				Default value: 0x00				
Bits	WCE	STSC1	STSC0	-	-	XIEN	E6EN	C6EN
R/W	R/W	R/W	R/W	-	-	R/W	R/W	R/W

Bit definition		
0	C6EN	The PC6 pin defaults to the reset function, setting this bit to 1 will disable the external reset function, and after the reset function is disabled, PC6 can be used as a normal I/O
1	E6EN	The PE6 pin defaults to the analog input function, setting this bit to 1 will turn off the analog input function, and this pin can be used as a GPIO
2	XIEN	External clock input enable control
4:3	-	Reserved
5	STSC0	Low-speed crystal oscillator start-up control
6	STSC1	High-speed crystal start-up control
7	WCE	IOCR Value Changes Enable Bit Before changing the value of the IOCR register, the user must first write this bit to 1, and change the value of the other IOCR

		bits for the next 6 clock cycles. After four cycles, the WCE automatically clears zero, and the update operation on the IOCR register is invalid.

MCU Status Register - MCUSR

MCUSR – IO special function control register								
MCUSR: 0x34(0x54)				Default value: 0x00				
Bits	SWDD	-	PDRF	OCDRF	WDRF	BORF	EXTRF	PORF
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W

Bit definition		
[0]	PORF	Power-on reset flag, write 0 to clear
[1]	EXTRF	External reset flag, power-on reset automatically clears, or write 0 to clear
[2]	BORF	Low voltage detection reset, power-on reset automatically clears, or write 0 to clear
[3]	WDRF	The watchdog reset flag, the power-on reset automatically clears, or writes 0 to clear zero
[4]	OCDRF	OCD debugger reset flag, power-on reset automatic zero, or write 0 clear
[5]	PDRF	Wake up flag from Power/off mode, please refer to the Power Management chapter for details.
[6]	-	Reserved
[7]	SWDD	SWD interface disable bits. Writing 1 will shut down the SWD interface. Once the SWD interface is down, debugging and ISP operation are not possible. If the SWD interface is closed in the user program, the operation of the internal program can be disabled by pulling RESET low during power-up, and then debugging and ISP operation can be performed. When the SWD interface is closed, the two I/O interfaces occupied by SWD can be used as general-purpose I/O. In order to avoid misoperation of SWDD, the user needs to write SWDD again within four cycles after the first update of the SWDD bit to take effect.

[Tips for use]:

In order to use the reset flag information more accurately and effectively, it is recommended that the user try to read the reset flag before the initialization of the program and then clear it.

Watchdog control status register - WDTCSR

WDTCSR – WDT control and status register								
address: 0x60					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	WDIF	WDIE	WDP3	WDTOE	WDE	WDP2	WDP1	WDP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
[7]	WDIF	WDT interrupt flag bit. When the WDT operates in interrupt mode and overflows, the WDIF bit is asserted. WDT interrupts occur when the WDT interrupt enable bit WDIE is "1" and the global interrupt is asserted. The WDIF bit is cleared when a WDT interrupt is executed, and it can also be cleared by writing "1" to the WDIF bit.
[6]	WDIE	WDT interrupt enable control bits. WDT interrupts are enabled when the WDIE bit is set to "1" and the global interrupt is asserted.

<table>
<tr><td></td><td></td><td>When the WDIE bit is set to "0", WDT interrupts are disabled.
The WDIE bit and the WDE bit together determine the watchdog's operating mode, as shown in the following table.
<table>
<tr><th>WDE</th><th>WDIE</th><th>mode</th><th>Actions</th></tr>
<tr><td>0</td><td>0</td><td>Halt mode</td><td>None</td></tr>
<tr><td>0</td><td>1</td><td>Interrupt mode</td><td>Interrupt</td></tr>
<tr><td>1</td><td>0</td><td>Reset mode</td><td>Reset</td></tr>
<tr><td>1</td><td>1</td><td>Int. reset mode</td><td>Interrupt+Reset</td></tr>
</table></td></tr>
<tr><td>[5]</td><td>WDP3</td><td>WDT pre-divider selection controls bit 3.
WDP [3] and WDP [2:0] form the WDT prescaler factor selection bit WDP [3:0], which is used to set the overflow period of the WDT.</td></tr>
<tr><td>[4]</td><td>WDTOE</td><td>WDT turns off the enable control bit.
When the WDE bit is to be cleared, the WDTOE bit must be set, otherwise the WDT will not be turned off.
When the WDTOE bit is asserted, the hardware zeros out the WDTOE bit after 4 clock cycles.</td></tr>
<tr><td>[3]</td><td>WDE</td><td>WDT enable control bit.
WDT is enabled when the WDE bit is set to "1". When the WDE bit is set to "0", WDT is disabled.
The WDE can only be cleared if the WDTOE position bit is bit. To shut down a WDT that has been enabled, you must follow the following timing:
1. Assert both WDTOE and WDE bits, even if the WDE has been set, the WDE bit must be written to "1" before the shutdown operation begins;
2. For the next 4 clock cycles, write "0" to the WDE bit. This will turn off WDT.

When the WDE bit is "1" and the WDT overflows the reset system, the WDT reset system flag WDRF (located in the MCUSR register) is asserted. When the WDRF bit is asserted, the WDE bit is asserted. Therefore, to clear the WDE bit, you must first clear the WDRF bit.</td></tr>
<tr><td>[2:0]</td><td>WDP</td><td>WDT pre-divider factor selection control.
Used to set the overflow period of the WDT. It is recommended to change the value of WDP when the WDT is not counted, and changing the value of WDP during the counting process will produce an unpredictable WDT overflow.</td></tr>
</table>

Watchdog pre-crossover selection list:

WDP3	WDP2	WDP1	WDP0	Watchdog time-out cycles	32KHz clock	2MHz clock
0	0	0	0	2K cycles	64ms	1ms
0	0	0	1	4K cycles	128ms	2ms
0	0	1	0	8K cycles	256ms	4ms
0	0	1	1	16K cycles	512ms	8ms
0	1	0	0	32K cycles	1s	16ms
0	1	0	1	64K cycles	2s	32ms
0	1	1	0	128K cycles	4s	64ms
0	1	1	1	256K cycles	8s	128ms
1	0	0	0	512K cycles	16s	256ms
1	0	0	1	1024K cycles	32s	512ms

1	0	1	0	Reserved
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Interrupts and interrupt vectors

- 28 interrupt sources
- Programmable vector start address

The interrupt resources of LGT8F88P/168P/328P are basically the same, the main difference is the interrupt vector size, the LGT8F88P use 1 word (16 bits), while the interrupt vectors of LGT8F168P/328P are 2 words wide (32bits).

LGT8F88P interrupt vector list

LGT8F88P interrupt vector list:

Number	Vector address	Interrupt source signal	Description of the nterrupt source
1	0x0000	RESET	External reset, power-on reset, watchdog reset, SWD debug reset, low voltage reset
2	0x0001	INT0	External interrupt request 0
3	0x0002	INT1	External interrupt request 1
4	0x0003	PCI0	Pin level interrupt 0
5	0x0004	PCI1	Pin level interrupt 1
6	0x0005	PCI2	Pin level interrupt 2
7	0x0006	WDT	Watchdog overflow interrupt
8	0x0007	TC2 COMPA	Timer 2 compares to match A interrupt
9	0x0008	TC2 COMPB	Timer 2 matches the B interrupt
10	0x0009	TC2 OVF	Timer 2 overflow interrupt
11	0x000A	TC1 CAPT	Timer 1 input captures interrupts
12	0x000B	TC1 COMPA	Timer 1 matches the A interrupt
13	0x000C	TC1 COMPB	Timer 1 matches the B interrupt
14	0x000D	TC1 OVF	Timer 1 overflow interrupt
15	0x000E	TC0 COMPA	Timer 0 compares to match A interrupt
16	0x000F	TC0 COMPB	Timer 0 compares to match B interrupt
17	0x0010	TC0 OVF	Timer 0 overflow interrupt
18	0x0011	SPI STC	SPI serial transmission end interrupt
19	0x0012	USART RXC	USART RX complete interrupt
20	0x0013	USART UDRE	USART data register empty interrupt
21	0x0014	USART TXC	USART TX complete interrupt
22	0x0015	ADC	ADC end of conversion interrupt
23	0x0016	EE_RDY	EEPROM is ready for outages
24	0x0017	ANA_COMP	Analog comparator 0 interrupt
25	0x0018	TWI	The two-wire serial interface interrupts
26	0x0019	ANA_COMP1	Analog comparator 1 interrupt
27	0x001A	-	Reserved
28	0x001B	PCI3	Pin level interrupt 3
29	0x001C	PCI4	Pin level interrupt 4
30	0x001D	TC3_INT	Timer 3 interrupts

LGT8F168P/328P interrupt vector list

LGT8F168P/328P interrupt vector list:

Number	Vector address	Interrupt source signal	Description of the nterrupt source
1	0x0000	RESET	External reset, power-on reset, watchdog reset, SWD debug reset, low voltage reset
2	0x0002	INT0	External interrupt request 0
3	0x0004	INT1	External interrupt request 1
4	0x0006	PCI0	Pin level interrupt 0
5	0x0008	PCI1	Pin level interrupt 1
6	0x000A	PCI2	Pin level interrupt 2
7	0x000C	WDT	Watchdog overflow interrupt
8	0x000E	TC2 COMPA	Timer 2 compares to match A interrupt
9	0x0010	TC2 COMPB	Timer 2 matches the B interrupt
10	0x0012	TC2 OVF	Timer 2 overflow interrupt
11	0x0014	TC1 CAPT	Timer 1 input captures interrupts
12	0x0016	TC1 COMPA	Timer 1 matches the A interrupt
13	0x0018	TC1 COMPB	Timer 1 matches the B interrupt
14	0x001A	TC1 OVF	Timer 1 overflow interrupt
15	0x001C	TC0 COMPA	Timer 0 compares to match A interrupt
16	0x001E	TC0 COMPB	Timer 0 compares to match B interrupt
17	0x0020	TC0 OVF	Timer 0 overflow interrupt
18	0x0022	SPI STC	SPI serial transmission end interrupt
19	0x0024	USART RXC	USART RX complete interrupt
20	0x0026	USART UDRE	USART data register empty interrupt
21	0x0028	USART TXC	USART TX complete interrupt
22	0x002A	ADC	ADC end of conversion interrupt
23	0x002C	EE_RDY	EEPROM ready interrupt
24	0x002E	ANA_COMP	Analog comparator interrupt
25	0x0030	TWI	Two-wire serial interface interrupts
26	0x0032	ANA_COMP1	Analog comparator 1 interrupt
27	0x0034	-	Reserved
28	0x0036	PCI3	Pin level interrupt 3
29	0x0038	PCI4	Pin level interrupt 4
30	0x003A	TC3_INT	Timer 3 interrupts

The reset vector of the LGT8FX8P is executed from the address 0x0000. With the exception of the reset vector, all other vector addresses can be redirected to a 512-byte-aligned start address via the IVSEL in the MCUCR register as well as the IVBASE register.

Interrupt vector processing

The following code only uses LGT8F88P as an example to illustrate reset and interrupt vector programming, for reference only:

Example of assembly code – LGT8F88P		
address	code	Comment
0x000	RJMP RESET	Reset vector
0x001	RJMP EXT_INT0	External interrupt 0
0x002	RJMP EXT_INT1	External interrupt 1
0x003	RJMP PCINT0	Pin level change interrupt 0
0x004	RJMP PCINT1	Pin level change interrupt 1
0x005	RJMP PCINT2	Pin level change interrupt 2
0x006	RJMP WDT	Watchdog timer interrupt
0x007	RJMP TIM2_COMPA	Timer 2 compares to match Group A interrupts
0x008	RJMP TIM2_COMPB	Timer 2 compares to match Group B interrupts
0x009	RJMP TIM2_OVF	Timer 2 overflow interrupt
0x00A	RJMP TIM1_CAPT	Timer 1 captures an interrupt
0x00B	RJMP TIM1_COMPA	Timer 1 compares to Group A interrupts
0x00C	RJMP TIM1_COMPB	Timer 1 compares to Group B interrupts
0x00D	RJMP TIM1_OVFR	Timer 1 overflow interrupt
0x00E	RJMP TIM0_COMPA	Timer 0 compares to match Group A interrupts
0x00F	RJMP TIM0_COMPB	Timer 0 compares to match Group B interrupts
0x010	RJMP TIM0_OVF	Timer 0 overflow interrupt
0x011	RJMP SPI_STC	SPI transfer completion interrupted
0x012	RJMP USART_RXC	USART receives a completion interrupt
0x013	RJMP USART_UDRE	USART data register null interrupt
0x014	RJMP USART_TXC	USART sends completion interrupt
0x015	RJMP ADC	ADC conversion completion interrupt
0x016	RJMP EE_RDY	The EEPROM controller is ready for interrupts
0x017	RJMP ANA_COMP	Comparator interrupt
0x018	RJMP TWI	TWI controller interrupt
0x019	NOP	Reserve the address
0x01A	NOP	Reserve the address
0x01B	RJMP PCI3	Pin level change interrupt 3
;		
0x01C (**RESET :**)	LDI r16, high(RAMEND)	The main program begins
0x01D	OUT SPH, r16	Set the stack pointer to the RAM top address
0x01E	LDI r16, low(RAMEND)	
0x01F	OUT SPL, r16	
0x020	SEI	Enable global interrupts
0x021	……	

Register definitions

MCU control register - MCUCR

MCUCR – MCU control register								
MCUCR: 0x35(0x55)				Default value: 0x00				
MCUCR	FWKEN	FPDEN	EXRFD	PUD	IRLD	IFAIL	IVSEL	WCE
R/W	R/W	R/W	R/W	R/W	W/O	R/O	R/W	R/W

Bit definition		
[0]	WCE	The MCUCR update enable bit, which needs to be set before updating the MCUCR, and then complete the update of the MCUCR register within 6 cycles
[1]	IVSEL	The interrupt vector selects bits, after this position 1, the interrupt vector address will be mapped to the new address based on the value of the IVBASE register
[2]	IFAIL	System configuration bit load failure flag bit, 0 = Configuration information validation passed 1 = Configuration information failed to load
[3]	IRLD	Writing 1 will reload the system configuration information
[4]	PUD	Global pull-up disable bits 0 = Global pull-up control enabled 1 = Turn off pull-up resistors for all IOs
[5]	EXRFD	External reset filter disable bits 0 = (190us) digital filter with external reset enabled 1 = Digital filter circuit with external reset disabled
[6]	FPDEN	Flash Power/down enable control 0: FLASH remains powered on after the system sleeps 1: FLASH power off after system SLEEP
[7]	FWKEN	Fast wake-up mode enables control and is only valid for Power/Off mode 0:260us filter delay 1:32us filter delay

Interrupt vector base address register - IVBASE

IVBASE – Interrupt vector base address register		
IVBASE: 0x75		Default value: 0x00
IVBASE	IVBASE[7:0]	
R/W	R/W	

Bit definition		
[7:0]	IVBASE	If IVSEL is 1, the interrupt vector (except the reset vector) will be remapped on a 512-byte page with the IVBASE address as the base. The base address of the interrupt vector after mapping is: (IVBASE << 8) + the corresponding vector address in Table 1

External interrupts

- ❑ 2 external interrupt sources
- ❑ Configurable level or edge trigger interrupts
- ❑ Can be used as a wake-up source in sleep mode

Overview

External interrupts are triggered by the INT0 and INT1 pins. As long as the external interrupt is enabled, the interrupt can be triggered even if the two pins are configured as outputs. This can be used to generate software interrupts. External interrupts can be triggered by rising edge, falling edge, or low level, configured by the external interrupt control register EICRA. When an external interrupt is enabled and configured for level triggering (INT0 and INT1 pins only), the interrupt is generated as long as the pin level is low. Rising or falling edge interrupt triggering on INT0 and INT1 pins requires the IO clock to function properly, while low-level triggered interrupts on INT0 and INT1 pins are detected asynchronously.
Except for idle mode, the IO clock stops working in all other sleep modes. Therefore, both external interrupts can be used as wake-up sources in sleep modes other than idle mode.

If a level-triggered interrupt is used as a wake-up source in power-down mode, the changed level must be maintained for a certain amount of time to wake up the MCU to reduce the MCU's sensitivity to noise. The required level must be held long enough for the MCU to end the wake-up process and then trigger a level interrupt.

Register definitions

List of registers

register	address	Default value	Description
EICRA	0x69	0x00	External interrupt control register A
EIMSK	0x3D	0x00	External interrupt mask register
EIFR	0x3C	0x00	External interrupt flag register

External interrupt control register A-EICRA

EICRA – External interrupt control register A								
address: 0x69				Default value: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	ISC11	ISC10	ISC01	ISC00
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	description
7:4	-	Reserved
3	ISC11	INT1 pin interrupt trigger mode MSbit.
2	ISC10	INT1 pin interrupt trigger mode LSbit. When the global interrupt is enabled and the corresponding interrupt mask control bit of the GICR register is set, the external interrupt 1 is triggered by the INT1 pin. The way an interrupt is triggered is described in the table. Before edge detection, the MCU first samples the level on the INT1 pin. If edge triggering or level change triggering is selected, pulses lasting more than 1 system clock cycle will trigger

		interrupts, while pulses that are too short do not guarantee interrupts. If the low trigger method is selected, the low level must be maintained until the current instruction execution is complete before the interrupt is triggered.
1	ISC01	INT0 pin interrupt trigger mode MSbit.
0	ISC00	INT0 pin interrupt trigger mode MSbit. When the global interrupt is enabled and the corresponding interrupt mask control bit of the GICR register is asserted, the external interrupt 0 is triggered by the INT0 pin. The way an interrupt is triggered is described in the table. Before edge detection, the MCU first samples the level on the INT0 pin. If edge triggering or level change triggering is selected, pulses lasting more than 1 system clock cycle will trigger interrupts, while pulses that are too short do not guarantee interrupts. If the low trigger method is selected, the low level must be maintained until the current instruction execution is complete before the interrupt is triggered.

The trigger method of external interrupt 1 is shown in the following table.

External interrupt 1 trigger mode control

ISC1[1:0]	description
0	External pin INT1 low level trigger
1	External pin INT1 rising or falling edge trigger
2	External pin INT1 falling edge trigger
3	External pin INT1 rising edge trigger

The external interrupt 0 trigger mode is shown in the table below.

External interrupt 0 trigger mode control

ISC0[1:0]	Description
0	External pin INT0 low level trigger
1	External pin INT0 rising or falling edge trigger
2	External pin INT0 falling edge trigger
3	External pin INT0 rising edge trigger

External Interrupt Mask Register - EIMSK

EIMSK – External Interrupt Mask Register								
address: 0x3D						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	-	-	INT1	INT0
R/W	-	-	-	-	-	-	R/W	R/W
Bit	Name	description						
7:2	-	Reserved						
1	INT1	External pin 1 interrupt enable control bit. When the INT1 bit is set to "1" and the global interrupt is set, the external pin 1 interrupt is enabled and the wake-up function is enabled. Even if the INT1 pin is configured as an output, an interrupt will be generated as long as the pin level changes accordingly. When setting the INT1 bit to "0", the external pin 1 interrupt is disabled and the wake-up function is also disabled.						

0	INT0	External pin 0 interrupt enable control bit. When the INT0 bit is set to "1" and the global interrupt is set, the external pin 0 interrupt is enabled and the wake-up function is enabled. Even if the INT0 pin is configured as an output, as long as the pin level changes accordingly, an interrupt will be generated. When setting the INT0 bit to "0", the external pin 0 interrupt is disabled, and the wake-up function is also disabled.

External Interrupt Flag Register - EIFR

EIFR – External Interrupt Flag Register								
address: 0x3C					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	-	-	INTF1	INTF0
R/W	-	-	-	-	-	-	R/W	R/W

Bit	Name	description
7:2	-	Reserved
1	INTF1	External pin 1 interrupt flag. INTF1 is set when an edge triggers an external pin 1 interrupt. When the external pin 1 interrupt is triggered low, the INTF1 bit is not set. If the external pin 1 interrupt enable INT1EN bit is "1" and the global interrupt flag is set, an external pin 1 interrupt will be generated. INTF1 will be automatically cleared when this interrupt service routine is executed, or it can also be cleared by writing "1" to the INTF1 bit.
0	INTF0	External pin 0 interrupt flag. INTF0 is set when an edge triggers an external pin 0 interrupt. The INTF0 bit is not set when the external pin 0 interrupt is triggered low. If the external pin 0 interrupt enable INT0EN bit is "1" and the global interrupt flag is set, an external pin 0 interrupt will be generated. INTF0 will be automatically cleared when this interrupt service routine is executed, or it can also be cleared by writing "1" to the INTF0 bit.

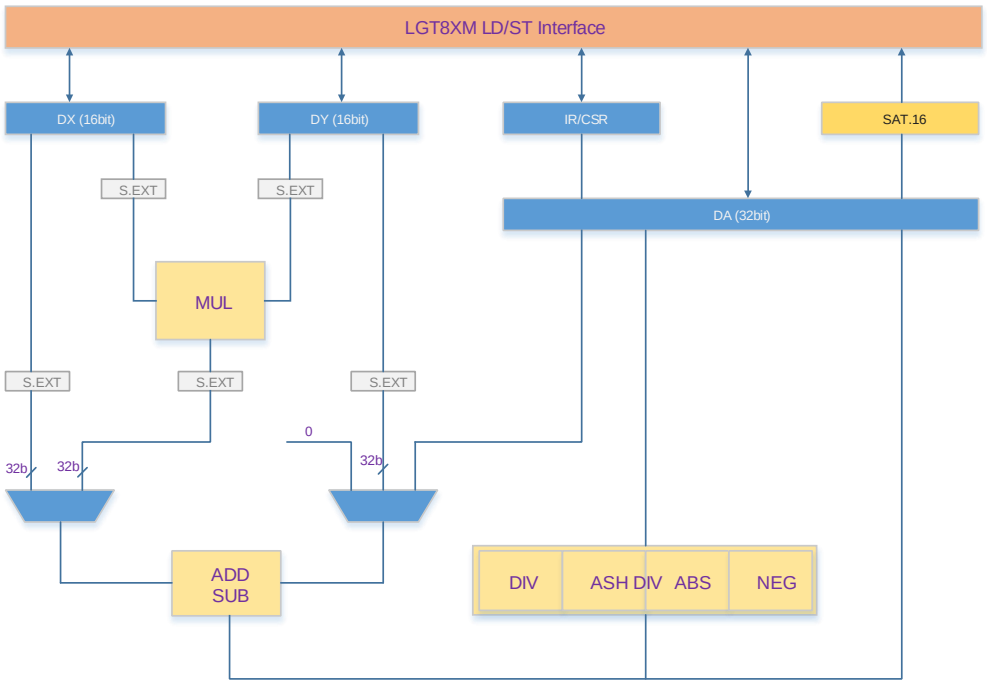
Computing Accelerator (uDSC)

- ❑ 16-bit storage mode (LD/ST)
- ❑ 32-bit accumulator (DX)
- ❑ Single-cycle 16-bit multiplier (MUL)
- ❑ 32-bit arithmetic logic unit (ALU)
- ❑ 16-bit saturation operation (SD)
- ❑ 8-cycle 32/16 divider
- ❑ Single-cycle multiply-add/multiply-subtract operations (MAC/MSC)

Overview

The digital computing accelerator (uDSC) is an operational co-processing module of the LGT8XM core, which cooperates with the 16-bit LD/ST mode of the LGT8XM core to realize a 16-bit digital signal processing unit. It can meet the processing requirements of almost all digital signal controls.
uDSC internal units and functions:

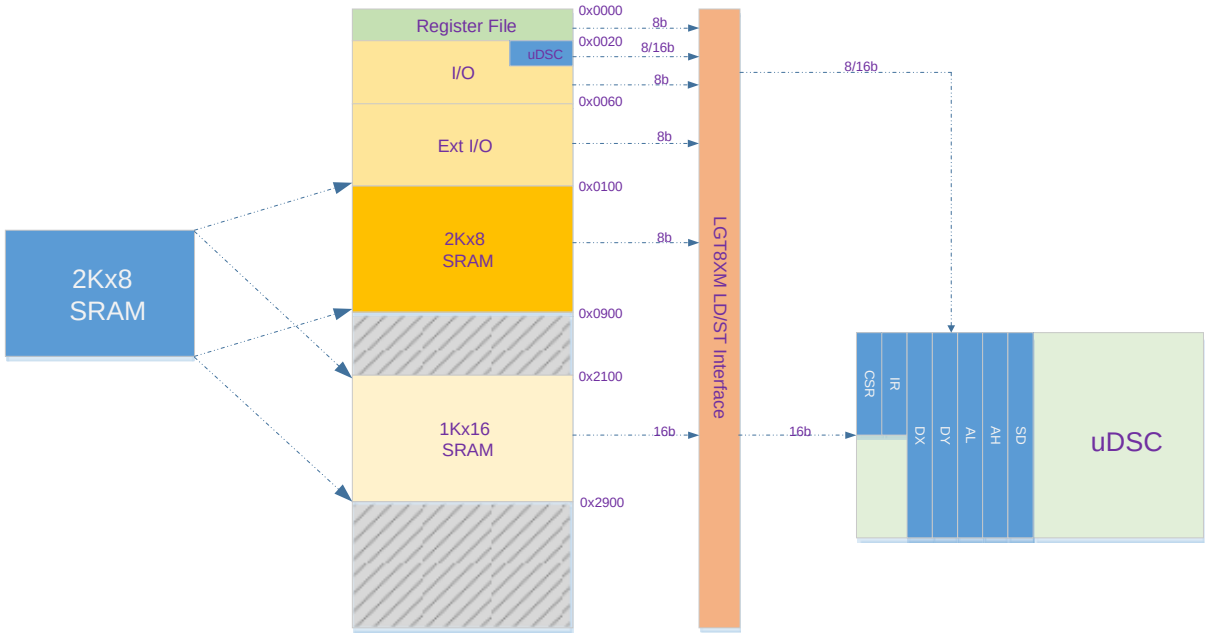
1. 16-bit operand register DX/DY
2. 32-bit accumulation register DA
3. Single-cycle 17-bit multiplier (can realize 16-bit signed/unsigned multiplication)
4. 32-bit ALU (can realize 16/32-bit addition, subtraction and shift operations)
5. 16-bit saturation operation (used to store the operation result in RAM space)
6. 32/16 divider, complete operation in 8 cycles



uDSC structure diagram

16-bit LD/ST working mode

In order to improve the efficiency of uDSC in processing a large amount of data operations, the LGT8XM core implements a dedicated 16-bit LD/ST storage channel, which can efficiently exchange 16-bit data between uDSC, SRAM and general-purpose register files using LDD/STD instructions.
In order not to destroy the normal LD/ST instruction system, the LGT8XM core remaps the SRAM space to 0x2100~0x28FF.
When using LD/ST instruction to access SRAM from 0x2100~0x28FF space, the kernel will automatically enable the 16-bit LD/ST function and open the direct access channel between SRAM and uDSC.
The following figure shows the data space address distribution of the LGT8XM core:



As shown in the figure above, the LGT8XM core can directly perform 16-bit data access between the DX/DY/DA registers of the uDSC and the SRAM by using the LD/ST instruction. At the same time, the internal registers of uDSC are also mapped to the I/O space, and there are two modes for accessing uDSC registers: 8/16.

In addition to the DX/DY/DA registers used for operations, uDSC also contains two other 8-bit registers: uDSC control status register CSR and operation instruction register IR. CSR/IR can only be accessed in bytes through I/O space; 16-bit mode when accessing DX/DY/AL/AH. It can be accessed by instructions such as IN/OUT and LD/ST/LDD/STD/LDS/STD.
The uDSC-related control status and data registers are mapped to the IO space, directly addressed by the IN/OU instruction, and 8/16-bit data access can be completed within one instruction cycle.

CSR is used to control the working mode of uDSC and record the status flag bit of current uDSC execution operation. IR controls the specific operation realized by uDSC. Most of the operations supported by uDSC will be completed in one cycle, and the division operation needs 7 waiting cycles. You can also judge whether the current division operation is completed through the flag bit in the CSR register.

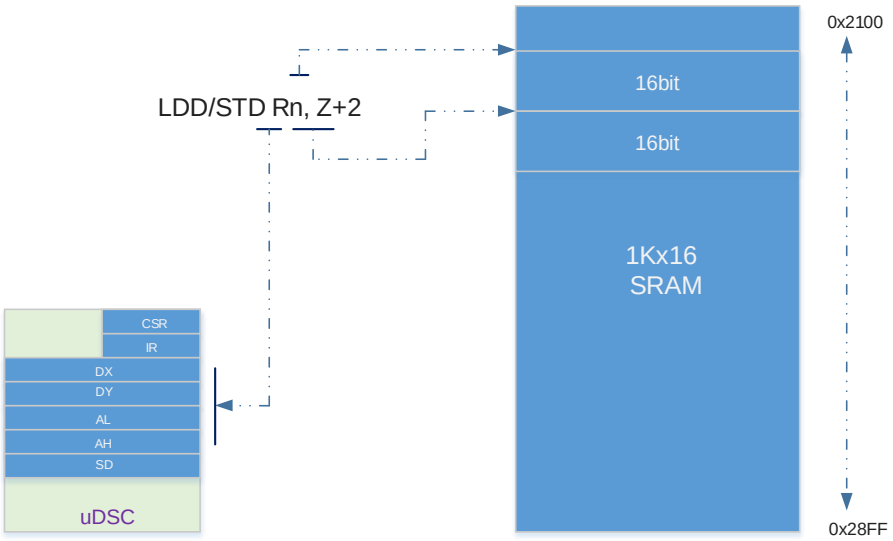
Standard LD/ST instructions use LGT8XM's internal general-purpose working registers as LD/ST data, and X/Y/Z as target addresses. When the target address falls in the 16-bit SRAM mapping space, the meaning of the operands of the LD/ST instruction changes, while X/Y/Z is still used as the target address, the meaning of the general working register addressing will conform to the uDSC mapping mode. Two processing methods. The mapping mode of uDSC is only used for accessing 0x2100~0x28FF address. The mapping mode is set by bit 6 (MM) of the CSR register.

In 16-bit LD/ST mode, the instruction "LDD Rn, Z+q" means to load the 16-bit data at address [Z] into the data register of uDSC, and then add an offset "q" to the value of Z . The relationship between the meaning of Rn and the mapping mode CSR[MM] here is as follows:

LDD Rn, Z/Y+q			
CSR[MM]	[Z+q]	Opcode	Operations
0	0x2100~0x28FF	LDD R0, Z+q	DX = [Z]; Z = Z + q; R0 kept unchanged
		LDD R1, Z+q	DY = [Z]; Z= Z + q; R1 kept unchanged
		LDD R2, Z+q	AL = [Z]; Z= Z + q; R2 kept unchanged
		LDD R3, Z+q	AH = [Z]; Z= Z + q; R3 kept unchanged
1	0x2100~0x28FF	LDD Rn, Z+q	{Rn} address for DX/DY/AL/AH in I/O region [DX/DY/AL/AY] = [Z]; Z = Z + q Rn keep unchanged
STD Rn, Z/Y+q			
0	0x2100~0x28FF	STD Z+q, R0	[Z] = DX; Z = Z + q; R0 kept unchanged
		STD Z+q, R1	[Z] = DY; Z = Z + q; R1 kept unchanged
		STD Z+q, R2	[Z] = AL; Z = Z + q; R2 kept unchanged
		STD Z+q, R3	[Z] = AH; Z = Z + q; R3 kept unchanged
		STD Z+q, R4	[Z] = SD; Z = Z + q; R4 kept unchanged
1	0x2100~0x28FF	STD Z+q, Rn	{Rn} address for DX/DY/AL/AH/SD in I/O region [Z] = [DX/DY/AL/AH/SD] addressed by {Rn} Rn keep unchanged

LD/ST, LDS/STS in the LGT8XM instruction set can all access the 0x2100~0x28FF area, but the Y/Z+q addressing mode of LDD/STD is more effective. The addressing of the LDD/STD method is based on a base address. We can set Y/Z as the base address of the data in the RAM. By using the Y/Z+q addressing mode of the LDD/STD instruction, the instruction can be executed in one cycle. and access data, and automatically move the address pointer to the next target address.

LGT8XM core standard LDD/STD instruction Y/Z+q offset addressing mode, when the instruction is executed, [Y/Z+q] is used as the address of 8-bit data, and the value of Y/Z does not increase after the execution is completed. When using LDD/STD to an address in the range 0x2100~0x28FF, the instruction behavior of LDD/STD changes: when the instruction is executed, [Y/Z] is used as the 16-bit data address, after execution the value of Y/Z Increments by the offset specified by "q". This feature can improve the efficiency of our continuous addressing, and the addressing of continuous 16-bit data can be realized by setting "q=2".



Mapping between variable addresses and 16-bit mode addresses

LGT8XM is an 8-bit processor, and data access is in bytes. The LGT8F328P carries 2K bytes of built-in data space. The 8-bit LD/ST accessed part of the space is mapped to the address 0x0100~0x08FF. C/C++ compilation automatically assigns variables between 0x0100~0x08FF. If we define a 16-bit array in C/C++ and need to use uDSC for operation, we need to first map the address of the variable to the address area (0x2100~0x28FF) accessed by 16-bit LD/ST. The method is very simple, just You need to increase the address of the variable by an offset of 0x2000.

uDSC operation instruction definition

The software specifies the operation to be realized through the IR register of uDSC. All computing operations of uDSC are performed between DX/DY/DA. Users can use 16-bit LD/ST channels to exchange data directly and quickly between DX/DY/DA and SRAM.

Instruction	IR[7:0]								Description
ADD/SUB	0	0	S^1	0	0	1	0	1	DA = DX + DY
	0	0	S^1	0	0	0	0	1	DA = DX – DY
	0	0	0	1	1	1	0	1	DA = DY
	0	0	S^1	1	1	0	0	1	DA = -DY
	0	0	S^1	1	0	1	1	1	DA = DA + DY
	0	0	S^1	1	0	0	1	1	DA = DA – DY
MAC/MSC	0	1	$S1^2$	$S0^2$	0	1	0	0	DA = DX * DY
	0	1	$S1^2$	$S0^2$	0	0	0	0	DA = -DX * DY
	0	1	$S1^2$	$S0^2$	1	1	0	0	DA = (DX * DY) >> 1
	0	1	$S1^2$	$S0^2$	1	0	0	0	DA = (-DX * DY) >> 1
	0	1	$S1^2$	$S0^2$	0	1	1	S	DA = DA + DX * DY
	0	1	$S1^2$	$S0^2$	1	1	1	S	DA = (DA + DX * DY) >> 1
	0	1	$S1^2$	$S0^2$	0	0	1	S	DA = DA – DX * DY
	0	1	$S1^2$	$S0^2$	1	0	1	S	DA = (DA – DX * DY) >> 1
MISC	1	0	0	0	0	0	0	0	DA = 0

	1	0	0	0	0	1	0	S	DA = NEG(DA)
	1	0	0	0	1	0	0	S	DA = DX^2
	1	0	0	0	1	0	1	S	DA = DY^2
	1	0	1	0	0	0	0	S	DA = ABS(DA)
	1	0	1	1	0	0	0	0	DA = DA/DY
	1	0	1	1	0	0	0	1	DA = DA/DY, DY = DA%DY
SHIFT	1	1	0	0	N3	N2	N1	N0	DA = DA << N
	1	1	S	1	N3	N2	N1	N0	DA = DA >> N

Legend :

1. **S** indicates whether the operation is a signed operation or an unsigned operation
2. **S1** indicates whether DX is a signed number, S2 indicates whether DY is a signed number
3. **N3...0** is a four-bit shift, which can realize up to 15 bit shift operations
4. **–** Indicates that the value of this bit is not meaningless, it can be set to 0 or 1, and it is recommended to set it to 0

Register definitions

name	IO address	Feature description
DCSR	0x20(0x00)	uDSC controls status registers
DSIR	0x21(0x01)	Arithmetic instruction registers
DSSD	0x22(0x02)	6-bit saturation operation result of accumulator DSA
DSDX	0x10(0x30)	Operand DSDX, 16-bit read and write access
DSDY	0x11(0x31)	Operand DSDY, 16-bit read and write access
DSAL	0x38(0x58)	32-bit accumulator DSA[15:0], 16-bit read and write access
DSAH	0x39(0x59)	32-bit accumulator DSA[15:0], 16-bit read and write access

DSCR - Control Status Register

DSCR – uDSC Control Status Register								
address: 0x20 (0x00)						Default value: 0010_xxxx		
Bit	7	6	5	4	3	2	1	0
Name	DSUEN	MM	D1	D0	-	N	Z	C
R/W	R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	description
7	DSUEN	uDSC module enable control; 1 = enable, 0 = disable
6	MM	uDSC register mapping mode; for detailed definition, please refer to the introduction of 16-bit working mode. 0 = fast access mode, 1 = IO mapped mode
5	D1	Division operation complete flag, 1 = operation complete
4	D0	Division operation divide by 0 flag bit
3	-	Unimplemented
2	N	Operation result negative flag bit
1	Z	Operation result zero flag
0	C	32 adder carry/borrow flag

DSIR – Arithmetic Instruction Register

DSIR – uDSC Arithmetic Instruction Register								
address: 0x21 (0x01)						Default value: 0000_0000		
Bit	7	6	5	4	3	2	1	0
Name	DSIR[7:0]							
R/W	R/W							
Bit	Name	description						
7:0	IR	uDSC operation instruction. Please refer to the description in the "Operation Instruction Definition" chapter						

DSDX - operand register DSDX

DSDX – uDSC operand register DX																
address: 0x30 (0x10)											Default value: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSDX[15:0]															
R/W	R/W															
Bit	Name	description														
15:0	DSDX	16-bit operand register DSDX														

DSDY - Operand register DSDY

DSDY – uDSC operand register DY																
address: 0x31 (0x11)											Default value: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSDY[15:0]															
R/W	R/W															
Bit	Name	description														
15:0	DSDY	16-bit operand register DSDY														

DSAL - Lower 16 bits of 32-bit accumulator DA

DSAL – Lower 16 bits of uDSC operand register DSA																
address: 0x58 (0x38)											Default value: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSA[15:0]															
R/W	R/W															
Bit	Name	description														
15:0	DSAL	The lower 16 bits of the 32-bit accumulator DSA														

DSAH - High 16 bits of 32-bit accumulator DA

DSAH – High 16 bits of uDSC operand register DSA																
address: 0x59 (0x39)											Default value: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSA[31:16]															
R/W	R/W															
Bit	Name	description														
15:0	DSAH	The upper 16 bits of the 32-bit accumulator DSA														

DSSD - DA Saturation Operation Register

DSSD– 16-bit DA saturation operation result																
address: 0x22 (0x02)											Default value: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSSD[15:0]															
R/W	R/W															
Bit	Name	description														
15:0	DSSD	16-bit saturation operation result of 32-bit accumulator DSA														

uDSC application example

Example 1. Basic configuration and operation

The following is a simple subroutine (AVRGCC) that implements a 16-bit multiplication operation and returns a 32-bit result.:

```
unsigned long dsu_xmuluu (unsigned short dy, unsigned short dx);
```

The following is the assembly implementation code of the C function above:

```
#include    “udsc_def.inc”          ; opcode definitions
            .global   dsu_xmuluu    ; declare for called from C/C++ code

dsu_xmuluu:
            out       DSDX, r24     ; load DX
            out       DSDY, r22     ; load DY
            ldi       r20, XMULUU   ; load opcode
            out       DSIR, r20     ; do multiply
            in        r22, DSAL     ; {r23, r22} = AL
            in        r24, DSAH     ; {r25, r24} = AH
            ret
```

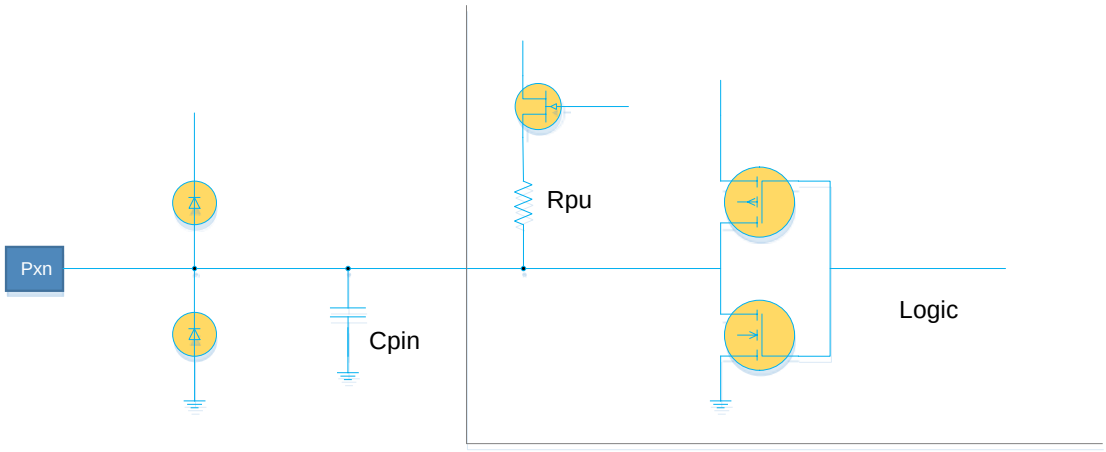
General Programmable Port (GPIO)

Overview

All MCUs implemented based on the LGT8XM core series have I/O port read-modify-write functions. This means that the state of a certain port can be changed independently using SBI and CBI instructions without affecting any other I/O. The same goes for changing the direction of a port or controlling its pull-up resistors.

Most of the I/Os of the LGT8FX8P have symmetrical drive characteristics and can drive and sink large currents. I/O has two levels of drive capability, and the user can control the drive capability of each group of I/O. The drive capability of the I/O can directly drive some LEDs.
Most of the I/Os of the LGT8FX8P can drive up to 30mA and can be used directly to drive segment LEDs.
The VCC and GND of all I/Os have independent ESD protection diodes directly, and the design can withstand ESD pulses up to 5000V at least.

I/O equivalent circuit diagram:



All the registers below in this chapter adopt a unified description method, the lowercase "x" indicates the alphabetical number name of the port, and the lowercase "n" indicates the bit number in the port. But when using port registers in a program, you must use the exact register name. For example, PORTB3, which represents the third digit of PORTB, is uniformly represented by PORTxn here. For detailed definitions of I/O-related registers, please refer to the register description section.

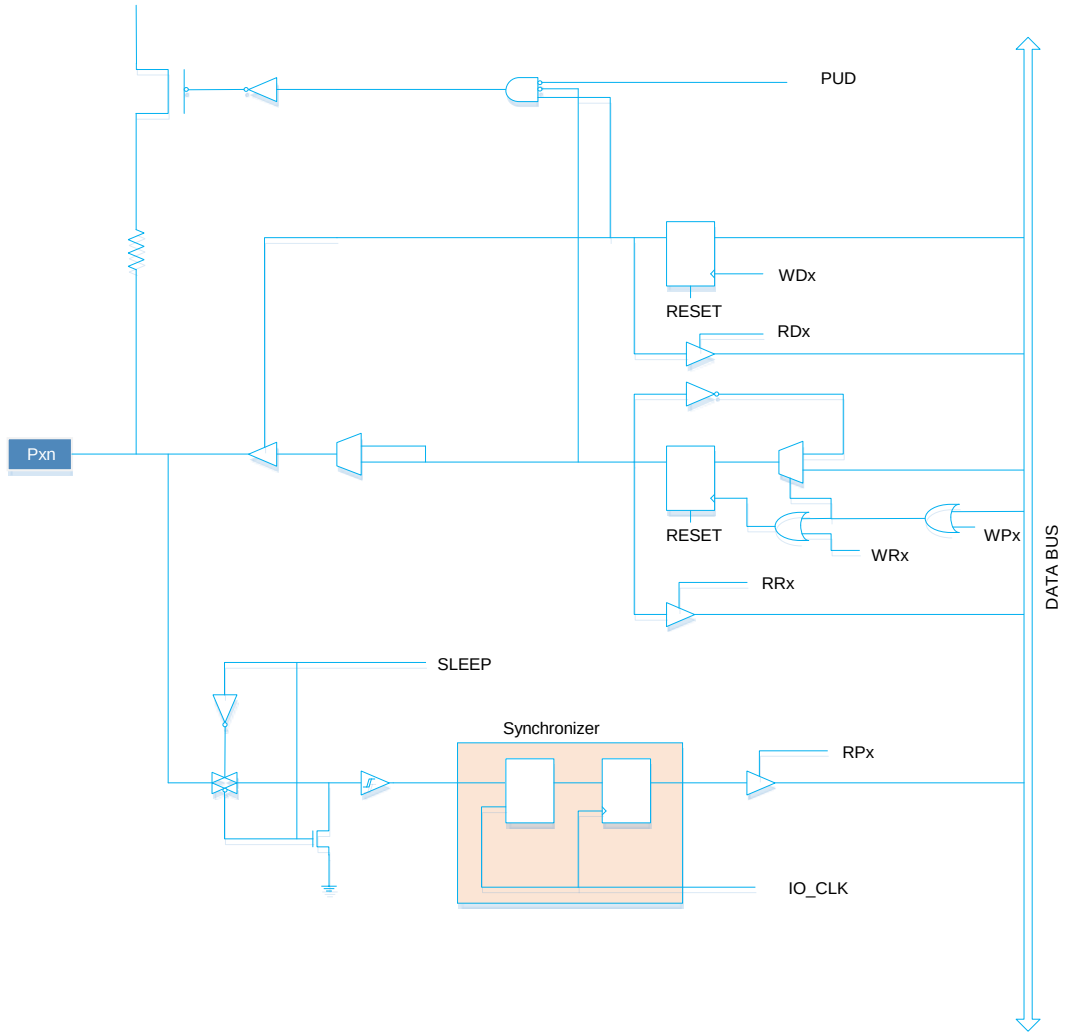
Each port is allocated three I/O register spaces, which are: Port Data Output Register (PORTx), Port Direction Register (DDRx), and Port Data Input Register (PINx). The port data input registers are read-only registers. The data output register and port direction register can be read and written. The PUD bit in the MCUCR register is used to control the pull-up resistors of all I/Os. When the PUD bit is 1, all I/O pull-up resistors will be disabled.

In addition to having general-purpose input/output functions, most I/Os will also be multiplexed as other peripheral functions. For specific multiplexing functions, please refer to the chapter on port function multiplexing.

It should be noted that enabling the multiplexing function of some ports will not affect the use of these ports as digital I/O. And some multiplexing functions may also need to control the input/output direction of the port through the I/O register. The specific settings will be introduced in the documentation of each multiplexing module.

General purpose input/output port

When used as a general-purpose I/O, the port is a bidirectional drive I/O port with internal programmable pull-up.
The following figure is the equivalent circuit diagram of the general-purpose I/O port:



PUD: PULLUP DISABLE

SLEEP: SLEEP CONTROL

IO_CLK: I/O CLOCK

WDx: WRITE DDRx

RDx: READ DDRx

WRx: WRITE PORTx

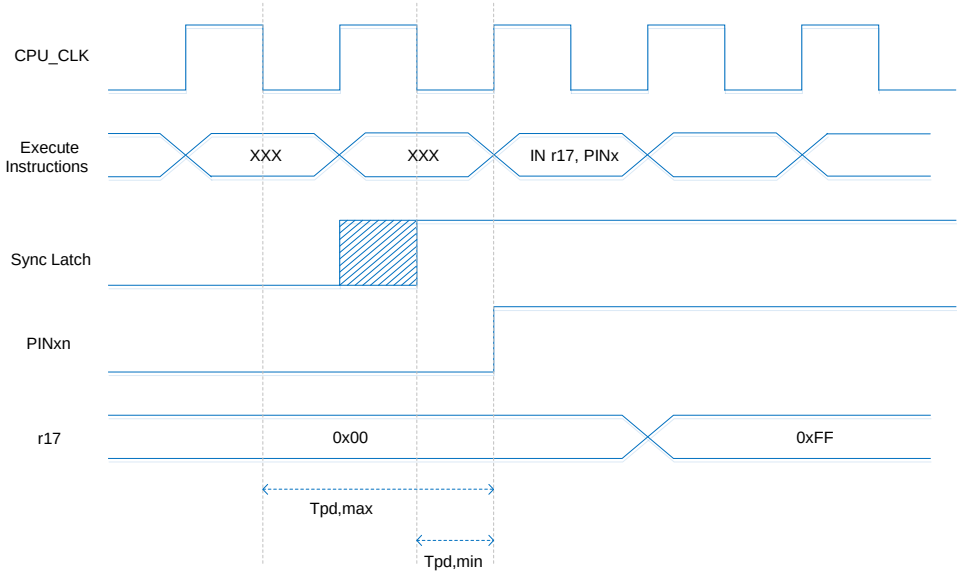
RRx: READ PORTx REGISTER

RPx: READ PORTx PIN

WPx: WRITE PINx REGISTER

Port usage configuration

Each port is controlled by three register bits: DDxn, PORTxn and PINxn. Among them, DDxn can be accessed through the DDRx register, PORTxn can be accessed through the PORTx register, and PINxn can be accessed through the PINx register.
The DDRxn register bits are used to set the input/output direction of the port. If DDxn is set to 1, the Pxn port is configured as an output port. If DDxn is set to 0, Pxn is configured as an input port.



If the PORTxn bit is written as 1 and this port is configured as an input port, the pull-up resistor of this port is valid.
If you want to disable the pull-up resistor of the port, PORTxn must be written as 0 or configure the port as an output port.
The reset initialization state of the port is the input state, and the pull-up resistor is invalid.
PORTxn is set to 1, and this port is configured as an output port, and the external port will be driven high.
If PORTxn is set to 0, the port will be driven low.

I/O switch

When the I/O state switches between tri-state ([DDxn, PORTxn]) = 0b00) and output high level ([DDxn, PORTxn] = 0b11), there will be an intermediate state where the port is pulled up or the output is low state. Usually, pull-up resistors are acceptable because in a high-impedance environment, the difference between driving high and pulling up is not important. If this is not the case, the pull-ups for all ports can be disabled via the PUD bit in the MCUCR register.
Also, the same problem occurs when switching between input with pull-up enabled and output low. User must use tri-state ([DDxn, PORTxn] = 0b00) or output high ([DDxn, PORTxn] = 0b11) as intermediate state.
Port driver configuration table:

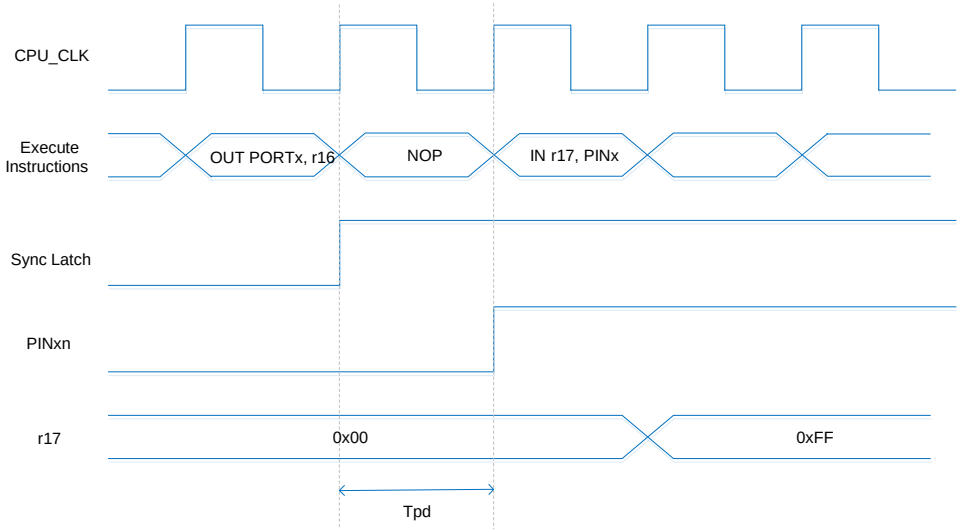
DDxn	PORTxn	PUD	Port status	Pull up	Function description
0	0	X	input	forbid	Three-state (High-Z)
0	1	0	input	Enable	Input + internal pull-up mode
0	1	1	input	forbid	Three-state (High-Z)
1	0	X	output	forbid	output low (fan-in)
1	1	X	output	forbid	output high (fan-out)

Read the port value

No matter how the port direction bit DDxn is set, the current state of the port can be read through the PINxn register bit.
To avoid directly reading the metastability generated by the port, the PINxn register bits are the result of the port passing through a synchronizer. The synchronizer is composed of a latch and a register, so there is a small delay between the value of PINxn and the current port.
This delay is due to the existence of the synchronizer, and the delay time is at most 1 and a half system cycles.
We assume that the system cycle starts from the first falling edge of the system clock, the latch latches the data when the clock is low, and the data passes through the latch when the clock is high, as shown in the shaded part of the figure above. When the clock is low, the port data is locked

stored, and is registered to the PINxn register on the rising edge of the next clock. Tpd,max and Tpd,min in the above figure are the maximum and minimum delays of port data, divided into 1.5 cycles and 0.5 cycles.

If you want to read the port value set by the software, you need to insert a null operation instruction (NOP) in the I/O write and read byte support. The timing is shown in the figure below:



The following code shows how to set the pin 0/1 of port B as high and 2/3 as low, define pin 4~7 as input and enable the pull-up resistors of pin 6 and 7. Then the value of the pin is read back into the general working register, and a NOP instruction is directly inserted between the output and input of the pin as described before.

assembly code

```
; Define Pull-ups and set outputs high
; Define directions for port pins
LDI r16, (1<<PB7)|(1<<PB6)|(1<<PB1)|1<<PB0)
LDI r17, (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
OUT PORTB, r16
OUT DDRB, r17
; Insert nop for synchronization
NOP
; Read port pins
IN r16, PINB
```

C language code

```
unsigned char I;
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* Insert nop for synchronization */
__no_operation();
/* Read port pins */
I = PINB;
```

Input Enable and Sleep Control

From the equivalent circuit diagram of I/O, we can see that the digital input can be clamped to the ground level under the control of the SLEEP signal. The SLEEP signal is controlled by the MCU's sleep controller and various sleep modes. This can ensure that after entering sleep mode, the system will not cause leakage due to floating port input.
The SLEEP control function of the port will be replaced by the external interrupt function. If the external interrupt request is invalid, SLEEP control can still work. The SLEEP control function will also be replaced by other secondary functions. For details, please refer to the introduction of the port secondary functions below.

Port fast state toggling

The port state is set as an output IO, and the port state can be changed through the PORTn register. If you need to toggle the output state of the current port, you usually need to read the current port state PINx first, then take the inversion and write it back to the PORTn register to complete the flip. LGT8FX8P provides another more efficient way to flip the port state, by directly writing 1 to the PINx register to flip the specified port state. For example, if we write PINB[3] as 1, the port status of PB3 can be reversed. This method is very practical for applications that need to generate an output clock.

Digital/analog multiplexing port

Some ports of LGT8FX8P are hybrid ports with digital and analog functions. Except for the output PD4 of the internal DAC, all other hybrid ports are used as analog inputs. When the port is used as an analog function, the software needs to set the port to input mode and turn off the internal pull-up as needed, so as not to affect the analog income. The DIDR0~2 registers are used to close the digital input channel of the mixed-function port to avoid excess power loss caused by the analog input to the digital circuit. DIDRx does not turn off the digital output function of the port.

High current push-pull drive port

LGT8FX8P supports up to 6 high-current push-pull drive terminals, and supports a maximum 80mA push-pull drive. Considering the limit of the maximum overcurrent capability of the chip VCC, it is not recommended to enable 6 high-current drives at the same time. Especially for the QFP32 package with only one set of power terminals, it is recommended not to turn on and drive more than 4 high-current loads at the same time.
The normal port drive is 12mA, and the software needs to enable the high current drive function of the port through the HDR register.
The ports with high current drive capability are as follows:

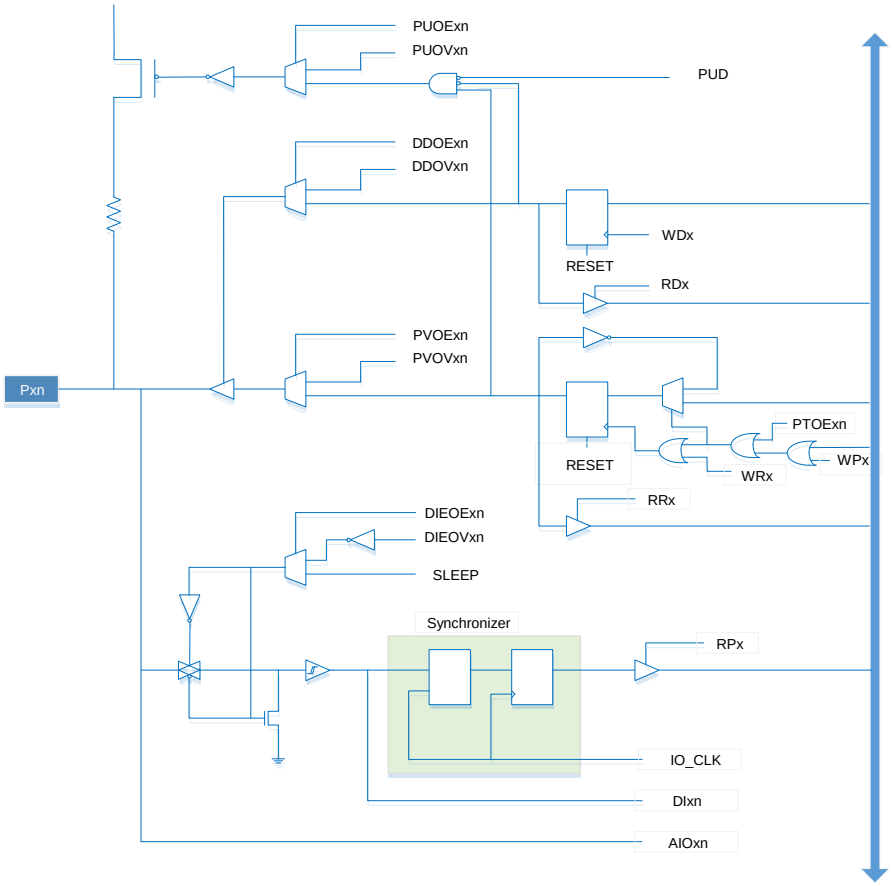
HDR port	QFP48	QFP32	HDR	Function description
PD5	PD5	PD5	HDR[0]	N/A
PD6	PD6	PD6	HDR[1]	N/A
PF1	PF1	PD1 PF1	HDR[2]	PD1 of the QFP32 package is internally equivalent to the PD1 and PF1 parallel of the QFP48
PF2	PF2	PD2 PF2	HDR[3]	PD2 of the QFP32 package is internally equivalent to the PD2 and PF2 parallel of the QFP48
PF4	PF4	PE4 PF4	HDR[4]	PE4 of the QFP32 package is internally equivalent to the PE4 and PF4 parallel of the QFP48
PF5	PF5	PE5 PF5	HDR[5]	PE5 of the QFP32 package is internally equivalent to the PE5 and PF5 parallel of the QFP48

Handling of unused ports

If some ports are not used, it is recommended to drive them to a fixed level. In any case, floating pins will cause more power consumption and cause the system to become unstable under strong interference.
The easiest way to give a port a fixed level is to turn on the port's pull-up resistor. It should be noted that the pull-up resistor is disabled during power-on reset. The way of pull-up resistors will also cause unnecessary leakage. Therefore it is recommended to use an external pull-up or pull-down resistor connection. It is not recommended to directly connect the port to power or ground, because if these pins are configured as outputs, it may cause very large current to pass through the port, causing destructive effects on the chip.

Port multiplexing function

Most of the ports have multiplexing function, and the equivalent circuit below illustrates the control of the port by the multiplexing function of the port. These alternate functions do not necessarily exist with all port pins.



PUOExn: Pxn PULL-UP OVERRIDE ENABLE
PUOVxn: Pxn PULL-UP OVERRIDE VALUE
DDOExn: Pxn DATA DIRECTION OVERRRIDE ENABLE
DDOVxn: Pxn DATA DIRECTION OVERRIDE VALUE
PVOExn: Pxn PORT VALUE OVERRIDE ENABLE
PVOVxn: Pxn PORT VALUE OVERRIDE VALUE
DIEOExn: Pxn INPUT-ENABLE OVERRIDE ENABLE
DIEOVxn: Pxn INPUT-ENABLE OVERRIDE VALUE
SLEEP: SLEEP CONTROL
PTOExn: Pxn PORT TOGGLE OVERRIDE ENABLE

PUD: PULLUP DISABLE
WDx: WRITE DDRx
RDx: READ DDRx
RRx: READ PORTx REGISTER
WRx: WRITE PORTx
RPx: READ PORTx PIN
WPx: WRITE PINx
IO_CLK: I/O CLOCK
DIxn: INPUT PIN n ON PORTx
AIOxn: ANALOG I/O PIN n ON PORTx

General description of multiplexing function control signals:

Signal	Full name	Feature description
PUOE	Pull-up multiplexing enable	If this bit is 1, the pull-up enable is controlled by PVOV; if this bit is 0, the pull-up enable is controlled by DDxn, PORTxn and PUD
PUOV	pull-up multiplexing value	If PUOE is 1, this bit will enable the pin's pull-up resistor, otherwise it will disable the pin's pull-up resistor
DDOE	Port direction multiplexing enable	The second bit is 1, the pin output enable is controlled by DDOE, otherwise it is controlled by DDxn
DDOV	Port direction multiplexing value	If DDOE is 1 and the second bit is 1, the output function of the pin will be enabled, otherwise the output of the pin will be turned off
PVOE	Port data multiplexing enable	If the second bit is 1 and the pin output is enabled, the output value of the pin will be controlled by PVOV, otherwise it will be controlled by PORTxn
PVOV	Port data multiplexing value	Refer to the PVOE function description
PTOE	Port Flip Multiplexing Enable	When the second bit is 1, the PORTxn bit will be flipped
DIEOE	Digital input enable mux enable	If the second bit is 1, the port digital input enable will be controlled by DIEOV; otherwise, it will be controlled by the running state of the MCU
DIEOV	Digital input enable multiplexed value	If DIEOE is 1, the digital input function of the port will be Control, independent of MCU operating status
DI	digital input	This is the digital input signal to the alternative function block. As can be seen from the lower circuit diagram of I/O, this value is after the Schmitt trigger, but before the I/O input synchronizer. This signal is connected to the peripheral module, and the peripheral module will perform synchronization processing as needed
AIO	analog input	Analog input/output signal, this signal is directly connected to the I/O PAD and can be used as an analog bidirectional signal. This signal is directly connected to the port of the internal ADC, comparator and other analog modules

The following section will briefly describe the alternate function of each pin and the related control signals.

Port B multiplexing function

Pin	Multiplexing function description
PB7	XTALI/TOSC2 (external main oscillator pin XI) PCINT7 (pin change interrupt 7)
PB6	XTALO/TOSC1 (external main oscillator pin XO) PCINT6 (pin change interrupt 6)
PB5	SCK (SPI bus master clock input) PCINT5 (pin change interrupt 5)
PB4	MISO (SPI bus master input/slave output) PCINT4 (pin change interrupt 4)

PB3	MOSI (SPI bus master output/slave input) OC2A (timer/counter 2 compare match output A) PCINT3 (pin change interrupt 3)
PB2	SSN (SPI bus slave select input) OC1B (Timer/Counter 1 compare match output B) PCINT2 (pin change interrupt 2)
PB1	OC1A (timer/counter 1 compare match output A) PCINT1 (pin change interrupt 1)
PB0	ICP1 (timer/counter 1 capture input) CLKO (system clock output) PCINT0 (pin change interrupt 0)

XTALI/TOSC2/PCINT7 – Port B pin 7

- XTALI: External crystal oscillator pin XI. When used as a clock signal for a crystal oscillator, this pin cannot be used as an I/O.
- TOSC2: Timer external crystal oscillator pin 2. When the internal RC is configured as the main working clock of the chip and the asynchronous timer function is enabled (ASSR register configuration), this pin will be used as the external crystal oscillator pin of the timer. When AS2 of the ASSR register is set to 1 and EXCLK is set to 0, the timer/counter 2 is enabled to use the asynchronous clock function of the external crystal oscillator. PB7 will be disconnected from the internal I/O port and become the reverse of the internal oscillator amplifier. output pin. In this mode, an external crystal oscillator is connected to the pin.
- PCINT7: Pin Change Interrupt 7. PB7 is an external interrupt source.
- If PB7 is used as a crystal oscillator pin, the values of DDB7, PORTB7 and PINB7 will have no meaning.

XTALO/TOSC1/PCINT6 - Port B pin 6

- **XTALO**: External crystal oscillator pin XO.
- **TOSC1**: Timer external crystal oscillator pin 1. When the internal RC is configured as the main working clock of the chip and the asynchronous timer function is enabled (ASSR register configuration), this pin will be used as the external crystal oscillator pin of the timer. When AS2 of the ASSR register is set to 1 and EXCLK is set to 0, the timer/counter 2 is enabled to use the asynchronous clock function of the external crystal oscillator, and PB6 will be connected to the internal I/O port port and become the input pin of the internal oscillator amplifier. . In this mode, an external crystal oscillator is connected to the pin.
- **PCINT6**: Pin Change Interrupt 6. PB6 is an external interrupt source.
- If PB6 is used as a crystal oscillator pin, the values of DDB6, PORTB6 and PINB6 will have no meaning.

SCK/PCINT5 - Port B pin 5

- **SCK**: SPI controller master device clock output, slave device clock input. When the SPI controller is configured as a slave device, this pin will be configured as an input pin, not controlled by DDB5. When the SPI controller is configured as a master, the direction of this pin is controlled by DDB5. When this pin is forced as an input by SPI, the pull-up resistor can still be controlled by the PORTB5 bit.
- **PCINT5**: Pin Change Interrupt. PB5 is an external interrupt source.

MISO/PCINT4 - Port B pin 4

- **MISO**: SPI controls the data input of the master device and the data output of the slave device. When the SPI is configured as a master device, this pin will be forced as an input and not controlled by DDB4. When the SPI is used as a slave device, the data direction of this pin is controlled by DDB4. When this pin is forced as an input by the SPI controller, its pull-up resistor can still be controlled by PROTB4.

- **PCINT4**: Pin Change Interrupt. PB4 is an external interrupt source.

MOSI/OC2A/PCINT3 - Port B pin 3

- **MOSI**: SPI controller master device data output, slave device data input. When the SPI is configured as a slave, this pin will be forced as an input and not controlled by DDB3. When the SPI controller is configured as a master, this pin method is controlled by DDB3. When this pin is forced as an input by SPI control, its pull-up resistor can still be controlled by PORTB3.
- **OC2A**: Group A compare match output of timer/counter 2. PB3 can be used as the external output of timer/counter 2 compare match. At this point the pin must be set as an output via DDB3. At the same time, OC2A is also the PWM mode output pin of Timer 2.
- **PCINT3**: Pin Change Interrupt. PB3 is an external interrupt source.

SSN/OC1B/PCINT2 - Port B pin 2

- **SSN**: SPI slave chip select input. When the SPI controller is configured as a slave, this pin will be forced as an input and not controlled by DDB2. As a slave device, the SPI controller is active when SSN is driven low. When the SPI controller is configured as a master, the direction of this pin is controlled by DDB2. When this pin is forced as an input by the SPI controller, the pull-up resistor can still be controlled by PORTB2.
- **OC1B**: Timer/Event Counter 1 group B compare match output. PB2 can be used as the external output of timer/counter 1 compare match. At this point the pin must be set as an output via DDB2. At the same time, **OC1B** is also the PWM mode output pin of Timer 1.
- PCINT2: Pin Change Interrupt. PB2 is an external interrupt source.

OC1A/PCINT1 - Port B pin 1

- **OC1A**: Group A compare match output of timer/counter 1. PB1 can be used as the external output of timer/counter 1 compare match. At this point the pin must be set as an output via DDB1. At the same time, OC1A is also the PWM mode output pin of Timer 1.
- **PCINT1**: Pin Change Interrupt. PB1 is an external interrupt source.

ICP1/CLKO/PCINT0 - Port B pin 0

- **ICP1**: Capture input pin of timer/counter 1
- **CLKO**: System working clock output, when the CLKOE bit in the CLKPR register is 1, this pin will be forced to output, not controlled by DDB0. The output frequency is the working clock frequency of the current system.
- **PCINT0**: Pin Change Interrupt. PB0 is an external interrupt source.

Port C multiplexing function

Pin	Multiplexing function description
PC7	ADC8 (ADC input channel 8) APN2 (DAP reverse input 2) PCINT15 (pin level change input 15)
PC6	RESETN (external reset input) PCINT14 (pin change input 14)
PC5	ADC5 (ADC input channel 5) SCL (TWI clock line) PCINT13 (pin change input 13)
PC4	ADC4 (ADC input channel 4) SDA (TWI data cable) PCINT12 (pin change input 12)
PC3	ADC3 (ADC input channel 3) PCINT11 (pin change input 11)
PC2	ADC2 (ADC input channel 2) PCINT10 (pin change input 10)
PC1	ADC1 (ADC input channel 1) PCINT9 (Pin Change Input 9)
PC0	ADC0 (ADC input channel 0) PCINT8 (pin change input 8)

ADC8/APN2/PCINT15 - Port C pin 6

- **ADC8**: ADC external input channel 8
- **APN2**: Inverting input port 2 of the differential amplifier
- **PCINT15**: Pin Change Interrupt. After closing the external reset input function of this pin, PC7 can be used as an external interrupt source.

RESETN/PCINT14 - Port C pin 6

- **RESETN**: External reset input pin. After power-on reset, this pin defaults to external reset function. The external reset function can be turned off through the IOCR register. After closing the external reset function, this pin can be used as a general-purpose I/O. But it should be noted that during power-on and other reset processes, this pin defaults to reset input, so if the user needs to use the general-purpose I/O function of this pin, the external circuit cannot affect the power-on and reset of the chip process, it is recommended to configure this pin as an output function I/O, and add an appropriate pull-up resistor externally.
- **PCINT14**: Pin Change Interrupt. After closing the external reset input function of this pin, PC6 can be used as an external interrupt source.

SCL/ADC5/PCINT13 - Port C pin 5

- **SCL**: TWI interface clock signal. After the TWEN bit in the TWCR register is set to 1, the TWI interface is enabled, and PC5 will be controlled by TWI and become the clock signal of the TWI interface.

- **ADC5**: ADC input channel 5. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- **PCINT13**: Pin Change Interrupt 13

SDA/ADC4/PCINT12 - Port C pin 4

- **SDA**: TWI interface data signal. After the TWEN bit in the TWCR register is set to 1, the TWI interface is enabled, and PC4 will be controlled by TWI and become the data signal of the TWI interface.
- **ADC4**: ADC input channel 4. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- **PCINT12**: Pin Change Interrupt 12

ADC3/APN1/PCINT11 - Port C pin 3

- **ADC3**: ADC input channel 3. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- **APN1**: Differential amplifier negative input 1
- **PCINT11**: Pin Change Interrupt 11

ADC2/APN0/PCINT10 - Port C pin 2

- **ADC2**: ADC input channel 2. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- **APN0**: Differential amplifier reverse input 0
- **PCINT10**: Pin Change Interrupt 10

ADC1/APP1/PCINT9 - Port C pin 1

- ADC1: ADC input channel 1. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- APP1: Differential amplifier positive input 1
- PCINT9: Pin Change Interrupt 9

ADC0/APP0/PCINT8 - Port C pin 0

- ADC0: ADC input channel 0. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC.
- APP0: Differential amplifier positive input 0
- PCINT8: Pin Change Interrupt 8

Port D multiplexing function

Pin	Multiplexing function description
PD7	ACXN (analog comparator 0/1 common negative terminal input) PCINT23 (pin change interrupt 23)
PD6	AC0P (QFP32: Analog Comparator 0 Positive Input) OC0A (timer/counter 0 compare match output A) OC3A (QFP32: Timer/Counter 3 compare match output A) PCINT22 (pin change interrupt 22)
PD5	T1 (timer/counter 1 external count clock input) OC0B (timer/counter 0 compare match output B) PCINT21 (pin change interrupt 21)
PD4	XCK (USART external clock input/output) DAO (internal 8bit DAC analog output) T0 (timer/counter 0 external count clock input) PCINT20 (pin change interrupt 20)
PD3	INT1 (external interrupt input 1) OC2B (timer/counter 2 compare match output B) PCINT19 (pin change interrupt 19)
PD2	INT0 (external interrupt input 0) AC0O (Comparator 0 output) OC3B (QFP32: Timer/Counter 3 compare match output B) PCINT18 (pin change interrupt 18)
PD1	TXD (USART data output) OC3A (QFP32: Timer/Counter 3 compare match output A) PCINT17 (pin change interrupt 17)
PD0	RXD (USART data input) PCINT16 (pin change interrupt 16)

ACXN/OC2B/PCINT23 - Port D pin 7

- **ACXN**: Analog Comparator 0/1 Common Negative Input
- **OC2B**: Timer/Event Counter 2 group B compare match output. PD7 can be used as the external output of timer/counter 2 compare match. At this point the pin must be set as an output via DDD7. At the same time, **OC2B** is also the PWM mode output pin of Timer 2;
- PCINT23: Pin Change Interrupt 23

AC0P/OC0A/PCINT22 - Port D pin 6

- **AC0P**: Analog comparator 0 positive input.
- **OC0A**: Timer/Event Counter 0 group A compare match output. PD6 can be used as the external output of timer/counter 0 compare match. At this point the pin must be set as an output via DDD6. At the same time, OC0A is also the PWM mode output pin of Timer 0
- **PCINT22**: Pin Change Interrupt 22

T1/OC0B/PCINT21 - Port D pin 5

- **T1**: External count clock input for timer/counter 1
- **OC0B**: Timer/Event Counter 0 Group B compare match output. PD5 can be used as the external output of timer/counter 0 compare match. At this point the pin must be set as an output via DDD5. At the same time, OC0B is also the PWM mode output pin of Timer 0
- **PCINT21**: Pin Change Interrupt 21

XCK/T0/DAO/PCINT20 - Port D pin 4

- **XCK**: External clock signal of USART in synchronous mode
- **T0**: External count clock input for timer/counter 0
- **DAO**: Internal 8-bit DAC analog output
- **PCINT20**: Pin Change Interrupt 20

INT1/OC2B/PCINT19 - Port D pin 3

- **INT1**: External interrupt input 1
- **OC2B**: Timer/Event Counter 2 group B compare match output. PD3 can be used as the external output of timer/counter 2 compare match. At this point the pin must be set as an output via DDD3. At the same time, OC2B is also the PWM mode output pin of Timer 2
- **PCINT19**: Pin Change Interrupt 19

INT0/OC3B/AC0O/PCINT18 - Port D Pin 2

- **INT0**: External interrupt input 0
- **OC3B**: Timing counter 3 comparison match output B. Only in QFP32 package, PD2 and QFP48/PF2 are combined into one IO, so the OC3B function on PF2 will also be output from PD2
- **AC0O**: The analog comparator 0 comparison result is output directly. Controlled by the AC0FR register
- **PCINT18**: Pin Change Interrupt 18

TXD/OC3A/PCINT17 - Port D pin 1

- **TXD**: Transmit data (USART data output). After the USART transmitter is enabled, PD1 will be forced to output, not controlled by DDD1
- **OC3A**: Timer counter 3 comparison match output A. Only in QFP32 package, PD1 and QFP48/PF1 are combined into one IO, so the OC3A function on PF1 will also be output from PD1
- **PCINT17**: Pin Change Interrupt 17

RXD/PCINT16 - Port D pin 0

- **RXD**: transmit data (USART data input). After the USART receiver is enabled, PD0 will be forced as an input, not controlled by DDD0. After the pin is forced as an input by the USART, the pull-up resistor can still be controlled by the PORTD0 bit
- **PCINT16**: Pin Change Interrupt 16

Port E multiplexing function

Pin	Multiplexing function description
PE7	ADC11 (ADC input channel 11) PCINT31 (pin change interrupt 31)
PE6	AVREF (QFP32: ADC external reference voltage) ADC10 (ADC input channel 10) PCINT30 (pin change interrupt 30)
PE5	CLKO (system clock output) AC1O (analog comparator 1 output) PCINT29 (pin change interrupt 29)
PE4	OC0A (timer/counter 0 compare configuration output A) PCINT28 (pin change interrupt 28)
PE3	ADC7 (ADC input channel 7) AC1N (analog comparator 1 negative input) PCINT27 (pin change interrupt 27)
PE2	SWD (SWD debugger data line) PCINT26 (pin change interrupt 26)
PE1	ADC6 (ADC input channel 6) ACXP (analog ratio machine 0/1 common positive input) PCINT25 (pin change interrupt 25)
PE0	SWC (SWD debugger clock input) APN4 (differential amplifier reverse input 4) PCINT24 (pin change interrupt 24)

ADC11/PCINT31 - Port E pin 7

ADC11: ADC external input channel 11
PCINT31: Pin Change Interrupt 30

AVREF/ADC10/PCINT30 - Port E pin 6

- AVREF: ADC external reference power input, when used as an analog function, the corresponding digital I/O needs to be set as an input, and the pull-up resistor should be turned off to avoid the digital circuit from interfering with the analog circuit
- ADC10: ADC analog input channel 10
- PCINT30: Pin Change Interrupt 30

CLKO/AC1O/PCINT29 - Port E pin 5

- CLKO: This function is the same as the CLKO function of PB0. Can be used as an alternate pin for PB0/CLKO
- AC1O: Analog Comparator 1 Output
- PCINT29: Pin Change Interrupt 29

OC0A/PCINT28 - Port E pin 4

- **OC0A**: Timer/Event Counter 0 group A compare match output. PE4 can be used as the external output of timer/counter 0 compare match. At this point the pin must be set as an output via DDE4. At the same time, OC0A is also the PWM mode output pin of Timer 0.
- **PCINT28**: Pin Change Interrupt 28

ADC7/ AC1N/PCINT27 - Port E pin 3

- **ADC7**: ADC input channel 7. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC
- **AC1N**: Analog Comparator 1 Negative Input
- **PCINT27**: Pin Change Interrupt 27

SWD/PCINT26 - Port E pin 2

- **SWD**: SWD debugger data line. PE2 defaults to the SWD function. The user can turn off the SWD debugger function by setting the SWDD bit of the MCUSR register to 1. After SWD is closed, the debugging function will not be available.
- **PCINT26**: Pin Change Interrupt 26

ADC6/ACXP/PCINT25 - Port E pin 1

- **ADC6**: ADC input channel 6. The DIDR register is used to turn off the digital function of the digital-analog multiplexing I/O to avoid the influence of the digital part on the analog circuit. For details, please refer to the relevant chapters of ADC
- **ACXP**: Analog Comparator 0/1 Common Positive Input
- **PCINT25**: Pin Change Interrupt 25

SWC/APN4/PCINT24 - Port E pin 0

- **SWC**: SWD debugger clock line. PE0 defaults to SWC function. The user can turn off the SWD debugger function by setting the SWDD bit of the MCUSR register to 1. After SWD is closed, the debugging function will not be available
- **APN4**: Differential Amplifier Inverting Input 4
- **PCINT24**: Pin Change Interrupt 24

Port F multiplexing function

Pin	Multiplexing function description
PF7	OC2B (timer/counter 2 compare match output B) PCINT39 (pin change interrupt 39)
PF6	T3 (timer/counter 3 external clock input) OC2A (timer/counter 2 compare match output A) PCINT38 (pin change interrupt 38)
PF5	OC1A (timer/counter 1 compare match output A) PCINT37 (pin change interrupt 37)
PF4	OC1B (timer/counter 1 comparison configuration output B) ICP3 (timer/counter 3 external capture input) PCINT36 (pin change interrupt 36)
PF3	OC0B (Timer/Event Counter 0 Compare Configuration Output B) PCINT35 (pin change interrupt 35)
PF2	OC3B (timer/counter 3 compare match output B) PCINT34 (pin change interrupt 34)
PF1	OC3A (timer/counter 3 compare match output A) PCINT33 (pin change interrupt 33)
PF0	ADC9 (ADC external input channel 9) APN3 (Differential Amplifier Inverting Input 3) PCINT32 (pin change interrupt 32)

OC2B/PCINT39 - Port F pin 7

- **OC2B**: Timer/Event Counter 2 compare match output B. Output selection is controlled by the PMX1 register
- **PCINT39**: Pin Change Interrupt 39

OC2A/T3/PCINT38 - Port F pin 6

- **OC2A**: Timer/Counter 2 compare match output A. Output selection is controlled by the PMX1 register
- **T3**: Timer/Event Counter 3 external clock input
- **PCINT38**: Pin Change Interrupt 38

0C1A/PCINT37 - Port F pin 5

- **OC1A**: Timer/Event Counter 1 Compare Match Output A. Output selection is controlled by the PMX0 register
- **PCINT37**: Pin Change Interrupt 37

ICP3/OC1B/PCINT36 - Port F pin 4

- **OC1B**: Timer/Event Counter 1 group B compare match output. Output selection is controlled by the PMX0 register
- **ICP3**: Timer/Event Counter 3 External Capture Input
- **PCINT36**: Pin Change Interrupt 36

OC3C/OC0B/PCINT35 - Port F pin 3

- **OC0B**: Timer/Event Counter 0 Group B compare match output. Output selection is controlled by the PMX0 register
- **OC3C**: Group C compare match output of timer/counter 3
- **PCINT35**: Pin Change Interrupt 35

OC3B/PCINT34- Port F pin 2

- **OC3B**: Group B comparison match output of timer/counter 3
- **PCINT34**: Pin Change Interrupt 34

OC3A/PCINT33 - Port F pin 1

- **OC3A**: Group B compare match output of timer/counter 3. Output selection is controlled by the PMX1 register
- **PCINT33**: Pin Change Interrupt 33

ADC9/APN3/PCINT32 - Port F pin 0

- **ADC9**: ADC external mode input channel 9
- **APN3**: Differential Amplifier Inverting Input 3
- **PCINT32**: Pin Change Interrupt 32

register definition

Port B Output Data Register - PORTB

PORTB – Port B output data register								
PORTB: 0x05(0x25)				Default value: 0x00				
Bits	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PORTB	Group B Port Output Registers						

Port B Direction Register - DDRB

DDRB – Port B Direction Register								
DDRB: 0x04(0x24)				Default value: 0x00				
DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	DDB	Port B group direction control bit; 1 = output, 0 = input						

Port B Input Data Register - PINB

PINB – Port B input data register								
PINB: 0x03(0x23)				Default value: 0x00				
PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PINB	Group B port status register. Read PINB to get the current state of the port directly; Writing PINBn bit 1 will toggle the output state of PORTBn						

Port C Output Data Register - PORTC

PORTC – Port C output data register								
PORTC: 0x08(0x28)				Default value: 0x00				
PORTC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PORTC	Group C Port Output Registers						

Port C Direction Register - DDRC

DDRC – Port C Direction Register								
DDRC: 0x07(0x27)				Default value: 0x00				
DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	DDC	Group C port direction control bit; 1 = output, 0 = input						

Port C Input Data Register - PINC

PINC – Port C input data register								
PINB: 0x06(0x26)				Default value: 0x00				
PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PINC	Group C port status register; read PINC to get the current port status Writing to PINC will flip the current port output						

Port D Output Data Register - PORTD

PORTD – Port D output data register								
PORTD: 0x0B(0x2B)				Default value: 0x00				
Bits	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								

[7:0]	PORTD	Bank D port output register

Port D Direction Register - DDRD

DDRD - Port D Direction Register								
DDRD: 0x0A(0x2A)				Default value: 0x00				
DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	DDD	Group D port output direction control register						

Port D Input Data Register - PIND

PING – input data register for port								
PIND: 0x09(0x29)				Default value: 0x00				
PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PIND	Group D Port Status Register Read PIND to get the current port level status Write PINDn as 1, flip the state of the corresponding bit of PORTDn						

Port E output data register - PORTE

PORTE – Port E output data register								
PORTE: 0x0E(0x2E)				Default value: 0x00				
Bits	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PORTE	Group E Port Output Registers						

Port E Direction Register - DDRE

DDRE – Port E Direction Register								
DDRE: 0x0D(0x2D)				Default value: 0x00				
DDRE	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	DDE	Group E port direction control register						

Port E Input Data Register - PINE

PINE – Port E input data register								
PINE: 0x0C(0x2C)				Default value: 0x00				
PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PINE	Group E port status register Read PINE to get the current port level status Write PINEn to 1, flip the state of PORTEn bit						

Port F Output Register - PORTF

PINF – Port F Input Data Register								
PORTF: 0x14(0x34)				Default value: 0x00				
Bits	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PORTF	Group F Port Status Register Input mode port, writing 1 to the corresponding bit will enable the internal pull-up Output mode port, writing 1 to the corresponding bit will drive the output high level						

Port F Direction Control Register - DDRF

DDRF – Port F Direction Control Register								
DDRF: 0x13(0x33)				Default value: 0x00				
Bits	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	DDRF	Group F Port Direction Control Register						

Port F Status Register - PINF

PINF – Port F Status Register								
PINF: 0x12(0x32)				Default value: 0x00				
Bits	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								
[7:0]	PINF	Group F Port Status Register Read PINF to get the current level status of port F Write 1 to PINFn to flip the state of the corresponding bit of PORTFn						

Port Driver Control Register - HDR

HDR0 – Port Driver Control Register								
HDR: 0xE0				Default value: 0x00				
Bit	-	-	HDR5	HDR4	HDR3	HDR2	HDR1	HDR0
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								

[7:6]	-	Reserved
5	HDR5	PF5 output drive control; 1=80mA drive, 0=12mA drive
4	HDR4	PF4 output drive control; 1=80mA drive, 0=12mA drive
3	HDR3	PF2 output drive control; 1=80mA drive, 0=12mA drive
2	HDR2	PF1 output drive control; 1=80mA drive, 0=12mA drive
1	HDR1	PD6 output drive control; 1=80mA drive, 0=12mA drive
0	HDR0	PD5 output drive control; 1=80mA drive, 0=12mA drive

Port Multiplexing Control Register 0 - PMX0

PMX0 – Port Mux Control Register 0								
PMX0: 0xEE					Default value: 0x00			
Bit	WCE	C1BF4	C1AF5	C0BF3	C0AC0	SSB1	TXD6	RXD5
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit definition								

7	WCE	PMX0/1 update enable control; before updating the PMX0/1 register, you need to write the WCE bit to 1, and complete the update of PMX0/1 within the next 6 system cycles.
6	C1BF4	OC1B auxiliary output control 1 = OC1B output to PF4 0 = OC1B output to PB2
5	C1AF5	OC1A auxiliary output control 1 = OC1A output to PF5 0 = OC1A output to PB1
4	C0BF3	OC0B auxiliary output control 1 = OC0B output to PF3 0 = OC0B output to PD5
3	C0AC0	OC0A auxiliary output control The 0C0A output is jointly controlled by the C0AC0 bit and the C0AS bit of the TCCR0B register: { C0AC0, C0AS } = 00 = OC0A output to PD6 01 = 0C0A output to PE4 10 = 0C0A output to PC0 11 = OC0A outputs to PE4 and PC0 simultaneously
2	SSB1	SPSS Auxiliary Output Control 1 = SPSS output to PB1 0 = SPSS output to PB2
1	TXD6	Serial port TXD auxiliary output control 1 = TXD output to PD6, 0 = TXD output to PD1
0	RXD5	Serial port RXD auxiliary input control 1 = RXD input is from PD5, 0 = RXD input is from PD0

Port Multiplexing Control Register 1 - PMX1

PMX1 – Port Multiplexing Control Register 1								
PMX1: 0xED					Default value: 0x00			
Bit	-	-	-	-	-	C3AC	C2BF7	C2AF6
R/W	-	-	-	-	-	R/W	R/W	R/W
Bit definition								
[7:3]	-	Reserved						
2	C3AC	OC3A auxiliary output control 1 = OC3A output to QFP48/AC0P 0 = OC3A output to PF1						
1	C2BF7	OC2B auxiliary output control 1 = OC2B output to PF7 0 = OC2B output to PD3						
0	C2AF6	OC2A auxiliary output control 1 = OC2A output to PF6 0 = OC2A output to PB3						
Instructions for use								
PMX0/1 shared register update protection control bit PMX0[7], when updating PMX1, please refer to the control description of PMX0[7] in the PMX0 register.								

Port Multiplexing Control Register 2 – PMX2

PMX2 – Port Multiplexing Control Register 2								
PMX2: 0xF0					Default value: 0x00			
Bit	WCE	STSC1	STSC0	-	-	XIEN	E6EN	C6EN
R/W	R/W	R/W	R/W	-	-	R/W	R/W	R/W
位定义								
[7]	WCE	PMX2 update enable control; before updating the PMX2 register, you need to write the WCE bit to 1, and complete the update of PMX2 within the next 6 system cycles.						
[6]	STSC1	High-speed crystal oscillator IO startup circuit control After the high-speed crystal oscillator is enabled through PMCR, STSC1 is automatically enabled. When the system clock is switched to an external high-speed crystal, STSC1 is automatically cleared. The software can also manually clear STSC1 after the crystal oscillator is stable, and turn off the crystal oscillator startup circuit to save power consumption.						
[5]	STSC0	Low-speed crystal oscillator IO startup circuit control After the low-speed crystal oscillator is enabled through PMCR, STSC0 is automatically enabled. When the system clock is switched to an external low-speed crystal oscillator, STSC0 is cleared automatically. The software can also manually clear STSC0 after the crystal oscillator is stable, and turn off the crystal oscillator startup circuit to save power consumption.						
[4:3]	-	Reserved						
[2]	XIEN	Enable external clock input, need to enable external crystal oscillator at the same time						
[1]	E6EN	Enable the general IO function of PE6; the default PE6 is AVREF function						
[0]	C6EN	Enable the general-purpose IO function of PC6; by default, PC6 is an external reset input						

Interrupt on pin change

- ❑ 40 pin change interrupt sources
- ❑ 5 interrupt entries

Overview

Pin change interrupts are triggered by PBn, PCn, PDn, PEn and PFn pins. Interrupts can be triggered even if these pins are configured as outputs, as long as the pin-change interrupt is enabled. This can be used to generate a software interrupt.

Any enabled PBn pin flip will trigger pin level interrupt PCI0, enabled PCn pin flip will trigger PCI1, enabled PDn pin flip will trigger PCI2, and enabled PEn pin flip will trigger PCI3 . The enablement of each pin change interrupt is controlled by the PCMSK0~4 registers. All pin-change interrupts are detected asynchronously and can be used as wake-up sources in certain sleep modes.

register definition

Pin Change Interrupt list of registers

register	address	Default value	description
PCICR	0x68	0x00	Pin Change Interrupt Control Register
PCIFR	0x3B	0x00	Pin Change Interrupt Flag Register
PCMSK0	0x6B	0x00	Pin Change Interrupt Mask Register 0
PCMSK1	0x6C	0x00	Pin Change Interrupt Mask Register 1
PCMSK2	0x6D	0x00	Pin Change Interrupt Mask Register 2
PCMSK3	0x73	0x00	Pin Change Interrupt Mask Register 3
PCMSK4	0x74	0x00	Pin Change Interrupt Mask Register 4

PCICR – Pin Change Interrupt Control Register

PCICR – Pin Change Interrupt Control Register								
address: 0x68						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	PCIE4	PCIE3	PCIE2	PCIE1	PCIE0
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:5	-	Reserved
4	PCIE4	Pin Change Interrupt Enable Control Bit 4. When the PCIE4 bit is set to "1" and the global interrupt is enabled, the pin change interrupt 4 is enabled. A change in any of the enabled PFn pins will generate a PCI4 interrupt. The enable of PFn pin interrupt can be controlled by PCMSK4 register respectively. When setting the PCIE3 bit to "0", the pin change interrupt 3 is disabled.
3	PCIE3	Pin Change Interrupt Enable Control Bit 3. When the PCIE3 bit is set to "1" and the global interrupt is enabled, the pin

		change interrupt 3 is enabled. A change in any of the enabled PEn pins will generate a PCI3 interrupt. The enable of PEn pin interrupt can be controlled by PCMSK3 register respectively. When setting the PCIE3 bit to "0", the pin change interrupt 3 is disabled.
2	PCIE2	Pin change interrupt enable control bit 2. When the PCIE2 bit is set to "1" and the global interrupt is enabled, the pin change interrupt 2 is enabled. A change in any of the enabled PDn pins will generate a PCI2 interrupt. The enable of PDn pin interrupt can be controlled by PCMSK2 register respectively. When setting the PCIE2 bit to "0", the pin change interrupt 2 is disabled.
1	PCIE1	Pin change interrupt enable control bit 1. When the PCIE1 bit is set to "1" and the global interrupt is enabled, the pin change interrupt 1 is enabled. A change on any of the enabled PCn pins will generate a PCI1 interrupt. The enable of PCn pin interrupt can be controlled by PCMSK1 register respectively. When setting the PCIE1 bit to "0", the pin change interrupt 1 is disabled.
0	PCIE0	Pin change interrupt enable control bit 0. When the PCIE0 bit is set to "1" and the global interrupt is enabled, the pin change interrupt 0 is enabled. A change in any of the enabled PBn pins will generate a PCI0 interrupt. The enable of PBn pin interrupt can be controlled by PCMSK0 register respectively. When setting the PCIE0 bit to "0", the pin change interrupt 0 is disabled.

PCIFR – Pin Change Interrupt Flag Register

PCIFR – Pin Change Interrupt Flag Register								
address: 0x3B					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	PCIF4	PCIF3	PCIF2	PCIF1	PCIF0
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:5	-	Reserved
4	PCIF4	Pin change interrupt flag bit 4. A change on any of the enabled PFn pins will set PCIF4. When both PCIE4 and global interrupts are set, the MCU will jump to the PCI4 interrupt entry address. The enable of PFn pin interrupt can be controlled by PCMSK4 register respectively. Executing an interrupt service routine or writing "1" to the PCIF4 bit will clear the PCIF4 bit.
3	PCIF3	Pin change interrupt flag bit 3. A change on any of the enabled PEn pins will set PCIF3. When both PCIE3 and the global interrupt are set, the MCU will jump to the PCI3 interrupt entry address. The enable of PEn pin interrupt can be controlled by PCMSK3 register respectively. Executing the interrupt service routine or writing "1" to the PCIF3 bit will clear the PCIF3 bit.
2	PCIF2	Pin change interrupt flag bit 2. A change on any of the enabled PDn pins will set PCIF2. When both PCIE2 and the global interrupt are set, the MCU will jump to the PCI2 interrupt entry address. The enable of PDn pin interrupt can be controlled by PCMSK2 register respectively. Executing the interrupt service routine or writing "1" to the PCIF2 bit will clear the PCIF2 bit.
1	PCIF1	Pin change interrupt flag bit 1.

		A change on any of the enabled PCn pins will set PCIF1. When both PCIE1 and the global interrupt are set, the MCU will jump to the PCI1 interrupt entry address. The enable of PCn pin interrupt can be controlled by PCMSK1 register respectively. Executing the interrupt service routine or writing "1" to the PCIF1 bit will clear the PCIF1 bit.
0	PCIF0	Pin change interrupt flag bit 0. A change on any of the enabled PBn pins will set PCIF0. When both PCIE0 and the global interrupt are set, the MCU will jump to the PCI0 interrupt entry address. The enable of PBn pin interrupt can be controlled by PCMSK0 register respectively. Executing the interrupt service routine or writing "1" to the PCIF0 bit will clear the PCIF0 bit.

PCMSK0 – Pin Change Interrupt Mask Register 0

PCMSK0 – Pin Change Mask Register 0								
address: 0x6B					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	PCINT7	Pin change enable mask bit 7. When setting the PCINT7 bit to "1", the PB7 pin level change interrupt is enabled. A level change on the PB7 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT7 bit to "0", the PB7 pin level change interrupt is disabled.
6	PCINT6	Pin Change Enable Mask Bit 6. When setting the PCINT6 bit to "1", the PB6 pin level change interrupt is enabled. A level change on the PB6 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT6 bit to "0", the PB6 pin level change interrupt is disabled.
5	PCINT5	Pin change enable mask bit 5. When setting the PCINT5 bit to "1", the PB5 pin level change interrupt is enabled. A level change on the PB5 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT5 bit to "0", the PB5 pin level change interrupt is disabled.
4	PCINT4	Pin change enable mask bit 4. When setting the PCINT4 bit to "1", the PB4 pin level change interrupt is enabled. A level change on the PB4 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT4 bit to "0", the PB4 pin level change interrupt is disabled.
3	PCINT3	Pin change enable mask bit 3. When setting the PCINT3 bit to "1", the PB3 pin level change interrupt is enabled. A level change on the PB3 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT3 bit to "0", the PB3 pin level change interrupt is disabled.
2	PCINT2	Pin change enable mask bit 2. When setting the PCINT2 bit to "1", the PB2 pin level change interrupt is enabled. A level change on the PB2 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT2 bit to "0", the PB2 pin level change interrupt is disabled.
1	PCINT1	Pin change enable mask bit 1. When setting the PCINT1 bit to "1", the PB1 pin level change interrupt is

		enabled. A level change on the PB1 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT1 bit to "0", the PB1 pin level change interrupt is disabled.
0	PCINT0	Pin change enable mask bit 0. When setting the PCINT0 bit to "1", the PB0 pin level change interrupt is enabled. A level change on the PB0 pin will set PCIF0, and if the PCIE0 bit and the global interrupt are set, a PCI0 interrupt will be generated. When setting the PCINT0 bit to "0", the PB0 pin level change interrupt is disabled.

PCMSK1 – Pin Change Interrupt Mask Register 1

PCMSK1 – Pin Change Mask Register 1								
address: 0x6C					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	PCINT15	Pin change enable mask bit 15. When setting the PCINT15 bit to "1", the PC7 pin level change interrupt is enabled. A level change on the PC7 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT15 bit to "0", the PC7 pin level change interrupt is disabled.
6	PCINT14	Pin change enable mask bit 14. When setting the PCINT14 bit to "1", the PC6 pin level change interrupt is enabled. A level change on the PC6 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT14 bit to "0", the PC6 pin level change interrupt is disabled.
5	PCINT13	Pin change enable mask bit 13. When setting the PCINT13 bit to "1", the PC5 pin level change interrupt is enabled. A level change on the PC5 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT13 bit to "0", the PC5 pin level change interrupt is disabled.
4	PCINT12	Pin change enable mask bit 12. When setting the PCINT12 bit to "1", the PC4 pin level change interrupt is enabled. A level change on the PC4 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT12 bit to "0", the PC4 pin level change interrupt is disabled.
3	PCINT11	Pin change enable mask bit 11. When setting the PCINT11 bit to "1", the PC3 pin level change interrupt is enabled. A level change on the PC3 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT11 bit to "0", the PC3 pin level change interrupt is disabled.
2	PCINT10	Pin change enable mask bit 2. When setting the PCINT10 bit to "1", the PC2 pin level change interrupt is enabled. A level change on the PC2 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT10 bit to "0", the PC2 pin level change interrupt is disabled.

1	PCINT9	Pin change enable mask bit 1. When setting the PCINT9 bit to "1", the PC1 pin level change interrupt is enabled. A level change on the PC1 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT9 bit to "0", the PC1 pin level change interrupt is disabled.
0	PCINT8	Pin change enable mask bit 0. When setting the PCINT8 bit to "1", the PC0 pin level change interrupt is enabled. A level change on the PC0 pin will set PCIF1, and if the PCIE1 bit and the global interrupt are set, a PCI1 interrupt will be generated. When setting the PCINT8 bit to "0", the PC0 pin level change interrupt is disabled.

PCMSK2 – Pin Change Interrupt Mask Register 2

PCMSK2 – Pin Change Mask Register 2								
address: 0x6D					Default value: 0x00			
Bits	7	6	5	4	3	2	1	0
	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	description						
7	PCINT23	Pin change enable mask bit 23. When setting the PCINT23 bit to "1", the PD7 pin level change interrupt is enabled. A level change on the PD7 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT23 bit to "0", the PD7 pin level change interrupt is disabled.						
6	PCINT22	Pin Change Enable Mask Bit 6. When setting the PCINT22 bit to "1", the PD6 pin level change interrupt is enabled. A level change on the PD6 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT22 bit to "0", the PD6 pin level change interrupt is disabled.						
5	PCINT21	Pin change enable mask bit 21. When setting the PCINT21 bit to "1", the PD5 pin level change interrupt is enabled. A level change on the PD5 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT21 bit to "0", the PD5 pin level change interrupt is disabled.						
4	PCINT20	Pin change enable mask bit 20. When setting the PCINT20 bit to "1", the PD4 pin level change interrupt is enabled. A level change on the PD4 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT20 bit to "0", the PD4 pin level change interrupt is disabled.						
3	PCINT19	Pin Change Enable Mask Bit 19. When setting the PCINT19 bit to "1", the PD3 pin level change interrupt is enabled. A level change on the PD3 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT19 bit to "0", the PD3 pin level change interrupt is disabled.						
2	PCINT18	Pin Change Enable Mask Bit 18.						

		When setting the PCINT18 bit to "1", the PD2 pin level change interrupt is enabled. A level change on the PD2 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT18 bit to "0", the PD2 pin level change interrupt is disabled.
1	PCINT17	Pin Change Enable Mask Bit 17. When setting the PCINT17 bit to "1", the PD1 pin level change interrupt is enabled. A level change on the PD1 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT17 bit to "0", the PD1 pin level change interrupt is disabled.
0	PCINT16	Pin change enable mask bit 16. When setting the PCINT16 bit to "1", the PD0 pin level change interrupt is enabled. A level change on the PD0 pin will set PCIF2, and if the PCIE2 bit and the global interrupt are set, a PCI2 interrupt will be generated. When setting the PCINT16 bit to "0", the PD0 pin level change interrupt is disabled.

PCMSK3 – Pin Change Interrupt Mask Register 3

PCMSK3 – Pin Change Mask Register 3								
address: 0x73					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT31	PCINT30	PCINT29	PCINT28	PCINT27	PCINT26	PCINT25	PCINT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	description						
7	PCINT31	Pin Change Enable Mask Bit 31. When setting the PCINT31 bit to "1", the PE7 pin level change interrupt is enabled. A level change on the PE7 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT31 bit to "0", the PE7 pin level change interrupt is disabled.						
6	PCINT30	Pin Change Enable Mask Bit 30. When setting the PCINT30 bit to "1", the PE6 pin level change interrupt is enabled. A level change on the PE6 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT30 bit to "0", the PE6 pin level change interrupt is disabled.						
5	PCINT29	Pin Change Enable Mask Bit 39. When setting the PCINT29 bit to "1", the PE5 pin level change interrupt is enabled. A level change on the PE5 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT29 bit to "0", the PE5 pin level change interrupt is disabled.						
4	PCINT28	Pin Change Enable Mask Bit 28. When setting the PCINT28 bit to "1", the PE4 pin level change interrupt is enabled. A level change on the PE4 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT28 bit to "0", the PE4 pin level change interrupt is disabled.						
3	PCINT27	Pin Change Enable Mask Bit 27.						

		When setting the PCINT27 bit to "1", the PE3 pin level change interrupt is enabled. A level change on the PE3 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT27 bit to "0", the PE3 pin level change interrupt is disabled.
2	PCINT26	Pin change enable mask bit 26. When setting the PCINT26 bit to "1", the PE2 pin level change interrupt is enabled. A level change on the PE2 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT26 bit to "0", the PE2 pin level change interrupt is disabled.
1	PCINT25	Pin Change Enable Mask Bit 25. When setting the PCINT25 bit to "1", the PE1 pin level change interrupt is enabled. A level change on the PE1 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT25 bit to "0", the PE1 pin level change interrupt is disabled.
0	PCINT24	Pin change enable mask bit 24. When setting the PCINT24 bit to "1", the PE0 pin level change interrupt is enabled. A level change on the PE0 pin will set PCIF3, and if the PCIE3 bit and the global interrupt are set, a PCI3 interrupt will be generated. When setting the PCINT24 bit to "0", the PE0 pin level change interrupt is disabled.

PCMSK4 – Pin Change Interrupt Mask Register 4

PCMSK4 – Pin Change Mask Register 4								
address: 0x74					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT39	PCINT38	PCINT37	PCINT36	PCINT35	PCINT34	PCINT33	PCINT32
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	PCINT39	Pin Change Enable Mask Bit 39. When setting the PCINT39 bit to "1", the PF7 pin level change interrupt is enabled. A level change on the PF7 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT39 bit to "0", the PF7 pin level change interrupt is disabled.
6	PCINT38	Pin Change Enable Mask Bit 38. When setting the PCINT38 bit to "1", the PF6 pin level change interrupt is enabled. A level change on the PF6 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT38 bit to "0", the PF6 pin level change interrupt is disabled.
5	PCINT37	Pin Change Enable Mask Bit 37. When setting the PCINT37 bit to "1", the PF5 pin level change interrupt is enabled. A level change on the PF5 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT37 bit to "0", the PF5 pin level change interrupt is disabled.
4	PCINT36	Pin change enable mask bit 36.

		When setting the PCINT36 bit to "1", the PF4 pin level change interrupt is enabled. A level change on the PF4 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT36 bit to "0", the PF4 pin level change interrupt is disabled.
3	PCINT35	Pin Change Enable Mask Bit 35. When setting the PCINT35 bit to "1", the PF3 pin level change interrupt is enabled. A level change on the PF3 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT35 bit to "0", the PF3 pin level change interrupt is disabled.
2	PCINT34	Pin change enable mask bit 34. When setting the PCINT34 bit to "1", the PF2 pin level change interrupt is enabled. A level change on the PF2 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT34 bit to "0", the PF2 pin level change interrupt is disabled.
1	PCINT33	Pin change enable mask bit 33. When setting the PCINT33 bit to "1", the PF1 pin level change interrupt is enabled. A level change on the PF1 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT33 bit to "0", the PF1 pin level change interrupt is disabled.
0	PCINT32	Pin change enable mask bit 32. When setting the PCINT31 bit to "1", the PF0 pin level change interrupt is enabled. A level change on the PF0 pin will set PCIF4, and if the PCIE4 bit and the global interrupt are set, a PCI4 interrupt will be generated. When setting the PCINT32 bit to "0", the PF0 pin level change interrupt is disabled.

Timer/Event Counter 0 (TMR0)

- ❑ 8-bit counter
- ❑ Two independent comparison units
- ❑ When a comparison match occurs, the counter is automatically cleared and automatically loaded
- ❑ Phase-corrected PWM output without interference pulse
- ❑ Frequency generator
- ❑ External event counter
- ❑ 10-bit clock prescaler
- ❑ Overflow and compare match interrupt
- ❑ With dead time control
- ❑ 6 optional trigger sources to automatically turn off the PWM output
- ❑ Generate high-speed and high-resolution (500KHz@7Bit) PWM in high-speed clock mode

Overview

TC0 is a general-purpose 8-bit timer counter module that supports PWM output and can generate waveforms precisely. TC0 includes a counting clock generation unit, an 8-bit counter, a waveform generation mode control unit and 2 output comparison units. At the same time, TC0 can share the 10-bit prescaler with TC1, or use the 10-bit prescaler independently. The prescaler divides the system clock clkio or the high-speed clock rcm2x (2 times the frequency of the internal 32M RC oscillator output clock rc32m) to generate the count clock Clkt0. The waveform generation mode control unit controls the working mode of the counter and the generation of the comparison output waveform. According to different working modes, the counter implements clearing, adding one or subtracting one operation for each counting clock Clkt0. Clkt0 can be generated by internal clock source or external clock source. When the count value TCNT0 of the counter reaches the maximum value (equal to the maximum value 0xFF or the output comparison register OCR0A, which is defined as TOP, and the maximum value is defined as MAX to show the difference), the counter will be cleared or subtracted by one. When the count value TCNT0 of the counter reaches the minimum value (equal to 0x00, defined as BOTTOM), the counter will add one. When the count value TCNT0 of the counter reaches OCR0A/OCR0B, also known as when a comparison match occurs, it will be cleared or set to output the comparison signal OC0A/OC0B to generate a PWM waveform. When the dead time insertion is enabled, the set dead time (the number of count clocks corresponding to the DTR0 register) will be inserted into the generated PWM waveform. The software can turn off the waveform output of OC0A/OC0B by clearing the COM0A/COM0B bit to zero, or set the corresponding trigger source. When the trigger event occurs, the hardware automatically clears the COM0A/COM0B bit to turn off the OC0A/OC0B waveform output.

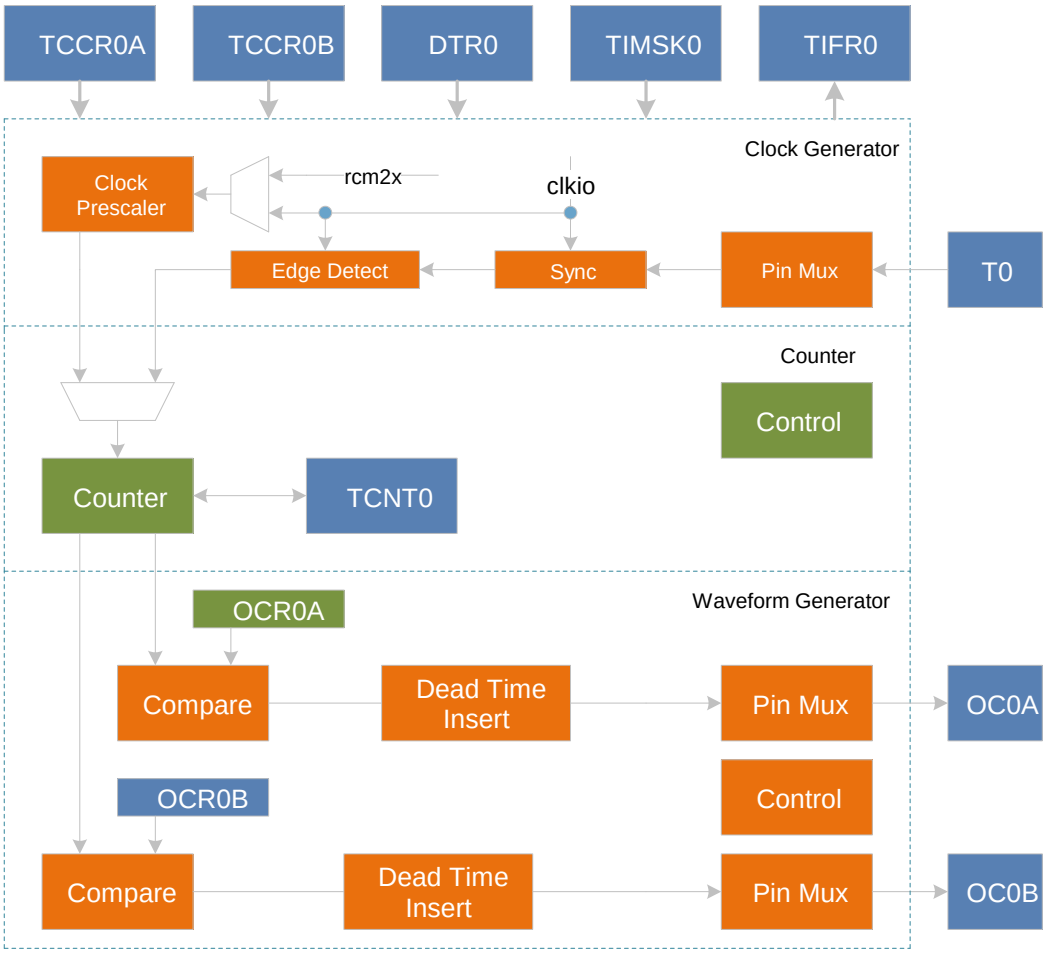
The counting clock can be generated by internal or external clock sources. The selection of clock source and frequency division is controlled by the CS0 bit in the TCCR0B register. For details, see the TC0 and TC1 prescaler chapters.

The length of the counter is 8 bits and supports bidirectional counting. The waveform generation mode, that is, the working mode of the counter, is controlled by the WGM0 bit located in the TCCR0A and TCCR0B registers. According to different working modes, the counter implements clearing, adding one or subtracting one operation for each counting clock Clkt0. When the count overflows, the count overflow flag TOV0 bit in the TIFR0 register will be set. TC0 count overflow interrupt can be generated when the interrupt is enabled.

The output comparison unit compares the count value TCNT0 with the values of the output comparison registers OCR0A and OCR0B. When TCNT0 is equal to OCR0A or OCR0B, it is called a comparison match, and the output comparison flag OCF0A or OCF0B in the TIFR0 register will be set. TC0 output compare match interrupt can be generated when the interrupt is enabled.

It should be noted that in PWM mode, the OCR0A and OCR0B registers are double-buffered registers. In normal mode and CTC mode, the double buffering function is invalid. When the count reaches the maximum or minimum value, the value in the buffer register is synchronously updated to compare registers OCR0A and OCR0B. For details, see the chapter description of working mode.

The waveform generator generates output comparison waveform signals OC0A and OC0B using compare match, count overflow, etc. according to the waveform generation mode control and the comparison output mode control. The specific generation method is described in the working mode and register chapter. When the output comparison waveform signals OC0A and OC0B are to be output to corresponding pins, the data direction register of the pin must also be set as output.
The figure below shows the internal structure of TC0. TC0 includes a counting clock generation unit, an 8-bit counter, 2 output comparison units and 2 waveform generation control units.



TC0 structure diagram

Operating mode

Timing counter 0 has four different working modes, including normal mode (Normal), clear when compare match (CTC) mode, fast pulse width modulation (FPWM) mode and phase correction pulse width modulation (PCPWM) mode, generated by waveform Mode control bits WGM0[2:0] to select. These four modes are described in detail below. Since there are two independent output comparison units, they are represented by "A" and "B" respectively, and the lowercase "x" is used to represent the two output comparison unit channels.

Normal mode

The normal mode is the simplest working mode of the timer counter. At this time, the waveform generation mode control bit WGM0[2:0]=0, and the maximum value TOP of the count is MAX (0xFF). In this mode, the counting method increases by one for each counting clock, and when the counter reaches TOP and overflows, it returns to BOTTOM and starts accumulating again. Set the timer counter overflow flag TOV0 in the same count clock that the count value TCNT0 becomes zero. In this mode, the TOV0 flag is like the 9th counting bit, but it will only be set and not cleared. The overflow interrupt service routine will automatically clear the TOV0 flag, which can be used by software to increase the resolution of the timer counter. In normal mode, there is no special situation to consider, and a new count value can be written at any time.
The waveform of the output comparison signal OC0x can only be obtained when the data direction register of the OC0x pin is set to output. When COM0x=1, the OC0x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{oc0xnormal} = f_{sys}/(2*N*256)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
The output compare unit can be used to generate interrupts, but it is not recommended to use interrupts in normal mode, which will take up too much CPU time.

CTC mode

When setting WGM0[2:0]=2, the timing counter 0 enters the CTC mode, and the maximum value TOP of counting is OCR0A. In this mode, the counting method is incremented by one for each count clock, and the counter is cleared when the counter value TCNT0 is equal to TOP. OCR0A defines the maximum value of the count, which is the resolution of the counter. This mode allows the user to easily control the frequency of the compare match output, and also simplifies the operation of counting external events.
When the counter reaches the maximum count value, the output comparison match flag OCF0 is set, and an interrupt will be generated when the corresponding interrupt enable is set. In the interrupt service routine, the OCR0A register can be updated, that is, the maximum value of the count. OCR0A does not use double buffering in this mode, be careful when updating the maximum value close to the minimum value when the counter is operating with no prescaler or a very low prescaler. If the value written into OCR0A is less than the current TCNT0 value, the counter will lose a compare match. Before the next compare match occurs, the counter has to count to TOP first, and then start counting from BOTTOM to the OCR0A value. Same as the normal mode, the count value returns to the count clock of BOTTOM to set the TOV0 flag.
The waveform of the output comparison signal OC0x can only be obtained when the data direction register of the OC0x pin is set to output. When COM0x=1, the OC0x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{oc0xctc} = f_{sys}/(2*N*(1+OCR0x))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
It can be seen from the formula that when OCR0A is set to 0x0 and there is no prescaler, an output waveform with a maximum frequency of fsys/2 can be obtained.

Fast PWM mode

When WGM0[2:0]=3 or 7, the timing counter 0 enters the fast PWM mode, which can be used to generate high-frequency PWM waveforms, and the maximum count TOP is MAX (0xFF) or OCR0x respectively. Fast PWM mode differs from other PWM modes in that it operates unidirectionally. The counter accumulates from the minimum value 0x00 to TOP and then returns to BOTTOM to count again.
When the count value TCNT0 reaches OCR0x or BOTTOM, the output comparison signal OC0x will be set or cleared, depending on the setting of the comparison output mode COM0x, see the register description for details. Due to the unidirectional operation, the fast PWM mode operates at twice the frequency of the phase-corrected PWM mode with bidirectional operation. The high frequency characteristics make the fast PWM mode suitable for power regulation, rectification and DAC applications. High-frequency signals can reduce the size of external components (inductance and capacitance, etc.), thereby reducing system cost.

When the count value reaches the maximum value, the timer counter overflow flag TOV0 will be set, and the value of the comparison buffer will be updated to the comparison value. If the interrupt is enabled, the compare buffer OCR0x register can be updated in the interrupt service routine.
The waveform of the output comparison signal OC0x can only be obtained when the data direction register of the OC0x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc0xfpwm} = f_{sys}/(N*(1+TOP))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
When a comparison match occurs between TCNT0 and OCR0x, the waveform generator will set (clear) the OC0x signal, and when TCNT0 is cleared, the waveform generator will clear (set) the OC0x signal to generate a PWM wave. Therefore, the extreme value of OCR0x will generate a special PWM waveform. When OCR0x is set to 0x00, the output PWM has a narrow peak pulse every (1+TOP) count clock. When OCR0x is set to the maximum value, the output waveform is a continuous high level or low level.

Phase corrected PWM mode

When setting WGM0[2:0]=1 or 5, the timing counter 0 enters the phase correction PWM mode, and the maximum value TOP of counting is MAX (0xFF) or OCR0A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT0 matches OCR0x, the output comparison signal OC0x will be cleared or set, depending on the setting of the comparison output mode COM0x. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.
In the phase correction PWM mode, when the count reaches BOTTOM, the TOV0 flag is set, and when the count reaches TOP, the value of the comparison buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR0x register can be updated in the interrupt service routine.
The waveform of the output comparison signal OC0x can only be obtained when the data direction register of the OC0x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc0xpcpwm} = f_{sys}/(N*TOP*2)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
During the count-up process, when TCNT0 matches OCR0x, the waveform generator clears (sets) the OC0x signal. During the countdown process, when TCNT0 matches OCR0x, the waveform generator sets (clears) the OC0x signal. Therefore, the extreme value of OCR0x will generate a special PWM wave. When OCR0x is set to the maximum or minimum value, the OC0x signal output will keep low or high.
In order to ensure the symmetry of the output PWM wave on both sides of the minimum value, when there is no comparison match, the OC0x signal will be inverted in two cases. The first case is when the value of OCR0x changes from the maximum value 0xFF to other data. When OCR0x is the maximum value and the count value reaches the maximum value, the output of OC0x is the same as the result of comparing and matching when counting in descending order, that is, keep OC0x unchanged. At this time, the comparison value will be updated to a new OCR0x value (not 0xFF), and the value of OC0x will remain until it is flipped when a comparison match occurs when counting up. At this time, the OC0x signal is not symmetrical about the minimum value, so it is necessary to flip the OC0x signal when TCNT0 reaches the maximum value, that is, the first case where the OC0x signal is flipped when there is no comparison match. The second case is that when TCNT0 starts counting from a value higher than OCR0x, a compare match will be lost, causing an asymmetrical situation. It is also necessary to flip the OC0x signal to achieve symmetry on both sides of the minimum.

Automatic shutdown and restart of PWM output

When the DOC0x bit of the TCCR0A register is set high, the automatic shutdown function of the PWM output will be enabled. When the trigger condition is met, the hardware will clear the corresponding COM0x bit, disconnect the PWM output signal OC0x from its output pin, and switch to General IO output to realize automatic shutdown of PWM output.
At this time, the state of the output pin can be controlled by the output of the general-purpose IO port.

After the automatic shutdown of the PWM output is enabled, the trigger condition needs to be set, and the trigger source is selected by the DSX0n bit of the TCCR0C register. Trigger sources include analog comparator interrupt, external interrupt, pin level change interrupt and timer overflow interrupt. For details, please refer to the description of the TCCR0C register. When one or some trigger sources are selected as the trigger conditions, when these interrupt flag bits are set, the hardware will clear the COM0x bit to turn off the PWM output.

When a trigger event occurs and the PWM output is turned off, the timer module does not have a corresponding interrupt flag bit, and the software needs to read the interrupt flag bit of the trigger source to know the trigger condition and trigger event.

When the PWM output is automatically turned off and the output needs to be restarted again, the software only needs to reset the COM0x bit to switch the OC0x signal output to the corresponding pin. It should be noted that after the automatic shutdown occurs, the timer does not stop working, and the status of the OC0x signal has been updated. The software can set the COM0x bit to output the OC0x signal after the timer overflows or a comparison match occurs, so that a clear PWM output state can be obtained.

Dead time control

When the DTEN0 bit is set to "1", the function of inserting dead time is enabled, and the output waveforms of OC0A and OC0B will insert the set dead time on the basis of the waveform generated by the comparison output of channel B, and the length of time is DTR0 The time value corresponding to the count clock number of the register. As shown in the figure below, the dead time insertion of OC0A and OC0B is based on the comparative output waveform of channel B. When COM0A and COM0B are both "2" or "3", the waveform polarity of OC0A is the same as that of OC0B. When COM0A and COM0B are "2" or "3" respectively, the waveform of OC0A is the same as that of OC0B opposite polarity.

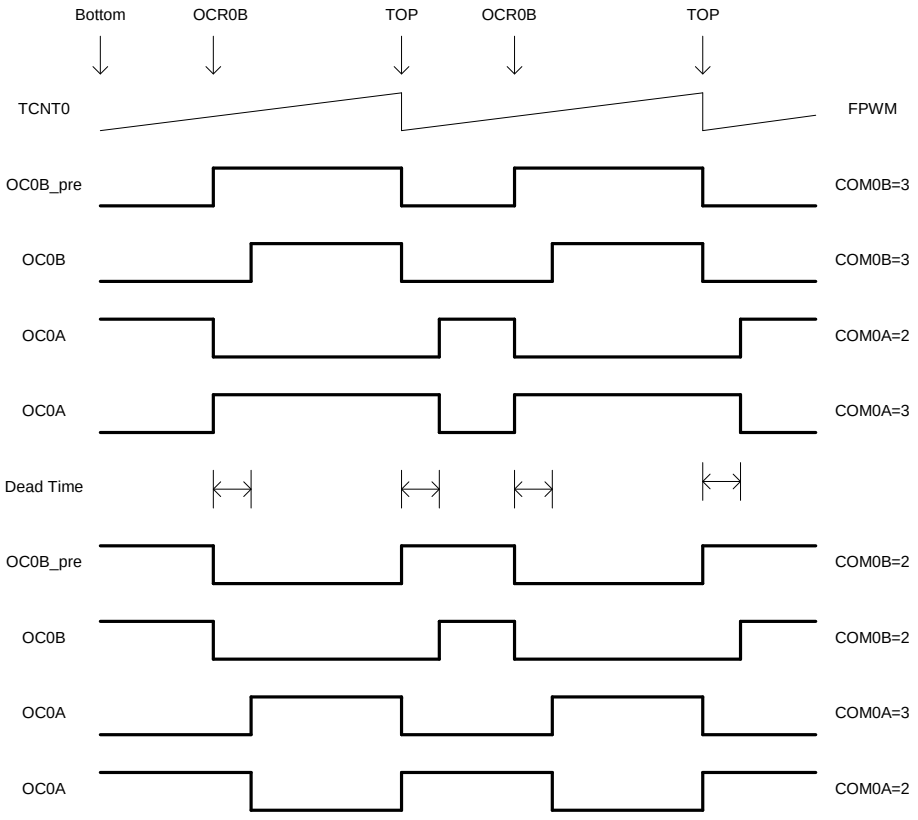


Figure 1 TC0 dead time control in FPWM mode

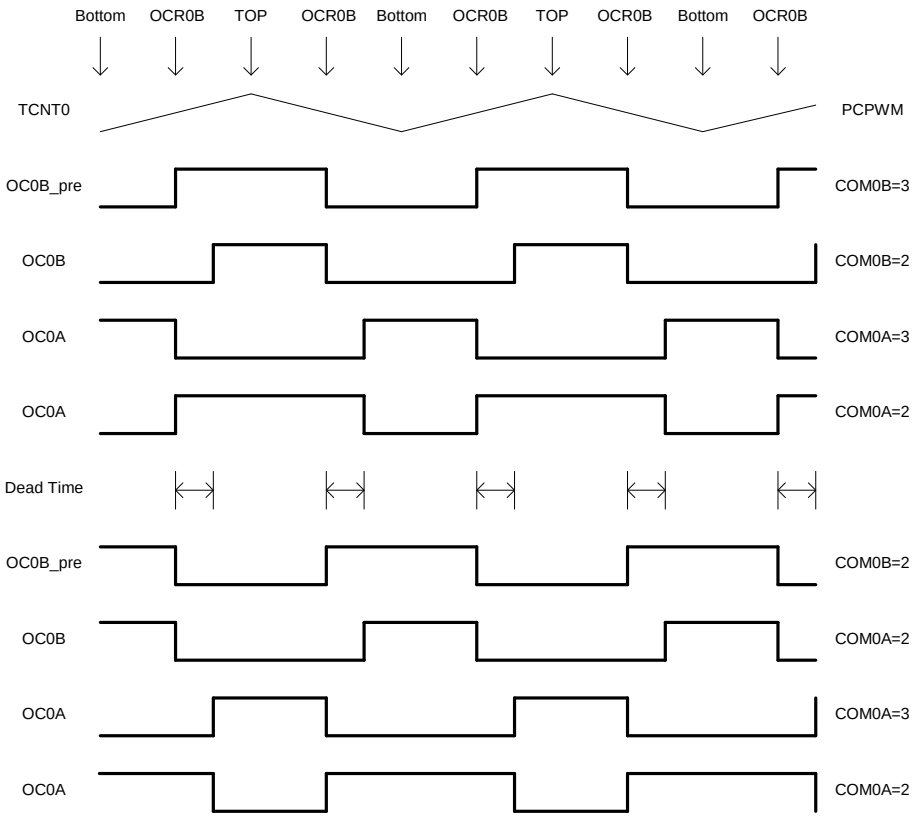


Figure 2 TC0 dead time control in PCPWM mode

When the DTEN0 bit is set to "0", the function of inserting dead time is disabled, and the output waveforms of OC0A and OC0B are the waveforms generated by their respective comparison outputs.

High speed clock mode

In the high-speed clock mode, a higher-frequency clock is used as the counting clock source to generate a higher-speed and higher-resolution PWM waveform. This high-frequency clock is generated by multiplying the output clock rc32m of the internal 32M RC oscillator by 2. Therefore, before entering the high-frequency mode, it is necessary to enable the frequency multiplication function of the internal 32M RC oscillator, that is, set the F2XEN bit of the TCKCSR register, and wait for a certain period of time until the frequency multiplication clock signal output is stable. Then, the TC2XS0 bit of TCKCSR can be set to make the timer counter enter the high-speed clock mode.

In this mode, the system clock is asynchronous to the high-speed clock, and some registers (see TC0 register list) work in the high-speed clock domain. Therefore, configuration and reading of such registers are also asynchronous, and attention should be paid to the operation.

There are no special requirements for discontinuous read and write operations on registers under the high-speed clock domain, but for continuous read and write operations, it is necessary to wait for a system clock, and the following steps can be followed:

1) Write register A;
2) Wait for a system clock (NOP or register under the operating system clock);
3) Read or write register A or B.
4) Wait for one system clock (NOP or register under OS clock).

When reading the registers in the high-speed clock domain, the registers except TCNT0 can be read directly. When the counter is still counting, the value of TCNT0 will change with the high-speed clock, and the counter can be suspended (set CS0 to zero) and then restarted. Read the value of TCNT0.

Register definition

TC0 register list

Register	Address	Default Value	Description
TCCR0A*	0x44	0x00	TC0 Control Register A
TCCR0B*	0x45	0x00	TC0 Control Register B
TCNT0*	0x46	0x00	TC0 count value register
OCR0A*	0x47	0x00	TC0 output compare register A
OCR0B*	0x48	0x00	TC0 output compare register B
DSX0*	0x49	0x00	TC0 trigger source control register
DTR0*	0x4F	0x00	TC0 Dead Time Register
TIMSK0	0x6E	0x00	Timer counter 0 interrupt mask register
TIFR0	0x35	0x00	Timer counter 0 interrupt flag register
TCKCSR	0xEC	0x00	TC Clock Control and Status Register

Notice

The registers with "*" work in the system clock and high-speed clock domain, and the registers without "*" only work in the system clock domain.

TC0 Control Register A - TCCR0A

TCCR0A – TC0 Control Register A								
address: 0x44						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
	COM0A1	COM0A0	COM0B1	COM0B0	DOC0B	DOC0A	WGM01	WGM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	COM0A1	TC0 compare match A output mode control high bit. COM0A1 and COM0A0 together form a comparative output mode to control COM0A[1:0], which is used to control the output waveform of OC0A. If both bits 1 or 2 of COM0A are set, the output comparison waveform occupies the OC0A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM0A controls the output comparison waveform differently, see the comparison output mode control table description for details.
6	COM0A0	TC0 compare match A output mode control low bit. COM0A0 and COM0A1 together form a comparison output mode control COM0A[1:0], which is used to control the output waveform of OC0A. If both bits 1 or 2 of COM0A are set, the output comparison waveform occupies the OC0A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM0A controls the output comparison waveform differently, see the comparison output mode control table description for details.
5	COM0B1	TC0 compare match B output mode control high bit. COM0B1 and COM0B0 together form a comparison output mode to control COM0B[1:0], which is used to control the output waveform of OC0B. If 1 or 2 bits of COM0B are set, the output comparison waveform occupies the OC0B pin, but the data direction

		register of this pin must be set high to output this waveform. In different working modes, COM0B controls the output comparison waveform differently, see the comparison output mode control table description for details.
4	COM0B0	TC0 compare match B output mode control low bit. COM0B0 and COM0B1 together form a comparative output mode control COM0B[1:0], which is used to control the output waveform of OC0B. If 1 or 2 bits of COM0B are set, the output comparison waveform occupies the OC0B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM0B controls the output comparison waveform differently, see the comparison output mode control table description for details.
3	DOC0B	TC0 closes the output comparison enable control high bit. When the DOC0B bit is "1", the trigger source is turned off and the output comparison signal OC0B is enabled. When a trigger event occurs, the hardware automatically clears the COM0B bit and turns off the waveform output of OC0B. The software can re-enable the PWM output by setting COMB. When the DOC0B bit is "0", the trigger source is turned off and the output comparison signal OC0B is disabled.
2	DOC0A	TC0 closes the output comparison enable control low bit. When the DOC0A bit is set to "1", the trigger source is turned off and the output comparison signal OC0A is enabled. When a trigger event occurs, the hardware automatically turns off the waveform output of OC0A. When the DOC0A bit is set to "0", the trigger source is turned off and the output comparison signal OC0A is disabled. When a trigger event occurs, the waveform output of OC0A is not turned off.
1	WGM01	TC0 waveform generation mode control bit. WGM01, WGM00, and WGM02 form a waveform generation mode to control WGM0[2:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
0	WGM00	TC0 waveform generation mode control low bit. WGM00, WGM01, and WGM02 form a waveform generation mode to control WGM0[2:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.

TC0 Control Register B - TCCR0B

TCCR0B – TC0 Control Register B								
address: 0x45					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC0A	FOC0B	OC0AS	DTEN0	WGM02	CS02	CS01	CS00
R/W	W	W	W/R	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	FOC0A	TC0 forces the output compare A control bit. When working in non-PWM mode, a compare match can be generated by writing "1" to the forced output compare bit FOC0A. Forced compare match will not set the OCF0A flag, nor will it reload or clear the timer, but the output pin OC0A will be updated according to the setting of COM0A, just like a compare match really happened. The return value of reading FOC0A is always zero.
6	FOC0B	TC0 forces the output compare B control bit.

<table>
<tr><td></td><td></td><td>When working in non-PWM mode, a compare match can be generated by writing "1" to the forced output compare bit FOC0B. Forced compare match will not set the OCF0B flag, nor will it reload or clear the timer, but the output pin OC0B will be updated accordingly according to the setting of COM0B, just like a compare match really happened.
The return value of reading FOC0B is always zero.</td></tr>
<tr><td>5</td><td>OC0AS</td><td>OC0A output port selection control bit. When the OC0AS bit is set to "0", the waveform of OC0A is output from pin PD6; when the bit of OC0AS is set to "1", the waveform of OC0A is output from pin PE4 (valid under QFP32 package).</td></tr>
<tr><td>4</td><td>DTEN0</td><td>TC0 dead time enable control bit.
When setting the DTEN0 bit to "1", the dead time insertion is enabled. Both OC0A and OC0B insert dead time on the basis of the waveform generated by the comparison output of channel B, and the inserted dead time interval is determined by the counting time corresponding to the DTR0 register. The polarity of the OC0A output waveform is determined by the corresponding relationship between COM0 and COM0B. For details, see the waveform polarity table after OC0A is inserted into the dead time.
When the DTEN0 bit is set to "0", dead time insertion is prohibited, and the waveforms of OC0A and OC0B are the waveforms generated by their respective comparison outputs.</td></tr>
<tr><td>3</td><td>WGM02</td><td>TC0 waveform generation mode control high bit.
WGM02, WGM00, and WGM01 form a waveform generation mode to control WGM0[2:0], control the counting mode of the counter and waveform generation mode, see the waveform generation mode table description for details.</td></tr>
<tr><td>2</td><td>CS02</td><td>TC0 clock selection control high.
Used to select the clock source for timer counter 0.</td></tr>
<tr><td>1</td><td>CS01</td><td>TC0 clock selection control bit.
Used to select the clock source for timer counter 0.</td></tr>
<tr><td>0</td><td>CS00</td><td>TC0 clock selection control low.
Used to select the clock source for timer counter 0.
<table>
<tr><th>CS0[2:0]</th><th>Description</th></tr>
<tr><td>0</td><td>No clock source, stop counting</td></tr>
<tr><td>1</td><td>clk_{sys}</td></tr>
<tr><td>2</td><td>$clk_{sys}/8$, From the prescaler</td></tr>
<tr><td>3</td><td>$clk_{sys}/64$, From the prescaler</td></tr>
<tr><td>4</td><td>$clk_{sys}/256$, From the prescaler</td></tr>
<tr><td>5</td><td>$clk_{sys}/1024$, From the prescaler</td></tr>
<tr><td>6</td><td>External clock T0 pin, falling edge trigger</td></tr>
<tr><td>7</td><td>External clock T0 pin, rising edge trigger</td></tr>
</table></td></tr>
</table>

The following table shows the control of the comparison output mode on the output comparison waveform in non-PWM mode (ie normal mode and CTC mode).

COM0x[1:0]	description
0	OC0x disconnected, general IO port operation
1	Toggle OC0x signal on compare match
2	Clear OC0x signal on compare match
3	Set OC0x signal on compare match

The following table shows the control of the comparison output mode on the output comparison waveform in the fast PWM mode.

COM0x[1:0]	description
0	OC0x disconnected, general IO port operation
1	Reserved
2	Clear OC0x on compare match, and set OC0x on max-value match
3	Set OC0x on compare match, and clear OC0x on max-value match

The following table shows the control of the output comparison waveform in the comparison output mode in the phase correction mode.

COM0x[1:0]	description
0	OC0x disconnected, general IO port operation
1	Reserved
2	The OC0x signal is cleared when the comparison matches in the ascending order counting, and the OC0x signal is set when the comparison matches the descending order counting.
3	The OC0x signal is set when the comparison matches in ascending order counting, and the OC0x signal is cleared when the comparison matches in descending order counting.

The table below shows the waveform generation mode controls.

WGM0[2:0]	Working mode	TOP value	OCR0X update	TOV0 set
0	Normal	0xFF	immediately	MAX
1	PCPWM	0xFF	TOP	BOTTOM
2	CTC	OCR0A	immediately	MAX
3	FPWM	0xFF	TOP	MAX
4	Reserved	-	-	-
5	PCPWM	OCR0A	TOP	BOTTOM
6	Reserved	-	-	-
7	FPWM	OCR0A	TOP	TOP

The following table shows the polarity control of the OC0A signal output waveform when the dead time is enabled.

Polarity control of OC0A signal output waveform in dead time enable mode

DTEN0	COM0A[1:0]	COM0B[1:0]	description
0	-	-	OC0A signal polarity is controlled by OC0A compare output mode
1	0	-	OC0A disconnected, general IO port operation
1	1	-	Reserved
1	2	2	The OC0A signal has the same polarity as the OC0B signal
		3	The polarity of the OC0A signal is opposite to that of the OC0B signal
1	3	2	The polarity of the OC0A signal is opposite to that of the OC0B signal
		3	The OC0A signal has the same polarity as the OC0B signal

【Notice】:
The polarity of the OC0B signal output waveform is controlled by the OC0B compare output mode, which is the same as the mode without dead time enabled.

TC0 Control Register C – TCCR0C

TCCR0C - TC0 Control Register C								
address: 0x49					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	DSX07	DSX06	DSX05	DSX04	-	-	DSX01	DSX00
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	description
7	DSX07	TC0 trigger source selection control enable bit 7. When the DSX07 bit is set to "1", TC1 overflow is enabled as the trigger source of the output comparison signal waveform OC0A/OC0B for closing. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B. When the DSX07 bit is set to "0", TC1 overflow is disabled as a trigger source for the output comparison signal waveform OC0A/OC0B.
6	DSX06	TC0 trigger source selection control enable bit 6. When the DSX06 bit is set to "1", TC2 overflow is enabled as the trigger source of the output comparison signal waveform OC0A/OC0B for closing. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B. When the DSX06 bit is set to "0", TC2 overflow is disabled as a trigger source for turning off the output comparison signal waveform OC0A/OC0B.
5	DSX05	TC0 trigger source selection control enable bit 5. When the DSX05 bit is set to "1", the pin level change 0 is enabled as the trigger source for turning off the output comparison signal waveform OC0A/OC0B. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B. When the DSX05 bit is set to "0", the pin level change 0 is disabled as the trigger source of the output comparison signal waveform OC0A/OC0B.
4	DSX04	TC0 trigger source selection control enable bit 4. When the DSX04 bit is set to "1", the external interrupt 0 is enabled as the trigger source for closing the output comparison signal waveform OC0A/OC0B. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B. When the DSX04 bit is set to "0", the external interrupt 0 is disabled as the trigger source for closing the output comparison signal waveform OC0A/OC0B.
3:2	-	Reserved
1	DSX01	TC0 trigger source selection control enable bit 1. When the DSX01 bit is set to "1", the analog comparator 1 is enabled as a trigger source for closing the output comparison signal waveform OC0A/OC0B. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B.

		When the DSX01 bit is set to "0", the analog comparator 1 is disabled as the trigger source of the output comparison signal waveform OC0A/OC0B.
0	DSX00	TC0 trigger source selection control enable bit 0. When the DSX00 bit is set to "1", the analog comparator 0 is enabled as a trigger source for closing the output comparison signal waveform OC0A/OC0B. When the DOC0A/DOC0B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC0A/OC0B. When the DSX00 bit is set to "0", the analog comparator 0 is disabled as the trigger source of the output comparison signal waveform OC0A/OC0B.

The following table shows the selection control of the trigger source of the waveform output.

Disable trigger source selection control for OC0A/OC0B waveform output

DOC0x	DSX0n=1	Trigger source	Description
0	-	-	The DOC0x bit is "0", the trigger source is off and the waveform output function is disabled
1	0	Analog Comparator 0	ACIF0 rising edge turn off OC0x waveform output
1	1	Analog Comparator 1	ACIF1 rising edge turn off OC0x waveform output
1	4	External interrupt 0	INTF0 rising edge turn off OC0x waveform output
1	5	Pin level change 0	PCIF0 rising edge turn off OC0x waveform output
1	6	TC2 overflow	TOV2 rising edge turn off OC0x waveform output
1	7	TC1 overflow	TOV rising edge turn off OC0x waveform output

Note:

1) DSX0n=1 means that when the nth bit of the DSX0 register is 1, each register bit can be set at the same time.

TC0 Count Value Register - TCNT0

TCNT0 – TC0 count value register								
address: 0x46					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT07	TCNT06	TCNT05	TCNT04	TCNT03	TCNT02	TCNT01	TCNT00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	TCNT0	TC0 count value register. The 8 count values of the counter can be read and written directly through the TCNT0 register. A CPU write operation to the TCNT0 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer is stopped. This allows initializing the TCNT0 register to match the OCR0 value without causing an interrupt. If the value written to TCNT0 equals or bypasses the OCR0 value, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT0. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

TC0 output comparison register A- OCR0A

OCR0A – TC0 output comparison register A								
address: 0x47					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR0A7	OCR0A6	OCR0A5	OCR0A4	OCR0A3	OCR0A2	OCR0A1	OCR0A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR0A	TC0 output compare register. OCR0A contains an 8-bit data, which is continuously compared with the counter value TCNT0. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC0A pin. When using PWM mode, the OCR0A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR0A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR0A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR0A itself.

TC0 Output Compare Register B - OCR0B

OCR0B – TC0 Output Compare Register B								
address: 0x48					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR0B7	OCR0B6	OCR0B5	OCR0B4	OCR0B3	OCR0B2	OCR0B1	OCR0B0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	description
7:0	OCR0B	TC0 output compare B register. OCR0B contains an 8-bit data, which is continuously compared with the counter value TCNT0. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC0B pin. When using PWM mode, the OCR0B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR0B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR0B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR0B itself.

TC0 Interrupt Mask Register - TIMSK0

TIMSK0 – TC0 Interrupt Mask Register								
address: 0x6E					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	description
7:3		Reserved
2	OCIE0B	TC0 output compare B match interrupt enable bit. When the OCIE0B bit is "1" and the global interrupt is set, the TC0 output compare B match interrupt is enabled. When a compare match occurs, that is, when the OCF0B bit in TIFR0 is set, an interrupt is generated. When the OCIE0B bit is "0", the TC0 output compare B match interrupt is disabled.
1	OCIE0A	TC0 output compare A match interrupt enable bit. When the OCIE0A bit is "1" and the global interrupt is set, the TC0 output compare A match interrupt is enabled. When a compare match occurs, that is, when the OCF0A bit in TIFR0 is set, an interrupt is generated. When the OCIE0A bit is "0", the TC0 output compare A match interrupt is disabled.
0	TOIE0	TC0 overflow interrupt enable bit. When the TOIE0 bit is "1" and the global interrupt is set, the TC0 overflow interrupt is enabled. An interrupt is generated when TC0 overflows, that is, the TOV0 bit in TIFR is set. When TOIE0 bit is "0", TC0 overflow interrupt is disabled.

TC0 Interrupt Flag Register - TIFR0

TIFR0 – TC0 interrupt flag register								
address: 0x35					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OC0A	OC0B	-	-	-	OCF0B	OCF0A	TOV0
R/W	R/O	R/O	-	-	-	R/W	R/W	R/W

Bit	Name	description
7	OC0A	Output comparison waveform signal OC0A. Output comparison waveform signal OC0A, software can read but not write. Before the OC0A signal is not enabled to be output to its corresponding IO pin, the software can first read the value of the OC0A bit to obtain the polarity of the comparison waveform signal to be output, and can be changed by configuring the COM0A bit and setting the FOC0A bit Its polarity avoids unnecessary interference pulses after enabling the OC0A signal output to its corresponding IO pin.
6	OC0B	Output comparison waveform signal OC0B. Output comparison waveform signal OC0B, software can read but not write. Before the OC0B signal is not enabled to be output to its corresponding IO pin, the software can read the value of the OC0B bit

		to obtain the polarity of the comparison waveform signal to be output, and can be changed by configuring the COM0B bit and setting the FOC0B bit Its polarity avoids unnecessary interference pulses after enabling the OC0B signal output to its corresponding IO pin.
5:3		Reserved
2	OCF0B	TC0 output compare B match flag. When TCNT0 is equal to OCR0B, the comparison unit gives a match signal and sets the comparison flag OCF0B. If the output compare B interrupt enable OCIE0B is "1" and the global interrupt flag is set at this time, an output compare B interrupt will be generated. OCF0B will be automatically cleared when this interrupt service routine is executed, or write "1" to the OCF0B bit to clear this bit.
1	OCF0A	TC0 output compare B match flag. When TCNT0 is equal to OCR0B, the comparison unit gives a match signal and sets the comparison flag OCF0B. If the output compare B interrupt enable OCIE0B is "1" and the global interrupt flag is set at this time, an output compare B interrupt will be generated. OCF0B will be automatically cleared when this interrupt service routine is executed, or write "1" to the OCF0B bit to clear this bit.
0	TOV0	TC0 overflow flag. When the counter overflows, the overflow flag TOV0 is set. If the overflow interrupt enable TOIE0 is "1" and the global interrupt flag is set at this time, an overflow interrupt will be generated. TOV0 will be automatically cleared when this interrupt service routine is executed, or it can also be cleared by writing "1" to the TOV0 bit.

DTR0 - TC0 Dead Time Control Register

DTR0 – TC0 dead time control register								
address: 0x4F					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	DTR07	DTR06	DTR05	DTR04	DTR03	DTR02	DTR01	DTR00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
[7:4]	DTR0H	TC0 dead time register high. When the DTEN0 bit of the TCCR0B register is "1", OC0A and OC0B form a complementary output, and the insertion dead time control is enabled. The dead time inserted on the OC0B channel is determined by DTR0H, and the length of time is determined by DTR0H counting clocks. corresponding time.
[3:0]	DTR0L	TC0 dead time register low. When the DTEN0 bit of the TCCR0B register is "1", OC0A and OC0B form a complementary output, and the insertion dead time control is enabled. The dead time inserted on the OC0A channel is determined by DTR0L, and the length of time is determined by DTR0H count clocks. corresponding time.

TCKCSR - TC Clock Control and Status Register

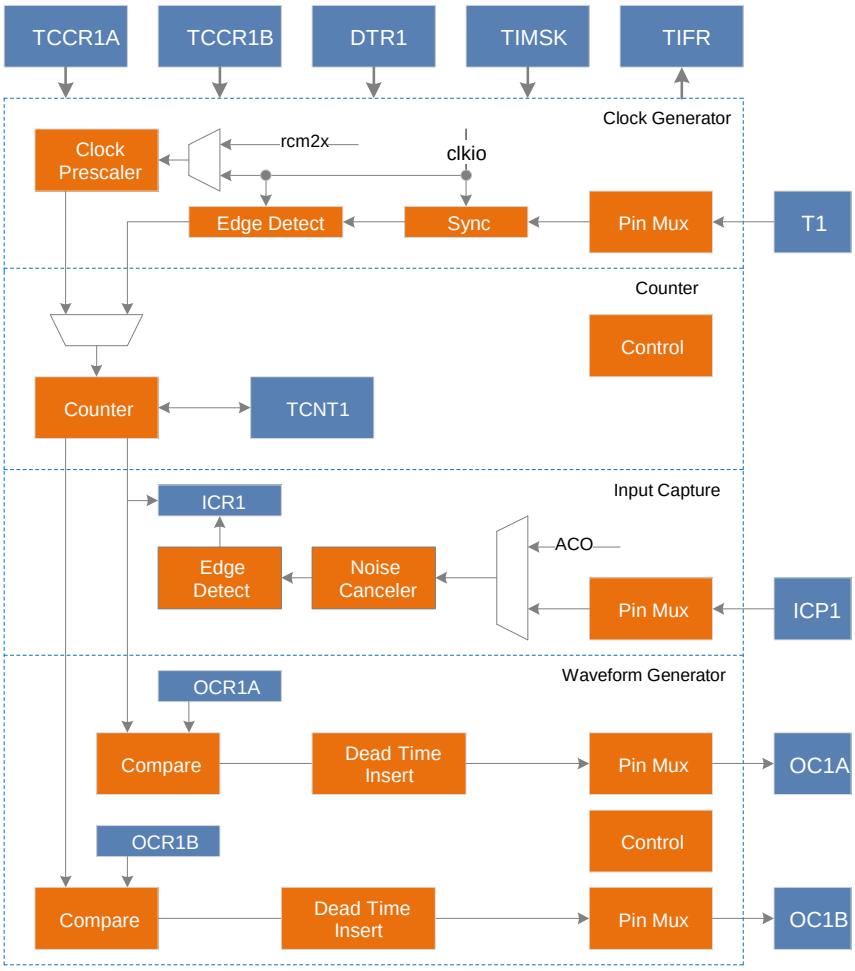
TCKSCR – TC Clock Control and Status Register								
address: 0xEC					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
R/W	-	R/W	R	R	-	R/W	R/W	R/W

Bit	Name	description
7	-	Reserved
6	F2XEN	RC 32M Multiplier output enable control bit. When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is enabled, and the 64M high-speed clock is output. When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is disabled, and the 64M high-speed clock cannot be output.
5	TC2XF1	TC high-speed clock mode flag bit 1. See Timer Counter 1 Register Description.
4	TC2XF0	TC high-speed clock mode flag bit 0. When the TC2XF0 bit is read as "1", it indicates that the timer counter 0 works in the high-speed clock mode, and when it is "0", it indicates that the timer counter 0 works in the system clock mode.
3:2	-	Reserved
1	TC2XS1	TC high-speed clock mode selection control bit 1. See Timer Counter 1 Register Description.
0	TC2XS0	TC high-speed clock mode selection control bit 0. When setting the TC2XS0 bit to "1", select the timing counter 0 to work in the high-speed clock mode. When setting the TC2XS0 bit to "0", select the timing counter 0 to work in the system clock mode.

Timer/Counter 1 (TMR1)

- ❑ True 16-bit design, allowing 16-bit PWM
- ❑ 2 independent output compare units
- ❑ Double buffered output compare register
- ❑ 1 input capture unit
- ❑ Input Capture Noise Suppressor
- ❑ The counter is automatically cleared and automatically loaded on a compare match
- ❑ Phase-corrected PWM without glitch pulses
- ❑ Variable PWM period
- ❑ frequency generator
- ❑ External event counter
- ❑ 4 independent interrupt sources
- ❑ PWM with dead time control
- ❑ 6 selectable trigger sources to automatically turn off the PWM output
- ❑ Generate high-speed and high-resolution (500KHZ@7BIT) PWM in high-speed clock mode

Overview



TC1 structure diagram

TC1 is a general-purpose 16-bit timer counter module that supports PWM output and can generate waveforms precisely. TC1 includes a 16-bit counter, waveform generation mode control unit, 2 independent output comparison units and an input capture unit. At the same time, TC1 can share the 10-bit prescaler with TC0, or use the 10-bit prescaler independently. The prescaler divides the system clock clkio or the high-speed clock rcm2x (2 times the frequency of the internal 32M RC oscillator output clock rc32m) to generate the count clock Clkt1. The waveform generation mode control unit controls the working mode of the counter and the generation of the comparison output waveform. According to different working modes, the counter realizes clearing, adding one or subtracting one operation for each counting clock Clkt1. Clkt1 can be generated by internal clock source or external clock source. When the count value TCNT1 of the counter reaches the maximum value (equal to the maximum value 0xFFFF or a fixed value or the output comparison register OCR1A or the input capture register ICR1, which is defined as TOP, and the maximum value is defined as MAX to show the difference), the counter will be cleared. or minus one operation. When the count value TCNT1 of the counter reaches the minimum value (equal to 0x0000, defined as BOTTOM), the counter will add one. When the count value TCNT1 of the counter reaches OCR1A or OCR1B, also known as when a comparison match occurs, it will be cleared or set to output the comparison signal OC1A or OC1B to generate a PWM waveform. When the dead time insertion is enabled, the set dead time (the number of counting clocks corresponding to the DTR1 register) will be inserted into the generated PWM waveform. When the input capture function is turned on, the counter starts or stops counting when it is triggered, and the ICR1 register will record the count value within the trigger period of the capture signal. The software can turn off the waveform output of OC1A/OC1B by clearing the COM1A/COM1B bit to zero, or set the corresponding trigger source. When the trigger event occurs, the hardware automatically clears the COM1A/COM1B bit to turn off the OC1A/OC1B waveform output.

The counting clock can be generated by an internal or external clock source. The selection of the clock source and frequency division is controlled by the CS1 bit in the TCCR1B register. For details, see the TC0 and TC1 prescaler chapters.

The counter has a length of 16 bits and supports bidirectional counting. The waveform generation mode, that is, the working mode of the counter, is controlled by the WGM1 bit located in the TCCR1A and TCCR1B registers. According to different working modes, the counter realizes clearing, adding one or subtracting one operation for each counting clock Clkt1. When the count overflows, the count overflow flag TOV1 bit in the TIFR1 register will be set. TC1 count overflow interrupt can be generated when the interrupt is enabled.

The output comparison unit compares the count value TCNT1 with the values of the output comparison registers OCR1A and OCR1B. When TCNT1 is equal to OCR1A or OCR1B, it is called a comparison match, and the output comparison flag OCF1A or OCF1B in the TIFR1 register will be set. TC1 output compare match interrupt can be generated when the interrupt is enabled.

It should be noted that in the PWM mode of operation, the OCR1A and OCR1B registers are double-buffered registers. In normal mode and CTC mode, the double buffer function is invalid. When the count reaches the maximum or minimum value, the value in the buffer register is synchronously updated to compare registers OCR1A and OCR1B. For details, see the chapter description of working mode.

The waveform generator generates output comparison waveform signals OC1A and OC1B using compare match, count overflow, etc. according to the waveform generation mode control and the comparison output mode control. The specific generation method is described in the working mode and register chapter. When the output comparison waveform signals OC1A and OC1B are to be output to corresponding pins, the data direction register of the pin must also be set as output.

Operating mode

Timing counter 1 has six different working modes, including normal mode (Normal), clear when compare match (CTC) mode, fast pulse width modulation (FPWM) mode, phase correction pulse width modulation (PCPWM) mode, phase frequency correction Pulse Width Modulation (PFCPWM) mode, and Input Capture (ICP) mode. It is selected by the waveform generation mode control bits WGM1[3:0]. The six modes are described in detail below. Since there are two independent output comparison units, they are represented by "A" and "B" respectively, and the lowercase "x" is used to represent the two output comparison unit channels.

Normal mode

The normal mode is the simplest working mode of the timer counter. At this time, the waveform generation mode control bit WGM1[3:0]=0, and the maximum value TOP of the count is MAX (0xFFFF). In this mode, the counting method increases by one for each counting clock, and when the counter reaches TOP and overflows, it returns to BOTTOM and starts accumulating again. The timer counter overflow flag TOV1 is set in the same count clock that the count value TCNT1 becomes zero. In this mode, the TOV1 flag is like the 17th counting bit, but it will only be set and not cleared. The overflow interrupt service routine will automatically clear the TOV1 flag, which can be used by software to increase the resolution of the timer counter. In normal mode, there is no special situation to consider, and a new count value can be written at any time.

The waveform of the output comparison signal OC1x can only be obtained when the data direction register of the OC1x pin is set to output. When COM1x=1, the OC1x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{oc1xnormal} = f_{sys}/(2*N*65536)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
The output compare unit can be used to generate interrupts, but it is not recommended to use interrupts in normal mode, which will take up too much CPU time.

CTC mode

When setting WGM1[3:0]=4 or 12, timer counter 1 enters CTC mode. When WGM1[3]=0, the maximum count TOP is OCR1A, when WGM1[3]=1, the maximum count TOP is ICR1. The following takes WGM1[3:0]=4 as an example to describe the CTC mode. In this mode, the counting method is incremented for each count clock. When the counter value TCNT1 is equal to TOP, the counter is cleared. This mode allows the user to easily control the frequency of the compare match output, and also simplifies the operation of counting external events.

When the counter reaches TOP, the output comparison match flag OCF1 is set, and an interrupt will be generated when the corresponding interrupt enable is set. The OCR1A register can be updated in the interrupt service routine. The OCR1A does not use double buffering in this mode, so be careful when updating the maximum value close to the minimum value when the counter is operating with no prescaler or a very low prescaler. If the value written into OCR1A is less than the current TCNT1 value, the counter will lose a comparison match. Before the next compare match occurs, the counter has to count to MAX first, and then start counting from BOTTOM to OCR1A. Same as the normal mode, the TOV1 flag is set in the count clock when the count value returns to 0x0.

The waveform of the output comparison signal OC1x can only be obtained when the data direction register of the OC1x pin is set to output. The frequency of the waveform can be calculated using the following formula:

$$f_{oc1xctc} = f_{sys}/(2*N*(1+OCR1A))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

It can be seen from the formula that when OCR1A is set to 0x0 and there is no prescaler, an output waveform with a maximum frequency of fsys/2 can be obtained.

When WGM1[3:0]=12, it is similar to WGM1[3:0]=4, just replace the one related to OCR1A with ICR1.

Fast PWM mode

When WGM1[3:0]=5, 6, 7, 14 or 15, the timing counter 1 enters the fast PWM mode, and the maximum count TOP is 0xFF, 0x1FF, 0x3FF, ICR1 or OCR1A, which can be used to generate high-frequency PWM waveform. Fast PWM mode differs from other PWM modes in that it operates unidirectionally. After the counter is accumulated from BOTTOM to TOP, it returns to BOTTOM to count again. When the count value TCNT1 reaches TOP or BOTTOM, the output comparison signal OC1x will be set or cleared, depending on the setting of the comparison output mode COM1, see the register description for details. Due to the unidirectional operation, the fast PWM mode operates at twice the frequency of the phase-corrected PWM mode with bidirectional operation. The high frequency characteristics make the fast PWM mode suitable for power regulation, rectification and DAC applications. High-frequency signals can reduce the size of external components (inductance and capacitance, etc.), thereby reducing system cost.
When the count value reaches TOP, the timer counter overflow flag TOV1 will be set, and the value of the comparison buffer will be updated to the comparison value. If the interrupt is enabled, the OCR1A register can be updated in the interrupt service routine.
The waveform of the output comparison signal OC1x can only be obtained when the data direction register of the OC1x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc1xfpwm} = f_{sys}/(N*(1+TOP))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

When a comparison match occurs between TCNT1 and OCR1x, the waveform generator will set (clear) the OC1x signal, and when TCNT1 is cleared, the waveform generator will clear (set) the OC1x signal to generate a PWM wave. Therefore, the extreme value of OCR1x will generate a special PWM waveform. When OCR1x is set to 0x00, the output PWM has a narrow spike every (1+TOP) count clock. When OCR1x is set to TOP, the output waveform is continuous high level or low level. If OCR1A is used as TOP and COM1A=1 is set, the output comparison signal OC1A will generate a PWM wave with a duty cycle of 50%.

Phase corrected PWM mode

When setting WGM0[3:0]=1, 2, 3, 10 or 11, the timing counter 1 enters the phase correction PWM mode, and the maximum counting TOP is 0xFF, 0x1FF, 0x3FF, ICR1 or OCR1A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT1 matches OCR1x, the output comparison signal OC1x will be cleared or set, depending on the setting of the comparison output mode COM1. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.

In the phase correction PWM mode, when the count reaches BOTTOM, the TOV1 flag is set, and when the count reaches TOP, the value of the comparison buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR1x register can be updated in the interrupt service routine.

The output comparison signal OC1x waveform can only be obtained when the data direction register of the OC1x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc1xcpcpwm} = f_{sys}/(N*TOP*2)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

During the count-up process, when TCNT1 matches OCR1x, the waveform generator clears (sets) the OC1x signal. During the countdown process, when TCNT1 matches OCR1x, the waveform generator sets

(clears) the OC1x signal. Therefore, the extreme value of OCR1x will generate a special PWM wave. When OCR1x is set to TOP or BOTTOM, the OC1x signal output will always keep low or high level. If OCR1A is used as TOP and COM1A=1 is set, the output comparison signal OC1A will generate a PWM wave with a duty cycle of 50%.

In order to ensure the symmetry of the output PWM wave on both sides of BOTTOM, when there is no comparison match, there are two cases where the OC1x signal will be inverted. The first case is when the value of OCR1x is changed from TOP to other data. When OCR1x is TOP and the count value reaches TOP, the output of OC1x is the same as the result of comparing and matching when counting in descending order, that is, keep OC1x unchanged. At this time, the comparison value will be updated to the new OCR1x value (not TOP), and the value of OC1x will remain until the comparison match occurs when counting up and flipped. At this time, the OC1x signal is not symmetrical about the minimum value, so it is necessary to flip the OC1x signal when TCNT1 reaches the maximum value, that is, the first case where the OC1x signal is flipped when there is no comparison match. The second case is that when TCNT1 starts counting from a value higher than OCR1x, a compare match will be lost, causing an asymmetrical situation. It is also necessary to flip the OC1x signal to achieve symmetry on both sides of the minimum.

Phase frequency correction PWM mode

When setting WGM0[3:0]=8 or 9, the timer counter 1 enters the phase frequency correction PWM mode, and the maximum value TOP of the count is ICR1 or OCR1A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT1 matches OCR1x, the output comparison signal OC1x will be cleared or set, depending on the setting of the comparison output mode COM1. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.

In the phase frequency correction PWM mode, when the count reaches BOTTOM, the TOV1 flag is set, and the value of the comparison buffer is updated to the comparison value. The time for updating the comparison value is the biggest difference between the phase frequency correction PWM mode and the phase correction PWM mode. If the interrupt is enabled, the compare buffer OCR1x register can be updated in the interrupt service routine. When the CPU changes the TOP value, that is, the value of ORC1A or ICR1, it must be ensured that the new TOP value is not smaller than the TOP value already in use, otherwise the comparison match will not happen again.

The output comparison signal OC1x waveform can only be obtained when the data direction register of the OC1x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc1xcpfcpwm} = f_{sys}/(N*TOP*2)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

During the count-up process, when TCNT1 matches OCR1x, the waveform generator clears (sets) the OC1x signal. During the countdown process, when TCNT1 matches OCR1x, the waveform generator sets (clears) the OC1x signal. Therefore, the extreme value of OCR1x will generate a special PWM wave. When OCR1x is set to TOP or BOTTOM, the OC1x signal output will always keep low or high level. If OCR1A is used as TOP and COM1A=1 is set, the output comparison signal OC1A will generate a PWM wave with a duty cycle of 50%.

Because the OCR1x register is updated at BOTTOM time, the counting lengths of the ascending and descending order on both sides of the TOP value are the same, and a symmetrical waveform with correct frequency and phase is produced.

When using a fixed TOP value, it is best to use the ICR1 register as the TOP value, that is, set WGM1[3:0]=8, at this time the OCR1A register is only used to generate PWM output. If you want to

generate PWM wave with frequency change, you must change the TOP value, and the double buffering feature of OCR1A will be more suitable for this application.

Input capture mode

Input capture is used to capture external events and give them time stamps to illustrate the moment when this event occurs. It can be performed in the previous counting mode, but the waveform generation mode that uses the ICR1 value as the counting TOP value must be removed.

The trigger signal of the external event is input by the pin ICP1, and it can also be realized by the analog comparator unit. When the logic level on the pin ICP1 changes, or the output ACO level of the analog comparator changes, and this level change is captured by the input capture unit, the input capture is triggered, and the 16-bit count value TCNT1 The data is copied to the input capture register ICR1, and the input capture flag ICF1 is set. If the ICIE1 bit is "1", the input capture flag will generate an input capture interrupt.

By setting the analog comparison control and status register ACSR analog comparison input capture control bit ACIC to select the input capture trigger source ICP1 or ACO. It should be noted that changing the trigger source may cause an input capture, so ICF1 must be cleared once after changing the trigger source to avoid erroneous results.

The input capture signal is sent to the edge detector after passing through an optional noise suppressor. According to the configuration of the input capture selection control bit ICES1, see whether the detected edge meets the trigger condition. The noise suppressor is a simple digital filter that samples the input signal 4 times, and its output is fed to the edge detector only when the values of the 4 samples are equal. The noise suppressor is enabled or disabled controlled by the ICNC1 bit of the TCCR1B register.

When using the input capture function, after ICF1 is set, the value of the ICR1 register should be read as early as possible, because the value of ICR1 will be updated after the next capture event occurs. It is recommended to enable the input capture interrupt. In any input capture mode, it is not recommended to change the count TOP value during operation.

The time stamps captured by the input can be used to calculate frequency, duty cycle, and other characteristics of the signal, as well as create logs for trigger events. When measuring the duty cycle of an external signal, it is required to change the trigger edge after each capture, so the trigger signal edge must be changed as soon as possible after reading the ICR1 value.

Automatic shutdown and restart of PWM output

When the DOC1x bit of the TCCR1C register is set high, the automatic shutdown function of the PWM output will be enabled. When the trigger condition is met, the hardware will clear the corresponding COM1x bit, disconnect the PWM output signal OC1x from its output pin, and switch to General IO output to realize automatic shutdown of PWM output. At this time, the state of the output pin can be controlled by the output of the general-purpose IO port.

After the automatic shutdown of the PWM output is enabled, the trigger condition needs to be set, and the trigger source is selected by the DSX1n bit of the TCCR1D register. Trigger sources include analog comparator interrupt, external interrupt, pin level change interrupt and timer overflow interrupt. For details, please refer to the description of the TCCR1D register. When one or some trigger sources are selected as trigger conditions, the
When these interrupt flags are set, the hardware will clear the COM1x bit to turn off the PWM output.

When a trigger event occurs and the PWM output is turned off, the timer module does not have a corresponding interrupt flag bit, and the software needs to read the interrupt flag bit of the trigger source to know the trigger condition and trigger event.

When the PWM output is automatically turned off and the output needs to be restarted again, the software only needs to reset the COM1x bit to switch the OC1x signal output to the corresponding pin. It

should be noted that after the automatic shutdown occurs, the timer does not stop working, and the status of the OC1x signal has been updated. The software can set the COM1x bit to output the OC1x signal after the timer overflows or compares a match, so that a clear PWM output state can be obtained.

Dead time control

When the DTEN1 bit is set to "1", the function of inserting dead time is enabled, and the output waveforms of OC1A and OC1B will insert the set dead time on the basis of the waveform generated by the comparison output of channel B, and the length of time is DTR1 The time value corresponding to the count clock number of the register. As shown in the figure below, the dead time insertion of OC1A and OC1B is based on the comparative output waveform of channel B. When COM1A and COM1B are both "2" or "3", the waveform polarity of OC1A is the same as that of OC1B; when COM1A and COM1B are "2" or "3" respectively, the waveform of OC1A is the same as that of OC1B opposite polarity.

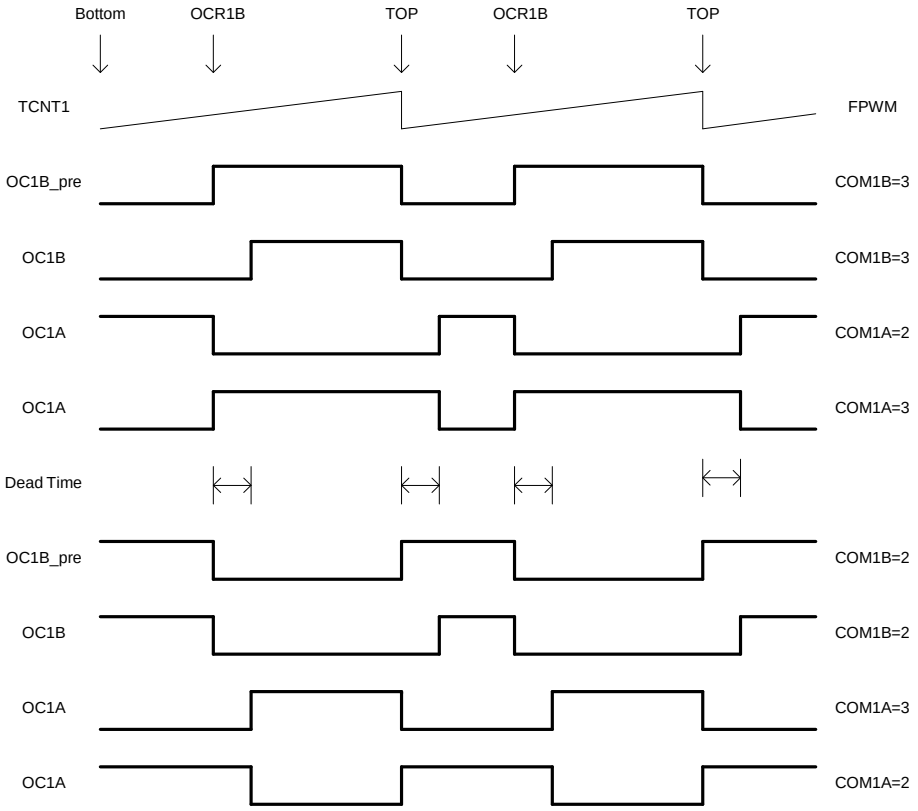


Figure 3 TC1 dead time control in FPWM mode

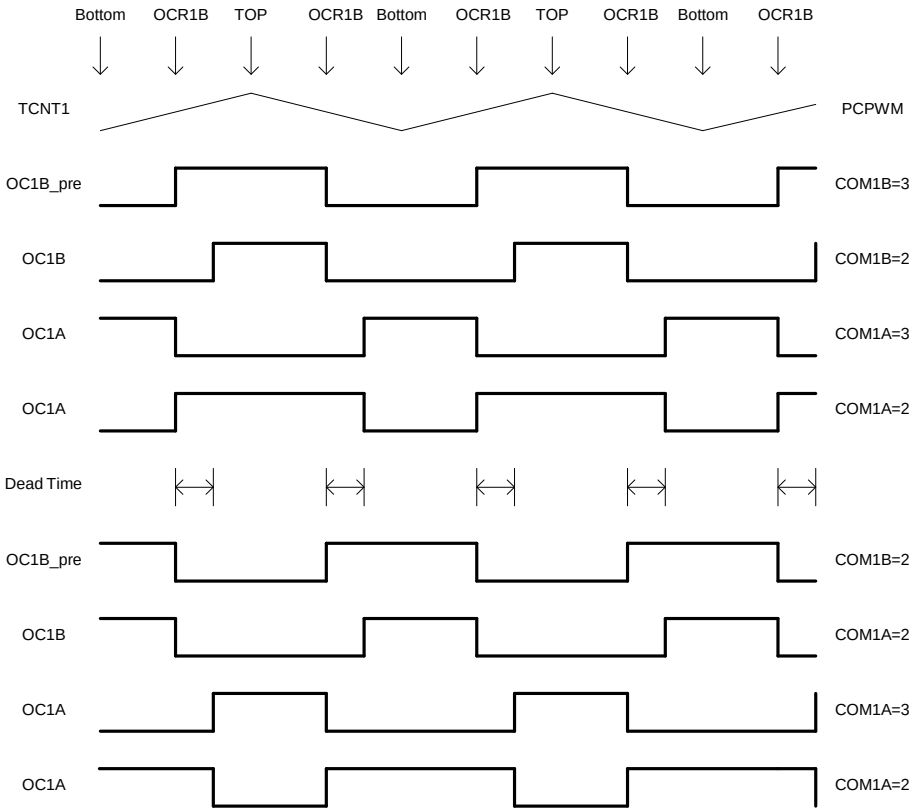


Figure 4 TC1 dead time control in PCPWM mode

When the DTEN1 bit is set to "0", the function of inserting dead time is disabled, and the output waveforms of OC1A and OC1B are the waveforms generated by their respective comparison outputs.

High-speed counting mode

In the high-speed clock mode, a higher-frequency clock is used as the counting clock source to generate a higher-speed and higher-resolution PWM waveform. This high-frequency clock is generated by multiplying the output clock rc32m of the internal 32M RC oscillator by 2. Therefore, before entering the high-frequency mode, it is necessary to enable the frequency multiplication function of the internal 32M RC oscillator, that is, set the F2XEN bit of the TCKCSR register, and wait for a certain period of time until the multiplier clock signal output is stable. Then, the TC2XS1 bit of TCKCSR can be set to make the timer counter enter the high-speed clock mode.

In this mode, the system clock is asynchronous to the high-speed clock, and some registers (see TC1 register list) work in the high-speed clock domain. Therefore, configuration and reading of such registers are also asynchronous, and attention should be paid to the operation.

There are no special requirements for discontinuous read and write operations on registers under the high-speed clock domain, but for continuous read and write operations, it is necessary to wait for a system clock, and the following steps can be followed:

5) Write register A;
6) Wait for a system clock (NOP or register under the operating system clock);
7) Read or write register A or B.
8) Wait for one system clock (NOP or register under OS clock).

When reading registers in the high-speed clock domain, registers with a width of 8 bits can be read directly, and when reading the value of a 16-bit register (OCR1A, OCR1B, ICR1, TCNT1), first read the

value of the lower register , wait for a system clock, and then read the value of the high register, and when reading the value of TCNT1, when the counter is still counting, the value of TCNT1 will change with the high-speed clock, and the counter can be suspended (set CS1 to zero) Then read the value of TCNT1.

When reading OCR1A, OCR1B and ICR1, follow the steps below:

1) Read OCR1AL/OCR1BL/ICR1L;
2) Wait for a system clock (NOP);
3) Read OCR1AH/OCR1BH/ICR1H.

When reading TCNT1, follow the steps below:

1) Set CS1 to zero;
2) Wait for a system clock (NOP);
3) Read the value of TCNT1L;
4) Wait for a system clock (NOP);

Read the value of TCNT1H.

Register definition

TC1 register list

Register	Address	Default value	Description
TCCR1A*	0x80	0x00	TC1 Control Register A
TCCR1B*	0x81	0x00	TC1 Control Register B
TCCR1C*	0x82	0x00	TC1 Control Register C
DSX1	0x83	0x00	TC1 trigger source control register
TCNT1L*	0x84	0x00	TC1 count value register low byte
TCNT1H*	0x85	0x00	TC1 count value register high byte
ICR1L*	0x86	0x00	TC1 input capture register low byte
ICR1H*	0x87	0x00	TC1 input capture register high byte
OCR1AL*	0x88	0x00	TC1 output compare register A low byte
OCR1AH*	0x89	0x00	TC1 output compare register A high byte
OCR1BL*	0x8A	0x00	TC1 output compare register B low byte
OCR1BH*	0x8B	0x00	TC1 output compare register B high byte
DTR1*	0x8C	0x00	TC1 Dead Time Control Register
TIMSK1	0x6F	0x00	Timer Counter Interrupt Mask Register
TIFR1	0x36	0x00	Timer Counter Interrupt Flag Register
TCKCSR1	0xEC	0x00	TC1 Clock Control Status Register

【Notice】
The registers with "*" work in the system clock and high-speed clock domain, and the registers without "*" only work in the system clock domain.

TCCR1A – TC1 Control Register A

TCCR1A –TC1 Control Register A								
address: 0x80					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	description
7	COM1A1	Compare match output A mode control high. COM1A1 and COM1A0 form COM1A[1:0] to control the output comparison waveform OC1A. If both bits 1 or 2 of COM1A are set, the output comparison waveform occupies the OC1A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM1A controls the output comparison waveform differently, see the comparison output mode control table description for details.
6	COM1A0	Compare match output A mode control low bit. COM1A1 and COM1A0 form COM1A[1:0] to control the output comparison waveform OC1A. If both bits 1 or 2 of COM1A are set, the output comparison waveform occupies the OC1A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM1A controls the output comparison waveform differently, see the comparison output mode control table description for details.
5	COM1B1	Compare match output B mode control high. COM1B1 and COM1B0 form COM1B[1:0] to control the output comparison waveform OC1B. If both bits 1 or 2 of COM1B are set, the output comparison waveform occupies the OC1B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM1B controls the output comparison waveform differently, see the comparison output mode control table description for details.
4	COM1B0	Compare match output B mode control low. COM1B1 and COM1B0 form COM1B[1:0] to control the output comparison waveform OC1B. If both bits 1 or 2 of COM1B are set, the output comparison waveform occupies the OC1B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM1B controls the output comparison waveform differently, see the comparison output mode control table description for details.
3:2	-	Reserved
1	WGM11	Waveform Generation Mode Control Second Low Bit. WGM11, WGM13, WGM12, and WGM10 together form a waveform generation mode to control WGM1[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
0	WGM10	The least significant bit of the waveform generation mode control. WGM10, WGM13, WGM12, and WGM11 form a waveform generation mode to control WGM1[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.

The following table shows the control of the comparison output mode on the output comparison waveform in non-PWM mode (that is, normal mode and CTC mode).

COM1x[1:0]	description
0	OC1x disconnected, general IO port operation
1	Invert OC1x signal on compare match
2	Clear OC1x signal on compare match
3	Assert OC1x signal on compare match

The following table shows the control of the comparison output mode on the output comparison waveform in the fast PWM mode.

COM1x[1:0]	description
0	OC1x disconnected, general IO port operation
1	WGM1==15: OC1A toggle on compare match and OC1B disconnected WGM1!=15: OC1x is disconnected, general IO port operation
2	Clear OC1x on compare and match, set on max-value match
3	OC1x set on comparison match; cleared on max-value match

The following table shows the control of the output comparison waveform in the comparison output mode in the phase correction mode.

COM1x[1:0]	description
0	OC1x disconnected, general IO port operation
1	WGM1==9 or 11: toggle OC1A on compare match, OC1B disconnect WGM1!=9 or 11: OC1x is disconnected, general IO port operation
2	Clear the OC1x signal on compare match in up-counting, and set it on compare match in down-counting
3	The comparison match sets the OC1x signal in up-counting, and the comparison match clears the OC1x signal in down-counting

TCCR1B – TC1 Control Register B

TCCR1B – TC1 Control Register B								
address: 0x81					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	ICNC1	Input capture noise suppressor enable control bit. When the ICNC1 bit is set to "1", the input capture noise suppressor is enabled. At this time, the input of the external pin ICP1 is filtered, and the input signal is only valid when 4 consecutive sampling values are equal. This function delays the input capture by 4 clock cycle. When the ICNC1 bit is set to "0", the input capture noise suppressor is prohibited, and the input of the external pin ICP1 is directly effective at this time.
6	ICES1	Input capture trigger edge selection control bit. When the ICES1 bit is set to "1", the rising edge of the selection level triggers the input capture; when the ICES1 bit is set to "0", the falling edge of the selection level triggers the input capture. When an event is captured, the value of the counter is copied to the ICR1 register and the input capture flag ICF1 is set. If the interrupt is enabled, an input capture interrupt is generated.
5	-	Reserved.

<table>
<tr><td>4</td><td>WGM13</td><td>Waveform generation mode control high.
WGM13, WGM12, WGM11, and WGM10 form a waveform generation mode to control WGM1[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.</td></tr>
<tr><td>3</td><td>WGM12</td><td>The second most significant bit is the waveform generation mode control. WGM12, WGM13, WGM11, and WGM10 together form a waveform generation mode to control WGM1[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.</td></tr>
<tr><td>2</td><td>CS12</td><td>Clock select control high. Used to select the clock source for timer counter 1.</td></tr>
<tr><td>1</td><td>CS11</td><td>Clock select control bit. Used to select the clock source for timer counter 1.</td></tr>
<tr><td>0</td><td>CS10</td><td>Clock select control low.
Used to select the clock source for timer counter 1.
<table>
<tr><th>CS1[2:0]</th><th>description</th></tr>
<tr><td>0</td><td>No clock source, stop counting</td></tr>
<tr><td>1</td><td>clk_{sys}</td></tr>
<tr><td>2</td><td>clk_{sys}/8, From the prescaler</td></tr>
<tr><td>3</td><td>clk_{sys}/64, From the prescaler</td></tr>
<tr><td>4</td><td>clk_{sys}/256, From the prescaler</td></tr>
<tr><td>5</td><td>clk_{sys}/1024, From the prescaler</td></tr>
<tr><td>6</td><td>Extern clock T1 pin, falling edge trigger</td></tr>
<tr><td>7</td><td>Extern clock T1 pin, rising edge trigger</td></tr>
</table></td></tr>
</table>

The table below shows the waveform generation mode controls.

WGM1[3:0]	Working mode	TOP value	Update OCR0 moment	Set TOV0 moment
0	Normal	0xFFFF	Immediate	MAX
1	8-bit PCPWM	0x00FF	TOP	BOTTOM
2	9-bit PCPWM	0x01FF	TOP	BOTTOM
3	10-bit PCPWM	0x03FF	TOP	BOTTOM
4	CTC	OCR1A	Immediate	MAX
5	8-bit FPWM	0x00FF	BOTTOM	TOP
6	9-bit FPWM	0x01FF	BOTTOM	TOP
7	10-bit FPWM	0x03FF	BOTTOM	TOP
8	PFCPWM	ICR1	BOTTOM	BOTTOM
9	PFCPWM	OCR1A	BOTTOM	BOTTOM
10	PCPWM	ICR1	TOP	BOTTOM
11	PCPWM	OCR1A	TOP	BOTTOM
12	CTC	ICR1	Immediate	MAX
13	Reserved	-	-	-
14	FPWM	ICR1	TOP	TOP
15	FPWM	OCR1A	TOP	TOP

TCCR1C – TC1 Control Register C

TCCR1C – TC1 Control Register C								
address: 0x82					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC1A	FOC1B	DOC1B	DOC1A	DTEN1	-	-	-
R/W	W	W	R/W	R/W	R/W	-	-	-

Bit	Name	description
7	FOC1A	Force output compare A. When working in non-PWM mode, a compare match can be generated by writing "1" to the forced output compare bit FOC1A. Forced compare match will not set the OCF1A flag, nor will it reload or clear the timer, but the output pin OC1A will be updated accordingly according to the setting of COM1A, just like a compare match really happened. When working in PWM mode, it must be cleared when writing the TCCR1A register. The return value of reading FOC1A is always zero.
6	FOC1B	Force output compare B. When working in non-PWM mode, a comparison match can be generated by writing "1" to the forced output comparison bit FOC1B. Forced compare match will not set the OCF1B flag, nor will it reload or clear the timer, but the output pin OC1B will be updated accordingly according to the setting of COM1B, just like a compare match really happened. When working in PWM mode, it must be cleared when writing the TCCR1A register. The return value of reading FOC1B is always zero.
5	DOC1B	TC1 closes the output comparison enable control high bit. When the DOC1B bit is set to "1", the trigger source is turned off and the output comparison signal OC1B is enabled. When a trigger event occurs, the hardware automatically clears the COM1B bit and turns off the waveform output of OC1B. The software can re-enable the PWM output by setting COM1B. When the DOC1B bit is set to "0", the trigger source is turned off and the output comparison signal OC1B is disabled.
4	DOC1A	TC1 closes the output comparison enable control low bit. When the DOC1A bit is set to "1", the trigger source is turned off and the output comparison signal OC1A is enabled. When a trigger event occurs, the hardware automatically clears the COM1A bit and turns off the waveform output of OC1A. The software can re-enable the PWM output by setting COM1A. When the DOC1A bit is set to "0", the trigger source is turned off and the output comparison signal OC1A is disabled.
3	DTEN1	TC1 dead time enable control bit. When setting the DTEN1 bit to "1", the dead time insertion is enabled. Both OC1A and OC1B insert dead time on the basis of the waveform generated by the comparison output of channel B, and the inserted dead time interval is determined by the counting time corresponding to the DTR1 register. The polarity of the OC1A output waveform is determined by the corresponding relationship between COM1A and COM1B. For details, see the waveform polarity table after OC1A inserts the dead time. When the DTEN1 bit is set to "0", dead time insertion is prohibited, and the waveforms of OC1A and OC1B are the waveforms generated by their respective comparison outputs.
2:0	-	Reserved.

The following table shows the polarity control of the OC1A signal output waveform when the dead time is enabled.

Polarity control of OC1A signal output waveform in dead time enable mode

DTEN1	COM1A[1:0]	COM1B[1:0]	Description
0	-	-	OC1A polarity controlled by OC1A compare output mode
1	0	-	OC1A disconnected, general IO port operation
1	1	-	Reserved
1	2	2	The OC1A polarity is the same of OC1B signal
		3	The OC1A polarity is opposite to that of the OC1B
1	3	2	The OC1A polarity is opposite to that of the OC1B
		3	The OC1A polarity is the same of OC1B signal

【Notice】:
The polarity of the OC1B signal output waveform is controlled by the OC1B compare output mode, which is the same as the mode without dead time enabled.

TCCR1D – TC1 Control Register D

TCCR1D – TC control register D								
address: 0x83					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	DSX17	DSX16	DSX15	DSX14	-	-	DSX11	DSX10
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	DSX17	TC1 trigger source selection control enable bit 7. When the DSX17 bit is set to "1", TC0 overflow is enabled as a trigger source for turning off the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX17 bit is set to "0", TC0 overflow is disabled as a trigger source for closing the output comparison signal waveform OC1A/OC1B.
6	DSX16	TC1 trigger source selection control enable bit 6. When the DSX16 bit is set to "1", TC2 overflow is enabled as a trigger source for closing the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX16 bit is set to "0", TC2 overflow is disabled as a trigger source for closing the output comparison signal waveform OC1A/OC1B.
5	DSX15	TC1 trigger source selection control enable bit 5. When the DSX15 bit is set to "1", the pin level change 1 is enabled as the trigger source for closing the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX15 bit is set to "0", the pin level change 1 is disabled as the trigger source of the output comparison signal waveform OC1A/OC1B.
4	DSX14	TC1 trigger source selection control enable bit 4. When the DSX14 bit is set to "1", the external interrupt 1 is enabled as the trigger source for closing the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the

		rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX14 bit is set to "0", the external interrupt 1 is disabled as the trigger source for closing the output comparison signal waveform OC1A/OC1B.
3:2	-	Reserved
1	DSX11	TC1 trigger source selection control enable bit 1. When the DSX11 bit is set to "1", the analog comparator 1 is enabled as a trigger source for closing the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX11 bit is set to "0", the analog comparator 1 is disabled as the trigger source of the output comparison signal waveform OC1A/OC1B.
0	DSX10	TC1 trigger source selection control enable bit 0. When the DSX10 bit is set to "1", the analog comparator 0 is enabled as a trigger source for closing the output comparison signal waveform OC1A/OC1B. When the DOC1A/DOC1B bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC1A/OC1B. When the DSX10 bit is set to "0", the analog comparator 0 is disabled as the trigger source of the output comparison signal waveform OC1A/OC1B.

The following table shows the selection control of the trigger source of the waveform output.

Turn off the trigger source selection control of OC1A/OC1B waveform output

DOC1x	DSX1n=1	Trigger source	description
0	-	-	The DOC1x bit is "0", the trigger source is off and the waveform output function is disabled
1	0	Analog comparator 0	ACIF0 rising edge turn off OC1x waveform output
1	1	Analog comparator 1	ACIF1 rising edge turn off OC1x waveform output
1	4	External interrupt 1	INTF1 rising edge turn off OC1x waveform output
1	5	Pin level change 1	PCIF1 rising edge turn off OC1x waveform output
1	6	TC2 overflow	TOV2 rising edge turn off OC1x waveform output
1	7	TC0 overflow	TOV0 rising edge turn off OC1x waveform output

【Notice】:

DSX1n=1 means that when the nth bit of the DSX1 register is 1, each register bit can be set at the same time.

TCNT1L – TC1 count value register low byte

TCNT1L –TC1 计数值寄存器低字节								
address: 0x84					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT1L7	TCNT1L6	TCNT1L5	TCNT1L4	TCNT1L3	TCNT1L2	TCNT1L1	TCNT1L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	description						
7:0	TCNT1	The low byte of the TC1 count value.						

		TCNT1H and TCNT1L are combined to form TCNT1, and the 16-bit count value of the counter can be read and written directly through the TCNT1 register. Reading and writing a 16-bit register requires two operations. When writing 16-bit TCNT1, write TCNT1H first. When reading 16-bit TCNT1, TCNT1L should be read first. A CPU write operation to the TCNT1 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer has been stopped. This allows initializing the value of the TCNT1 register to match the value of OCR1x without causing an interrupt. If the value written to TCNT1 equals or bypasses the OCR1x value, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT1. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

TCNT1H – TC1 count value register high byte

TCNT1H – TC1 count value register high byte								
address: 0x85					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT1H7	TCNT1H6	TCNT1H5	TCNT1H4	TCNT1H3	TCNT1H2	TCNT1H1	TCNT1H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	TCNT1H	High byte of TC1 count value. TCNT1H and TCNT1L are combined to form TCNT1, and the 16-bit count value of the counter can be read and written directly through the TCNT1 register. Reading and writing a 16-bit register requires two operations. When writing 16-bit TCNT1, write TCNT1H first. When reading 16-bit TCNT1, TCNT1L should be read first. A CPU write operation to the TCNT1 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer has been stopped. This allows initializing the value of the TCNT1 register to match the value of OCR1x without causing an interrupt. If the value written to TCNT1 equals or bypasses the OCR1x value, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT1. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

ICR1L – TC1 Input Capture Register Low Byte

ICR1L – TC1 Input Capture Register Low Byte								
address: 0x86					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICR1L7	ICR1L6	ICR1L5	ICR1L4	ICR1L3	ICR1L2	ICR1L1	ICR1L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	ICR1L	Low byte of TC1 input capture value. ICR1H and ICR1L combine to form the 16-position ICR1. Reading and writing a 16-bit register requires two operations. When writing 16-bit ICR1, ICR1H should be written first. When reading 16-bit ICR1, ICR1L

		should be read first. When the input capture is triggered, the count value TCNT1 will be updated and copied to the ICR1 register. The ICR1 register can also be used to define the TOP value of the count.

ICR1H – TC1 Input Capture Register High Byte

ICR1H – TC1 Input Capture Register High Byte								
address: 0x87					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICR1H7	ICR1H6	ICR1H5	ICR1H4	ICR1H3	ICR1H2	ICR1H1	ICR1H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	ICR1H	TC1 inputs the high byte of the capture value. ICR1H and ICR1L combine to form the 16-position ICR1. Reading and writing a 16-bit register requires two operations. When writing 16-bit ICR1, ICR1H should be written first. When reading 16-bit ICR1, ICR1L should be read first. When the input capture is triggered, the count value TCNT1 will be updated and copied to the ICR1 register. The ICR1 register can also be used to define the TOP value of the count.

OCR1AL – TC1 output compare register A low byte

OCR1AL – TC1 output compare register A low byte								
address: 0x88					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1AL7	OCR1AL6	OCR1AL5	OCR1AL4	OCR1AL3	OCR1AL2	OCR1AL1	OCR1AL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR1AL	Low byte of output compare register A. OCR1AL and OCR1AH combine to form the 16-position OCR1A. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR1A, OCR1AH should be written first. When reading 16-bit OCR1A, OCR1AL should be read first. OCR1A is continuously compared with the counter value TCNT1. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC1A pin. When using PWM mode, the OCR1A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR1A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR1A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR1A itself.

OCR1AH – TC1 output compare register A high byte

OCR1AH – TC1 output compare register A high byte								
address: 0x89					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1AH7	OCR1AH6	OCR1AH5	OCR1AH4	OCR1AH3	OCR1AH2	OCR1AH1	OCR1AH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR1AH	High Byte of Output Compare Register A. OCR1AL and OCR1AH combine to form the 16-position OCR1A. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR1A, OCR1AH should be written first. When reading 16-bit OCR1A, OCR1AL should be read first. OCR1A is continuously compared with the counter value TCNT1. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC1A pin. When using PWM mode, the OCR1A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR1A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR1A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR1A itself.

OCR1BL – TC1 output compare register B low byte

OCR1BL – TC1 output compare register B low byte								
address: 0x8A					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1BL7	OCR1BL6	OCR1BL5	OCR1BL4	OCR1BL3	OCR1BL2	OCR1BL1	OCR1BL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR1BL	Low byte of output compare register B. OCR1BL and OCR1BH combine to form the 16-position OCR1B. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR1B, OCR1BH should be written first. When reading 16-bit OCR1B, OCR1BL should be read first. OCR1B is continuously compared with the counter value TCNT1. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC1B pin. When using PWM mode, the OCR1B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR1B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR1B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR1B itself.

OCR1BH – TC1 output compare register B high byte

OCR1BH – TC1 output compare register B high byte								
address: 0x8B					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1BH7	OCR1BH6	OCR1BH5	OCR1BH4	OCR1BH3	OCR1BH2	OCR1BH1	OCR1BH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR1BH	High Byte of Output Compare Register B. OCR1BL and OCR1BH combine to form the 16-position OCR1B. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR1B, OCR1BH should be written first. When reading 16-bit OCR1B, OCR1BL should be read first. OCR1B is continuously compared with the counter value TCNT1. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC1B pin. When using PWM mode, the OCR1B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR1B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR1B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR1B itself.

TIMSK1 – TC1 Interrupt Mask Register

TIMSK1 – TC1 Interrupt Mask Register								
address: 0x6F					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	TICIE1	-	-	OCIE1A	OCIE1B	TOIE1
R/W	-	-	R/W	-	-	R/W	R/W	R/W

Bit	Name	description
7:6	-	Reserved
5	TICIE1	TC1 input capture interrupt enable control bit. When the ICIE1 bit is "1" and the global interrupt is set, the TC1 input capture interrupt is enabled. When the input capture is triggered, that is, the ICF1 flag of TIFR1 is set, an interrupt occurs. When ICIE1 bit is "0", TC1 input capture interrupt is disabled.
4:3	-	Reserved.
2	OCIE1B	TC1 output compare B match interrupt enable bit. When the OCIE1B bit is "1" and the global interrupt is set, the TC1 output compare B match interrupt is enabled. When a compare match occurs, that is, when the OCF1B bit in TIFR is set, an interrupt is generated. When the OCIE1B bit is "0", the TC1 output compare B match interrupt is disabled.
1	OCIE1A	TC1 output compare A match interrupt enable bit. When the OCIE1A bit is "1" and the global interrupt is set, the TC1 output compare A match interrupt is enabled. When a compare match

		occurs, that is, when the OCF1A bit in TIFR is set, an interrupt is generated. When the OCIE1A bit is "0", the TC1 output compare A match interrupt is disabled.
0	TOIE1	TC1 overflow interrupt enable bit. When the TOIE1 bit is "1" and the global interrupt is set, the TC1 overflow interrupt is enabled. An interrupt is generated when TC1 overflows, ie the TOV1 bit in TIFR is set. When the TOIE1 bit is "0", the TC1 overflow interrupt is disabled.

TIFR1 – TC1 Interrupt Flag Register

TIFR1 – TC1 Interrupt Flag Register								
address: 0x36					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	ICF1	-	-	OCF1B	OCF1A	TOV1
R/W	-	-	R/W	-	-	R/W	R/W	R/W

Bit	Name	description
7:6	-	Reserved.
5	ICF1	Input capture flag. The ICF1 flag is set when an input capture event occurs. When ICR1 is used as the TOP value for counting and the count value reaches the TOP value, the ICF1 flag is set. If ICIE1 is "1" and the global interrupt flag is set, an input capture interrupt will be generated. ICF1 will be automatically cleared when this interrupt service routine is executed, or writing "1" to the ICF1 bit can also clear this bit.
4:3	-	Reserved.
2	OCF1B	Output compare B match flag. When TCNT1 is equal to OCR1B, the comparison unit gives a match signal and sets the comparison flag OCF1B. If the output comparison interrupt enable OCIE1B is "1" and the global interrupt flag is set at this time, an output comparison interrupt will be generated. OCF1B will be automatically cleared when this interrupt service routine is executed, or writing "1" to the OCF1B bit can also clear this bit.
1	OCF1A	Output compare A match flag. When TCNT1 is equal to OCR1A, the comparison unit gives a match signal and sets the comparison flag OCF1A. If the output comparison interrupt enable OCIE1A is "1" and the global interrupt flag is set at this time, an output comparison interrupt will be generated. OCF1A will be automatically cleared when this interrupt service routine is executed, or writing "1" to the OCF1A bit can also clear this bit.
0	TOV1	overflow flag. When the counter overflows, the overflow flag TOV1 is set. If the overflow interrupt enable TOIE1 is "1" and the global interrupt flag is set, an overflow interrupt will be generated. TOV1 will be automatically cleared when this interrupt service routine is executed, or it can also be cleared by writing "1" to the TOV1 bit.

DTR1L – TC1 Dead Time Register Low Byte

DTR1–TC1 Dead Time Register								
address: 0x8C					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0

	DTR1L							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	description						
7:0	DTR1L	Dead Time Register High Byte. When the DTEN1 bit is high, OC1A and OC1B are complementary outputs, and the dead time inserted on the OC1A output is determined by DTR1L counting clocks.						

DTR1H – TC1 Dead Time Register High Byte

DTR1H – TC1 Dead Time Register High Byte								
address: 0x8D					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	DTR1H							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	description						
7:0	DTR1H	Dead Time Register High Byte. When the DTEN1 bit is high, OC1A and OC1B are complementary outputs, and all OC1B outputs The inserted dead time is determined by DTR1H count clocks.						

TCKCSR – TC Clock Control Status Register

TCKCSR – TC Clock Control Status Register								
address: 0xEC					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
R/W	-	R/W	R/O	R/O	-	R/W	R/W	R/W

Bit	Name	description
7	-	Reserved.
6	F2XEN	RC 32M multiplier output enable control bit. When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is enabled, Output 64M high-speed clock When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is disabled. Can not output 64M high-speed clock
5	TC2XF1	TC high speed clock mode flag bit 1 . When the TC2XF1 bit is read as "1", it indicates that the timer counter 1 works in the high-speed clock mode, and when it is "0", it indicates that the timer counter 1 works in the system clock mode
4	TC2XF0	TC high-speed clock mode flag bit 0. See TC0 register description.
3:2	-	Reserved.
1	TC2XS1	TC high-speed clock mode selection control bit 1 When setting the TC2XS1 bit to "1", select the timing counter 1 to work in the high-speed clock mode When setting the TC2XS1 bit to "0", select the timing counter 1 to work in the system clock mode
0	TC2XS0	TC high-speed clock mode selection control bit 0. See TC0 Description

TMR0/1/3 prescaler

- ❑ Three 10-bit prescalers
- ❑ TC0, TC1 and TC3 multiplexing prescaler CPS310 in multiplexing mode
- ❑ In independent mode, TC0 exclusive prescaler CPS310, TC1 exclusive prescaler CPS1, TC3 exclusive prescaler CPS3
- ❑ Support software reset

Overview

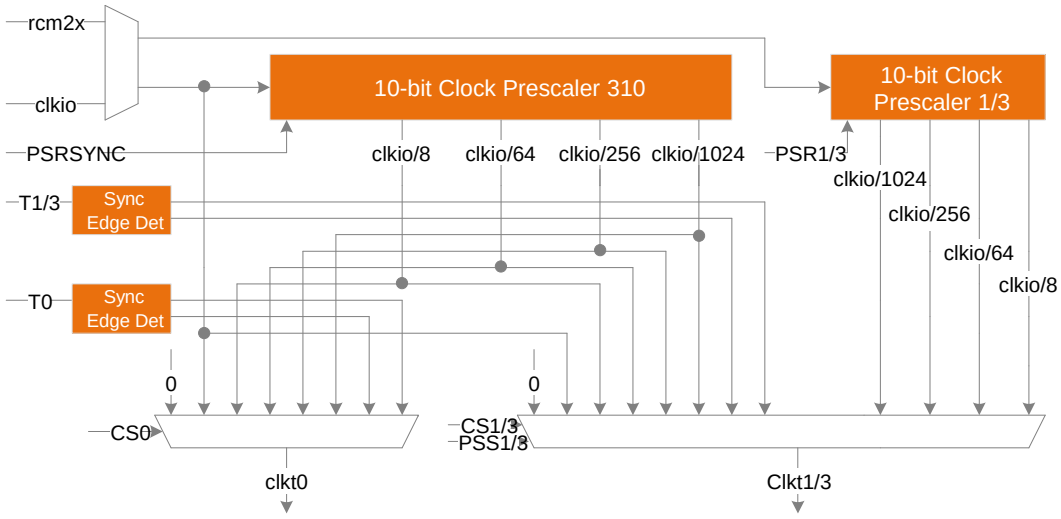
In multiplexing mode (PSS1=0 and PSS3=0), TC0, TC1 and TC3 share a 10-bit prescaler CPS310, but they have different frequency division settings.

In single-use mode (PSS1=1 and PSS3=0), TC1 independently uses a 10-bit prescaler CPS1, TC0 and TC3 share a 10-bit prescaler CPS310, but they have different frequency division settings.

In single-use mode (PSS1=0 and PSS3=1), TC3 independently uses a 10-bit prescaler CPS3, and TC0 and TC1 share a 10-bit prescaler CPS310, but they have different frequency division settings.

In independent mode (PSS1=1 and PSS3=1), TC0 independently uses a 10-bit prescaler CPS310, TC1 independently uses a 10-bit prescaler CPS1, and TC3 independently uses prescaler CPS3.

The following figure applies to TCO, TC1 and TC3, where n represents 0, 1 or 3.



TC0/TC1/TC3 Prescaler block diagram

Internal clock source

When setting CSn[2:0]=1, timer 3 can only be driven by the system clock clkio, timer counter 0 or 1 can be directly driven by the system clock clkio or high-speed clock rcm2x (2 times the frequency of the internal 32M RC oscillator output clock) drive. The prescaler can output 4 different clock frequencies, namely clkio/8, clkio/64, clkio/256 and clkio/1024.

Divider reset

Multiplexing mode

When setting the PSS1 bit to "0" and the PSS3 bit to "0", TC0, TC1 and TC3 share a prescaler CPS310.
The prescaler is independent, its operation is independent of the clock selection logic of TC, and it is shared by TC0, TC1 and TC3. Since it is not affected by the clock selection control, the state of the prescaler has an effect on the application of the divided clock. The effect occurs when the timer is enabled and the output of the prescaler is selected as the count clock source (6>CSn[2:0]>1).
It may take 1 to N+1 system clocks from the timer enable to the first count, where N is the prescaler factor (8, 64, 256 or 1024).
It is possible to synchronize timer and program operation by resetting the prescaler. But it must be noted that if another timer is using this prescaler, resetting the prescaler will affect all timers connected to it.

Single use mode

When the PSS1 bit is set to "1", TC1 uses the prescaler CPS1 independently, and the reset of the prescaler is controlled by the PSR1 bit. The respective resets act independently and do not affect other prescalers.
When setting the PSS3 bit to "1", TC3 uses the prescaler CPS3 independently, and the reset of the prescaler is controlled by the PSR3 bit. The respective resets act independently and do not affect other prescalers.
When the PSS1 bit is set to "1" and the PSS3 bit is "1", TC0 uses the prescaler CPS310 independently, the reset of the prescaler is controlled by the PSRSYNC bit, TC1 uses the prescaler CPS1 independently, and TC3 uses independently For the prescaler CPS3, the respective reset functions independently and will not affect other prescalers.
When the PSS1 bit is set to "1", TC1 uses the prescaler CPS1 independently, and the reset of the prescaler is controlled by the PSR1 bit. The respective resets act independently and do not affect other prescalers.
When setting the PSS3 bit to "1", TC3 uses the prescaler CPS3 independently, and the reset of the prescaler is controlled by the PSR3 bit. The respective resets act independently and do not affect other prescalers.
When the PSS1 bit is set to "1" and the PSS3 bit is "1", TC0 uses the prescaler CPS310 independently, the reset of the prescaler is controlled by the PSRSYNC bit, TC1 uses the prescaler CPS1 independently, and TC3 uses independently For the prescaler CPS3, the respective reset functions independently and will not affect other prescalers.

External clock source

External clock source provided by T0/T1/T3 pin can be used as count clock source. The signal of the T0/T1/T3 pin is used as the clock source of the counter after passing through the synchronous logic and the edge detector. Each rising edge (CSn[2:0]=7) or falling edge (CSn[2:0]=6) will generate a count pulse. External clock sources are not fed into the prescaler.

Due to the existence of synchronization and edge detection circuits on the pins, the level changes on T0/T1/T3 need to be delayed by 2.5 to 3.5 system clocks to update the counter.
Disable or enable the clock input must be at least one system clock cycle after T0/T1/T3 remains stable, otherwise there may be a possibility of wrong counting clock pulses.

In order to ensure correct sampling, the external clock pulse width must be greater than one system clock period, and the external clock frequency must be less than half of the system clock frequency when the duty cycle is 50%. Due to the difference in system clock frequency and duty cycle caused by the error of the oscillator itself, it is recommended that the highest frequency of the external clock should not be greater than fsys/2.5.

Register definitions

GTCCR – General Timer Counter Control Register

GTCCR – General Timer Counter Control Register								
address: 0x43					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TSM	-	-	-	-	-	PSRASY	PSRSYNC
R/W	R/W	-	-	-	-	-	W	W

Bit	Name	description
7	TSM	Timer counter synchronization mode control bit. When setting the TSM bit to "1", it is the timing counter synchronous mode. In synchronous mode, the value written to the PSRASY bit and PSRSYNC bit will be retained, allowing the corresponding prescaler to always be reset. This ensures that the corresponding timer counters are aborted and configured to the same value. When setting the TSM bit to "0", the value of the PSRASY bit and PSRSYNC bit will be cleared by hardware, and the timing counter will start working at the same time.
6:2	-	Reserved
1	PSRASY	See timer TC2 register description.
0	PSRSYNC	Prescaler CPS310 reset control bit. ➢ When setting the PSRSYNC bit to "1", the prescaler CPS310 will be reset. When the TSM bit is not set, the hardware will clear the PSRSYNC bit after reset. ➢ When setting the PSRSYNC bit to "0", the setting is invalid. ➢ In multiplexing mode, TC0/TC1/TC3 share the prescaler, reset will affect these three timers. ➢ In standalone mode, reset only affects TC0. ➢ The read value of this bit will always be "0".

PSSR – Prescaler Select Register

PSSR – Prescaler Select Register								
address: 0xE2					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	PSS1	PSS3	-	-	-	-	PSR3	PSR1
R/W	R/W	R/W	-	-	-	-	R/W	R/W

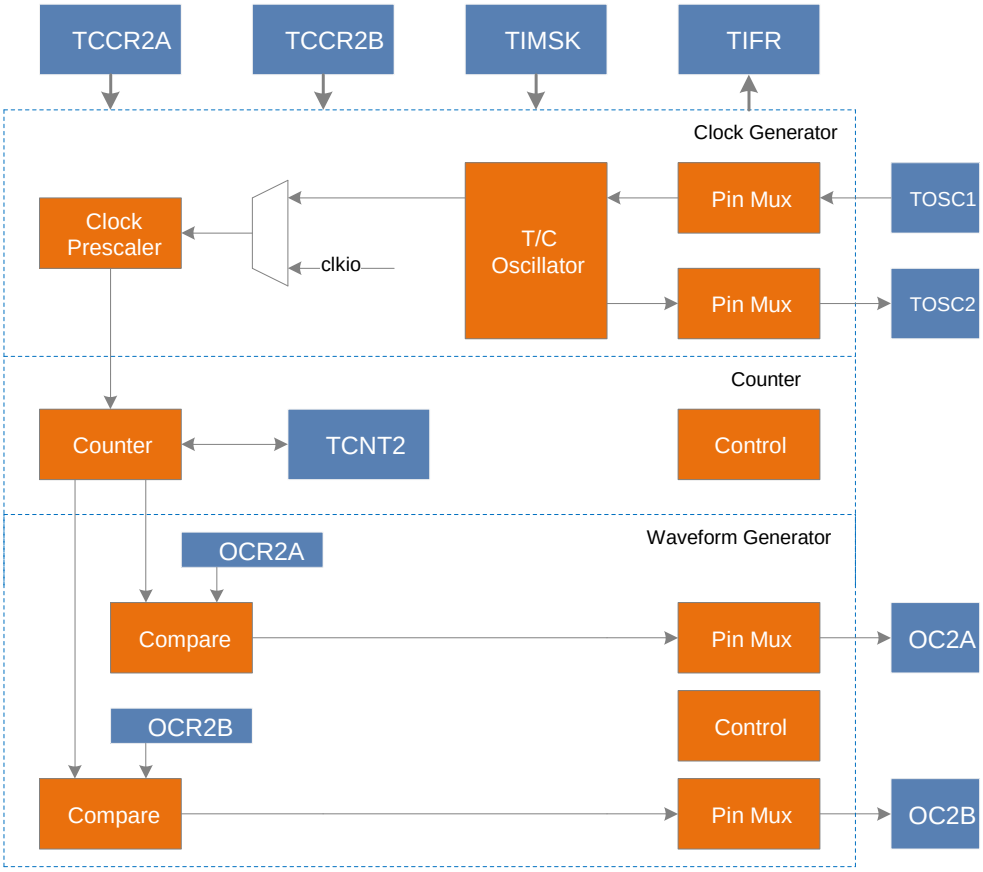
Bit	Name	description
7	PSS1	Prescaler selection control bits. When setting the PSS1 bit to "1", TC1 uses prescaler CPS1 alone. When setting the PSS1 bit to "0", it is the prescaler multiplexing mode. TC0 and TC1 share the prescaler CPS310. The prescaler CPS1 is invalid and will always be reset. If the PSS3 bit is "0" at the same time, TC3 shares prescaler CPS310 with TC0 and TC1. Both prescalers CPS1 and CPS3 are invalid and will always be reset.
6	PSS3	Prescaler selection control bits. When setting the PSS3 bit to "1", TC3 uses prescaler CPS3 alone.

		When setting the PSS3 bit to "0", it is the prescaler multiplexing mode. TC0 and TC3 share the prescaler CPS310. The prescaler CPS3 is invalid and will always be reset. If the PSS1 bit is "0" at the same time, TC1, TC0, and TC3 share the prescaler CPS310. Both prescalers CPS1 and CPS3 are invalid and will always be reset.
5:2	-	Reserved.
1	PSR3	Prescaler CPS3 reset control bit. The PSR3 bit is only valid in TC3 single-use mode. When setting the PSR3 bit to "1", the prescaler CPS3 will be reset. Hardware will clear PSR3 bit after reset. When setting the PSR3 bit to "0", the setting is invalid. The read value of this bit will always be "0".
0	PSR1	Prescaler CPS1 reset control bit. The PSR1 bit is only valid in TC1 single-use mode. When setting the PSR1 bit to "1", the prescaler CPS1 will be reset. The hardware will clear the PSR1 bit after reset. When setting the PSR1 bit to "0", the setting is invalid. The read value of this bit will always be "0".

Timer/Counter 2 (TMR2)

- ❑ 8-bit counter
- ❑ Two independent compare units
- ❑ The counter is automatically cleared and automatically loaded when a compare match occurs
- ❑ Phase-corrected PWM output without glitch pulses
- ❑ frequency generator
- ❑ External event counter
- ❑ 10-bit clock prescaler
- ❑ Overflow and compare match interrupt
- ❑ Allows the use of an external 32.768KHz RTC crystal to count

Overview



TC2 structure diagram

TC2 is a general-purpose 8-bit timer counter module that supports PWM output and can generate waveforms precisely. TC2 includes an 8-bit counter, waveform generation mode control unit and 2 output comparison units. The waveform generation mode control unit controls the working mode of the counter and the generation of the comparison output waveform. According to different working modes, the counter realizes clearing, adding one or subtracting one operation for each counting clock Clkt2. Clkt2 can be generated by internal clock source or external clock source.
When using an external 32.768KHz crystal to count, TC2 can be used as an RTC counter. When the count value of the counter TCNT2

When the maximum value is reached (equal to the maximum value 0xFF or the output comparison register OCR2A, which is defined as TOP, and the maximum value is defined as MAX to show the difference), the counter will be cleared or subtracted by one. When the count value TCNT2 of the counter reaches the minimum value (equal to 0x00, defined as BOTTOM), the counter will add one. When the count value TCNT2 of the counter reaches OCR2A/OCR2B, also known as when a comparison match occurs, it will be cleared or set to output the comparison signal OC2A/OCR2B to generate a PWM waveform.

Operating mode

Timing counter 2 has four different working modes, including normal mode (Normal), clear when compare match (CTC) mode, fast pulse width modulation (FPWM) mode and phase correction pulse width modulation (PCPWM) mode, generated by waveform Mode control bits WGM2[2:0] to select. These four modes are described in detail below. Since there are two independent output comparison units, they are represented by "A" and "B" respectively, and the lowercase "x" is used to represent the two output comparison unit channels.

Normal mode

The normal mode is the simplest working mode of the timer counter. At this time, the waveform generation mode control bit WGM2[2:0]=0, and the maximum value TOP of the count is MAX (0xFF). In this mode, the counting method increases by one for each counting clock, and when the counter reaches TOP and overflows, it returns to BOTTOM and starts accumulating again. The timer counter overflow flag TOV2 is set in the same count clock that the count value TCNT2 becomes zero. In this mode, the TOV2 flag is like the 9th counting bit, but it will only be set and not cleared. The overflow interrupt service routine will automatically clear the TOV2 flag, which can be used by software to increase the resolution of the timer counter. In normal mode, there is no special situation to consider, and a new count value can be written at any time.

The waveform of the output comparison signal OC2x can only be obtained when the data direction register of the OC2x pin is set to output. When COM2x=1, the OC2x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{oc2xnormal} = f_{sys}/(2*N*256)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

The output compare unit can be used to generate interrupts, but it is not recommended to use interrupts in normal mode, which will take up too much CPU time.

CTC mode

When setting WGM2[2:0]=2, the timer counter 2 enters the CTC mode, and the maximum value TOP of the count is OCR2A. In this mode, the counting method is incremented by one for each count clock, and the counter is cleared when the counter value TCNT2 is equal to TOP. OCR2A defines the maximum value of the count, which is the resolution of the counter. This mode allows the user to easily control the frequency of the compare match output, and also simplifies the operation of counting external events.

When the counter reaches the maximum value of the count, the output comparison match flag OCF2 is set, and an interrupt will be generated when the corresponding interrupt enable is set. In the interrupt service routine, the OCR2A register can be updated, that is, the maximum value of the count. OCR2A does not use double buffering in this mode, be careful when updating the maximum value close to the minimum value when the counter is operating with no prescaler or a very low prescaler. If the value written into OCR2A is less than the current TCNT2 value, the counter will lose a compare match. Before the next compare match occurs, the counter has to count to TOP first, and then start counting from BOTTOM to the OCR2A value. Same as the normal mode, the count value returns to the count clock of BOTTOM to set the TOV2 flag.

The waveform of the output comparison signal OC2x can only be obtained when the data direction register of the OC2x pin is set to output. When COM2x=1, the OC2x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{oc2xctc} = f_{sys}/(2*N*(1+OCR2A))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024). It can be seen from the formula that when OCR2x is set to 0x0 and there is no prescaler, an output waveform with a maximum frequency of fsys/2 can be obtained.

Fast PWM mode

When WGM2[2:0]=3 or 7, the timer counter 2 enters the fast PWM mode, which can be used to generate high-frequency PWM waveforms, and the maximum count TOP is MAX (0xFF) or OCR2A respectively. Fast PWM mode differs from other PWM modes in that it operates unidirectionally. The counter accumulates from the minimum value 0x00 to TOP and then returns to BOTTOM to count again.
When the count value TCNT2 reaches OCR2x or BOTTOM, the output comparison signal OC2x will be set or cleared, depending on the setting of the comparison output mode COM2x, see the register description for details. Due to the unidirectional operation, the fast PWM mode operates at twice the frequency of the phase-corrected PWM mode with bidirectional operation. The high frequency characteristics make the fast PWM mode suitable for power regulation, rectification and DAC applications. High-frequency signals can reduce the size of external components (inductance and capacitance, etc.), thereby reducing system cost.
When the count value reaches the maximum value, the timer counter overflow flag TOV2 will be set, and the value of the comparison buffer will be updated to the comparison value. If the interrupt is enabled, the compare buffer OCR2x register can be updated in the interrupt service routine.
The waveform of the output comparison signal OC2x can only be obtained when the data direction register of the OC2x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc2xfpwm} = f_{sys}/(N*(1+TOP))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
When a comparison match occurs between TCNT2 and OCR2x, the waveform generator will set (clear) the OC2x signal, and when TCNT2 is cleared, the waveform generator will clear (set) the OC2x signal to generate a PWM wave. Therefore, the extreme value of OCR2x will generate a special PWM waveform. When OCR2x is set to 0x00, the output PWM has a narrow spike every (1+TOP) count clock. When OCR2x is set to the maximum value, the output waveform is a continuous high level or low level.

Phase corrected PWM mode

When setting WGM2[2:0]=1 or 5, the timing counter 2 enters the phase correction PWM mode, and the maximum value TOP of counting is MAX (0xFF) or OCR2A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT2 matches OCR2x, the output comparison signal OC2x will be cleared or set, depending on the setting of the comparison output mode COM2x. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.
In the phase correction PWM mode, when the count reaches BOTTOM, the TOV2 flag is set, and when the count reaches TOP, the value of the comparison buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR2x register can be updated in the interrupt service routine.
The waveform of the output comparison signal OC2x can only be obtained when the data direction register of the OC2x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{oc2xpcpwm} = f_{sys}/(N*TOP*2)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).
During the count-up process, when TCNT2 matches OCR2x, the waveform generator clears (sets) the OC2x signal. During counting down, when TCNT2 matches OCR2x, the waveform generator sets (clears) the OC2x signal. Therefore, the extreme value of OCR2x will generate a special PWM wave. When OCR2x is set to the maximum or minimum value, the OC2x signal output will always remain low or high.
In order to ensure the symmetry of the output PWM wave on both sides of the minimum value, when there is no comparison match, the OC2x signal will be inverted in two cases. The first case is when the value of OCR2x changes from the maximum value of 0xFF to other data. When OCR2x is the maximum value and the count value reaches the maximum value, the output of OC2x is the same as the result of comparing and matching when counting in descending order, that is, keep OC2x unchanged. At this time, the comparison value will be updated to a new OCR2x value (not 0xFF), and the value of OC2x will remain until a comparison match occurs when counting up and flipped. At this time, the OC2x signal is not symmetrical about the minimum value, so it is necessary to flip the OC2x signal when TCNT2 reaches the

maximum value, that is, the first case where the OC2x signal is flipped when there is no comparison match. The second case is that when TCNT2 starts counting from a value higher than OCR2x, a compare match will be lost, causing an asymmetrical situation. It is also necessary to flip the OC2x signal to achieve symmetry on both sides of the minimum.

Asynchronous operation mode of TC2

When the AS2 bit in the ASSR register is "1", TC2 works in asynchronous mode, and the clock source of the counter comes from the oscillator of the external timing counter. The operation of TC2 in asynchronous mode should consider the following points.

- Switching between synchronous and asynchronous modes has the potential to corrupt TCNT2, OCR2A, OCR2B, TCCR2A, and TCCR2B data. The safe course of action is as follows:
  1. Clear the OCIE2A, TOIE2 and OCIE2B register bits to disable the TC2 interrupt;
  2. Set the AS2 bit to select the appropriate clock source;
  3. Write new data to TCNT2, OCR2A, TCCR2A, OCR2B and TCCR2B registers;
  4. When switching to asynchronous mode, it is necessary to wait for the TCN2UB, OCR2AUB, TCR2AUB, OCR2BUB and TCR2BUB bits to be cleared;
  5. Clear the interrupt flag bit of TC2;
  6. Enable interrupts that need to be used.
- The oscillator is best to use a 32.768KHz watch crystal oscillator. The system clock frequency must be more than 4 times higher than the crystal oscillator frequency.
- When the CPU writes TCNT2, OCR2A, TCCR2A, OCR2B, and TCCR2B, the hardware will put the data into the temporary register first, and then latch it into the corresponding register after two rising edges of the TOSC1 clock. New data write operations cannot be performed until the data is latched from the scratchpad into the destination register. Each register has its own independent scratchpad, so writing TCNT2 will not interfere with writing OCR2. The asynchronous status register ASSR is used to check whether the data has been written to the destination register.
- If TC2 is used as the wake-up condition of the MCU sleep mode, the sleep mode cannot be entered before the update of each register is completed, otherwise the MCU may enter the sleep mode before the TC2 setting takes effect, so TC2 cannot wake up the system.
- If TC2 is used as the wake-up condition of the MCU sleep mode, care must be taken in the process of re-entering the sleep mode. The interrupt logic requires one TOSC1 clock cycle to reset, if the time from wake-up to re-entering sleep is less than one TOSC1 clock cycle, the interrupt will no longer occur and the device will not wake up. The following operation method is recommended:
  1. Write appropriate data to each register;
  2. Wait for the corresponding update busy flag of ASSR to be cleared;
  3. Enter hibernation mode.
- If the asynchronous mode of operation is selected, the oscillator of TC2 will always work unless it enters power-down mode. The user must be aware that the oscillator stabilization time may be as long as 1 second, therefore, it is recommended that the user wait at least 1 second after enabling the oscillator of TC2 before using the asynchronous operation mode of TC2.
- The process of waking up in sleep mode in asynchronous working mode: After the interrupt condition is met, the wake-up process is started on the next timer clock. That is, the counter increments for at least one more clock before the processor can read the value of the counter. After waking up, the MCU executes the interrupt service routine, and then begins to execute the program after the SLEEP statement.
- Reading the value of TCNT2 shortly after waking up from sleep mode may return incorrect data. Because TCNT2 is driven by the asynchronous TOSC1 clock, reading TCNT2 must be done through an internal system clock synchronous register, and synchronization occurs on each rising edge of TOSC1. After waking up from sleep mode, the system clock is reactivated, and the read value of TCNT2 is the value before entering sleep mode, and will not be updated until the next rising edge of TOSC1 arrives. The phase of TOSC1 when waking up from sleep mode is completely unpredictable and depends on the wake-up time. Therefore, the recommended sequence to read the TCNT2 value is:
  1. Write an arbitrary value to OCR2A or TCCR2A;

1. Wait for the corresponding update busy flag to be cleared;
2. Read TCNT2.

- In asynchronous mode, the synchronization of interrupt flag bits requires 3 system clock cycles plus 1 timer cycle. The counter increments for at least one more clock before the MCU can read the counter value that caused the interrupt flag to be set. Changes of the output compare signal are synchronized with the timer clock, not the system clock.

TC2 Prescaler

The input clock of the TC2 prescaler is called clkt2s. The AS2 bit in the ASSR register selects the internal system clock clkio or the external TOSC1 clock source, which is connected to the system clock clkio by default. If AS2 is set, TC2 will be asynchronously driven by TOSC1. When TOSC1 pin and TOSC2 pin are externally connected with a 32.768KHz watch crystal oscillator, TC2 can be used as an RTC counter. Applying an external clock signal directly on the TOSC1 pin is not recommended.

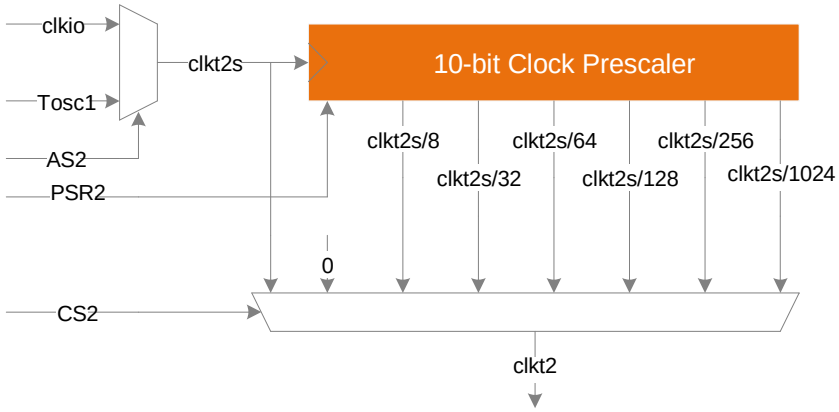


Figure 5 TC2 prescaler block diagram

The figure above shows the TC2 prescaler. As shown in the figure, the possible prescaler options are: clkt2s/8, clkt2s/32, clkt2s/64, clkt2s/128, clkt2s/256, and clkt2s/1024. In addition clkt2s and 0 (stop counting) can be selected. Setting the PSR2 bit of the SFIOR register will reset the prescaler, allowing the user to start with a predictable prescaler.

register definition

TC2 register list

register	Address	Default value	Description
TCCR2A	0xB0	0x00	TC2 Control Register A
TCCR2B	0xB1	0x00	TC2 Control Register B
TCNT2	0xB2	0x00	TC2 count value register
OCR2A	0xB3	0x00	TC2 output compare register A
OCR2B	0xB4	0x00	TC2 output compare register B
ASSR	0xB6	0x00	TC2 Asynchronous Status Register
TIMSK2	0x70	0x00	Timer Counter Interrupt Mask Register
TIFR2	0x37	0x00	Timer Counter Interrupt Flag Register

TCCR2A – TC2 Control Register A

TCCR2 A – TC2 control register A								
address: 0xB0						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20
R/W	W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	description
7	COM2A1	TC2 compare match output A mode control high bit. COM2A1 and COM2A0 form an output comparison mode to control COM2A[1:0] and control the output waveform of OC2A. If both bits 1 or 2 of COM2A are set, the output comparison waveform occupies the OC2A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM2A controls the output comparison waveform differently, see the comparison output mode control table description for details.
6	COM2A0	TC2 compare match output A mode control low bit. COM2A0 and COM2A1 form an output comparison mode to control COM2A[1:0] and control the output waveform of OC2A. If both bits 1 or 2 of COM2A are set, the output comparison waveform occupies the OC2A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM2A controls the output comparison waveform differently, see the comparison output mode control table description for details.
5	COM2B1	TC2 compare match output B mode control high bit. COM2B1 and COM2B0 together form an output comparison mode to control COM2B[1:0] and control the output waveform of OC2B. If both bits 1 or 2 of COM2B are set, the output comparison waveform occupies the OC2B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM2B controls the output comparison waveform differently, see the comparison output mode control table description for details.
4	COM2B0	TC2 compare match output B mode control low bit. COM2B0 and COM2B1 form an output comparison mode to control COM2B[1:0] and control the output waveform of OC2B. If both bits 1 or 2 of COM2B are set, the output comparison waveform occupies the OC2B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM2B controls the output comparison waveform differently, see the comparison output mode control table description for details.
3:2	-	Reserved.
1	WGM21	TC2 waveform generation mode control high bit. WGM20, WGM21, and WGM22 form a waveform generation mode to control WGM2[2:0] to control the counting mode of the counter and the waveform generation mode. For details, see the waveform generation mode table description.
0	WGM20	TC2 waveform generation mode control low bit. WGM21, WGM20, and WGM22 form a waveform generation mode to control WGM2[2:0] to control the counting mode of the counter and the waveform generation mode. For details, see the waveform generation mode table description.

TCCR2B – TC2 Control Register B

TCCR2B – TC2 control register B								
address: 0xB1					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC2A	FOC2B	-	-	WGM22	CS22	CS21	CS20
R/W	W	W	-	-	R/W	R/W	R/W	R/W

Bit	Name	description
7	FOC2A	TC2 forces the output compare A control bit. When working in non-PWM mode, a compare match can be generated by writing "1" to the forced output compare bit FOC2A. Forced compare match will not set the OCF2A flag, nor will it reload or clear the timer, but the output pin OC2A will be updated accordingly according to the COM2A setting, just like a compare match really happened. The return value of reading FOC2A is always zero.
6	FOC2B	TC2 forces the output compare B control bit. When working in non-PWM mode, a compare match can be generated by writing "1" to the forced output compare bit FOC2B. Forced compare match will not set the OCF2B flag, nor will it reload or clear the timer, but the output pin OC2B will be updated accordingly according to the settings of COM2B, just like a compare match really happened. The return value of reading FOC2B is always zero.
5:4	-	Reserved
3	WGM22	TC2 waveform generation mode control high bit. WGM22, WGM20, and WGM21 form a waveform generation mode to control WGM2[2:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
2	CS22	TC2 clock selection control high. Used to select the clock source for timer counter 2.
1	CS21	TC2 clock selection control bit. Used to select the clock source for timer counter 2.
0	CS20	TC2 clock selection control low. Used to select the clock source for timer counter 2.

CS2[2:0]	description
0	No clock source, stop counting
1	clk_{t2s}
2	$clk_{t2s}/8$, From the prescaler
3	$clk_{t2s}/32$, From the prescaler
4	$clk_{t2s}/64$, From the prescaler
5	$clk_{t2s}/128$, From the prescaler
6	$clk_{t2s}/256$, From the prescaler
7	$clk_{t2s}/1024$, From the prescaler

The following table shows the control of the comparison output mode on the output comparison waveform in non-PWM mode (that is, normal mode and CTC mode).

Table 1 OC2x compare output mode control in non-PWM mode

COM2x[1:0]	description
0	OC2x disconnected, general IO port operation
1	Toggle OC2x signal on compare match
2	Toggle OC2x signal on compare match
3	Set OC2x signal on compare match

The following table shows the control of the comparison output mode on the output comparison waveform in the fast PWM mode.

Table 2 OC2x compare output mode control in fast PWM mode

COM2x[1:0]	description
0	OC2x disconnected, general I/O port operation
1	Reserved
2	Clear OC2x on compare match, and set OC2x on max match
3	Set OC2x on compare match, and clear OC2x on max match

The following table shows the control of the output comparison waveform in the comparison output mode in the phase correction mode.

Table 3 OC2x compare output mode control in phase correction PWM mode

COM2x[1:0]	description
0	OC2x disconnected, general I/O port operation
1	Reserved
2	OC2x cleared on compare match in up-counting, and OC2x set on compare match in down-counting.
3	OC2x set on compare match in up-counting, and OC2x cleared on compare match in down-counting.

The table below shows the waveform generation mode controls.

Table 4 Waveform Generation Mode Control

WGM2[2:0]	Working mode	TOP value	Update OCR2x moments	Set TOV2 moments
0	Normal	0xFF	Immediate	MAX
1	PCPWM	0xFF	TOP	BOTTOM
2	CTC	OCR2A	Immediate	MAX
3	FPWM	0xFF	TOP	MAX
4	Reserved	-	-	-
5	PCPWM	OCR2A	TOP	BOTTOM
6	Reserved	-	-	-
7	FPWM	OCR2A	TOP	TOP

TCNT2 – TC2 count value register

TCNT2 – TC2 count value register								
address: 0xB2					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT27	TCNT26	TCNT25	TCNT24	TCNT23	TCNT22	TCNT21	TCNT20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	TCNT2	TC2 count value register. The 8 count values of the counter can be read and written directly through the TCNT2 register. A CPU write operation to the TCNT2 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer has been stopped. This allows initializing the value of the TCNT2 register to match the value of OCR2 without causing an interrupt. If the value written to TCNT2 equals or bypasses the value of OCR2, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT2. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

OCR2A – TC2 Output Compare Register A

OCR2A – TC2 Output Compare Register A								
address: 0xB3					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR2A7	OCR2A6	OCR2A5	OCR2A4	OCR2A3	OCR2A2	OCR2A1	OCR2A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR2A	TC2 output compare register A. OCR2A contains an 8-bit data, which is continuously compared with the counter value TCNT2. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC2A pin. When using PWM mode, the OCR2A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR2A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR2A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR2A itself.

OCR2B – TC2 Output Compare Register B

OCR2B – TC2 Output Compare Register B								
address: 0xB4					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR2B7	OCR2B6	OCR2B5	OCR2B4	OCR2B3	OCR2B2	OCR2B1	OCR2B0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7:0	OCR2B	TC2 output compare B register. OCR2B contains an 8-bit data, which is continuously compared with the counter value TCNT2. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC2B pin. When using PWM mode, the OCR2B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR2B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR2B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR2B itself.

TIMSK2 – TC2 Interrupt Mask Register

TIMSK2 – TC2 Interrupt Mask Register								
address: 0x70					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	description
7:3		Reserved.
2	OCIE2B	TC2 output compare B match interrupt enable bit. When the OCIE2B bit is "1" and the global interrupt is set, the TC2 output compare B match interrupt is enabled. When a compare match occurs, that is, when the OCF2B bit in TIFR2 is set, an interrupt is generated. When the OCIE2B bit is "0", the TC2 output compare B match interrupt is disabled.
1	OCIE2A	TC2 output compare A match interrupt enable bit. When the OCIE2A bit is "1" and the global interrupt is set, the TC2 output compare A match interrupt is enabled. When a compare match occurs, that is, when the OCF2A bit in TIFR2 is set, an interrupt is generated. When the OCIE2A bit is "0", the TC2 output compare A match interrupt is disabled.
0	TOIE2	TC2 overflow interrupt enable bit. When the TOIE2 bit is "1" and the global interrupt is set, the TC2 overflow interrupt is enabled. An interrupt is generated when TC2 overflows, that is, the TOV2 bit in TIFR2 is set. When the TOIE2 bit is "0", the TC2 overflow interrupt is disabled.

TIFR2 – TC2 Interrupt Flag Register

TIFR2 – TC2 Interrupt Flag Register								
address: 0x37					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCF2B	OCF2A	TOV2
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	description
7:3	-	Reserved.
2	OCF2B	TC2 output compare B match flag.

		When TCNT2 is equal to OCR2B, the comparison unit gives a matching signal and sets the comparison flag OCF2B. If the output compare B interrupt enable OCIE2B is "1" and the global interrupt flag is set at this time, an output compare B interrupt will be generated. OCF2B will be automatically cleared when this interrupt service routine is executed, or write "1" to the OCF2B bit to clear this bit.
1	OCF2A	TC2 outputs compare A match flag. When TCNT2 is equal to OCR2A, the comparison unit gives a matching signal and sets the comparison flag OCF2A. If the output compare A interrupt enable OCIE2A is "1" and the global interrupt flag is set at this time, an output compare A interrupt will be generated. OCF2A will be automatically cleared when this interrupt service routine is executed, or writing "1" to the OCF2A bit can also clear this bit.
0	TOV2	TC2 overflow flag. When the counter overflows, the overflow flag TOV2 is set. If the overflow interrupt enable TOIE2 is "1" and the global interrupt flag is set at this time, an overflow interrupt will be generated. TOV2 will be automatically cleared when this interrupt service routine is executed, or it can be cleared by writing "1" to the TOV2 bit.

ASSR – Asynchronous Interface Status Register

ASSR – TC2 Asynchronous Interface Status Register								
address: 0xB6					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	INTCK	-	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	description
7	INTCK	Asynchronous clock selection control bit. When setting the INTCK bit to 1, select the internal RC32K as the asynchronous clock source. When setting the INTCK bit to 0, the external crystal oscillator clock is selected as the asynchronous clock source.
6	-	Reserved.
5	AS2	Timer 2 asynchronous mode selection control bit. When setting the AS2 bit to 1, Timer 2 works in asynchronous mode, and its clock source is selected by the INTCK bit. When setting the AS2 bit to 0, Timer 2 works in synchronous mode, and its clock source is Clkio. When the value of AS2 changes, the values of TCNT2, OCR2A, OCR2B, TCCR2A and TCCR2B registers may be incorrect and need to be reconfigured.
4	TCN2UB	TCNT2 register update flag. When Timer 2 works in asynchronous mode and writes to TCNT2, the TCN2UB bit will be set. When the value of TCNT2 is updated, the hardware will clear the TCN2UB bit. TCNT2 can only be updated when the TCN2UB bit is 0.
3	OCR2AUB	OCR2A register update flag. When Timer 2 works in asynchronous mode and writes to OCR2A, the OCR2AUB bit will be set. When the value of OCR2A is updated, the hardware will clear the OCR2AUB bit. OCR2A can only be updated when the OCR2AUB bit is 0.
2	OCR2BUB	OCR2B register update flag. When Timer 2 works in asynchronous mode and writes to OCR2B, the OCR2BUB bit will be set. When the value of OCR2B is updated, the hardware will clear the OCR2BUB bit. OCR2B can only be updated when the OCR2BUB bit is 0.

1	TCR2AUB	TCCR2A register update flag. When Timer 2 works in asynchronous mode and writes to TCCR2A, the TCR2AUB bit will be set. When the value of TCCR2A is updated, the hardware will clear the TCR2AUB bit. TCCR2A can be updated only when the TCR2AUB bit is 0.
0	TCR2BUB	TCCR2B register update flag. When Timer 2 works in asynchronous mode and writes to TCCR2B, the TCR2BUB bit will be set. When the value of TCCR2B is updated, the hardware will clear the TCR2BUB bit. TCCR2B can only be updated when the TCR2BUB bit is 0.

Timer/Counter 3 (TMR3)

- ❑ True 16-bit design, allowing 16-bit PWM
- ❑ 3 Independent Output Compare Units
- ❑ Double buffered output compare register
- ❑ 1 input capture unit
- ❑ Input Capture Noise Suppressor
- ❑ The counter is automatically cleared and automatically loaded on a compare match
- ❑ Phase-corrected PWM without glitch pulses
- ❑ Variable PWM period
- ❑ frequency generator
- ❑ External event counter
- ❑ 5 independent interrupt sources
- ❑ With dead time control
- ❑ 6 selectable trigger sources to automatically turn off the PWM output

Overview

TC3 is a general-purpose 16-bit timer counter module that supports PWM output and can generate waveforms precisely. TC3 includes a 16-bit counter, waveform generation mode control unit, 2 independent output comparison units and an input capture unit. The waveform generation mode control unit controls the working mode of the counter and the generation of the comparison output waveform. According to different working modes, the counter realizes clearing, adding one or subtracting one operation for each counting clock Clkt3. Clkt3 can be generated by internal clock source or external clock source. When the count value TCNT3 of the counter reaches the maximum value (equal to the maximum value 0xFFFF or a fixed value or the output comparison register OCR3A or the input capture register ICR3, which is defined as TOP, and the maximum value is defined as MAX to show the difference), the counter will be cleared. or minus one operation. When the count value TCNT3 of the counter reaches the minimum value (equal to 0x0000, defined as BOTTOM), the counter will add one. When the count value TCNT3 of the counter reaches OCR3A or OCR3B or OCR3C, also known as when a compare match occurs, it will be cleared or set to output the comparison signal OC3A or OC3B or OC3C to generate a PWM waveform. When the input capture function is turned on, the counter starts or stops counting when it is triggered, and the ICR3 register will record the count value within the trigger period of the capture signal.

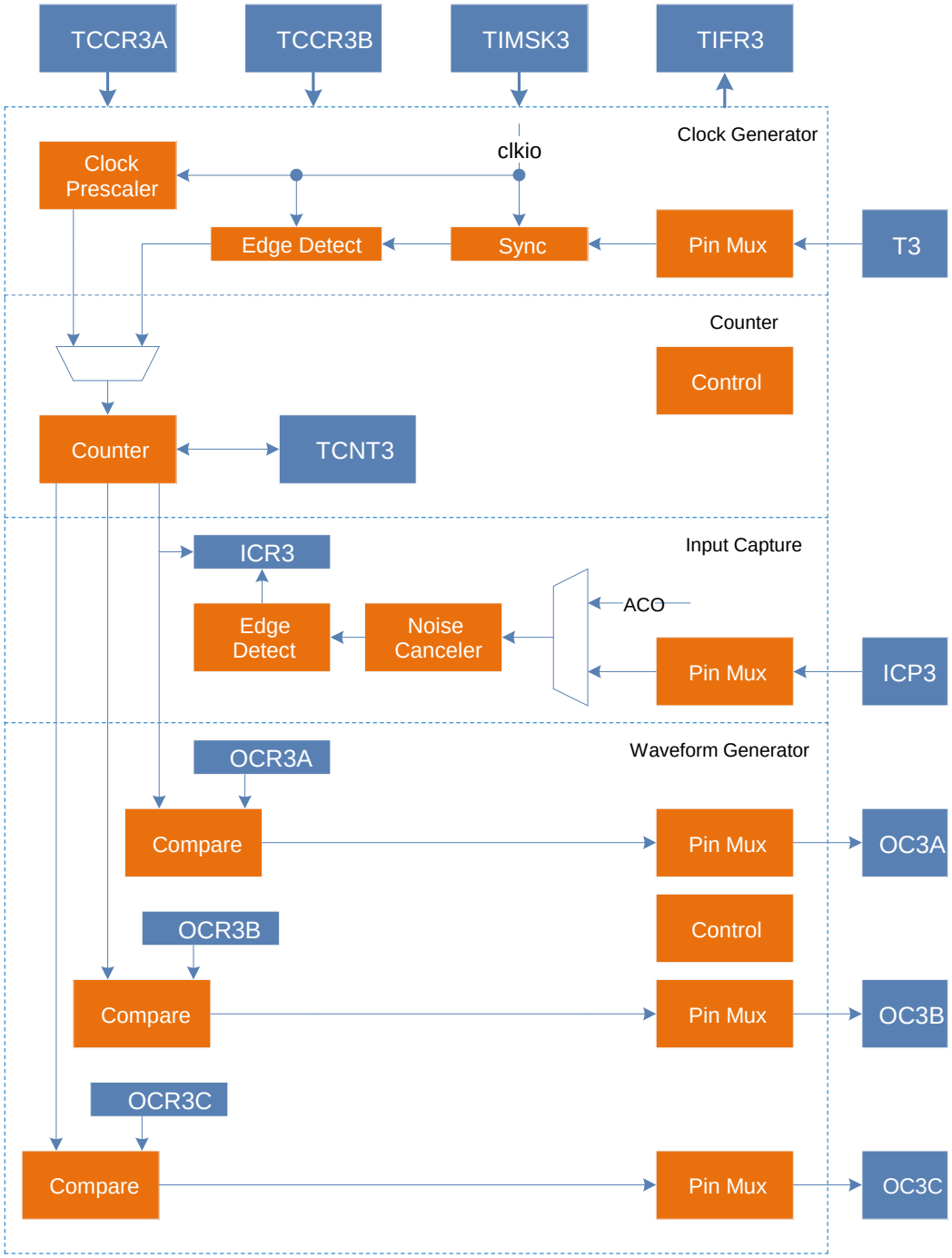


Figure 6 TC3 structure diagram

Operating mode

Timing counter 1 has six different working modes, including normal mode (Normal), clear when compare match (CTC) mode, fast pulse width modulation (FPWM) mode, phase correction pulse width modulation (PCPWM) mode, phase frequency correction Pulse Width Modulation (PFCPWM) mode, and Input Capture (ICP) mode. It is selected by the waveform generation mode control bits WGM3[3:0]. The six modes are described in detail below. Since there are three independent output comparison units, which are represented by "A", "B" and "C" respectively, the two output comparison unit channels are represented by a lowercase "x".

Normal mode

The normal mode is the simplest working mode of the timer counter. At this time, the waveform generation mode control bit WGM3[3:0]=0, and the maximum value TOP of the count is MAX (0xFFFF). In this mode, the counting method increases by one for each counting clock, and when the counter reaches TOP and overflows, it returns to BOTTOM and starts accumulating again. The timer counter overflow flag TOV3 is set in the same count clock that the count value TCNT3 becomes zero. In this mode, the TOV3 flag is like the 17th counting bit, but it will only be set and not cleared. The overflow interrupt service routine will automatically clear the TOV3 flag, which can be used by software to increase the resolution of the timer counter. In normal mode, there is no special situation to consider, and a new count value can be written at any time.
The waveform of the output comparison signal OC3x can only be obtained when the data direction register of the OC3x pin is set to output. When COM3x=1, the OC3x signal will be inverted when a compare match occurs. In this case, the frequency of the waveform can be calculated by the following formula:

$$f_{OC3xnormal} = f_{sys}/(2*N*65536)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

The output compare unit can be used to generate interrupts, but it is not recommended to use interrupts in normal mode, which will take up too much CPU
time.

CTC Mode

When setting WGM3[3:0]=4 or 12, timer counter 1 enters CTC mode. When WGM3[3]=0, the maximum count TOP is OCR3A, when WGM3[3]=1, the maximum count TOP is ICR3. The following takes WGM3[3:0]=4 as an example to describe the CTC mode. In this mode, the counting method is incremented for each count clock. When the counter value TCNT3 is equal to TOP, the counter is cleared. This mode allows the user to easily control the frequency of the compare match output, and also simplifies the operation of counting external events.

When the counter reaches TOP=OCR3A, the output comparison match flag OCF3A is set, when the counter reaches TOP=ICR3, the output comparison match flag ICF3 is set, and an interrupt will be generated when the corresponding interrupt enable is set. The OCR3A register can be updated in the interrupt service routine. The OCR3A does not use double buffering in this mode, so be careful when updating the maximum value close to the minimum value when the counter is operating with no prescaler or a very low prescaler. If the value written into OCR3A is less than the current TCNT3 value, the counter will lose a comparison match. Before the next compare match occurs, the counter has to count to MAX first, and then start counting from BOTTOM to OCR3A. Same as the normal mode, the TOV3 flag is set in the count clock when the count value returns to 0x0.

The waveform of the output comparison signal OC3x can only be obtained when the data direction register of the OC3x pin is set to output. The frequency of the waveform can be calculated using the following formula:

$$f_{OC3xctc} = f_{sys}/(2*N*(1+OCR3A))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

It can be seen from the formula that when OCR3A is set to 0x0 and there is no prescaler, an output waveform with a maximum frequency of fsys/2 can be obtained.

When WGM3[3:0]=12, it is similar to WGM3[3:0]=4, just replace the one related to OCR3A with ICR3.

快速PWM 模式

When WGM3[3:0]=5, 6, 7, 14 or 15, the timing counter 1 enters the fast PWM mode, and the maximum count TOP is 0xFF, 0x1FF, 0x3FF, ICR3 or OCR3A, which can be used to generate high-frequency PWM waveform. Fast PWM mode differs from other PWM modes in that it operates unidirectionally.

After the counter is accumulated from BOTTOM to TOP, it returns to BOTTOM to count again. When the count value TCNT3 reaches TOP or BOTTOM, the output comparison signal OC3x will be set or cleared, depending on the setting of the comparison output mode COM3, see the register description for details. Due to the unidirectional operation, the fast PWM mode operates at twice the frequency of the phase-corrected PWM mode with bidirectional operation. The high frequency characteristics make the fast PWM mode suitable for power regulation, rectification and DAC applications. High-frequency signals can reduce the size of external components (inductance and capacitance, etc.), thereby reducing system cost.

When the count value reaches TOP, the timer counter overflow flag TOV3 will be set, and the value of the comparison buffer will be updated to the comparison value. If the interrupt is enabled, the OCR3A register can be updated in the interrupt service routine.

The waveform of the output comparison signal OC3x can only be obtained when the data direction register of the OC3x pin is set to output. The frequency of the waveform can be calculated with the following formula:

$$f_{OC3xfpwm} = f_{sys}/(N*(1+TOP))$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

When a comparison match occurs between TCNT3 and OCR3x, the waveform generator will set (clear) the OC3x signal, and when TCNT3 is cleared, the waveform generator will clear (set) the OC3x signal to generate a PWM wave. Therefore, the extreme value of OCR3x will generate a special PWM waveform. When OCR3x is set to 0x00, the output PWM has a narrow peak pulse every (1+TOP) count clock. When OCR3x is set to TOP, the output waveform is continuous high level or low level. If OCR3A is used as TOP and COM3A=1 is set, the output comparison signal OC3A will generate a PWM wave with a duty cycle of 50%.

Phase corrected PWM mode

When setting WGM3[3:0]=1, 2, 3, 10 or 11, the timing counter 1 enters the phase correction PWM mode, and the maximum counting TOP is 0xFF, 0x1FF, 0x3FF, ICR3 or OCR3A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT3 matches OCR3x, the output comparison signal OC3x will be cleared or set, depending on the setting of the comparison output mode COM3. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.

In the phase correction PWM mode, when the count reaches BOTTOM, the TOV3 flag is set, and when the count reaches TOP, the value of the comparison buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR3x register can be updated in the interrupt service routine.

The output comparison signal OC3x waveform can only be obtained when the data direction register of the OC3x pin is set as an output. The frequency of the waveform can be calculated with the following formula:

$$f_{OC3xcpcpwm} = f_{sys}/(N*TOP*2)$$

Among them, N represents the prescaler factor (1, 8, 64, 256 or 1024).

During counting up, when TCNT3 matches OCR3x, the waveform generator clears (sets) the OC3x signal. During the countdown process, when TCNT3 matches OCR3x, the waveform generator sets (clears) the OC3x signal. Therefore, the extreme value of OCR3x will generate a special PWM wave. When OCR3x is set to TOP or BOTTOM, the OC3x signal output will keep low or high level.

If OCR3A is used as TOP and COM3A=1 is set, the output comparison signal OC3A will generate a PWM wave with a duty cycle of 50%.
In order to ensure the symmetry of the output PWM wave on both sides of BOTTOM, when there is no comparison match, there are two cases where the OC3x signal will be inverted. The first case is when the value of OCR3x is changed from TOP to other data. When OCR3x is TOP and the count value reaches TOP, the output of OC3x is the same as the result of comparing and matching when counting in descending order, that is, keep OC3x unchanged. At this time, the comparison value will be updated to the new OCR3x value (not TOP), and the OC3x value will remain until the comparison match occurs when counting up and flipped. At this time, the OC3x signal is not symmetrical about the minimum value, so it is necessary to flip the OC3x signal when TCNT3 reaches the maximum value, that is, the first case of flipping the OC3x signal when there is no comparison match. The second case is that when TCNT3 starts counting from a value higher than OCR3x, a compare match will be lost, causing an asymmetrical situation. It is also necessary to flip the OC3x signal to achieve symmetry on both sides of the minimum.

Phase frequency correct PWM mode

When setting WGM3[3:0]=8 or 9, the timing counter 1 enters the phase frequency correction PWM mode, and the maximum value TOP of the count is ICR3 or OCR3A respectively. The counter adopts two-way operation, increments from BOTTOM to TOP, then decrements to BOTTOM, and repeats this operation. When the counting reaches TOP and BOTTOM, the counting direction is changed, and the counting value only stays on TOP or BOTTOM for one counting clock. During the increment or decrement process, when the count value TCNT3 matches OCR3x, the output comparison signal OC3x will be cleared or set, depending on the setting of the comparison output mode COM3. Compared with unidirectional operation, the maximum frequency obtainable by bidirectional operation is smaller, but its excellent symmetry is more suitable for motor control.

In the phase frequency correction PWM mode, when the count reaches BOTTOM, the TOV3 flag is set, and the value of the comparison buffer is updated to the comparison value. The time for updating the comparison value is the biggest difference between the phase frequency correction PWM mode and the phase correction PWM mode. If the interrupt is enabled, the compare buffer OCR3x register can be updated in the interrupt service routine. When the CPU changes the TOP value, that is, the value of OCR3A or ICR3, it must be ensured that the new TOP value is not smaller than the TOP value already in use, otherwise the comparison match will not happen again.

The output comparison signal OC3x waveform can only be obtained when the data direction register of the OC3x pin is set as an output. The frequency of the waveform can be calculated with the following formula:

$$f_{OC3xcpfcpwm} = f_{sys}/(N*TOP*2)$$

Where N represents the prescaler factor (1, 8, 64, 256 or 1024).

During count-up, when TCNT3 matches OCR3x, the waveform generator clears (sets) the OC3x signal.
During the countdown process, when TCNT3 matches OCR3x, the waveform generator sets (clears) the OC3x signal. Therefore, the extreme value of OCR3x will generate a special PWM wave. When OCR3x is set to TOP or BOTTOM, the OC3x signal output will keep low or high level. If OCR3A is used as TOP and COM3A=1 is set, the output comparison signal OC3A will generate a PWM wave with a duty cycle of 50%.

Because the OCR3x register is updated at BOTTOM time, the counting lengths of the ascending and descending order on both sides of the TOP value are the same, and a symmetrical waveform with correct frequency and phase is produced.

When using a fixed TOP value, it is best to use the ICR3 register as the TOP value, that is, set WGM3[3:0]=8, at this time the OCR3A register is only used to generate PWM output. If you want to generate PWM waves with frequency changes, you must change the TOP value, and the double buffering feature of OCR3A will be more suitable for this application.

Input capture mode

Input capture is used to capture external events and give them time stamps to illustrate the moment when this event occurs. It can be performed in the previous counting mode, but the waveform generation mode that uses the ICR3 value as the counting TOP value must be removed.

The trigger signal of the external event is input by the pin ICP3, and it can also be realized by the analog comparator unit. When the logic level on the pin ICP3 changes, or the output ACO level of the analog comparator changes, and this level change is captured by the input capture unit, the input capture is triggered, and the 16-bit count value TCNT3 data is copied to the input capture register ICR3, and the input capture flag ICF3 is set. If the ICIE1 bit is "1", the input capture flag will generate an input capture interrupt.

By setting the analog comparison control and status register ACSR analog comparison input capture control bit ACIC to select the input capture trigger source ICP3 or ACO. It should be noted that changing the trigger source may cause an input capture, so ICF3 must be cleared once after changing the trigger source to avoid erroneous results.

The input capture signal is sent to the edge detector after passing through an optional noise suppressor. According to the configuration of the input capture selection control bit ICES1, see whether the detected edge meets the trigger condition. The noise suppressor is a simple digital filter that samples the input signal 4 times, and its output is fed to the edge detector only when the values of the 4 samples are equal. The noise suppressor is enabled or disabled controlled by the ICNC1 bit of the TCCR3B register.

When using the input capture function, after ICF3 is set, the value of the ICR3 register should be read as early as possible, because the value of ICR3 will be updated after the next capture event occurs. It is recommended to enable the input capture interrupt. In any input capture mode, it is not recommended to change the count TOP value during operation.

The time stamps captured by the input can be used to calculate frequency, duty cycle, and other characteristics of the signal, as well as create logs for trigger events. When measuring the duty cycle of an external signal, it is required to change the trigger edge after each capture, so the trigger signal edge must be changed as soon as possible after reading the ICR3 value.

Automatic shutdown and restart of PWM output

When the DOC3x bit of the TCCR3C register is set high, the automatic shutdown function of the PWM output will be enabled. When the trigger condition is met, the hardware will clear the corresponding COM3x bit, disconnect the PWM output signal OC3x from its output pin, and switch to General IO output to realize automatic shutdown of PWM output. At this time, the state of the output pin can be controlled by the output of the general-purpose IO port.

After the automatic shutdown of the PWM output is enabled, the trigger condition needs to be set, and the trigger source is selected by the DSX3n bit of the TCCR3D register. Trigger sources include analog comparator interrupt, external interrupt, pin level change interrupt and timer overflow interrupt. For details, please refer to the TCCR3D register description. When one or some trigger sources are selected as trigger conditions, when these interrupt flag bits are set, the hardware will clear the COM3x bit to turn off the PWM output.

When a trigger event occurs and the PWM output is turned off, the timer module does not have a corresponding interrupt flag bit, and the software needs to read the interrupt flag bit of the trigger source to know the trigger condition and trigger event.

When the PWM output is automatically turned off and the output needs to be restarted again, the software only needs to reset the COM3x bit to switch the OC3x signal output to the corresponding pin. It should be noted that after the automatic shutdown occurs, the timer does not stop working, and the status of the OC3x signal has been updated. The software can set the COM3x bit to output the OC3x signal after the timer overflows or compares a match, so that a clear PWM output state can be obtained.

Dead time control

When the DTEN3 bit is set to "1", the function of inserting dead time is enabled, and the output waveforms of OC3A and OC3B will insert the set dead time on the basis of the waveform generated by the comparison output of channel B, and the length of time is DTR3 The time value corresponding to the count clock number of the register. As shown in the figure below, the dead time insertion of OC3A and OC3B is based on the comparative output waveform of channel B. When COM3A and COM3B are both "2" or "3", the waveform polarity of OC3A is the same as that of OC3B; when COM3A and COM3B are "2" or "3" respectively, the waveform of OC3A is the same as that of OC3B opposite polarity.

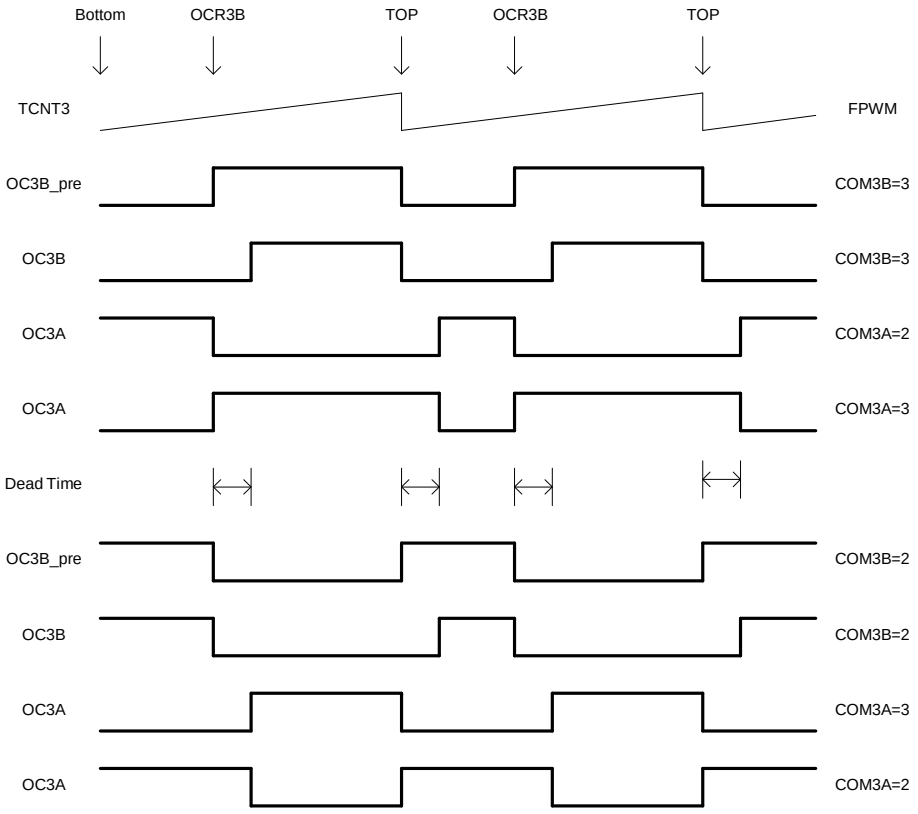


Figure 7 TC3 dead time control in FPWM mode

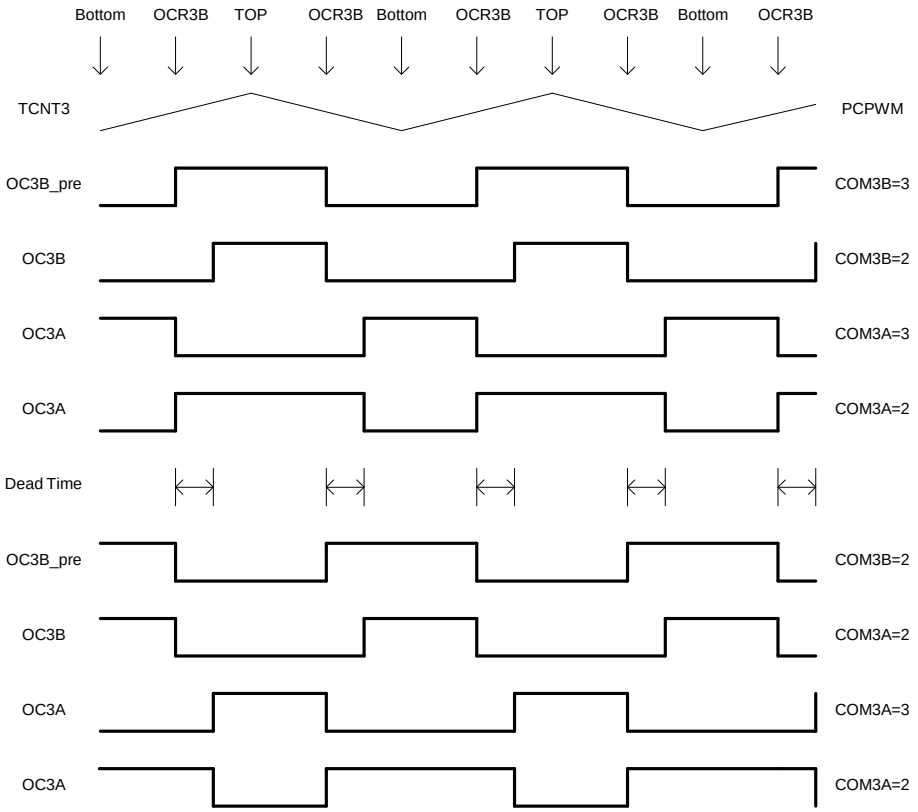


Figure 8 TC3 dead time control in PCPWM mode

When the DTEN3 bit is set to "0", the function of inserting dead time is disabled, and the output waveforms of OC3A and OC3B are the waveforms generated by their respective comparison outputs.

Registers definition

TC3 register list

Register	Address	Default value	Description
TCCR3A	0x90	0x00	TC3 Control Register A
TCCR3B	0x91	0x00	TC3 Control Register B
TCCR3C	0x92	0x00	TC3 Control Register C
TCCR3D	0x93	0x00	TC3 Control Register D
TCNT3L	0x94	0x00	TC3 count value register low byte
TCNT3H	0x95	0x00	TC3 count value register high byte
ICR3L	0x96	0x00	TC3 input capture register low byte
ICR3H	0x97	0x00	TC3 input capture register high byte
OCR3AL	0x98	0x00	TC3 output compare register A low byte
OCR3AH	0x99	0x00	TC3 output compare register A high byte
OCR3BL	0x9A	0x00	TC3 output compare register B low byte
OCR3BH	0x9B	0x00	TC3 output compare register B high byte
DTR3L	0x9C	0x00	TC3 Dead Time Register Low Byte
DTR3H	0x9D	0x00	TC3 Dead Time Register High Byte
OCR3CL	0x9E	0x00	TC3 output compare register C low byte

OCR3CH	0x9F	0x00	TC3 output compare register C high byte
TIMSK3	0x71	0x00	Timer Counter Interrupt Mask Register
TIFR3	0x38	0x00	Timer Counter Interrupt Flag Register

TCCR3A – TC3 Control Register A

TCCR3A – TC3 Control Register A								
address: 0x90					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	COM3A1	COM3A0	COM3B1	COM3B0	COM3C1	COM3C0	WGM31	WGM30
R/W	R/W	R/W	R/W	R/W	W	W	R/W	R/W

Bit	Name	description
7	COM3A1	Compare match output A mode control high. COM3A1 and COM3A0 form COM3A[1:0] to control the output comparison waveform OC3A. If both bits 1 or 2 of COM3A are set, the output comparison waveform occupies the OC3A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3A controls the output comparison waveform differently, see the comparison output mode control table description for details.
6	COM3A0	Compare match output A mode control low bit. COM3A1 and COM3A0 form COM3A[1:0] to control the output comparison waveform OC3A. If both bits 1 or 2 of COM3A are set, the output comparison waveform occupies the OC3A pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3A controls the output comparison waveform differently, see the comparison output mode control table description for details.
5	COM3B1	Compare match output B mode control high. COM3B1 and COM3B0 form COM3B[1:0] to control the output comparison waveform OC3B. If both bits 1 or 2 of COM3B are set, the output comparison waveform occupies the OC3B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3B controls the output comparison waveform differently, see the comparison output mode control table description for details.
4	COM3B0	Compare match output B mode control low. COM3B1 and COM3B0 form COM3B[1:0] to control the output comparison waveform OC3B. If both bits 1 or 2 of COM3B are set, the output comparison waveform occupies the OC3B pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3B controls the output comparison waveform differently, see the comparison output mode control table description for details.
3	COM3C1	Compare match output C mode control high. COM3C1 and COM3C0 form COM3C[1:0] to control the output comparison waveform OC3C. If both bits 1 or 2 of COM3C are set, the output comparison waveform occupies the OC3C pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3C controls the output comparison waveform differently, see the comparison output mode control table description for details.
2	COM3C0	Compare match output C mode control low. COM3C1 and COM3C0 form COM3C[1:0] to control the output comparison

		waveform OC3C. If both bits 1 or 2 of COM3C are set, the output comparison waveform occupies the OC3C pin, but the data direction register of this pin must be set high to output this waveform. In different working modes, COM3C controls the output comparison waveform differently, see the comparison output mode control table description for details.
1	WGM31	Waveform Generation Mode Control Second Low Bit. WGM31, WGM33, WGM32, and WGM30 form a waveform generation mode to control WGM3[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
0	WGM30	The least significant bit of the waveform generation mode control. WGM30, WGM33, WGM32, and WGM31 form a waveform generation mode to control WGM3[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.

The following table shows the control of the comparison output mode on the output comparison waveform in non-PWM mode (that is, normal mode and CTC mode).
Comparative output mode control in non-PWM mode

COM3x[1:0]	Description
0	OC3x disconnected, general IO port operation
1	Invert OC3x signal on compare match
2	Clear OC3x signal on compare match
3	Assert OC3x signal on compare match

The following table shows the control of the comparison output mode on the output comparison waveform in the fast PWM mode.

Comparative output mode control in fast PWM mode

COM3x[1:0]	Description
0	OC3x disconnected, general IO port operation
1	When WGM3 is 15: OC3A signal is flipped when compare match, OC3B is disconnected When WGM3 is other values: OC3x disconnected, general IO port operation
2	Clear OC3x on compare match, set OC3x on max-value match
3	Set OC3x on compare match, Clear OC3x on max-value match

The following table shows the control of the output comparison waveform in the comparison output mode in the phase correction mode.
Comparator output mode control in phase correction and phase frequency correction PWM mode

COM3x[1:0]	Description
0	OC3x disconnected, general IO port operation
1	When WGM3 == 9 or 11: toggle OC3A on compare match, OC3B disconnect When WGM3 != 9 or 11: OC3x disconnected, general IO port operation
2	Compare match clears OC3x when counting up, and compare match sets OC3x when counting down.
3	Comparison match sets OC3x when counting up, and compare match clears OC3x when counting down.

TCCR3B – TC3 Control Register B

TCCR3B – TC3 Control Register B								
Address: 0x91					default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICNC3	ICES3	-	WGM33	WGM32	CS32	CS31	CS30
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	ICNC3	Input capture noise suppressor enable control bit. When the ICNC3 bit is set to "1", the input capture noise suppressor is enabled. At this time, the input of the external pin ICP3 is filtered, and the input signal is only valid when 4 consecutive sampling values are equal. This function delays the input capture by 4 clock cycle. When the ICNC3 bit is set to "0", the input capture noise suppressor is prohibited, and the input of the external pin ICP3 is directly effective at this time.
6	ICES3	Input capture trigger edge selection control bit. When the ICES3 bit is set to "1", the rising edge of the selection level triggers the input capture; when the ICES3 bit is set to "0", the falling edge of the selection level triggers the input capture. When an event is captured, the counter value is copied to the ICR3 register and the input capture flag ICF3 is set. If the interrupt is enabled, an input capture interrupt is generated.
5	-	Reserved.
4	WGM33	Waveform generation mode control high. WGM33, WGM32, WGM31, and WGM30 form a waveform generation mode to control WGM3[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
3	WGM32	The second most significant bit is the waveform generation mode control. WGM32, WGM33, WGM31, and WGM30 form a waveform generation mode to control WGM3[3:0], control the counting mode of the counter and the waveform generation mode, see the waveform generation mode table description for details.
2	CS32	Clock select control high. It is used to select the clock source of timer counter 3.
1	CS31	Clock select control bit. It is used to select the clock source of timer counter 3.
0	CS30	Clock select control low. It is used to select the clock source of timer counter 3.

CS3[2:0]	Description
0	No clock source, stop counting
1	clk_{sys}
2	$clk_{sys}/8$, From the prescaler
3	$clk_{sys}/64$, From the prescaler
4	$clk_{sys}/256$, From the prescaler
5	$clk_{sys}/1024$, From the prescaler
6	External clock T3 pin, falling edge trigger
7	External clock T3 pin, rising edge trigger

The table below shows the waveform generation mode controls.

Table 5 Waveform Generation Mode Control

WGM3[3:0]	Operating mode	TOP value	Update OCR1A moment	Set TOV3 moment
0	Normal	0xFFFF	Immediate	MAX
1	8-bit PCPWM	0x00FF	TOP	BOTTOM
2	9-bit PCPWM	0x01FF	TOP	BOTTOM
3	10-bit PCPWM	0x03FF	TOP	BOTTOM
4	CTC	OCR3A	Immediate	MAX
5	8-bit FPWM	0x00FF	BOTTOM	TOP
6	9-bit FPWM	0x01FF	BOTTOM	TOP
7	10-bit FPWM	0x03FF	BOTTOM	TOP
8	PFCPWM	ICR3	BOTTOM	BOTTOM
9	PFCPWM	OCR3A	BOTTOM	BOTTOM
10	PCPWM	ICR3	TOP	BOTTOM
11	PCPWM	OCR3A	TOP	BOTTOM
12	CTC	ICR3	Immediate	MAX
13	Reserved	-	-	-
14	FPWM	ICR3	TOP	TOP
15	FPWM	OCR3A	TOP	TOP

TCCR3C – TC3 Control Register C

TCCR3C – TC3 Control Register C								
Address: 0x92					default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	FOC3A	FOC3B	DOC3B	DOC3A	DTEN3	-	DOC3C	FOC3C
R/W	W	W	-	-	-	-	-	-

Bit	Name	description
7	FOC3A	Force output compare A. When working in non-PWM mode, a comparison match can be generated by writing "1" to the forced output comparison bit FOC3A. Forced compare match will not set the OCF3A flag, nor will it reload or clear the timer, but the output pin OC3A will be updated accordingly according to the setting of COM3A, just like a compare match really happened. When working in PWM mode, it must be cleared when writing the TCCR3A register. The return value of reading FOC3A is always zero.
6	FOC3B	Force output compare B. When working in non-PWM mode, a comparison match can be generated by writing "1" to the forced output comparison bit FOC3B. Forced compare match will not set the OCF3B flag, nor will it reload or clear the timer, but the output pin OC3B will be updated accordingly according to the setting of COM3B, just like a compare match really happened. When working in PWM mode, it must be cleared when writing the TCCR3A register. The return value of reading FOC3B is always zero.
5	DOC3B	Disable output compare B enable control bit.

		When the DOC3B bit is high, the hardware prohibits the output comparison B to be enabled. After the condition for prohibiting the output is met, the COM3B bit will be cleared, the output pin OC3B is disconnected, and the pin becomes a general-purpose IO operation. When the DOC3B bit is low, the hardware disables output compare B function is invalid.
4	DOC3A	Disable output compare A enable control bit. When the DOC3A bit is high, the hardware disables the output comparison A and is enabled. After meeting the conditions for prohibiting the output, the COM3A bit will be cleared, the output pin OC3A is disconnected, and the pin becomes a general-purpose IO operation. When the DOC3A bit is low, the hardware disables the output compare A function is invalid.
3	DTEN3	Dead time enable control bit. When the DTEN3 bit is high, the dead time is enabled, OC3A and OC3B become complementary outputs, and the dead time is inserted as set by DTR3L and DTR3H. Dead time is disabled when DTEN3 bit is low. Both OC3A and OC3B are single output.
2	-	
1	DOC3C	Disable output compare C enable control bit. When the DOC3C bit is high, the hardware prohibits the output comparison C to be enabled. After the condition of prohibiting the output is satisfied, the COM3C bit will be cleared, the output pin OC3C is disconnected, and the pin becomes a general IO operation. When the DOC3C bit is low, the hardware disable output compare C function is invalid.
0	FOC3C	Force output compare C. When working in non-PWM mode, you can generate a comparison match by writing "1" to the forced output comparison bit FOC3C. Forced compare match will not set the OCF3C flag, nor will it reload or clear the timer, but the output pin OC3C will be updated accordingly according to the setting of COM3C, just like a compare match really happened. When working in PWM mode, it must be cleared when writing the TCCR3A register. The return value of reading FOC3C is always zero.

TCCR3D – TC3 Control Register D

TCCR3D – TC3 Control Register D								
Address: 0x93					Default time: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	DSX37	DSX36	DSX35	DSX34	-	-	DSX31	DSX30
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	Description
7	DSX37	TC3 trigger source selection control enable bit 7. When the DSX37 bit is set to "1", TC0 overflow is enabled as a trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX37 bit is set to "0", TC0 overflow is disabled as a trigger source for closing the output comparison signal waveform OC3x.
6	DSX36	TC3 trigger source selection control enable bit 6. When the DSX36 bit is set to "1", TC2 overflow is enabled as a trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX36 bit is set to "0", TC2 overflow is disabled as a trigger source for closing the output comparison signal waveform OC3x.

5	DSX35	TC3 trigger source selection control enable bit 5. When the DSX35 bit is set to "1", the pin level change 1 is enabled as the trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX35 bit is set to "0", the pin level change 1 is disabled as the trigger source for closing the output comparison signal waveform OC3x.
4	DSX34	TC3 trigger source selection control enable bit 4. When the DSX34 bit is set to "1", the external interrupt 1 is enabled as the trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX34 bit is set to "0", the external interrupt 1 is disabled as the trigger source for closing the output comparison signal waveform OC3x.
3:2	-	Reserved.
1	DSX31	TC3 trigger source selection control enable bit 1. When the DSX31 bit is set to "1", the analog comparator 1 is enabled as a trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX31 bit is set to "0", the analog comparator 1 is disabled as a trigger source for closing the output comparison signal waveform OC3x.
0	DSX30	TC3 trigger source selection control enable bit 0. When the DSX30 bit is set to "1", the analog comparator 0 is enabled as a trigger source for closing the output comparison signal waveform OC3x. When the DOC3x bit is "1", the rising edge of the interrupt flag register bit of the selected trigger source will automatically turn off the waveform output of OC3x. When the DSX30 bit is set to "0", the analog comparator 0 is disabled as a trigger source for closing the output comparison signal waveform OC3x.

The following table shows the selection control of the trigger source of the waveform output.

Turn off the trigger source selection control for the OC3x waveform output

DOC3x	DSX3n=1	Trigger source	Description
0	-	-	DOC3x bit="0", trigger source off and the waveform out function disabled
1	0	Analog comparator 0	Rising edge of ACIF0 will turn off OC3x waveform output
1	1	Analog comparator 1	Rising edge of ACIF1 will turn off OC3x waveform output
1	4	External interrupt 1	Rising edge of INTF1 will turn off OC3x waveform output
1	5	Pin level change 1	Rising edge of PCIF1 will turn off OC3x waveform output
1	6	TC2 overflow	Rising edge of TOV2 will turn off OC3x waveform output
1	7	TC0 overflow	Rising edge of TOV0 will turn off OC3x waveform output

note：

2) DSX3n=1 means that when the nth bit of the TCCR1D register is 1, each register bit can be set at the same time.

TCNT3L–TC3 counter register low byte

TCNT3L – TC3 count value register low byte								
Address: 0x94					default value: 0x00			
Bit	7	6	5	4	3	2	1	0

Name	TCNT3L7	TCNT3L6	TCNT3L5	TCNT3L4	TCNT3L3	TCNT3L2	TCNT3L1	TCNT3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	TCNT3L	Low byte of TC3 count value. TCNT3H and TCNT3L are combined to form TCNT3, and the 16-bit count value of the counter can be read and written directly through the TCNT3 register. Reading and writing a 16-bit register requires two operations. When writing 16-bit TCNT3, write TCNT3H first. When reading 16-bit TCNT3, TCNT3L should be read first. A CPU write operation to the TCNT3 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer has been stopped. This allows initializing the value of the TCNT3 register to match the value of OCR3x without causing an interrupt. If the value written to TCNT3 equals or bypasses the OCR3x value, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT3. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

TCNT3H – TC3 Counter Register High Byte

TCNT3H – TC3 count value register high byte								
Address: 0x95					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TCNT3H7	TCNT3H6	TCNT3H5	TCNT3H4	TCNT3H3	TCNT3H2	TCNT3H1	TCNT3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	TCNT3H	High byte of TC3 count value. TCNT3H and TCNT3L are combined to form TCNT3, and the 16-bit count value of the counter can be read and written directly through the TCNT3 register. Reading and writing a 16-bit register requires two operations. When writing 16-bit TCNT3, write TCNT3H first. When reading 16-bit TCNT3, TCNT3L should be read first. A CPU write operation to the TCNT3 register will prevent a compare match from occurring on the next timer clock cycle, even if the timer has been stopped. This allows initializing the value of the TCNT3 register to match the value of OCR3x without causing an interrupt. If the value written to TCNT3 equals or bypasses the OCR3x value, the compare match will be lost, resulting in incorrect waveform generation results. The timer stops counting when no clock source is selected, but the CPU can still access TCNT3. The CPU write counter has a higher priority than clearing or adding and subtracting operations.

ICR3L – TC3 Capture Register Low Byte

ICR3L – TC3 Input Capture Register Low Byte								
Address: 0x96					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ICR3L7	ICR3L6	ICR3L5	ICR3L4	ICR3L3	ICR3L2	ICR3L1	ICR3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	ICR3L	TC3 inputs the low byte of the capture value. ICR3H and ICR3L combine to form the 16-position ICR3. Reading and writing a 16-bit register requires two operations. When writing 16-bit ICR3, ICR3H should be written first. When reading 16-bit ICR3, ICR3L should be read first.

		When the input capture is triggered, the count value TCNT3 will be updated and copied to the ICR3 register. The ICR3 register can also be used to define the TOP value of the count.

ICR3H – TC3 Capture Register High Byte

ICR3H – TC3 Input Capture Register High Byte								
Address: 0x97					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ICR3H7	ICR3H6	ICR3H5	ICR3H4	ICR3H3	ICR3H2	ICR3H1	ICR3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	ICR3H	TC3 inputs the high byte of the capture value. ICR3H and ICR3L combine to form the 16-position ICR3. Reading and writing a 16-bit register requires two operations. When writing 16-bit ICR3, ICR3H should be written first. When reading 16-bit ICR3, ICR3L should be read first. When the input capture is triggered, the count value TCNT3 will be updated and copied to the ICR3 register. The ICR3 register can also be used to define the TOP value of the count.						

OCR3AL–TC3 Output Compare Register A Low Byte

OCR3AL – TC3 output compare register A low byte								
Address: 0x98					Default Value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3AL7	OCR3AL6	OCR3AL5	OCR3AL4	OCR3AL3	OCR3AL2	OCR3AL1	OCR3AL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	OCR3AL	Low byte of output compare register A. OCR3AL and OCR3AH combine to form the 16-position OCR3A. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3A, OCR3AH should be written first. When reading 16-bit OCR3A, OCR3AL should be read first. OCR3A is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3A pin. When using PWM mode, the OCR3A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR3A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3A itself.							

OCR3AH – TC3 Output Compare Register A High Byte

OCR3AH – TC3 output compare register A high byte								
Address: 0x99					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR3AH7	OCR3AH6	OCR3AH5	OCR3AH4	OCR3AH3	OCR3AH2	OCR3AH1	OCR3AH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	OCR3AH	High Byte of Output Compare Register A.						

		OCR3AL and OCR3AH combine to form the 16-position OCR3A. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3A, OCR3AH should be written first. When reading 16-bit OCR3A, OCR3AL should be read first. OCR3A is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3A pin. When using PWM mode, the OCR3A register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3A register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR3A buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3A itself.

OCR3BL–TC3 Output Compare Register B Low Byte

OCR3BL – TC3 output compare register B low byte								
Address: 0x9A					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3BL7	OCR3BL6	OCR3BL5	OCR3BL4	OCR3BL3	OCR3BL2	OCR3BL1	OCR3BL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	OCR3BL	Low byte of output compare register B. OCR3BL and OCR3BH combine to form the 16-position OCR3B. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3B, OCR3BH should be written first. When reading 16-bit OCR3B, OCR3BL should be read first. OCR3B is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3B pin. When using PWM mode, the OCR3B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR3B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3B itself.							

OCR3BH – TC3 Output Compare Register B High Byte

OCR3BH – TC3 output compare register B high byte								
Address: 0x9B					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3BH7	OCR3BH6	OCR3BH5	OCR3BH4	OCR3BH3	OCR3BH2	OCR3BH1	OCR3BH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	OCR3BH	High Byte of Output Compare Register B. OCR3BL and OCR3BH combine to form the 16-position OCR3B. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3B, OCR3BH should be written first. When reading 16-bit OCR3B, OCR3BL should be read first. OCR3B is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3B pin. When using PWM mode, the OCR3B register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3B register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses.							

		When using the double buffering function, the CPU accesses the OCR3B buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3B itself.

OCR3CL–TC3 Output Compare Register C Low Byte

OCR3CL – TC3 output compare register C low byte								
Address: 0x9E					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3CL7	OCR3CL6	OCR3CL5	OCR3CL4	OCR3CL3	OCR3CL2	OCR3CL1	OCR3CL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	OCR3CL	Low byte of output compare register C. OCR3CL and OCR3CH combine to form the 16-position OCR3C. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3C, OCR3CH should be written first. When reading 16-bit OCR3C, OCR3CL should be read first. OCR3C is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3C pin. When using PWM mode, the OCR3C register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3C register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR3C buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3C itself.

OCR3CH – TC3 Output Compare Register C High Byte

OCR3CH –TC3 output compares register C high byte								
Address: 0x9F					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3CH7	OCR3CH6	OCR3CH5	OCR3CH4	OCR3CH3	OCR3CH2	OCR3CH1	OCR3CH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	OCR3CH	High Byte of Output Compare Register C. OCR3CL and OCR3CH combine to form the 16-position OCR3C. Reading and writing a 16-bit register requires two operations. When writing 16-bit OCR3C, OCR3CH should be written first. When reading 16-bit OCR3C, OCR3CL should be read first. OCR3C is continuously compared with the counter value TCNT3. A compare match can be used to generate an output compare interrupt, or to generate a waveform on the OC3C pin. When using PWM mode, the OCR3C register uses a double-buffered register. In the normal working mode and match clear mode, the double buffering function is disabled. Double buffering can synchronize the update of the OCR3C register with the maximum or minimum value of the count, thereby preventing asymmetrical PWM pulses and eliminating interference pulses. When using the double buffering function, the CPU accesses the OCR3C buffer register, and when the double buffering function is disabled, the CPU accesses the OCR3C itself.

DTR3L – TC3 Dead Time Register Low Byte

DTR3L – TC3 Dead Time Register Low Byte	
Address: 0x9C	Default value: 0x00

Bit	7	6	5	4	3	2	1	0
Name	DTR3L7	DTR3L6	DTR3L5	DTR3L4	DTR3L3	DTR3L2	DTR3L1	DTR3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	DTR3L	Dead Time Register Low Byte. When the DTEN3 bit is high, OC3A and OC3B are complementary outputs, and the dead time inserted on the OC3A output is determined by DTR3L counting clocks.						

DTR3H – TC3 Dead Time Register High Byte

DTR3H – TC3 Dead Time Register High Byte								
Address: 0x9D					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	DTR3H7	DTR3H6	DTR3H5	DTR3H4	DTR3H3	DTR3H2	DTR3H1	DTR3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	DTR3H	Dead Time Register High Byte. When the DTEN3 bit is high, OC3A and OC3B are complementary outputs, and the dead time inserted on the OC3B output is determined by DTR3H counting clocks.						

TIMSK3–TC3 Interrupt Mask Register

TIMSK3 – TC3 Interrupt Mask Register								
Address: 0x71					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	ICIE3	-	OCIE3C	OCIE3B	OCIE3A	TOIE3
R/W	-	-	R/W	-	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:6	-	Reserved.						
5	ICIE3	TC3 input capture interrupt enable control bit. When the ICIE3 bit is "1" and the global interrupt is set, the TC3 input capture interrupt is enabled. When the input capture is triggered, that is, the ICF3 flag of TIFR3 is set, an interrupt occurs. When the ICIE3 bit is "0", the TC3 input capture interrupt is disabled.						
4	-	Reserved.						
3	OCIE3C	TC3 output compare C match interrupt enable bit. When the OCIE3C bit is "1" and the global interrupt is set, the TC3 output compare C match interrupt is enabled. When a compare match occurs, that is, when the OCF3C bit in TIFR3 is set, an interrupt is generated. When the OCIE3C bit is "0", the TC3 output compare C match interrupt is disabled.						
2	OCIE3B	TC3 output compare B match interrupt enable bit. When the OCIE3B bit is "1" and the global interrupt is set, the TC3 output compare B match interrupt is enabled. When a compare match occurs, that is, when the OCF3B bit in TIFR3 is set, an interrupt is generated. When the OCIE3B bit is "0", the TC3 output compare B match interrupt is disabled.						
1	OCIE3A	TC3 output compare A match interrupt enable bit. When the OCIE3A bit is "1" and the global interrupt is set, the TC3 output						

		compare A match interrupt is enabled. When a compare match occurs, that is, when the OCF3A bit in TIFR3 is set, an interrupt is generated. When the OCIE3A bit is "0", the TC3 output compare A match interrupt is disabled.
0	TOIE3	TC3 overflow interrupt enable bit. When the TOIE3 bit is "1" and the global interrupt is set, the TC3 overflow interrupt is enabled. An interrupt is generated when TC3 overflows, ie the TOV3 bit in TIFR3 is set. When the TOIE3 bit is "0", the TC3 overflow interrupt is disabled.

TIFR3–TC3 Interrupt Flag Register

TIFR3 – TC3 Interrupt Flag Register								
Address: 0x38					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	ICF3	-	-	OCF3B	OCF3A	TOV3
R/W	-	-	R/W	-	-	R/W	R/W	R/W

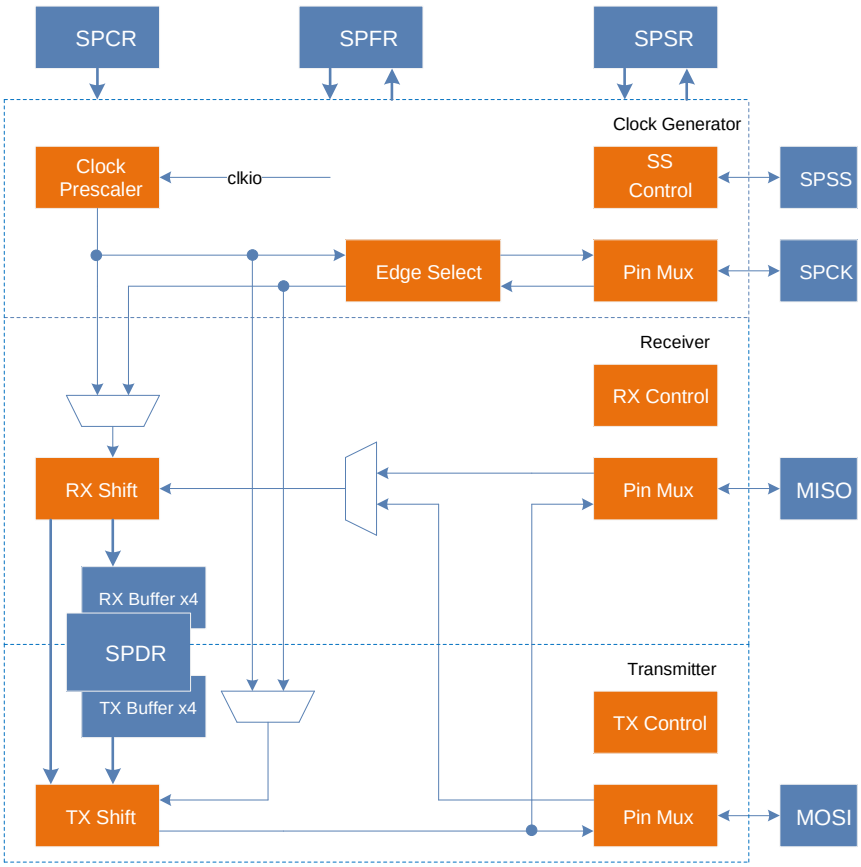
Bit	Name	Description
7:6	-	Reserved.
5	ICF3	Input capture flag. The ICF3 flag is set when an input capture event occurs. When ICR3 is used as the TOP value for counting and the count value reaches the TOP value, the ICF3 flag is set. If ICIE1 is "1" and the global interrupt flag is set, an input capture interrupt will be generated. The ICF3 flag will not be automatically cleared, and software needs to write "1" to the ICF3 bit to clear it.
4	-	Reserved.
3	OCF3C	Output compare C match flag. When TCNT3 is equal to OCR3C, the comparison unit gives a match signal and sets the comparison flag OCF3C. If the output comparison interrupt enable OCIE3C is "1" and the global interrupt flag is set at this time, an output comparison interrupt will be generated. The OCF3 flag bit will not be automatically cleared, and the software needs to write "1" to the OCF3C bit to clear the bit.
2	OCF3B	Output compare B match flag. When TCNT3 is equal to OCR3B, the comparison unit gives a match signal and sets the comparison flag OCF3B. If the output comparison interrupt enable OCIE3B is "1" and the global interrupt flag is set at this time, an output comparison interrupt will be generated. The OCF3B flag bit will not be automatically cleared, and software needs to write "1" to the OCF3B bit to clear it.
1	OCF3A	Output compare A match flag. When TCNT3 is equal to OCR3A, the comparison unit gives a match signal and sets the comparison flag OCF3A. If the output comparison interrupt enable OCIE3A is "1" and the global interrupt flag is set at this time, an output comparison interrupt will be generated. The OCF3A flag bit will not be automatically cleared, and software needs to write "1" to the OCF3A bit to clear it.
0	TOV3	overflow flag. When the counter overflows, the overflow flag TOV3 is set. If the overflow interrupt enable TOIE3 is "1" and the global interrupt flag is set, an overflow interrupt will be generated. The TOV3 flag will not be automatically cleared, and software needs to write "1" to the TOV3 bit to clear it.

Synchronous Serial Peripheral Interface (SPI)

- ❑ Full-duplex, three-wire synchronous data transmission
- ❑ Master or slave operation
- ❑ LSB or MSB first
- ❑ 7 programmable bit rates
- ❑ send end interrupt flag
- ❑ Write Conflict Flag Protection Mechanism
- ❑ Wake up from idle mode
- ❑ Double-speed mode for Master operation
- ❑ Support Master two-wire input mode
- ❑ 4 buffer registers for input/output

Overview

SPI mainly includes three parts: clock prescaler, clock detector, slave select detector, transmitter and receiver.



SPI structure diagram

Control and status registers are shared by all three parts. The clock prescaler only works in the Master operating mode, and the frequency division coefficient is selected by the bit rate control bit, thereby generating the corresponding frequency division clock and outputting it to the SPCK pin. The clock detector only works in the slave operation mode, detects the clock edge input from the SPCK pin, and performs shift operations on the sending and receiving shift registers according to the data transmission mode of the SPI.

The slave machine selection detector detects the slave machine selection signal SPSS to obtain the transmission status to control the operation of the transmitter and receiver. The transmitter consists of a shift register and transmit control logic.
The receiver consists of a shift register, four receive buffers and receive control logic.

Clock generation

The clock generation logic is divided into a master clock prescaler and a slave clock detector, which work in master and slave modes respectively. The clock prescaler selects the frequency division coefficient by the bit rate control bit and the double speed control bit, and generates the corresponding frequency division clock (there are 7 optional frequency division coefficients, see the register description for details), and the output to the SPCK pin is The clock is provided for the communication and also provides the shift clock for the internal transmit and receive shift registers. The clock detector detects the edge of the input clock SPCK, and shifts the transmitter and receiver according to the data transmission mode of SPI. In order to ensure the correct sampling of the clock signal, the width of the high level and low level of the SPCK clock must be greater than 2 system clock cycles.

Send and receive

The SPI module supports simultaneous transmission and reception in single-wire mode, and only supports dual-wire reception by the Master in dual-wire mode.

Single wire send and receive

The SPI master pulls down the slave selection signal SPSS that needs to communicate, and then a transmission process can be started. The Master and the slave prepare the data to be transmitted, the Master generates clock pulses on the clock signal SPCK to exchange data, the data of the Master is shifted out from MOSI, shifted in from MISO, the data of the slave is shifted out from MISO, shifted in from MOSI, and the exchange is completed After the data, the Master pulls up the SPSS signal to complete the communication.

When configured as a master, the SPI module does not control the SPSS pins and must be handled by user software. The software pulls down the SPSS pin, selects the slave to communicate, and initiates the transfer. The software writes the data to be transmitted into the SPDR register to start the clock generator, and the hardware generates the clock for communication, and shifts the 8-bit data out to the slave, and at the same time shifts the data of the slave in. After shifting one byte of data, stop the clock generator and set the transfer complete flag SPIF. The software can write data to the SPDR register again to continue to transmit the next byte, or pull the SPSS signal high to end the current transmission. The last incoming data will be kept in the receive buffer.

When configured as a slave, the SPI module will remain asleep as long as the SPSS signal remains high and keep the MISO pin tri-stated. At this point software can update the contents of the SPDR register. Even if there is an input clock pulse on the SPCK pin at this time, the data of SPDR will not be shifted out until the SPSS signal is pulled low. When the data transmission of one byte is completed, the hardware sets the transmission completion flag SPIF. At this time, the software can continue to write data to the SPDR register before reading the shifted data, and the last incoming data will be saved in the receive buffer.

The SPI module has only four buffers in the transmit direction and four buffers in the receive direction. When sending data, when the sending buffer is not full (that is, the sending buffer full flag bit WRFULL is low), the SPDR register can be written. When receiving data, when the receiving buffer is in a non-empty state (that is, the receiving buffer empty flag bit RDEMPT is low), the received characters can be read by accessing the SPDR register.

Master two-wire reception

The two-wire mode of the SPI module is only valid in the Master operation mode. The difference from the single-wire mode is that both MOSI and MISO are used for the Master to receive data, and each SPCK clock pulse simultaneously receives 2 bits of data (the data on the MISO line comes first, the

data on the MOSI line is behind), after receiving two bytes of data, the hardware sets the transmission completion flag SPIF, and the data is saved to the receive buffer and shift register. At this time, the software must read the SPDR register twice to obtain the received two bytes of data. It should be noted that although the master does not send data to the slave in the two-wire mode, the software still needs to write data to the SPDR register to start the clock generator to generate the communication clock, and write to the SPDR register once to receive two bytes of data .

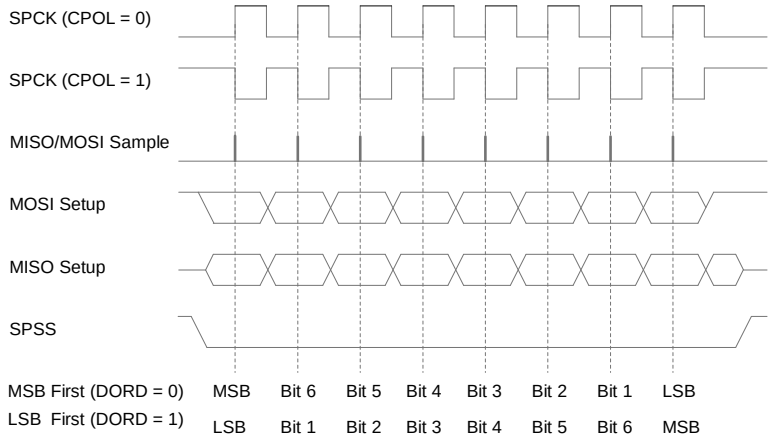
Data schema

In single-wire mode, relative to serial data, SPI has 4 combinations of SPCK phase and polarity, which are controlled by CPHA and CPOL, as shown in the following table.

CPHA and CPOL select data transmission mode

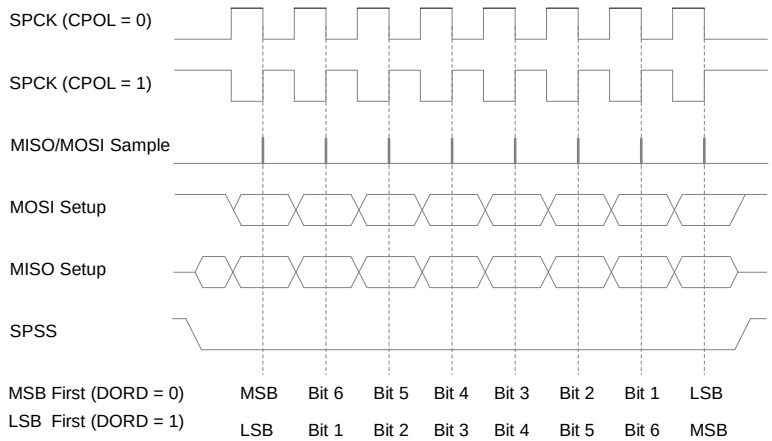
CPOL	CPHA	Start edge	End edge	SPI mode
0	0	Sampling (rising edge)	Set (falling edge)	0
0	1	Set (rising edge)	Sampling (falling edge)	1
1	0	Sampling (falling edge)	Set (rising edge)	2
1	1	Set (falling edge)	Sampling (rising edge)	3

When CPHA = 0, the clock edge of data sampling and setting is shown in the figure below:



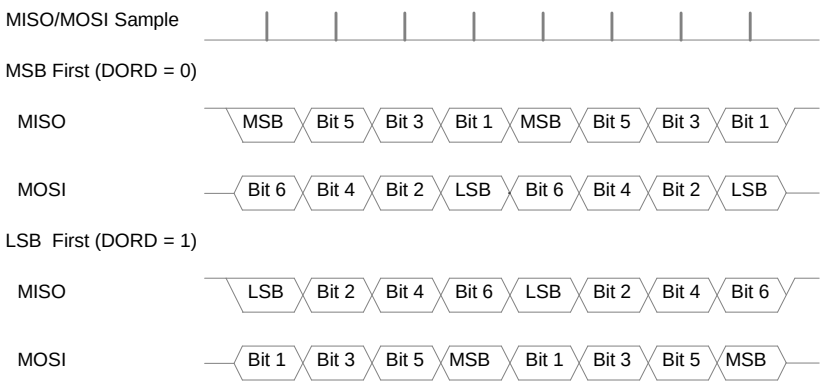
SPI data transmission mode when CPHA is "0"

When CPHA = 1, the clock edge of data sampling and setting is shown in the figure below:



SPI data transmission mode when CPHA is "1"

In the two-wire mode, both MISO and MISO are used as the input of the Master, and the time of data sampling is still determined by the data transmission mode. The sampling method is shown in the figure below:



SPI data sampling mode when DUAL is "1" in Master mode

SPSS pin function

When configured as a slave, the slave select signal SPSS pin is always used as an input. When the SPSS pin is kept low, the SPI interface is activated, the MISO pin becomes an output pin (the software configures the corresponding port), and other pins are inputs. When the SPSS pin is held high, the SPI module is reset and no longer receives data. The SPSS pin is useful for packet/byte synchronization to synchronize the slave's bit counter with the master's clock generator. When SPSS is pulled high, the SPI slave immediately resets the receive and transmit logic, and discards incomplete data in the shift register.

When configured as a master, user software can determine the direction of the SPSS pins.

If SPSS is configured as an output, it can be used to drive the SPSS pin of the slave. If SPSS is configured as an input, it must be kept high to ensure the normal operation of the Master. When configured as a master and the SPSS pin is an input, when the external circuit pulls down the SPSS pin, the SPI module will think that another master chooses itself as a slave and starts to transmit data. In order to prevent bus conflicts, the SPI module will perform the following actions:

1. Clear the MSTR bit in the SPCR register to convert to a slave, so that MOSI and SPCK become inputs;
2. Set the SPIF bit in the SPSR register to generate an SPI interrupt if the interrupt is enabled.

Therefore, when the data transmission of the SPI master is processed in an interrupt mode, and there is a possibility that the SPSS is pulled low, the interrupt service routine should check whether the MSTR bit is "1". If cleared, software must set it to re-enable SPI master mode.

SPI initialization

Before communicating, the SPI must be initialized first. The initialization process usually includes the selection of the Master and slave operation, the setting of the data transmission mode, the selection of the bit rate, and the direction control of each pin. Among them, the control of the pin direction under the operation of the Master and the slave is different, as shown in the following table:

Pin Direction Control

Pins	Orientation in Master mode	Orientation in slave mode
MOSI	User software defined	input
MISO	input	User software defined
SPCK	User software defined	input
SPSS	User software defined	input

SPI Master initialization

The initialization process of SPI master mode is as follows:

1. Set the MSTR bit, set the bit rate selection control bit, data transmission mode, data transmission sequence, interrupt enable or not, and dual-wire enable or not;
2. Set the MOSI and SPCK pins as outputs;
3. Set the SPE bit.

In master mode, when you do not want the SPI module to be selected as a slave by other masters, you can set the SPSS pin as an output.

SPI slave initialization

The SPI slave mode initialization process is as follows:

1. Clear the MSTR bit, set the data transmission mode, data transmission sequence, and enable or disable interrupts;
2. Set MISO pin as output;
3. Set the SPE bit.

SPI interrupt

When one or more of the following events occur, the SPI interrupt flag bit SPIF will be set:

1. When configured as a Master and the SPSS pin is an input, the external circuit pulls down the SPSS pin;
2. When the send buffer status is full, the software continues to write data to the SPDR register;
3. When the receive buffer status is full;
4. When the data written in the sending buffer has been sent out, the sending buffer status is empty.

When the SPIF bit is set, and the SPI interrupt enable bit SPIE and the global interrupt enable bit are both high, an SPI interrupt will be generated. After entering the interrupt service routine, the hardware will clear the SPIF. If the SPIF bit is set by 1 and 2 in the above events, SPIF will be cleared; If the SPIF bit is set by 3 and 4 in the above events, SPIF will not be cleared, because the SPIF bit will still be set when the status of the receive or transmit buffer has not changed, and software operation is required at this time to clear it.

In the SPI interrupt service routine, the operation sequence of software clearing the SPIF bit is as follows:

1) Read the state of the SPIF bit, if it is low, it indicates that the SPIF bit has been cleared by hardware, and there is no need to clear it again by software; if it is high, continue the operation;

2) Read the SPFR register, if the RDFULL bit is high, it indicates that the current receive buffer status is full, read the SPDR register to get the received data, the RDFULL bit will become low, and the software can continue to read the SPDR register to get the received data until the RDEMPT bit is high;

3) Read the SPFR register, if the RDFULL bit is low and the WREMPT bit is high, it indicates that the current receiving buffer status is not full, while the sending buffer status is empty, the software can read the SPDR register to obtain the received data until the RDEMPT bit is high ;

4) After the software obtains the received data, it executes to clear the SPIF bit. Because the SPIF bit is a read-only bit, the SPIF bit cannot be cleared directly. Instead, it is necessary to read the SPSR register first, and then access the SPDR (read or write the SPDR register) to clear the SPIF bit.

Registers definition

List of SPI registers

Register	Address	Default value	Description
SPCR	0x4C	0x00	SPI control register
SPSR	0x4D	0x00	SPI status register
SPDR	0x4E	0x00	SPI data register
SDFR	0x39	0x00	SPI buffer register

SPCR – SPI control register

SPCR – SPI control register								
Address: 0x4C				Default value: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	SPIE	SPI interrupt enable bit. When setting the SPIE bit to "1", the SPI interrupt is enabled. An SPI interrupt is generated when the SPIF bit in the SPSR register is set and the global interrupt is enabled. When setting the SPIE bit to "0", the SPI interrupt is disabled.
6	SPE	SPI enable bit. When setting the SPE bit to "1", the SPI module is enabled. SPE must be set before any SPI operation. When setting the SPE bit to "0", the SPI module is disabled.
5	DORD	Data sequence control bit. When setting the DORD bit to "1", the LSB of the data is transmitted first. When setting the DORD bit to "0", the MSB of the data is transmitted first.
4	MSTR	Master slave select control bit. When setting the MSTR bit to "1", the Master mode is selected. When setting the MSTR bit to "0", the slave mode is selected. In Master mode, when the SPSS pin is configured as an input and pulled low, the MSTR bit will be cleared, and the SPIF in the SPSR register will be set, and the user must reset MSTR to enter the Master mode.
3	CPOL	Clock polarity control bit. When setting the CPOL bit to "1", SPCK is high in idle state. When setting the CPOL bit to "0", SPCK is low in idle state.

CPOL	Start edge	End edge
0	Rising edge	Falling edge
1	Falling edge	Rising edge

Bit	Name	Description
2	CPHA	Clock phase control bit. When the CPHA bit is set to "1", the data is set along the start edge, and the data is sampled along the end edge. When the CPHA bit is set to "0", the data is sampled along the start edge, and the data is set along the end edge.

CPHA	Start edge	End edge

		0	Sampling	Setting
		1	Setting	Sampling
1	SPR1	Clock rate select bit 1. SPR1 and SPR0 are used to select the clock rate for SPI transfer. For specific control methods, see the relationship table between SPCK and system clock.		
0	SPR0	Clock rate select bit 0. SPR1 and SPR0 are used to select the clock rate for SPI transfer. For specific control methods, see the relationship table between SPCK and system clock.		

SPSR – SPI Status Register

SPSR – SPI status register								
Address: 0x4D					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	-	-	-	DUAL	-	SPI2X
R/W	R	R	R	R	R	R/W	R	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	Description
7	SPIF	SPI interrupt flag bit. Set the SPIF flag after the serial transmission is completed. In the Master mode, when the SPSS pin is configured as an input and pulled low, SPIF will also be set. If the SPIE bit and the global interrupt enable bit of the SPCR register are both set at this time, an SPI interrupt is generated. The SPIF bit is cleared automatically after entering the interrupt service routine, or by reading the SPSR register and then accessing the SPDR register to clear the SPIF bit.
6	WCOL	Write conflict flag. Writing to the SPDR register during data transfer will set the WCOL bit. The WCOL bit can be cleared by first reading the SPSR register and then accessing the SPDR register.
5	-	Reserved.
4	-	Reserved.
3	-	Reserved.
2	DUAL	Two-wire mode control bit. When the DUAL bit is set to "1", the SPI dual-wire transmission mode is enabled. When the DUAL bit is set to "0", the SPI dual-wire transmission mode is prohibited. The two-wire transmission mode is only valid in the SPI master mode. Both MISO and MOSI are used as data input for the master. For the data transmission method, see the description in the master two-wire reception and data mode chapters.
1	-	Reserved.
0	SPI2X	SPI double speed control bit. When setting the SPI2X bit to "1", the transmission speed of SPI is doubled. When setting the SPI2X bit to "0", the transmission speed of SPI is not doubled. For specific control methods, see the relationship table between SPCK and system clock.

The following table shows the relationship between SPCK and system clock.

Relationship between SPCK and system clock

SPI2X	SPR1	SPR0	SPCK Frequency
0	0	0	$f_{sys}/4$
0	0	1	$f_{sys}/16$
0	1	0	$f_{sys}/64$
0	1	1	$f_{sys}/128$
1	0	0	$f_{sys}/2$
1	0	1	$f_{sys}/8$
1	1	0	$f_{sys}/32$
1	1	1	$f_{sys}/64$

SPDR – SPI Data Register

SPDR – SPI Data Register								
Address: 0x4E					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	SPDR7	SPDR6	SPDR5	SPDR4	SPDR3	SPDR2	SPDR1	SPDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	SPDR	Data sent and received by SPI. SPI transmit data and receive data share the SPI data register SPDR. Writing data to SPDR means writing to the sending data shift register, and reading data from SPDR means reading the receiving data buffer.

SPFR – SPI buffer register

SPFR – SPI buffer register								
Address: 0x39					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RDFULI	RDEMPT	RDPTR1	RDPTR0	WRFULL	WREMPT	WRPTR1	WRPTR0
R/W	R	R/W	R	R	R	R/W	R	R

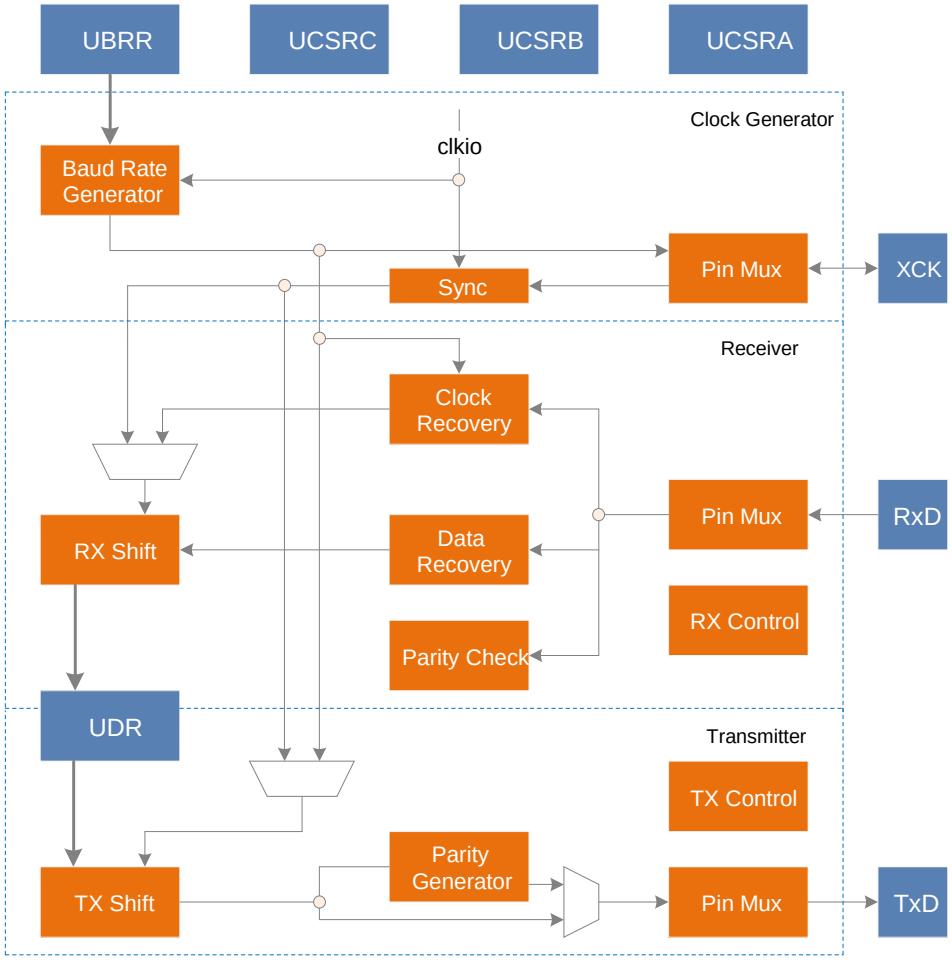
Bit	Name	description
7	RDFULL	Receive buffer full flag. When the data in the receiving buffer reaches four bytes, the RDFULL bit is high, indicating that the receiving buffer is full, and the interrupt flag will be set at the same time. If the software does not read the data in the receiving buffer in time, when the data is received again, the receiving buffer overflows, and the previous data will be overwritten by new data. When the data in the receiving buffer is less than four bytes, the RDFULL bit is low, indicating that the receiving buffer is not full and data can also be received. When setting the RDEMPT bit and the WREMPT bit at the same time, the receiving and sending buffer addresses and the SPI shift register pointer will be reset to zero, and the RDFULL bit will be low.
6	RDEMPT	Receive buffer empty flag. When no data is received, the RDEMPT bit is high, indicating that the receive buffer is empty. When data is received, it will be stored in the receive buffer, and the RDEMPT bit is low, indicating that the receive buffer is not empty. At this

		time, the MCU can read the data in the receive buffer by accessing the SPDR register. In order to ensure that the received data will not be lost, the software can read the data in the receive buffer when the receive buffer is not empty, that is, the RDEMPT bit is low. When the RDEMPT bit is set (write 1), the receive buffer address will be reset to zero. When setting the RDEMPT bit and the WREMPT bit at the same time, the receiving and sending buffer addresses and the SPI shift register pointer will be reset to zero, and the RDEMPT bit is high.
5	RDPTR1	Receive buffer address high.
4	RDPTR0	Receive buffer address low bits. When the SPDR register is read, the MCU will read the received data from the receive buffer, and the address of the receive buffer will be accumulated. When the RDEMPT bit is set (write 1), the receive buffer address will be reset to zero.
3	WRFULL	Transmit buffer full flag. When the data in the sending buffer reaches four bytes, the WRFULL bit is high, indicating that the sending buffer is full. When the data in the sending buffer is less than four bytes, the WRFULL bit is low, indicating that the sending buffer is not full. If you want to increase the transmission speed, the software can write data when the send buffer is not full, that is, when the WRFULL bit is low, and the SPI controller will send the data out in turn.
2	WREMPT	Transmit buffer empty flag. When the data written into the sending buffer has been sent, the WREMPT bit is high, indicating that the sending buffer is empty, and the interrupt flag bit SPIF will be set at the same time. When the SPDR register is written, the sending buffer address will be accumulated. When the data written in the sending buffer is not all sent, there is at least one byte of data in the receiving buffer, and the WREMPT bit is low, indicating that the sending buffer non empty. When the WREMPT bit is set (write 1), the transmit buffer address will be reset to zero. When setting the RDEMPT bit and the WREMPT bit at the same time, the receiving and sending buffer addresses and the SPI shift register pointer will be reset to zero, and the WREMPT bit is high.
1	WRPTR1	Transmit buffer address high bit.
0	WRPTR0	Low bit of transmit buffer address. When the SPDR register is written, the data in the SPDR will be written into the transmit buffer, and the transmit buffer address will be accumulated. When the WREMPT bit is set (write 1), the transmit buffer address will be reset to zero.

USART0 - Universal Synchronous/Asynchronous Serial Transceiver

- ❑ Full-duplex operation (separate serial receive and transmit registers)
- ❑ asynchronous or synchronous operation
- ❑ Master or slave operation
- ❑ High precision baud rate generator
- ❑ Supports 5, 6, 7, 8, or 9 data bits and 1, or 2 stop bits
- ❑ Parity generation and check mechanism supported by hardware
- ❑ Data overspeed detection
- ❑ frame error detection
- ❑ Noise filtering, including false start bit detection and digital low-pass filter
- ❑ Three independent interrupts: Transmit End Interrupt, Transmit Data Register Empty Interrupt, and Receive End Interrupt
- ❑ Multiprocessor communication mode
- ❑ Double-speed asynchronous communication mode

Overview



USART structure diagram

USART mainly consists of three parts: clock generator, transmitter and receiver. Control and status registers are shared by all three parts. The clock generator consists of a baud rate generator and synchronization logic for an external input clock in synchronous slave mode of operation.
The XCK pin is only used in synchronous transfer mode. The transmitter includes a write data buffer, serial shift register, parity generator, and control logic required to handle different frame formats. The write data buffer allows data to be sent continuously without introducing delay between data frames. The receiver has a clock and data recovery unit for asynchronous data reception. In addition to the recovery unit, the receiver includes parity, control logic, serial shift registers and a two-stage receive buffer UDR. The receiver supports the same frame formats as the transmitter, and can detect framing errors, data overruns, and parity errors.

Clock generation

The clock generation logic generates the base clock for the transmitter and receiver. The USART supports 4 clock modes: normal asynchronous mode, double-speed asynchronous mode, master synchronous mode, and slave synchronous mode. The UMSEL bit of the USCRC is used to select synchronous or asynchronous mode. The U2X bit of USCRA controls the double speed enable in asynchronous mode. The data direction register (multiplexed with IO) of the XCK pin, which is only valid in synchronous mode, determines whether the clock source is internally generated (master mode) or externally generated (slave mode).

Baud rate generator

The baud rate register UBRR and the down-counter are connected together as a programmable prescaler or baud rate generator for the USART. The down counter works under the system clock (fsys), and when it counts to zero or the UBRRL register is written, it will automatically load the value of the UBRR register. When the count reaches zero, a clock is generated, which is used as the output clock of the baud rate generator, and the frequency is fsys/(UBRR+1).

The following table gives the formulas for calculating the baud rate (bit/s) and UBRR value in various working modes.

Operating mode	Baud rate calculation formula (1)	UBRR value calculation formula
Asynchronous normal mode	$BAUD = f_{sys}/(16*(UBRR+1))$	$UBRR = f_{sys}/(16*BAUD) - 1$
Asynchronous speed mode	$BAUD = f_{sys}/(8*(UBRR+1))$	$UBRR = f_{sys}/(8*BAUD) - 1$
Synchronize master mode	$BAUD = f_{sys}/(2*(UBRR+1))$	$UBRR = f_{sys}/(2*BAUD) - 1$

Explanation:
1. The baud rate is defined as the bit transmission speed per second (bps);
2. BAUD is the baud rate, fsys is the system clock, and UBRR is the combined value of the baud rate registers UBRRH and UBRRL.

Double speed operation mode

By setting the U2X bit of the UCSRA register, the transmission rate can be doubled. This bit is only valid in the asynchronous working mode, and it is set to "0" in the synchronous working mode.
Setting this bit will halve the baud rate divider value, effectively doubling the transfer rate for asynchronous communication. In this case, the receiver uses only half the number of samples to sample the data and recover the clock, so a more precise baud rate setting and system clock are required. The transmitter is unchanged.

External clock

The synchronous slave mode of operation is driven by an external clock. The external clock is used by the transmitter and receiver after passing through the synchronization register and the edge detector.

This process will introduce a delay of two system clocks, so the maximum clock frequency of the external XCK is limited by the following formula:

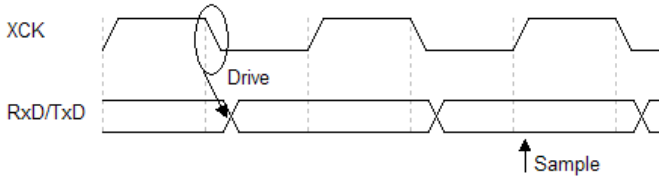
fXCK < fsys/4

It should be noted that fsys is determined by the stability of the system clock. In order to prevent data loss due to frequency drift, it is recommended to reserve enough margin.

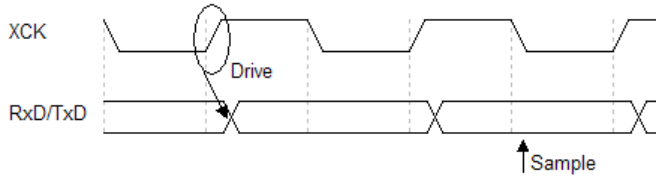
Synchronous Clock Operation

In synchronous mode, the XCK pin is used as clock input (slave mode) or clock output (master mode). The basic law of the relationship between the edge of the clock and the data sampling and data change is: the clock edge used for sampling the data input terminal (RxD) is opposite to the clock edge used for the change of the data output terminal.

UCPOL = 1



UCPOL = 0



XCK Timing in Synchronous Mode

As shown in the figure above, when the value of UCPOL is "1", the data output is changed on the falling edge of XCK, and data sampling is performed on the rising edge of XCK; when the value of UCPOL is "0", the data output is changed on the rising edge of XCK, Data is sampled on the falling edge of XCK.

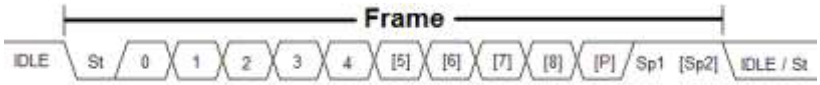
Frame format

A serial data frame consists of data words plus synchronization bits (start and stop bits) and parity bits for error correction.

The USART accepts the following 30 combinations of data frame formats:

- ❑ 1 start bit
- ❑ 5, 6, 7, 8 or 9 data bits
- ❑ No parity, odd parity, or even parity
- ❑ 1 or 2 stop bits

The data frame starts with the start bit, followed by the lowest bit of the data word, followed by other data bits, and ends with the highest bit of the data word, up to 9 bits of data can be successfully transmitted. If parity is enabled, the parity bit will follow the data word, followed by the stop bit. When a complete data frame is transmitted, the next new data frame can be transmitted immediately, or the transmission line can be idle (high level). The following figure shows a possible data frame structure, the bits in square brackets are optional.



USART frame structure diagram

Description:

1) IDLE There is no data transmission on the communication line (RxD or TxD), and the line must be high when the line is idle
2) St start bit, always low
3) 0-8 data bits
4) P parity bit, odd or even
5) Sp stop bit, always high

The structure of the data frame is set by UCSZ[2:0], UPM[1:0] and USBS in the UCSRB and UCSRC registers. Receiving uses the same settings as sending. Any changes to settings may disrupt ongoing data transfers. Among them, UCSZ[2:0] determines the number of data bits in the data frame, UPM[1:0] is used to enable and determine the type of checksum, and the USBS setting frame has one or two end bits. The receiver ignores the second stop bit, so framing errors are only detected when the first stop bit is "0".

Check digit calculation

The calculation of the parity bit is to perform an XOR operation on each bit of the data. If odd parity is selected, the XOR result also needs to be reversed.
The relationship between the parity bit and the data bit is as follows:

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$
$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

Description：

1）P_{even} even parity result
2）P_{odd} odd parity result
3）d_n nth data bit

USART initialization

Before communicating, the USART must be initialized first. The initialization process usually includes setting the baud rate, setting the frame structure, and enabling the receiver or transmitter as needed. For interrupt-driven USART operations, the global interrupt flag must be cleared and all USART interrupts disabled during initialization.
When performing reinitialization such as changing the baud rate or frame structure, it must be ensured that no data is transferred. The TXC flag bit can be used to detect whether the transmitter has completed all transmissions, and the RXC flag bit can be used to detect whether there is data in the receive buffer that has not been read. If the TXC flag is used for this purpose, the TXC flag must be cleared before each data transmission (before writing the UDR register).

Transmitter

Setting the TXEN bit in the UCSRB register will enable data transmission from the USART. After enabling, the general-purpose IO function of the TxD pin is replaced by the USART function and becomes the serial output of the transmitter. Before sending data, set the baud rate, working mode and frame format. If the synchronous transmission mode is used, the clock signal applied to the XCK pin is the clock for data transmission.

Send 5 to 8 frames of data

Load the data to be sent into the send buffer to start data transmit. The CPU loads data by writing to the UDR register. When the sending shift register can send a new frame of data, the data in the buffer will be transferred to the shift register.
When the shift register is idle (no data transfer in progress), or the last stop bit of the previous frame of data has been sent, it will be loaded with new data. Once the shift register is loaded with new data, it will transmit a full frame as programmed.

Send a frame of 9-bit data

If a frame of 9-bit data is sent, the 9th bit of the data should be written into the TXB8 bit of the register UCSRB first, and then the lower 8-bit data should be written into the sending data register UDR. The ninth bit of data is used to represent the address frame in multi-computer communication, and can be used for protocol processing in synchronous communication.

Send parity bit

The parity generation circuit generates corresponding parity bits for the serial data frame. When the parity bit is enabled (UPM1 = 1), the transmit control logic inserts a parity bit between the last bit and the first stop bit of the data word.

Send flag and interrupt processing

The USART transmitter has two flag bits: the USART data register empty flag UDRE and the transmission end flag TXC, both of which can generate interrupts.
The empty flag UDRE of the data register is used to indicate whether a new data can be written into the sending buffer. This bit is set to "1" when the transmit buffer is empty and to "0" when it is full. When the UDRE bit is "1", the CPU can write new data to the data register UDR, otherwise it cannot.
When the data register empty interrupt enable bit UDRIE in the UCSRB register is "1", as long as UDRE is set (and the global interrupt is enabled), a USART data register empty interrupt request will be generated. Writing to register UDR will clear UDRE.
When using the interrupt method to transmit data, in the data register empty interrupt service program, a new data must be written to UDR to clear UDRE, or the data register empty interrupt must be disabled. Otherwise, once the interrupt service routine is finished, a new interrupt will be generated again.
When the entire data frame is shifted out of the sending shift register and there is no new data in the sending register at the same time, the sending end flag TXC will be set. When the transmission end interrupt enable bit TXCIE (and global interrupt enable) on UCSRB is set to "1", the USART transmission end interrupt will be executed as the TXC flag bit is set. Once entering the interrupt service routine, the TXC flag bit is automatically cleared, and the CPU can also write "1" to this bit to clear it.

Disable transmitter

When TXEN is cleared, the transmitter can only be disabled after all the data is sent, that is, there is no data to be transmitted in the transmission shift register and the transmission buffer register. After the transmitter is disabled, the TxD pin resumes its general-purpose IO function.

Receiver

The USART receiver can be started by setting the receive enable bit (RXEN) of the UCSRB register. After enabling, the general-purpose IO function of the RxD pin is replaced by the USART function and becomes the serial input port of the receiver. Before data reception, the baud rate, operation mode and frame format must be set. If synchronous receive mode is used, the clock on the XCK pin is used as the transmit clock.

Receive frames of 5 to 8 bits of data

Once the receiver detects a valid start bit, it begins to accept data. Each bit of data after the start bit will be received at the set baud rate or XCK clock until the first stop bit of a frame of data is received, and the second stop bit will be ignored by the receiver. Each bit of data received is sent to the

receiving shift register. After receiving the first stop bit, the receiver sets the received data completion flag RXC bit in the UCSRA register and transfers the complete data frame in the shift register. In the receive buffer, the CPU can obtain the received data by reading the UDR register.

Receive a frame of 9-bit data

If the data frame of 9-bit data is set, before reading the lower 8-bit data from UDR, the RXB8 bit of the register UCSRB must be read first to obtain the 9th-bit data. This rule also applies to status flags FE, DOR, and PE. Reading the UDR location changes the state of the receive buffer, which in turn changes the TXB8, FE, DOR, and PE bits also stored in the buffer.

Receive end flag and interrupt processing

The USART receiver has a flag: the receiving end flag RXC, which is used to indicate whether there is unread data in the receiving buffer. When there is unread data in the receiving buffer, this bit is "1", otherwise it is "0". If the receiver is disabled, the receive buffer will be flushed and RXC will be cleared.
After setting the receive end interrupt enable bit RXCIE of UCSRB, as long as the RXC flag is set (and the global interrupt is enabled), a USART receive end interrupt will be generated. When using the interrupt method to receive data, the interrupt service program must read data from UDR to clear the RXC flag after the data reception is completed, otherwise, a new interrupt will be generated as soon as the interrupt handler ends.

Receive error flag

The USART receiver has three error flags: frame error FE, data overflow DOR and parity error PE. They are both located in the UCSRA register. Error flags are saved together with the data frame in the receive buffer. All error flags cannot generate interrupts.
The frame error flag FE indicates the status of the first stop bit of the next readable frame stored in the receive buffer. If the stop bit is correct (the value is "1"), the FE flag is "0", otherwise the FE flag is "1". This flag can be used to detect loss of synchronization, interruption of transmission, and also for protocol processing.
The data overflow flag DOR indicates that data is lost due to the receive buffer being full. When the receive buffer is full and there is already data in the receive shift register, if a new start bit is detected at this time, data overflow occurs. The DOR flag is set to indicate that one or more data frames were lost between the last UDR read and the next UDR read. When the data frame is successfully transferred from the shift register to the receive buffer, the DOR flag is cleared.
The parity error flag PE indicates that the next frame of data in the receiving buffer has a parity error when it is received. If parity is not enabled, PE is cleared.

Parity checker

Setting the parity mode bit UPM1 will enable the parity checker. The parity mode (even parity or odd parity) is determined by UPM0. When parity checking is enabled, the checker will calculate the parity of the input data and compare the result with the parity bits of the data frame.
The result of the checksum will be stored in the receive buffer along with the data and stop bits. The CPU checks the received frame for parity errors by reading the PE bit. If the next data read from the receive buffer has a parity error and the parity
If yes, then UPE is set and remains valid until the receive buffer UDR is read.

Disable receiver

Inhibiting the receiver is immediate compared to the transmitter. Data being received will be lost. After the receiver is disabled (RXEN is cleared), the receiver will no longer occupy the RxD pin and the receive buffer will be flushed.

Asynchronous data reception

The USART has a clock recovery unit and data recovery unit to handle asynchronous data reception. The clock recovery logic is used to synchronize the asynchronous serial data input from the RxD pin with the internal baud rate clock. Data recovery logic is used to collect data and filter every bit of incoming data through a low-pass filter to improve the receiver's anti-jamming performance. The working range of asynchronous reception depends on the accuracy of the internal baud rate clock, the frame input rate and the number of data bits contained in a frame.

Asynchronous operation scope

The operating range of the receiver depends on the degree of mismatch between the received data rate and the internal baud rate. If the transmitter transmits data at a bit rate that is too fast or too slow, or if the baud rate generated internally by the receiver does not have the same frequency, then the receiver cannot synchronize to the start bit. In order to ensure that the receiver will not miss the sampling of the start bit of the next frame, the data input rate and the internal receiver baud rate cannot be too different, and the ratio between them is used to describe the error range of the baud rate. The following two tables respectively give the allowable maximum baud rate error range in normal mode and double speed mode.

Maximum receiver baud rate error range in normal mode

Data bits + parity bit length	Max error margin（%）	Recommended error margin （%）
5	+6.7/-6.8	±3.0
6	+5.8/-5.9	±2.5
7	+5.1/-5.2	±2.0
8	+4.6/-4.5	±3.0
9	+4.1/-4.2	±1.5
10	+3.8/-3.8	±1.5

Maximum Receiver Baud Rate Error Range in Double Speed Mode

Data bits + parity bit length	Max error margin（%）	Recommended error margin （%）
5	+5.7/-5.9	±2.5
6	+4.9/-5.1	±2.0
7	+4.4/-4.5	±1.5
8	+3.9/-4.0	±1.5
9	+3.5/-3.6	±1.0
10	+3.2/-3.3	±1.0

It can be seen from the table that the baud rate allows a larger range of variation in normal mode. The recommended baud rate error ranges above are based on the assumption that the receiver and transmitter contribute equally to the maximum total error. There are two possible causes of a receiver baud rate error. First, the stability of the receiver system clock is related to operating voltage and temperature. Generally, there is no such problem when the crystal oscillator is used to generate the system clock, but when the internal oscillator is used, the system clock may be skewed. The second reason is that the baud rate generator may not be able to obtain exactly the desired baud rate by dividing the frequency of the system clock. At this time, the value of UBRR can be adjusted so that the error is as low as acceptable.

Baud rate setting and introducing error

For the standard crystal oscillator and resonator frequency, the actual communication baud rate in asynchronous mode can be obtained through the baud rate calculation formula, and the error between it and the common communication baud rate can be calculated by the following formula:

$$Error[\%] = (Baudreal/Baud - 1)*100\%$$

Where Baud is the commonly used communication baud rate, Baudreal is the baud rate calculated by the calculation formula, and the baud rate calculation formula can be used to obtain the difference between the baud rate error and the system clock fsys and the baud rate register UBRR value The relationship is as follows:

Normal mode:

$$Error[\%] = (fsys/(16*(UBRR+1))/Baud - 1)*100\%$$

Double speed mode:

$$Error[\%] = (fsys/(8*(UBRR+1))/Baud - 1)*100\%$$

When the clock error on both sides of the communication is not considered, that is, when the system clock fsys is a standard clock, the relationship between the baud rate error UBRR values can be obtained. The following table is the baud rate error under different UBRR value settings under the 16MHz system clock.

The error caused by setting the UBRR value under the 16MHz system clock

baud rate (bps)	f_{sys} = 16.000MHz			
	Normal mode (U2X = 0)		double speed mode(U2X = 1)	
	UBRR	error	UBRR	error
2400	416	-0.1%	832	0.0%
4800	207	0.2%	416	-0.1%
9600	103	0.2%	207	0.2%
14.4K	68	0.6%	138	-0.1%
19.2K	51	0.2%	103	0.2%
28.8K	34	-0.8%	68	0.6%
38.4K	25	2.1%	34	-0.8%
57.6K	16	0.2%	51	0.2%
76.8K	12	0.2%	25	0.2%
115.2K	8	-3.5%	16	2.1%
230.4K	3	8.5%	8	-3.5%
250K	3	0%	7	0%
0.5M	1	0%	3	0%
1M	0	0%	1	0%

Multiprocessor communication mode

Data frames received by the USART receiver can be filtered by setting the Multiprocessor Communication Mode (MPCM) bit in UCSRA.

Frames without address information will be ignored and will not be stored in the receive buffer. In a multiprocessor system, where the processors communicate over the same serial bus, this filtering effectively reduces the number of data frames that need to be processed by the CPU.

The setting of the MPCM bit does not affect the work of the transmitter, but in a multiprocessor communication system, its usage method will be different.

If the data frame length received by the receiver is 5 to 8 bits, then the first stop bit will be used to indicate whether the current frame contains data or address information. If the length of the data frame received by the receiver is 9 bits, then the 9th bit determines whether it is data or address information. If the frame type flag is "1", then this is an address frame, otherwise it is a data frame.

In multiprocessor communication mode, multiple slave processors are allowed to receive data from one master processor. The first step is to determine which slave processor is being addressed by decoding the address frame. The addressed slave processor will normally receive subsequent data frames, while other slave processors will ignore these data frames until the next address frame is received.

For a processor as a master, it can use 9-bit data frame format, and use the 9th bit of data to identify the frame format. In this communication mode, the slave processor must also work in 9-bit data frame format. The following are the steps for data exchange in the multiprocessor communication mode:

1. All slave processors work in multiprocessor communication mode (set MPCM);
2. The master processor sends an address frame, and all slave processors receive this frame. The RXC bit of the slave processor UCSRA register is normally set;
3. Each slave processor reads the contents of the UDR register and decodes the address frame to determine if it is selected. If selected, clear the MPCM bit of the UCSRA register, if not selected, keep MPCM as "1" and wait for the arrival of the next address frame;
4. The addressed slave processor receives all data frames until a new address frame is received. Slave processors that are not being addressed ignore these data frames;
5. After the addressed slave processor receives the last data frame, it sets the MPCM bit and waits for the arrival of the next address frame. Then repeat from step two.

A frame format using 5 to 8 bits of data is possible, but impractical because the receiver must switch between using n and n+1 frame formats. This setup makes full-duplex operation difficult since the receiver and transmitter use the same character length setting. If a frame format of 5 to 8 bits of data is used, the transmitter should set two stop bits, where the first stop bit is used to determine the frame type.

Registers definition

UCSRA – USART Control and Status Register A

UCSRA – USART Control and Status Register A								
Address: 0xC0					Default value: 0x20			
Bit	7	6	5	4	3	2	1	0
Name	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPME
R/W	R	R/W	R	R	R	R	R/W	R/W

Bit	Name	Description
7	RXC	Receive end flag. When the value of RXC is "1", it indicates that there is unread data in the receive buffer. When the value of RXC is "0", it indicates that there is no unread data in the receive buffer. When the receiver is disabled, the receive buffer is flushed, causing RXC to be cleared. When the receiving end interrupt enable bit RXCIE is "1", RXC can be used to generate the receiving end interrupt.
6	TXC	Send end flag. The data in the transmit shift register is sent out and TXC is set when the transmit buffer is empty. TXC is automatically cleared when the transmission end interrupt is executed, and it can also be cleared by writing "1" to TXC. When the transmission end interrupt enable bit TXCIE is "1", TXC can be used to generate a transmission end interrupt.

5	UDRE	Data register empty flag. When UDRE is "1", it indicates that the USART transmit data buffer is empty and data can be written. When UDRE is "0", it indicates that the USART send data buffer is full and cannot write data. When the data register empty interrupt enable bit UDRIE is "1", UDRE can be used to generate a data register empty interrupt.
4	FE	Framing error flag. When FE is "1", it indicates that the data received by the receiving data buffer has a frame error, that is, the first stop bit is "0". When FE is "0", it indicates that the data received by the receiving data buffer has no frame error, that is, the first stop bit is "1". After FE is set, it will be valid until UDR is read. When writing to UCSRA, the FE bit should be written as "0".
3	DOR	Data overflow flag. When the receive buffer is full (contains two data), there is data in the receive shift register, if a new start bit is detected at this time, data overflow occurs, DOR is set, and is valid until UDR is read . When writing to UCSRA, the DOR bit should be written as "0".
2	PE	Parity error flag. When the parity check is enabled (UPM1 is "1"), and the data frame received in the receive buffer has a parity check error, PE is set and remains valid until UDR is read. When writing to UCSRA, the bit of PE should be written as "0".
1	U2X	Double speed transmission enable bit. When U2X is "1", the transmission rate of the asynchronous communication mode is doubled. When U2X is "0", the transmission rate of the asynchronous communication mode is the normal rate. This bit is valid only in asynchronous mode of operation, clear this bit to 0 when using synchronous mode of operation.
0	MPCM	Multiprocessor communication mode enable bit. Setting the MPCM bit will enable multiprocessor communication mode. When MPCM is set, those input frames received by the USART receiver that do not contain address information will be ignored. The transmitter is not affected by MPCM settings.

UCSRB – USART Control and Status Register B

UCSRB – USART Control and Status Register B								
Address: 0xC1					Default Value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Bit	Name	Description
7	RXCIE	Receive end interrupt enable bit. Set to enable RXC interrupt, cleared to disable RXC interrupt. When RXCIE is "1", the global interrupt is enabled, and the RXC of the UCSRA register is "1", the USART reception end interrupt can be generated.
6	TXCIE	Transmit end interrupt enable bit. Set to enable TXC interrupt, cleared to disable TXC interrupt. When TXCIE is "1", the global interrupt is enabled, and the TXC of the UCSRA register is "1", the USART transmission end interrupt can be generated.
5	UDRIE	Data register empty interrupt enable bit. Set to enable UDRE interrupt, cleared to disable UDRE interrupt. When UDRIE is "1", the global interrupt is enabled, and the USART data register empty interrupt can be generated when the UDRE of the UCSRA register is "1".

4	RXEN	Receive enable bit. Enables the USART receiver when asserted. The general-purpose IO function of the RxD pin is replaced by USART reception. Disabling the receiver flushes the receive buffer and invalidates the FE, DOR, and PE flags.
3	TXEN	Send enable bit. Enables the USART transmitter when asserted. The general purpose IO function of the TxD pin is replaced by the USART transmit. After TXEN is cleared, only after all data transmission is completed can the USART transmission be disabled.
2	UCSZ2	Character length controls bit 2. UCSZ2 is combined with UCSZ1:0 of the UCSRC register to set the number of data bits contained in the data frame.
1	RXB8	The 8th bit of received data. When the data frame length is 9 bits, RXB8 is the highest bit of the received data. Read RXB8 before reading the lower 8-bit data contained in UDR.
0	TXB8	Send the 8th bit of data. When the data frame length is 9 bits, TXB8 is the highest bit of the sent data. Write TXB8 before writing the lower 8-bit data contained in UDR.

UCSRC – USART Control and Status Register C

UCSRC – USART Control and Status Register C								
Address: 0xC2					Default value: 0x06			
Bit	7	6	5	4	3	2	1	0
Name	UMSEL1	UMSEL0	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description	
7:6	UMSEL1:0	USART mode select bit. UMSEL selects synchronous or asynchronous mode of operation.	
		UMSEL	mode
		0	USART asynchronous mode of operation
		1	USART synchronous mode of operation
		2	SPI slave mode of operation
		3	SPI master mode of operation
5:4	UPM1:0	Parity mode selection bits. The high bit UPM1 selects to enable or disable parity check, and the low bit UPM0 selects odd check or even check.	
		UPM1:0	mode
		0	Disable parity
		1	Reserved
		2	Enable even parity
		3	Enable odd parity
3	USBS	Stop bit select bit. Select the number of stop bits.	
		USBS	Stop bits
		0	1

		1	2
2:1	UCSZ1:0	Data frame character length selection bit. UCSZ1:0 is combined with UCSZ2 of the UCSRB register to set the number of data bits contained in the data frame.	
		UCSZ2:0	Data frame length
		0	5 bit
		1	6 bit
		2	7 bit
		3	8 bit
		4	Reserved
		5	Reserved
		6	Reserved
		7	9 bit
0	UCPOL	Clock polarity select bit. In the USART synchronous mode of operation, UCPOL sets the relationship between the change of the output data and the sampling of the input data and the synchronous clock XCK. When using asynchronous working mode, it has nothing to do with UCPOL, clear this bit to 0.	
		UCPOL	send data change / Receive data samples
		0	XCK rising edge / XCK falling edge
		1	XCK falling edge / XCK rising edge

UBRRL – USART Baud Rate Register Low Byte

UBRRL – USART Baud Rate Register Low Byte								
Address: 0xC4					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UBRR7	UBRR6	UBRR5	UBRR4	UBRR3	UBRR2	UBRR1	UBRR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	UBRR[7:0]	Low byte portion of the USART baud rate register. The USART baud rate register contains two parts, UBRRL and UBRRH, which are combined to set the baud rate for communication.

UBRRH – USART Baud Rate Register High Byte

UBRRH – USART Baud Rate Register High Byte								
Address: 0xC5					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	UBRR11	UBRR10	UBRR9	UBRR8
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	Description
7:4	-	reserved.

<table>
<tr><td rowspan="4">3:0</td><td rowspan="4">UBRR[11:8]</td><td colspan="2">High byte portion of the USART baud rate register.
The USART baud rate register contains two parts, UBRRL and UBRRH, which are combined to set the baud rate for communication.

UBRR = {UBRR[11:8], UBRRL}</td></tr>
<tr><td>Operating mode</td><td>Baud rate calculation formula</td></tr>
<tr><td>async normal mode</td><td>BAUD = f_{sys}/(16*(UBRR+1))</td></tr>
<tr><td>Async double speed mode</td><td>BAUD = f_{sys}/(8*(UBRR+1))</td></tr>
<tr><td></td><td></td><td>Synchronous master mode</td><td>BAUD = f_{sys}/(2*(UBRR+1))</td></tr>
</table>

UDR – USART Data Register

UDR – USART Data Register								
Address: 0xC6					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UDR7	UDR6	UDR5	UDR4	UDR3	UDR2	UDR1	UDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

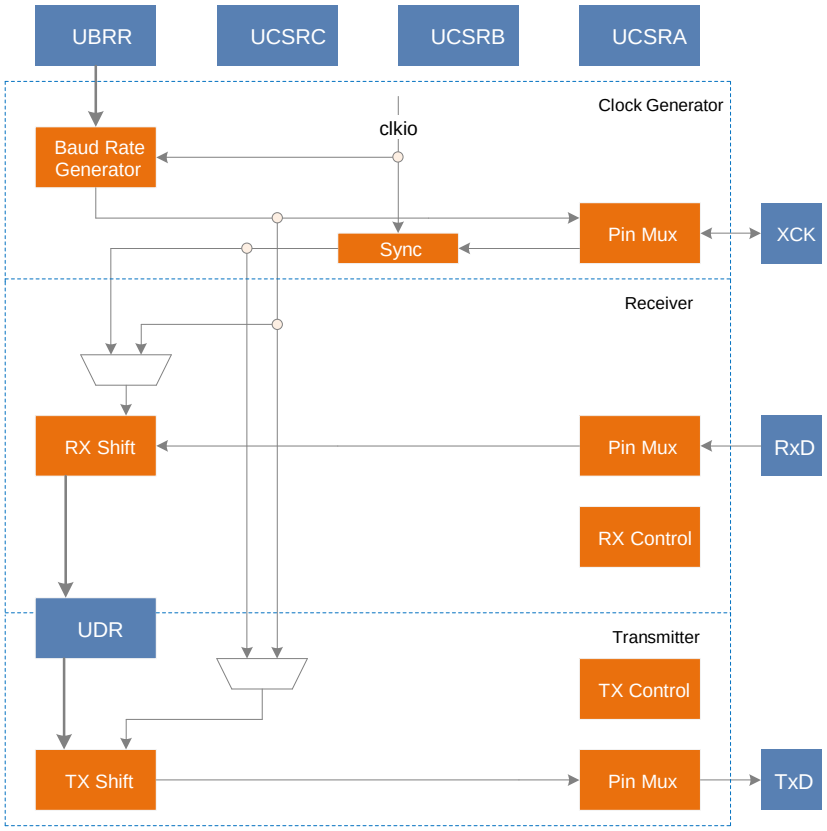
Bit	Name	Description
7:0	UDR	Data sent and received by the USART. The USART transmit data buffer and receive data buffer share the USART data register UDR. Writing data into the UDR means writing to the sending data buffer, and reading data from the UDR means reading the receiving data buffer. In 5 to 8-bit data frame mode, the unused 9th bits are ignored by the transmitter, and the receiver sets them to 0. Only when the UDRE flag of the UCSRA register is "1" can the send buffer be written, otherwise the operation of the transmitter will be wrong. When the transmit shift register is empty, the transmitter will load the data in the transmit buffer into the transmit shift register, and then the data will be serially output from the TxD pin. The receive buffer contains a two-stage FIFO that changes its state once the receive buffer is read.

USART0 - SPI working mode

- ❑ Full-duplex operation, three-wire synchronous data transmission
- ❑ Master or slave operation
- ❑ Supports all four operating modes (Mode 0, 1, 2 and 3)
- ❑ LSB or MSB transmitted first (configurable data transfer order)
- ❑ Queue Operations (Double Buffer)
- ❑ High Resolution Baud Rate Generator

Overview

When the UMSEL1 bit of USCRC is set to "1", the SPI working mode is enabled, represented by USPI. This SPI module is a three-wire SPI working mode. Compared with the four-wire SPI mode, it lacks a slave selection line, and the other three lines are consistent. USPI occupies USART resources, including transmit and receive shift registers and buffers, and a baud rate generator. Parity generation and checking logic, data and clock recovery logic are all disabled. The addresses of the control and status registers are the same, but the functions of the register bits change as required by the SPI operating mode.



USART in SPI Structure diagram

Clock generator

When SPI works in master mode, it needs to provide a clock for communication, and multiplex the baud rate generator of USART to generate this clock. This clock is output from the XCK pin, so the data direction register (DDR_XCK) of the XCK pin must be set to "1".

The clock frequency is determined by the following calculation formula:

BAUD = fsys/(2*(UBRR+1))

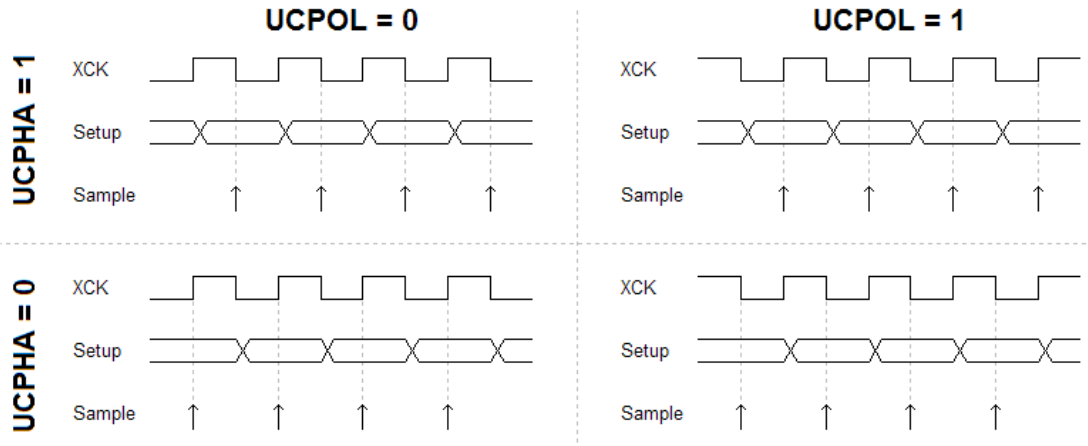
When the SPI works in slave mode, the communication clock is provided by the external master and input from the XCK pin, so the data direction register (DDR_XCK) of the XCK pin must be set to "0".

SPI Data Patterns and Timing

SPI has four combinations of clock phases and polarities, which are determined by the control bits UCPHA and UCPOL. The specific control is shown in the following table and the figure below:

SPI Operating mode

SPI mode	UCPOL	UCPHA	Start edge	End edge
0	0	0	Sample on rising edge	Set on falling edge
1	0	1	Set on rising edge	Sample on falling edge
2	1	0	Sample on falling edge	Set on rising edge
3	1	1	Set on falling edge	Sample on rising edge



SPI Operation mode diagrams

Frame format

A serial frame of SPI can start from the lowest bit or the highest bit, and end with the highest bit or the lowest bit, with a total of 8 bits of data. After the end of one frame, a new frame can be transmitted immediately, and the data line can be pulled high to be in an idle state after the transmission is completed.

Data transmission

The SPI sets the TXEN bit of the UCSRB register to "1" to enable the transmitter, and the TxD pin is occupied by the transmitter to send serial output data. The receiver can be disabled at this time.
The SPI sets the RXEN bit of the UCSRB register to "1" to enable the receiver, and the RxD pin is occupied by the receiver to receive serial input data. The transmitter must be enabled at this time.
Both SPI sending and receiving use XCK as the transmission clock.

Before communicating, the SPI must be initialized first. The initialization process usually includes the setting of the baud rate, the setting of the frame data bit transmission sequence, and enabling the receiver or transmitter as required. For interrupt-driven SPI operations, the global interrupt flag must be cleared and all SPI interrupts disabled during initialization.

When performing reinitialization such as changing the baud rate or frame structure, it must be ensured that no data is transferred. The TXC flag bit can be used to detect whether the transmitter has completed all transmissions, and the RXC flag bit can be used to detect whether there is data in the receive buffer that has not been read. If the TXC flag is used for this purpose, the TXC flag must be cleared before each data transmission (before writing the UDR register).

After initializing the SPI, write data to the UDR register to start data transmission. Since the transmitter controls the transmit clock, both sending and receiving data do so. When the sending shift register is ready to send a new frame of data, the transmitter will move the data written into the UDR register from the sending buffer to the sending shift register and send it out. In order to ensure the synchronization of the input buffer and the sent data, the UDR register must be read once after each byte of data is sent. When data overflow occurs, the most recently received data will be lost, not the oldest received data.

Transmission flags and interrupts

The SPI transmitter has two flags: SPI data register empty flag UDRE and transmission end flag TXC, both flags can generate interrupts.

The empty flag UDRE of the data register is used to indicate whether a new data can be written into the sending buffer. This bit is set to "1" when the transmit buffer is empty and to "0" when it is full. When the UDRE bit is "1", the CPU can write new data to the data register UDR, otherwise it cannot.

When the data register empty interrupt enable bit UDRIE in the UCSRB register is "1", as long as UDRE is set (and the global interrupt is enabled), an SPI data register empty interrupt request will be generated. Writing to register UDR will clear UDRE. When using the interrupt method to transmit data, in the data register empty interrupt service program, a new data must be written to UDR to clear UDRE, or the data register empty interrupt must be disabled. Otherwise, once the interrupt service routine is finished, a new interrupt will be generated again.

When the entire data frame is shifted out of the sending shift register and there is no new data in the sending register at the same time, the sending end flag TXC will be set. When the transmission end interrupt enable bit TXCIE (and the global interrupt enable) on UCSRB is set to "1", the SPI transmission end interrupt will be executed as the TXC flag bit is set. Once entering the interrupt service routine, the TXC flag bit is automatically cleared, and the CPU can also write "1" to this bit to clear it.

TX disable

When TXEN is cleared, the transmitter can only be disabled after all the data is sent, that is, there is no data to be transmitted in the transmission shift register and the transmission buffer register. After the transmitter is disabled, the TxD pin resumes its general-purpose IO function.

Receive end flag and interrupt

The SPI receiver has a flag: the receiving end flag RXC is used to indicate whether there is unread data in the receiving buffer. When there is unread data in the receiving buffer, this bit is "1", otherwise it is "0". If the receiver is disabled, the receive buffer will be flushed and RXC will be cleared. After setting the receive end interrupt enable bit RXCIE of UCSRB, as long as the RXC flag is set (and the global interrupt is enabled), the SPI receive end interrupt will be generated. When using the interrupt method to receive data, the interrupt service program must read data from UDR to clear the RXC flag after data reception is completed, otherwise, a new interrupt will be generated as soon as the interrupt handler ends.

RX disable

Inhibiting the receiver is immediate compared to the transmitter. Data being received will be lost. After the receiver is disabled (RXEN is cleared), the receiver will no longer occupy the RxD pin and the receive buffer will be flushed.

Registers definition

USART List of registers

register	Address	Default value	Description
UCSRA	0xC0	0x20	USPI Control and Status Register A
UCSRB	0xC1	0x00	USPI Control and Status Register B
UCSRC	0xC2	0x06	USPI Control and Status Register C
UBRRL	0xC4	0x0	USPI baud rate register low byte
UBRRH	0xC5	0x0	USPI baud rate register high byte
UDR	0xC6	0x0	USPI data register

UCSRA – USPI Control and Status Register A

UCSRA – USPI Control and Status Register A								
Address: 0xC0					Default value: 0x20			
Bit	7	6	5	4	3	2	1	0
Name	RXC	TXC	UDRE	-	-	-	-	-
R/W	R	R/W	R	-	-	-	-	-

Bit	Name	Description
7	RXC	Receive end flag. When the value of RXC is "1", it indicates that there is unread data in the receive buffer. When the value of RXC is "0", it indicates that there is no unread data in the receive buffer. When the receiver is disabled, the receive buffer is flushed, causing RXC to be cleared. When the receiving end interrupt enable bit RXCIE is "1", RXC can be used to generate the receiving end interrupt.
6	TXC	Send end flag. The data in the transmit shift register is sent out and TXC is set when the transmit buffer is empty. TXC is automatically cleared when the transmission end interrupt is executed, and it can also be cleared by writing "1" to TXC. When the transmission end interrupt enable bit TXCIE is "1", TXC can be used to generate a transmission end interrupt.
5	UDRE	Data register empty flag. When UDRE is "1", it indicates that the USPI send data buffer is empty and data can be written. When UDRE is "0", it indicates that the USPI send data buffer is full and cannot write data. When the data register empty interrupt enable bit UDRIE is "1", UDRE can be used to generate a data register empty interrupt.
4:0	-	Reserved.

UCSRB – USPI Control and Status Register B

UCSRB – USPI Control and Status Register B								
Address: 0xC1					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RXCIE	TXCIE	UDRIE	RXEN	TXEN	-	-	-
R/W	R/W	R/W	R/W	R/W	R/W	-	-	-

Bit	Name	Description
7	RXCIE	Receive end interrupt enable bit. Set to enable RXC interrupt, cleared to disable RXC interrupt. When RXCIE is "1", the global interrupt is enabled, and the RXC of the UCSRA register is "1", the USPI reception end interrupt can be generated.
6	TXCIE	Transmit end interrupt enable bit. Set to enable TXC interrupt, cleared to disable TXC interrupt. When TXCIE is "1", the global interrupt is enabled, and the TXC of the UCSRA register is "1", the USPI transmission end interrupt can be generated.
5	UDRIE	Data register empty interrupt enable bit. Set to enable UDRE interrupt, cleared to disable UDRE interrupt. When UDRIE is "1", the global interrupt is enabled, and the USPI data register empty interrupt can be generated when the UDRE of the UCSRA register is "1".
4	RXEN	Receive enable bit. When asserted, starts the USPI receiver. The general purpose IO function of the RxD pin is replaced by USPI receive. Disabling the receiver will flush the receive buffer.
3	TXEN	Send enable bit. When asserted, the USPI transmitter is started. The general purpose IO function of the TxD pin is replaced by a USPI transmit. After TXEN is cleared, only after all data transmission is completed can the USART transmission be disabled.
2:0	-	Reserved.

UCSRC – USART Control and Status Register C

UCSRC – USART Control and Status Register C								
Address: 0xC2					Default value: 0x86			
Bit	7	6	5	4	3	2	1	0
Name	UMSEL1	UMSEL0	-	-	-	DORD	UCPHA	UCPOL
R/W	R/W	R/W	-	-	-	R/W	R/W	R/W

Bit	Name	Description	
7:6	UMSEL1:0	USART mode select bit. UMSEL selects synchronous or asynchronous mode of operation.	
		UMSEL	mode
		0	USART asynchronous operation mode
		1	USART synchronous mode of operation
		2	SPI slave operation mode
		3	SPI master operating mode
5:3	-	Reserved.	

2	DORD	Data transfer order select bit.		
		DORD	Data order	
		0	MSbit first	
		1	LSbit first	
1	UCPHA	Clock phase selection. UCPHA selects data sampling to occur on the start or end edge.		
		UCPHA	sampling time	
		0	Start edge	
		1	End edge	
0	UCPOL	Clock polarity selection. UCPOL selects data changes and sampling to occur on rising or falling edges.		
		UCPOL	Send data change	RX data sampling
		0	XCK rising edge	XCK falling edge
		1	XCK falling edge	XCK rising edge

UBRRL – USPI baud rate register low byte

UBRRL – USPI baud rate register low byte								
Address: 0xC4					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UBRR7	UBRR6	UBRR5	UBRR4	UBRR3	UBRR2	UBRR1	UBRR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	UBRR[7:0]	Low byte portion of the USPI baud rate register. The USPI baud rate register contains two parts, UBRRL and UBRRH, which are combined to set the baud rate for communication.

UBRRH – USPI baud rate register high byte

UBRRH – USPI baud rate register high byte								
Address: 0xC5					Default data: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	UBRR11	UBRR10	UBRR9	UBRR8
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	Description
7:4	-	Reserved.
3:0	UBRR[11:8]	High byte portion of the USPI baud rate register. The USPI baud rate register contains two parts, UBRRL and UBRRH, which are combined to set the baud rate for communication. ***UBRR = {UBRR[11:8], UBRRL}***

		Operating mode	Baud rate calculation formula
		Slave mode	Baud rate determined by the external master
		Master mode	BAUD = f_{sys}/(2*(UBRR+1))

UDR – USPI Data Register

UDR – USPI Data Register								
Address: 0xC6					Default mode: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UDR7	UDR6	UDR5	UDR4	UDR3	UDR2	UDR1	UDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	UDR	Data sent and received by USPI. The USPI transmit data buffer and receive data buffer share the USPI data register UDR. Writing data into UDR means writing into the sending data buffer, and reading data from UDR means reading into the receiving data buffer. In 5 to 8 bit data frame mode, the unused 9th bits are ignored by the transmitter, while the receiver sets them to 0. Only when the UDRE flag of the UCSRA register is "1" can the sending buffer be written, otherwise the operation of the transmitter will be wrong. When the transmit shift register is empty, the transmitter will load the data in the transmit buffer into the transmit shift register, and then the data will be serially output from the TxD pin. The receive buffer contains a two-stage FIFO that changes its state once the receive buffer is read.

TWI – Two-Wire Serial Bus (I2C)

- ❑ Simple yet powerful and flexible communication interface requiring only 2 wires
- ❑ Support master and slave operation
- ❑ The device can operate in either transmitter mode or receiver mode
- ❑ 7-bit address space allows 128 slaves
- ❑ Support multi-master arbitration
- ❑ Up to 400Kbps data transfer rate
- ❑ Fully programmable slave address as well as common address
- ❑ Can wake up on address match in sleep mode

TWI Introduction to the bus

The two-wire serial interface TWI is well suited for typical processor applications. The TWI protocol allows system designers to interconnect 128 different devices with only two bidirectional transmission lines. These two lines are clock SCL and data SDA. External hardware only requires two pull-up resistors on each line. All devices connected to the bus have their own addresses. The TWI protocol solves the problem of bus arbitration.

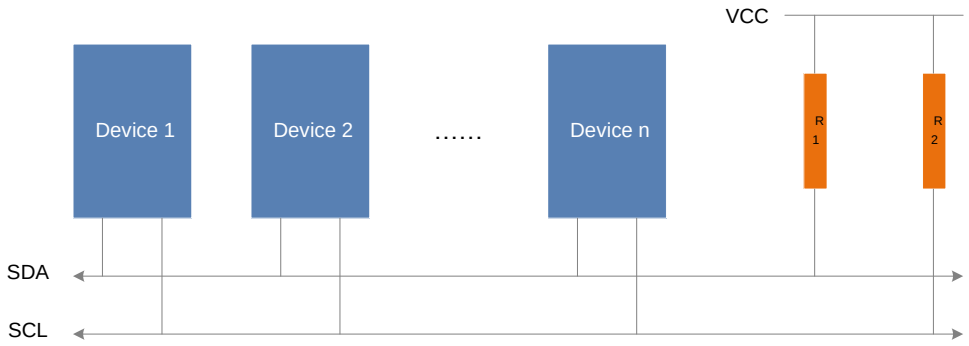
TWI Terminology

The following defined terms appear frequently in this section.

Term	Description
Master	Devices that start and stop transfers. The master is also responsible for generating the SCL clock.
Slave	device addressed by the master
Transmitter	A device that puts data on the bus
Receiver	A device that receives data from the bus

Electrical connections

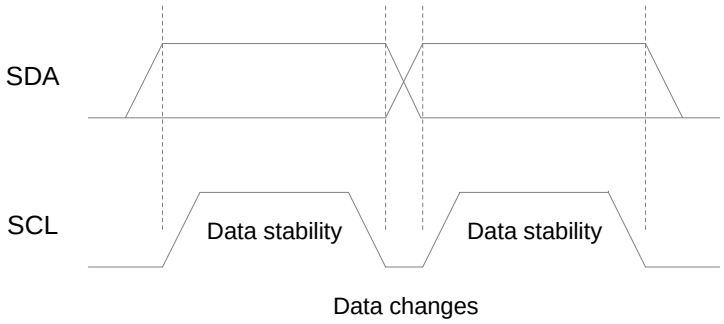
As shown in the figure below, both lines of the TWI interface are connected to the positive power supply through pull-up resistors. The bus drivers of all TWI-compatible devices are open-drain or open-collector, so that the wired-AND function of the interface operation is realized. When the TWI device output is "0", the TWI bus will generate a low level. When all TWI device outputs are tri-stated, the bus allows pull-up resistors to pull the voltage high. To ensure all bus operations, all devices connected to the TWI bus must be powered on.



TWI bus interconnection diagram

Data transfer and frame structure

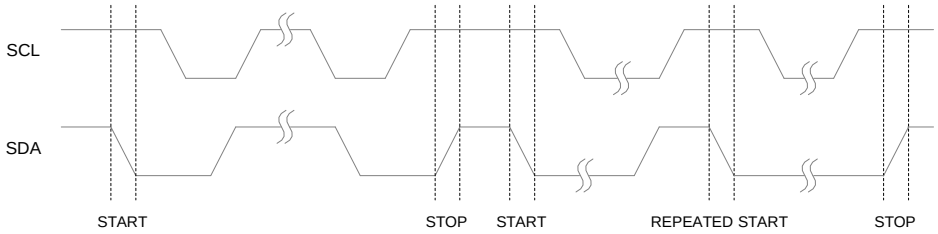
Every bit of data transmission on the TWI bus is synchronized with the clock. When the clock line is high, the level on the data line must remain stable, except to generate a start or stop condition.



TWI Data Validity Map

Start and stop states

TWI transmissions are started and stopped by the master. The master issues a START status on the bus to send data transfers, and a STOP status to stop data transfers. Between START and STOP states, the bus is considered busy and no other master is allowed to attempt to occupy control of the bus. There is a special case that only allows a new START state to occur between the START and STOP states. This is called the REPEATED START state, which is suitable for the current master to start a new transfer without giving up bus control. After a REPEATED START until the next STOP, the bus is still considered busy. This is consistent with START, so in this document, unless otherwise specified, START and REPEATED START are used to describe START and REPEATED START. As shown in the figure below, the START and STOP conditions are to change the level state of the SDA line when the SCL line is high.



START, REPEATED START and STOP state diagram

Address packet format

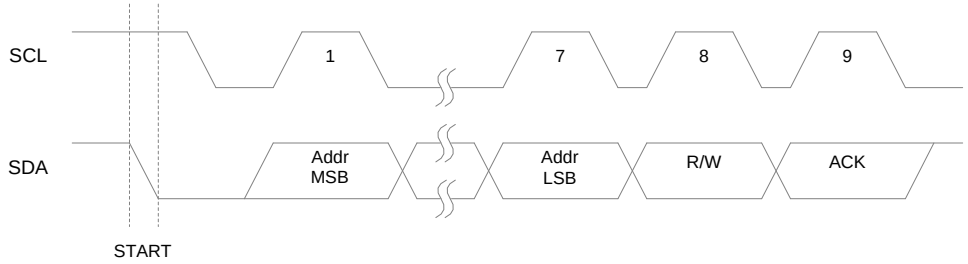
All address packets transmitted on the TWI bus are 9-bit data length, consisting of 7-bit address, 1-bit READ/WRITE control bit and 1-bit response bit. When the READ/WRITE bit is "1", the read operation is performed; when the READ/WRITE bit is "0", the write operation is performed. After the slave is addressed, it must acknowledge by pulling the SDA line low during the ninth SCL (ACK) cycle. If the slave is busy or cannot respond to the master for other reasons, it should keep the SDA line high during the ACK cycle. The master can then issue a STOP condition or a REPEATED START condition to resume transmission.

The address packet includes a slave address and a read or write control bit, represented by SLA+R or SLA+W respectively.

The MSB bit of the address byte occurs first. Except for the reserved address "00000000" which is reserved for general calls and all addresses in the form of "1111xxxx" which need to be reserved for future use, other slave addresses can be freely assigned by the designer.

When a general call occurs, all slaves should respond by pulling the SDA line low during the ACK cycle. The broadcast function can be used when the master needs to send the same information to multiple slaves. After the general call address plus the WRITE bit is sent to the bus, all slaves that need to respond to the general call will pull the SDA line low during the ACK cycle. All those slaves that responded to the general call will receive the following data packets. It should be noted that it is meaningless to send the general call address plus the READ bit, because if several slaves send different data at the same time, it will cause bus conflict.

The address packet format is shown in the figure below:



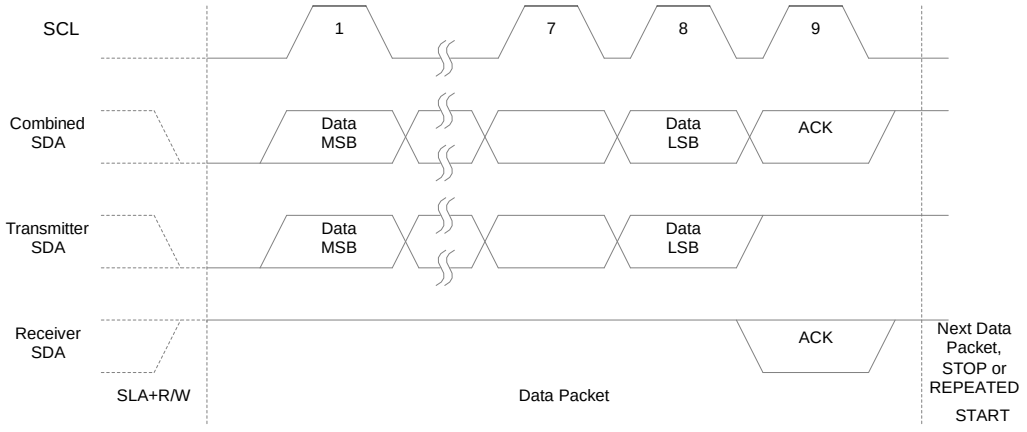
TWI address packet format diagram

Data packet format

All data packets transmitted on the TWI bus are 9-bit data length, consisting of 1 data byte and 1 acknowledge bit. During data transmission, the master is responsible for generating the transmission clock SCL and START and STOP states, the transmitter sends the byte data to be transmitted, and the receiver generates a reception response. The confirmation signal ACK is generated by the receiver by pulling down the SDA line in the 9th SCL (ACK) cycle. If the receiver keeps the SDA line high during the ACK cycle, a non-acknowledgement signal NACK is sent.

When the receiver has collected the last byte, or cannot receive any more data for some reason, it should inform the sender by sending a NACK after receiving the last byte. The MSB bit of the data byte is transmitted first.

The packet format is shown in the figure below:



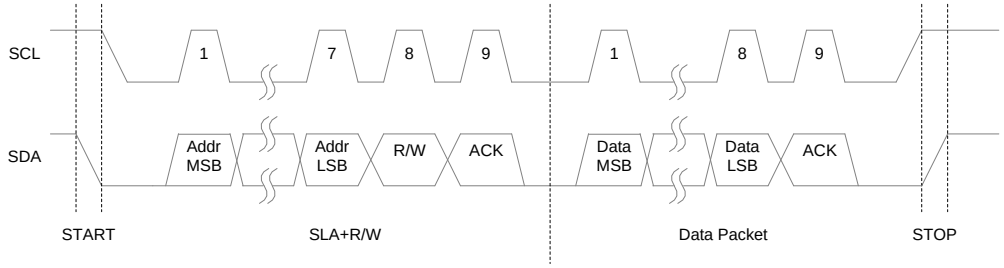
TWI data packet format diagram

Combined address and data packet transmission, a transmission basically consists of 1 START, 1

SLA+R/W, 1 or more data packets and 1 STOP. Only empty messages for START and STOP are illegal. The handshake between the master and slave can be achieved using the wired-AND function of the SCL line. A slave can extend the SCL ground period by pulling the SCL line low. This feature is useful when the master is clocked much faster than the slave, or the slave needs extra time to process data.

The extension of the low-level period of SCL by the slave will not affect the high-level period of SCL, which is still determined by the master. It can be seen that the slave can reduce the data transmission speed of TWI by changing the duty cycle of SCL.

The figure below shows a typical data transfer. Note that multiple bytes can be transmitted between SLA+R/W and STOP, depending on the implementation protocol of the application software.



Typical TWI transmission

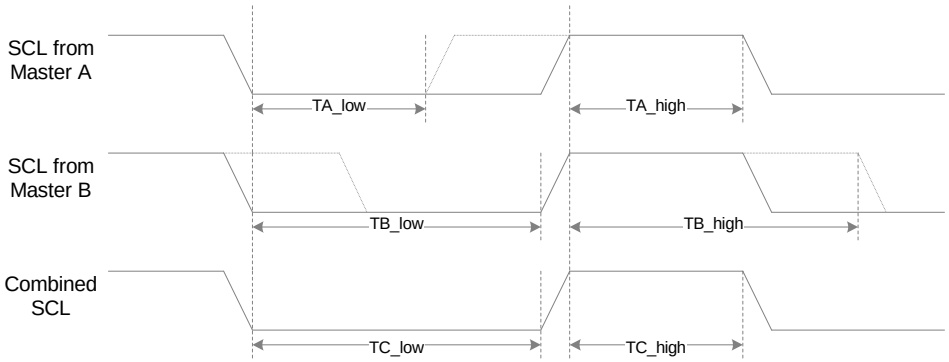
Multi-master systems and their arbitration and synchronization

The TWI protocol allows multiple masters on the bus, and uses special measures to ensure that even if two or more masters start transmission at the same time, it can be processed like a normal transmission. There are two problems with multi-master systems:

1. The implemented algorithm allows only one master among multiple masters to complete the transfer. Other masters must stop their transmissions when they discover that they have lost the option. This selection process is called arbitration. When the master in the competition finds that it has lost the arbitration, it should immediately switch to the slave mode to detect whether it is addressed by the master that has obtained the bus control right. In fact, when multiple masters start transmitting at the same time, it should not be detected by the slave, that is, it is not allowed to destroy the data being transmitted on the bus.
2. Different masters may use different SCL frequencies. In order to ensure the consistency of transmission, it is necessary to design a scheme to synchronize the serial clock of the master. This simplifies the arbitration process.

The line and function of the bus is used to solve the above problems. The serial clocks of all masters are wired together to generate a combined clock, whose high level time is equal to the shortest one among all master clocks, and whose low level is equal to the longest one among all master clocks. All masters listen to SCL, and when the combined SCL clock goes high or low, they can effectively start counting their respective SCL high and low overflow periods, respectively.

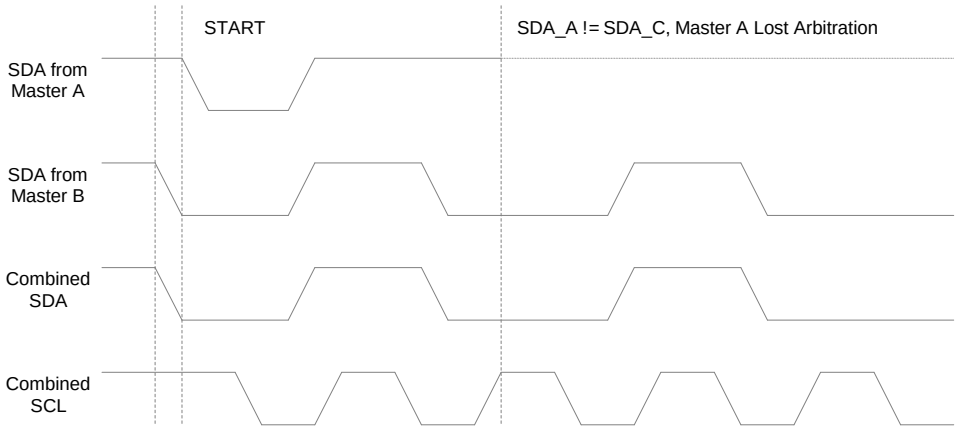
The SCL clock synchronization mechanism of multi-master is shown in the figure below:



Multi-master SCL clock synchronization timing diagram

After outputting data, all masters continue to monitor the SDA line to achieve arbitration. If the value read back from SDA does not match the value output by the master, the master loses arbitration. It should be noted that the master outputs high-level SDA and the other master outputs low-level SDA to lose arbitration. A master that loses arbitration should immediately transition to slave mode and detect if it is being addressed. A master that loses arbitration must deassert the SDA line, but not until the end of the current data or address packet.

to generate a clock signal. Arbitration will continue until there is only one master left in the system, which may take up multiple bits. If multiple masters address the same slave, arbitration will continue until the packet is reached.



Arbitration between two master

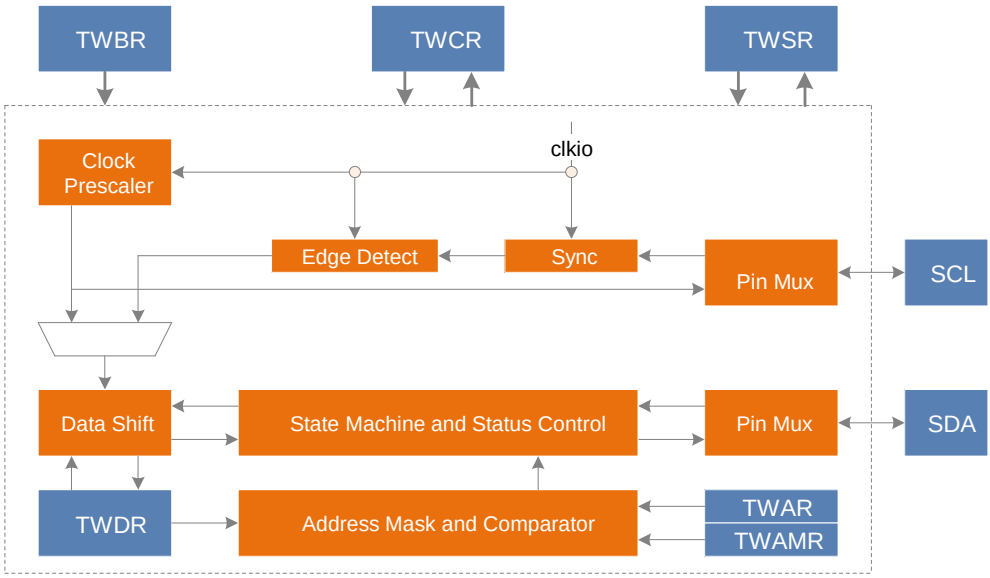
Note that arbitration is not permitted in the following situations:

- ❑ Between a REPEATED START state and a data bit;
- ❑ Between a STOP state and a data bit;
- ❑ Between a REPEATED START state and STOP state;

Application software must take the above into consideration to ensure that these illegal arbitration situations will not occur. This means that in a multi-master system, all data transfers must consist of the same SLA+R/W and data packets. In other words, all transfers must contain the same number of packets, otherwise the arbitration result is undefined.

Overview of TWI modules

The structure diagram of the TWI module is shown in the figure below.



TWI Block structure diagram

The TWI module mainly includes a bit rate generator, a bus interface unit, an address comparator and a control unit, etc. For details, see the following detailed description.

Bit rate generator unit

The bit rate generator unit mainly controls the SCL clock period in master mode. The SCL clock period is jointly determined by the prescaler control bits in the TWI bit rate register TWBR and the TWI status register TWSR. The operation of the slave is not affected by the bit rate or prescaler settings, but it is necessary to ensure that the operating clock of the slave is at least 16 times the SCL frequency. Note that the slave may extend the low period of SCL, thereby reducing the average clock frequency of the TWI bus. The SCL clock frequency is generated by the following formula:

$$fscl = fsys/(16 + 2*TWBR*4TWPS)$$

Among them, TWBR is the value of the TWI bit rate register, and TWPS is the prescaler control bit in the TWI status register.

Bus interface unit

The bus interface unit includes data and address shift register TWDR, START/STOP controller and arbitration decision hardware circuit.

TWDR contains the address or data byte to be sent, or the address or data byte that has been received. In addition to containing 8-bit TWDR, the bus interface unit also includes transmit or receive ACK/NACK registers. This ACK/NACK register cannot be directly accessed by application software. When receiving data, it can be set or cleared by the TWI control register TWCR. When sending data, the received ACK/NACK value is reflected by the TWS value in the TWI status register TWSR.

The START/STOP controller is responsible for generating and detecting START, REPEATED START and STOP states. When the MCU is in certain sleep modes, the START/STOP controller can still detect the START and STOP states, and wake up the MCU from the sleep mode when addressed by the master on the TWI bus.

If TWI starts data transmission in master mode, the arbitration detection circuit will continue to monitor the bus to determine whether it still has bus control. When the TWI module loses control of the bus, the control unit will perform correct actions and generate appropriate status codes to notify the MCU.

Address matching unit

The address matching unit is used to check whether the received address byte matches the 7-bit address in the TWI address register. When the TWI General Call Recognition Enable bit (TWGCE) in the TWAR register is set, the address received from the bus is also compared with the general address. Once the address is matched, the control unit will perform the correct action. The TWI module can respond or not respond to the master's addressing, depending on the setting of the TWCR register. Even in sleep mode, the address matching unit can compare addresses and wake up the MCU from sleep mode if addressed by a master on the bus.

Control unit

The control unit is responsible for listening to the bus and generating corresponding responses according to the settings of the TWCR. When an event that requires application software to participate occurs on the TWI bus, the TWI interrupt flag TWINT will be set. In the next clock cycle, the TWI status register TWSR will be updated to indicate the status code of the event. TWSR contains exact status information when TWINT is asserted. At other times, TWSR is a special status code, indicating that there is no exact status information. Once the TWINT flag is set, the SCL line remains low, suspending the TWI transfer on the bus and allowing the application software to process the event.

The TWINT flag will be set in the following situations:

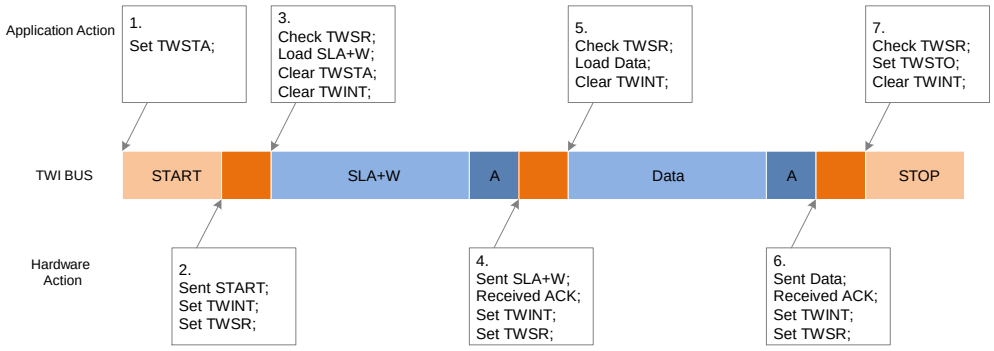
- ❑ After TWI transmits START/REPEATED START status
- ❑ After TWI transmits SLA+R/W
- ❑ After TWI transmits an address byte
- ❑ After the TWI bus arbitration fails
- ❑ After TWI is addressed by the master (slave address matching or broadcast mode)
- ❑ When being addressed as a slave, after receiving STOP or REPEATED START
- ❑ When a bus error is caused by an illegal START or STOP state

Use of TWI

The TWI interface is byte oriented and interrupt based. All bus events, such as receiving a byte or sending a START signal, etc., will generate a TWI interrupt. Since TWI is interrupt-based, the application software can freely perform other operations during the transmission of TWI bytes. The TWI interrupt enable bit TWIE in the TWCR register and the global interrupt enable bit together control whether a TWI interrupt is generated when the TWINT flag is set. If the TWIE bit is cleared, the application software must poll the TWINT flag to detect activity on the TWI bus.

When the TWINT flag is set, it means that the TWI interface has completed the current operation and is waiting for the response of the application software. In this case, the TWI status register TWSR contains the status code reflecting the current bus status. The application software can determine how the TWI interface should work in the next TWI bus cycle by setting the TWCR and TWDR registers.

The figure below shows an example of the connection between the application program and the TWI interface. In this example, the master expects to send one byte of data to the slave. The description here is very simple, and the next chapter will show it in more detail.



TWI typical transmission process diagram

The TWI transmission process shown in the figure is:

1. The first step in a TWI transmission is to send START. Instruct the TWI hardware to send a START signal by writing a specific value to the TWCR register. The written value will be specified later. It is very important that TWINT is set in the written value, writing a "1" to the TWINT bit will clear the bit. TWI will not start any operation while TWINT in TWCR register is set. Once the software clears the TWINT bit, the TWI module immediately starts the transmission of the START signal.
2. When the START status is sent, the TWINT flag of TWCR will be set, and the TWSR will be updated with a new status code, indicating that the START signal was sent successfully.
3. The application checks the value of TWSR to confirm that the START status has been sent successfully. If TWSR shows other values, the application can perform some special operations, such as calling the error handler. When it is determined that the status code is consistent with expectations, the program loads the value of SLA+W into the TWDR register. The TWDR register can be used in both address and data. Then the software writes a specific value to the TWCR register, instructing the TWI hardware to send the value of SLA+W in TWDR. The written value will be specified later. TWINT should be set in the written value to clear the TWINT flag. TWI will not start any operation while TWINT in TWCR register is set. Once the software clears the TWINT bit, the TWI module immediately starts the transmission of the address packet.
4. When the address packet is sent, the TWINT flag of TWCR will be set, and TWSR will be updated with a new status code, indicating that the address packet was sent successfully. The status code will also reflect whether the slave responds to the address packet.
5. The application checks the value of TWSR to confirm that the address packet has been successfully sent, and the received ACK is the expected value. If TWSR shows other values, the application can perform some special operations, such as calling the error handler. After confirming that the status code is consistent with expectations, the program loads the value of Data into the TWDR register. Then the software writes a specific value to the TWCR register, instructing the TWI hardware to send the value of Data in TWDR. The written value will be specified later. TWINT should be set in the written value to clear the TWINT flag. TWI will not start any operation while TWINT in TWCR register is set. Once the software clears the TWINT bit, the TWI module immediately starts the transmission of the data packet.
6. When the data packet is sent, the TWINT flag of TWCR will be set, and TWSR will be updated with a new status code, indicating that the data packet was successfully sent. The status code will also reflect whether the slave responds to the packet.
7. The application checks the value of TWSR to confirm that the data packet has been successfully sent, and the received ACK is the expected value. If TWSR shows other values, the application can perform some special operations, such as calling the error handler. When it is determined that the status code is consistent with expectations, the software writes a specific value to the TWCR register, instructing the TWI hardware to send a STOP signal. The written value will be specified later. TWINT should be set in the written value to clear the TWINT flag. TWI will not start any operation while TWINT in TWCR register is set. Once the software clears the TWINT bit, the TWI module immediately starts the transmission of the STOP signal. It should be noted that TWINT will not be set after the STOP signal is sent.

Although the example is relatively simple, it contains all the rules in the process of TWI data transmission. Summarized as follows:

- ❑ The TWINT flag is set when TWI has completed an operation and is waiting for feedback from the application. The SCL clock line will be pulled low until TWINT is cleared;

- When the TWINT flag is set, the user must update all TWI registers with the values associated with the next TWI bus cycle. For example, the TWDR register must be loaded with the value to be transmitted on the next bus cycle.
- After updating all registers and completing other necessary operations, the application program writes the TWCR register. When writing TWCR, the TWINT bit must be set to clear the TWINT flag. After TWINT is cleared, TWI starts to execute the operation set by TWCR.

Transfer mode

TWI can work in the following 4 main modes: Master Transmitter (MT), Master Receiver (MR), Slave Transmitter (ST) and Slave Receiver (SR). Multiple modes can be used under the same application. For example, TWI can use MT mode to write data to TWI EEPROM, and use MR mode to read data from EEPROM. If there are other masters on the system, some of which may also send data to TWI, SR mode will be used. It is up to the application software to decide which mode to use.

These modes are described in detail below. In the data transmission in each mode, the possible status codes will be described with pictures. These pictures contain the following abbreviations:

S： Start state
Rs： REPEATED START state
R： Read operation flag (SDA is high)
W： Write flag bit (SDA is low)
A: Acknowledgment bit (SDA is low)
NA： No response bit (SDA is high)
Data：8 bits of data bytes
P： STOP status
SLA： Slave address

The circle in the picture is used to indicate that the TWINT flag is set, and the number in the circle indicates the status code in the TWSR register, in which the prescaler control bit is masked as "0". In these places, the application must perform appropriate actions to continue or complete the TWI transmission. The TWI transfer will be suspended until the TWINT flag is cleared.

When the TWINT flag is set, the status code in the TWSR is used to determine the appropriate software action. Details of the required software actions and subsequent serial transfers for each status code are given in the individual tables. Note that the prescaler control bit in TWSR in the table is masked as "0".

Master send mode

In master transmit mode, TWI will send a certain number of data bytes to the slave receiver. In order to enter master mode, a START signal must be sent. The format of the address packet that follows determines whether the TWI enters master-transmitter mode or master-receiver mode. If SLA+W is sent, enter master send mode. If SLA+R is sent, it enters master receive mode. The status codes mentioned in this chapter all assume that the prescaler control bit is "0".

A START signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

The TWEN bit must be set to "1" to enable the TWI interface, TWSTA to be set to "1" to send the START signal, and TWINT to be set to "1" to clear the TWINT flag. The TWI module detects the bus status and sends a START signal immediately when the bus is free. After sending START, the hardware sets the TWINT flag, and at the same time updates the status code of TWSR to 0x08.

In order to enter master transmit mode, SLA+W must be transmitted. This can be done by doing the following. First write SLA+W to the TWDR register, then write "1" to the TWINT bit to clear the TWINT flag to continue the transmission, that is, write the following values to the TWCR register to send SLA+W:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

When the SLA+W transmission is completed and the response signal is received, TWINT is set again, and the status code of TWSR is updated at the same time. Possible status codes are 0x18, 0x20 or 0x38. The appropriate response under each status code will be described in detail in the status code table.
When the SLA+W is successfully sent, the data packet can be sent. This is done by writing data to the TWDR register.

TWDR can only be written when the TWINT flag is high. Otherwise, the access is ignored and the write conflict flag TWWC will be set. After updating the TWDR, write "1" to the TWINT bit to clear the TWINT flag to continue the transmission. That is, write the following values to the TWCR register to send data:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

When the data packet is sent and the response signal is received, TWINT is set again, and the status code of TWSR is updated at the same time. Possible status codes are 0x28 or 0x30. The appropriate response under each status code will be described in detail in the status code table.

When the data is sent successfully, you can continue to send data packets. This process is repeated until the last byte is sent.
The master generates a STOP signal or a REPEATED START signal before the entire transmission ends.
A STOP signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

A REPEATED START signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

After sending REPEATED START (status code 0x10), the TWI interface can access the same slave again, or access a new slave without sending a STOP signal. REPEATED START allows the master to switch between different slaves, master-transmitter and master-receiver modes without losing control of the bus.

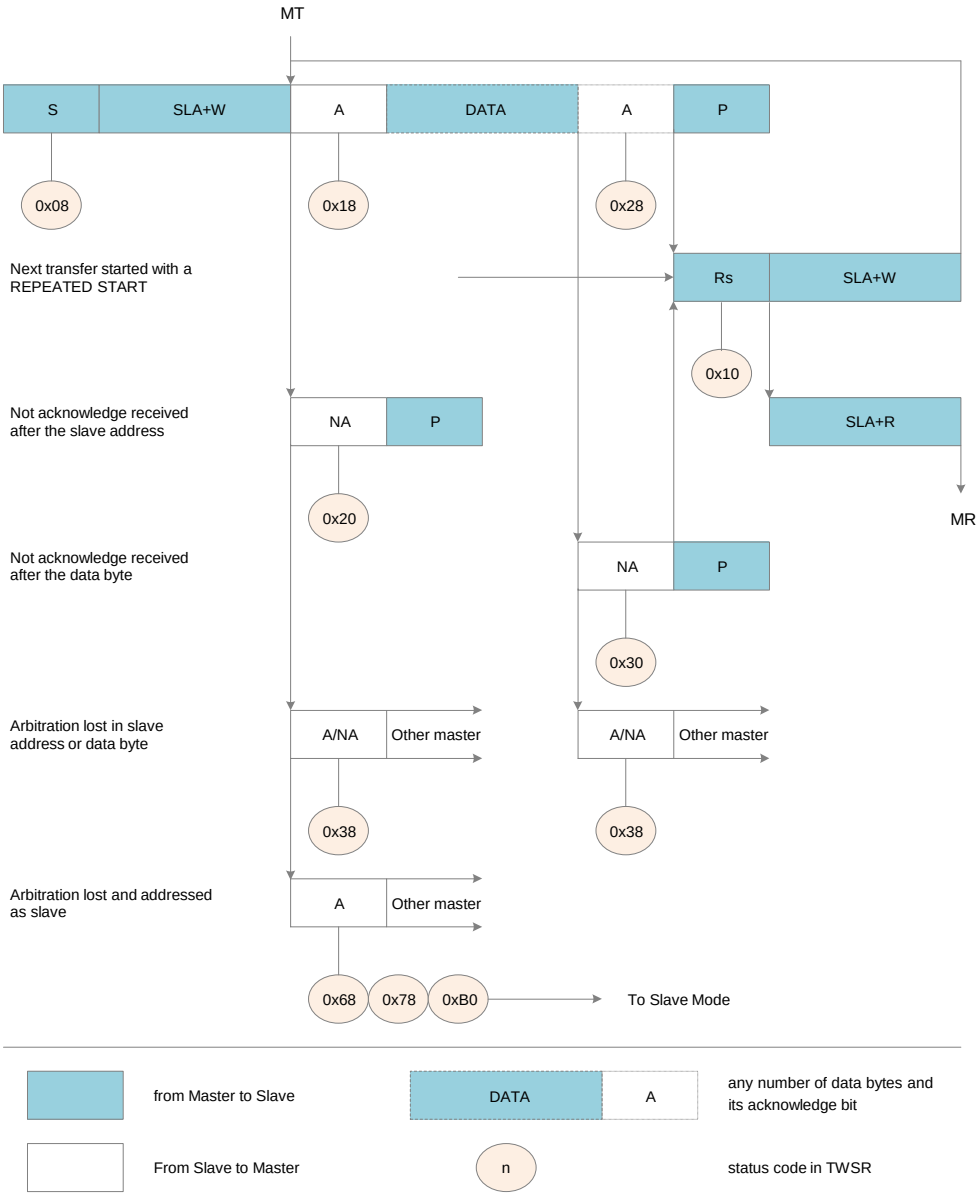
The status codes and corresponding operations in the master sending mode are shown in the following table:

Status code table of master sending mode

Status code	Bus and Hardware Status	Application software response					HW next steps
		R/W TWDR	Operations on TWCR				
			STA	STO	TWINT	TWEA	
0x08	START sent	Load SLA+W	0	0	1	x	SLA+W will be sent; Will receive ACK or NACK
0x10	REPEATED START sent	Load SLA+W	0	0	1	x	SLA+W will be sent; Will receive ACK or NACK

		Load SLA+R	0	0	1	x	SLA+R will be sent; will receive ACK or NACK; will switch to MR mode
0x18	SLA+W sent; ACK received	Load data	0	0	1	x	Data will be sent; Will receive ACK or NACK
		No action	1	0	1	x	REPEATED START is sent
		No action	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		No action	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent
0x20	SLA+W sent; NACK received	Load data	0	0	1	x	Data will be sent; Will receive ACK or NACK
		No action	1	0	1	x	REPEATED START is sent
		No action	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		No action	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent
0x28	data bytes sent ACK received	Load data	0	0	1	x	Data will be sent; Will receive ACK or NACK
		No action	1	0	1	x	REPEATED START is sent
		No action	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		No action	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent
0x30	data bytes sent NACK received	Load data	0	0	1	x	Data will be sent; Will receive ACK or NACK
		No action	1	0	1	x	REPEATED START is sent
		No action	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		No action	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent
0x38	SLA+W or data arbitration failure	No action	0	0	1	x	The bus is released; Unaddressed slave mode is entered
		No action	1	0	1	x	START will be sent when idle

The format and status of the master sending mode are shown in the following figure:



Format and State Diagram of master Transmit Mode

Master Receive Mode

In master receive mode, the TWI will receive a certain number of data bytes from the slave transmitter. In order to enter master mode, a START signal must be sent. The format of the address packet that follows determines whether the TWI enters master-transmitter mode or master-receiver mode. If SLA+W is sent, enter master send mode. If SLA+R is sent, it enters master receive mode. The status codes mentioned in this chapter all assume that the prescaler control bit is "0".

A START signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

The TWEN bit must be set to "1" to enable the TWI interface, TWSTA to be set to "1" to send the START signal, and TWINT to be set to "1" to clear the TWINT flag. The TWI module detects the bus status and sends a START signal immediately when the bus is free. After sending START, the hardware sets the TWINT flag, and at the same time updates the status code of TWSR to 0x08.

In order to enter master receive mode, SLA+R must be sent. This can be done by doing the following. First write SLA+R to the TWDR register, then write "1" to the TWINT bit to clear the TWINT flag to continue the transmission, that is, write the following values to the TWCR register to send SLA+R:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

When the SLA+R transmission is completed and the response signal is received, TWINT is set again, and the status code of TWSR is updated at the same time. Possible status codes are 0x38, 0x40 or 0x48. The appropriate response under each status code will be described in detail in the status code table.

When the SLA+R is successfully sent, it can start receiving data packets. To continue receiving, clear the TWINT flag by writing "1" to the TWINT bit. That is, write the following values to the TWCR register to start receiving:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

When the data packet is received and the response signal is sent, TWINT is set again, and the status code of TWSR is updated at the same time. Possible status codes are 0x50 or 0x58. The appropriate response under each status code will be described in detail in the status code table.

When the data is received successfully, you can continue to receive data packets. This process is repeated until the last byte is received.

After the master receives the last byte, it must send a NACK response signal to the slave transmitter. The master generates a STOP signal or a REPEATED START signal before the entire reception ends.

A STOP signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

A REPEATED START signal is issued by writing the following value to the TWCR register:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

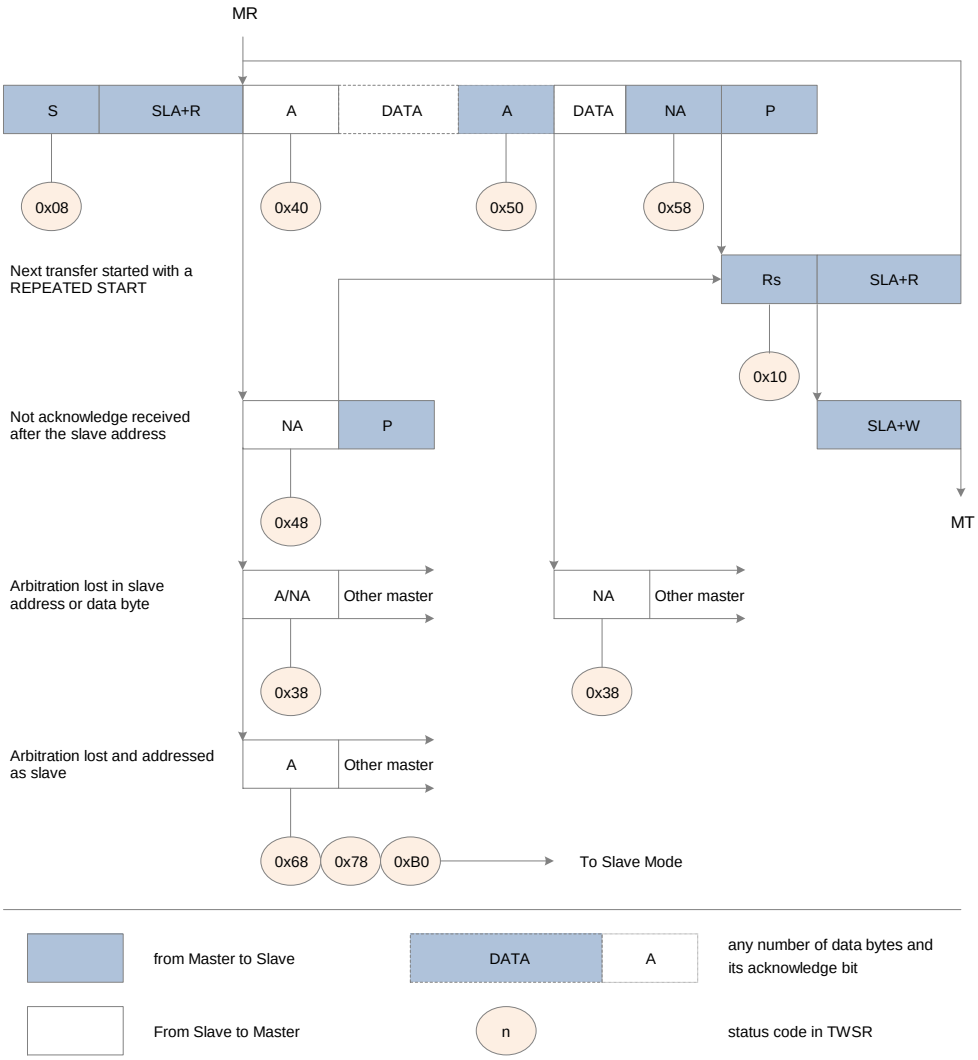
After sending REPEATED START (status code 0x10), the TWI interface can access the same master again, or access a new master without sending a STOP signal. REPEATED START allows the master to switch between different slaves, master-transmitter and master-receiver modes without losing control of the bus.
The status codes and corresponding operations in the master receiving mode are shown in the following table:

Status code table for master receive mode

Status code	Bus and Hardware Status	Application software response					HW next steps
		R/W TWDR	Operations on TWCR				
			STA	STO	TWINT	TWEA	
0x08	START sent	Load SLA+R	0	0	1	x	SLA+R will be sent; Will receive ACK or NACK
0x10	REPEATED START sent	Load SLA+R	0	0	1	x	SLA+R will be sent; Will receive ACK or NACK

		Load SLA+W	0	0	1	x	• SLA+W will be sent; • will receive ACK or NACK; • will be switched to MT mode
0x38	SLA+R or data arbitration failure	No action	0	0	1	x	The bus is released; Unaddressed slave mode is entered
		No action	1	0	1	x	START will be sent when idle
0x40	SLA+R has been sent; ACK received	No action	0	0	1	0	will receive data; NACK will be sent
		No action	0	0	1	1	will receive data; The ACK will be sent
0x48	SLA+R has been sent; NACK received	No action	1	0	1	x	REPEATED START is sent
		No action	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		No action	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent
0x50	data bytes received; ACK sent	Read Data	0	0	1	0	will receive data; NACK will be sent
		Read Data	0	0	1	1	will receive data; The ACK will be sent
0x58	data bytes received; NACK sent	Read Data	1	0	1	x	REPEATED START is sent
		Read Data	0	1	1	x	STOP will be sent; The TWSTO flag will be reset
		Read Data	1	1	1	x	STOP will be sent; The TWSTO flag will be reset; START will be sent

The format and status of the master receiving mode are shown in the following figure:



Format and state diagram for master receive mode

Slave Receive Mode

In slave receive mode, a certain number of data bytes can be received from the master transmitter. The status codes mentioned in this chapter all assume that the prescaler control bit is "0".

To enable slave receive mode, set the TWAR and TWCR registers.
TWAR needs to be set as follows:

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Device slave address							

The upper 7 bits of TWAR are the slave address that the TWI interface will respond to when the master is addressing. If the LSB is set, TWI will respond to the general call address (0x00), otherwise ignore the general call address.
TWCR needs to be set as follows:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

TWEN must be set to enable the TWI interface, and TWEA must be set to enable the master to return confirmation information ACK when addressing (slave address or general call) to itself. TWSTA and TWSTO must be cleared.

After initializing TWAR and TWCR, the TWI interface starts to wait until its own slave address (or broadcast address) is addressed. When the data direction bit following the slave address is "0" (indicating a write operation), the TWI enters slave receive mode. When the data direction bit is "1" (indicating a read operation), the TWI enters the slave transmitter mode. After receiving its own slave address and write operation flag, the TWINT flag is set, and the valid status code is also updated to TWSR. The appropriate response under each status code will be described in detail in the status code table. It should be noted that when the TWI arbitration in the master mode fails, it can also enter the slave receiver mode (see status code 0x68 and 0x78).

If the TWEA bit is reset during transmission, TWI will return NACK (high level) to the SDA line after receiving a byte. This can be used to indicate that the slave cannot receive any more data. When the TWEA bit is "0", TWI will not respond to its own slave address. However, TWI will still monitor the bus, and once TWEA is set, it can resume address recognition and respond. That is to say, TWEA can be used to temporarily isolate the TWI interface from the bus.

In sleep modes other than idle mode, the clock of the TWI interface can be turned off. If the slave receiver mode is enabled, the interface will continue to respond to the slave address or broadcast address using the bus clock. An address match will wake up the MCU. During wake-up, the TWI interface will keep SCL low until the TWINT flag is cleared. When the TWI interface clock returns to normal, more data can be received.

The status codes of the slave receiving mode are shown in the following table:

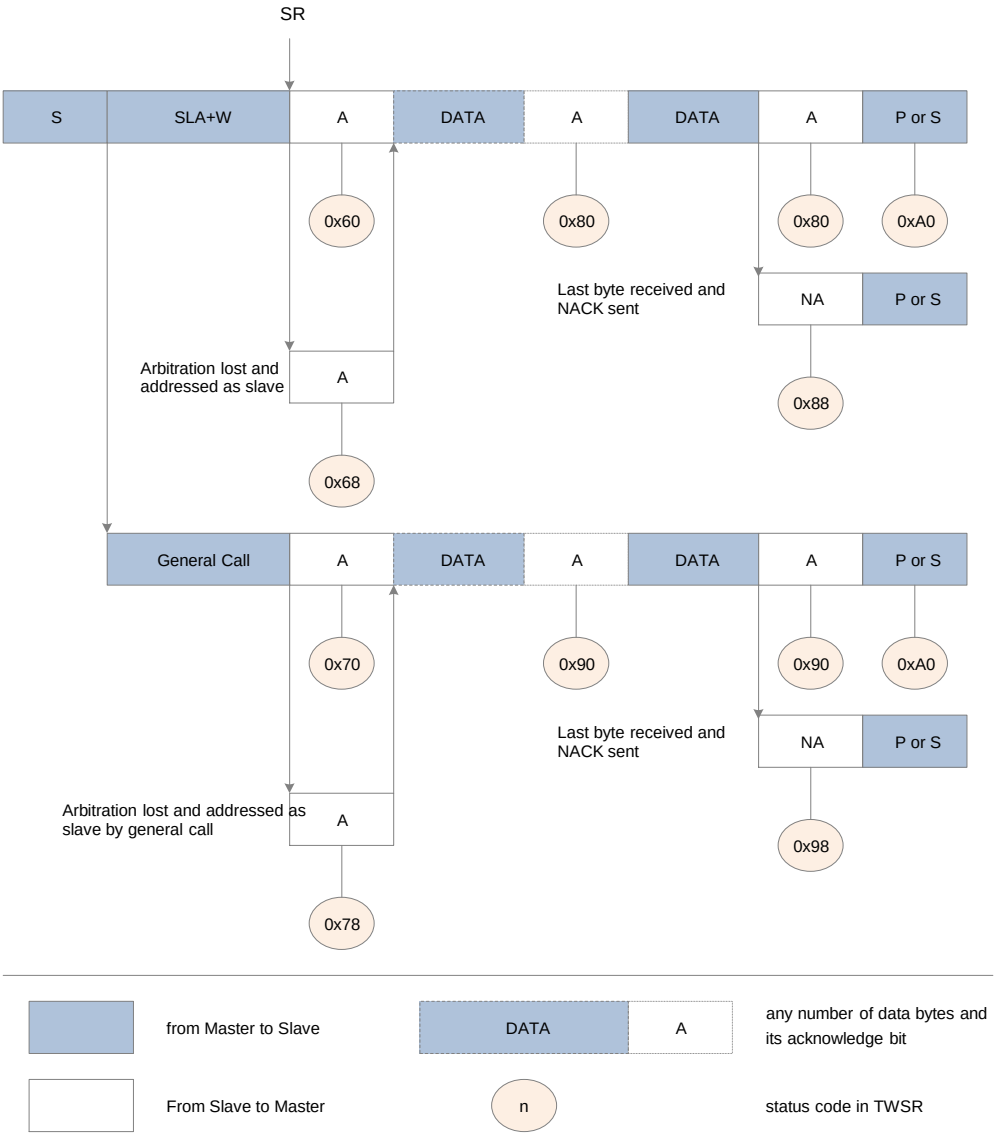
Status Code Table of Slave Receive Mode

Status code	Bus and Hardware Status	Application software response					HW next steps
		R/W TWDR	Operations on TWCR				
			STA	STO	TWINT	TWEA	
0x60	• SLA+W received; • ACK sent	No action	x	0	1	0	• will receive data; • will send NACK
		No action	x	0	1	1	• will receive data; • ACK will be sent
0x68	• Arbitration fails when sending SLA+R/W; • SLA+W received; • ACK sent	No action	x	0	1	0	• will receive data; • NACK will be sent
		No action	x	0	1	1	• will receive data; • ACK will be sent
0x70	• The broadcast address has been received; • ACK sent	No action	x	0	1	0	• will receive data; • NACK will be sent
		No action	x	0	1	1	• will receive data; • ACK will be sent
0x78	• Arbitration fails when sending SLA+R/W; • SLA+W received; • ACK sent	No action	x	0	1	0	• will receive data; • NACK will be sent
		No action	x	0	1	1	• will receive data; • ACK will be sent
0x80	• Own data has been received; • ACK sent	Read Data	x	0	1	0	• will receive data; • NACK will be sent
		Read Data	x	0	1	1	• will receive data; • ACK will be sent

0x88	• Own data has been received; • NACK sent data	Read data	0	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave addresses and broadcasts
		Read data	0	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • TWGCE=1 will respond to broadcast
		Read data	1	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave address and broadcast; • A START will be sent when the bus is free
		Read data	1	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • When TWGCE=1, it will respond to the broadcast; • A START will be sent when the bus is free
0x90	• Broadcast data has been received; • ACK sent	Read data	x	0	1	0	• will receive data; • will send NACK
		Read data	x	0	1	1	• will receive data; • ACK will be sent
0x98	• Broadcast data has been received; • NACK sent	Read data	0	0	1	0	• will switch to unaddressed slave mode; • will not respond to slave addresses and broadcast
		Read data	0	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • TWGCE=1 will respond to broadcast
		Read data	1	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave address and broadcast; • A START will be sent when the bus is free

		Read data	1	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • When TWGCE=1, it will respond to the broadcast; • A START will be sent when the bus is free
0xA0	Received STOP or REPEATED START while the slave is working	No action	0	0	1	0	• will not respond to slave addresses and broadcast • will switch to unaddressed slave mode;
			0	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • TWGCE=1 will respond to broadcast
			1	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave address and broadcast; • A START will be sent when the bus is free
			1	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • When TWGCE=1, it will respond to the broadcast; • A START will be sent when the bus is free

The format and state diagram of the slave receive mode are as follows:



Format and State Diagram for Slave Receive Mode

Slave send mode

In slave transmit mode, a certain number of data bytes can be sent to the master receiver. The status codes mentioned in this chapter all assume that the prescaler control bit is "0".

To enable slave receive mode, set the TWAR and TWCR registers.
TWAR needs to be set as follows:

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Device Slave Address							

The upper 7 bits of TWAR are the slave address that the TWI interface will respond to when the master is addressing. If the LSB is set, TWI will respond to the general call address (0x00), otherwise ignore the general call address.
TWCR needs to be set as follows:

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

TWEN must be set to enable the TWI interface, and TWEA must be set to enable the master to return confirmation information ACK when addressing (slave address or general call) to itself. TWSTA and TWSTO must be cleared.

After initializing TWAR and TWCR, the TWI interface starts to wait until its own slave address (or broadcast address) is addressed. When the data direction bit following the slave address is "0" (indicating a write operation), the TWI enters slave receive mode. When the data direction bit is "1" (indicating a read operation), the TWI enters the slave transmitter mode. After receiving its own slave address and read operation flag, the TWINT flag is set, and the valid status code is also updated to TWSR. The appropriate response under each status code will be described in detail in the status code table. It should be noted that when the TWI arbitration in the master mode fails, it can also enter the slave sending mode (see status code 0xB0).

If the TWEA bit is reset during transmission, the TWI will switch to unaddressed slave mode after sending the last byte. After the master receiver gives NACK or ACK for the last byte transmission, the status code in the TWSR register will be updated to 0xC0 or 0xC8. If the master receiver continues to transmit, the slave transmitter will not respond, and the master will receive all "1" data (ie 0xFF). When the slave sends the last byte of data (TWEA is cleared) and expects a NACK response, and the master wants to receive more data and sends an ACK as a response, TWSR will be updated to 0xC8.

When the TWEA bit is "0", TWI will not respond to its own slave address. However, TWI will still monitor the bus, and once TWEA is set, it can resume address recognition and respond. That is to say, TWEA can be used to temporarily isolate the TWI interface from the bus.

In sleep modes other than idle mode, the clock of the TWI interface can be turned off. If the slave receiver mode is enabled, the interface will continue to respond to the slave address or broadcast address using the bus clock. An address match will wake up the MCU. During wake-up, the TWI interface will keep SCL low until the TWINT flag is cleared. When the TWI interface clock returns to normal, more data can be received.

The status codes of the slave sending mode are shown in the following table:

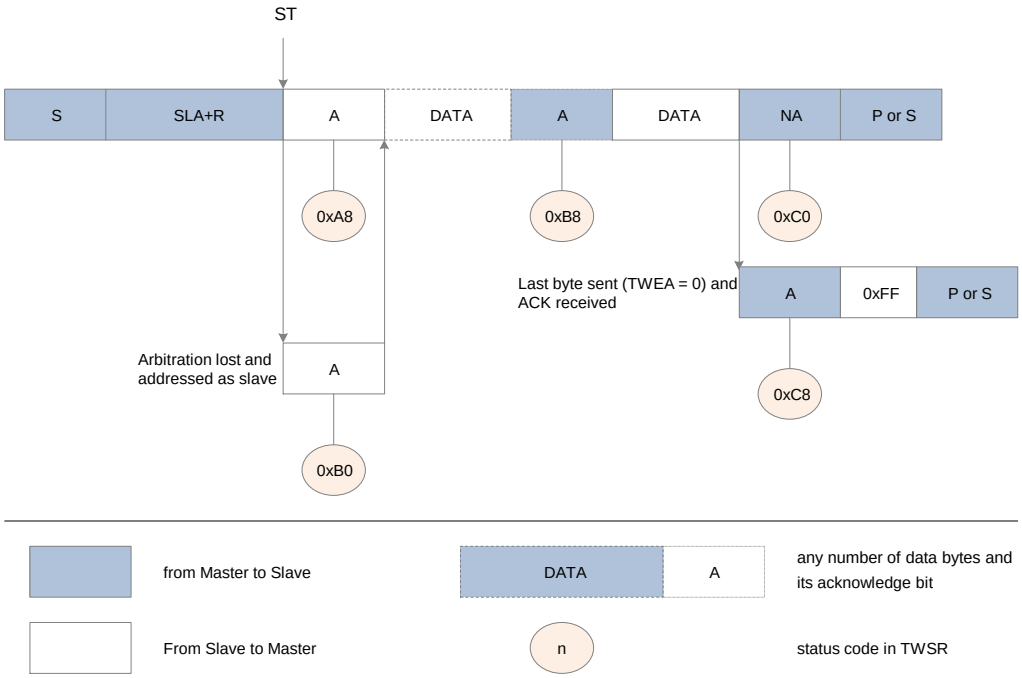
Status code table of slave sending mode

Status code	Bus and Hardware Status	Application software response					HW next steps
		R/W TWDR	Operations on TWCR				
			STA	STO	TWINT	TWEA	
0xA8	• SLA+R Received; • ACK sent	Load data	x	0	1	0	• The last data will be sent; • Expect to receive NACK
		Load data	x	0	1	1	• data will be sent; • will receive ACK
0xB0	• Arbitration failed when sending SLA+R/W; • SLA+R received; • ACK sent	Load data	x	0	1	0	• The last data will be sent; • Expect to receive NACK
		Load data	x	0	1	1	• data will be sent; • will receive ACK

0xB8	• data sent; • ACK received	Load data	x	0	1	0	• The last data will be sent; • Expect to receive NACK
		Load data	x	0	1	1	• data will be sent; • will receive ACK
0xC0	• data sent; • NACK received	No action	0	0	1	0	• will switch to unaddressed slave mode; • will not respond to slave addresses and broadcast
		No action	0	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • TWGCE=1 will respond to broadcast
		No action	1	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave address and broadcast; • A START will be sent when the bus is free
		No action	1	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • When TWGCE=1, it will respond to the broadcast; • A START will be sent when the bus is free
0xC8	• The last data has been sent; • ACK received	No action	0	0	1	0	• will switch to unaddressed slave mode; • will not respond to slave addresses and broadcast
		No action	0	0	1	1	• will switch to unaddressed slave mode; • will respond to the slave address; • TWGCE=1 will respond to broadcast
		No action	1	0	1	0	• will switch to unaddressed slave mode; • Will not respond to slave address and broadcast; • A START will be sent when the bus is free
		No action	1	0	1	1	• will switch to unaddressed slave mode;

							• will respond to the slave address; • When TWGCE=1, it will respond to the broadcast; • A START will be sent when the bus is free

The format and status of the slave sending mode are shown in the figure below:



Format and State Diagram for Slave Transmit Mode

Other statuses

There are two status codes that do not have corresponding TWI status definitions, as shown in the following table:

Other status codes table

Status code	Bus and Hardware Status	Application software response					HW next steps
		R/W TWDR	Operations on TWCR				
			STA	STO	TWINT	TWEA	
0xF8	• no state information; • TWINT = 0	No action	Do not operate TWCR				Wait or proceed with the current operation
0x00	• Bus error caused by illegal START or STOP	No action	0	1	1	x	• Only affects internal hardware; • Will not send STOP to the bus; • The bus is released and the TWSTO bit is cleared

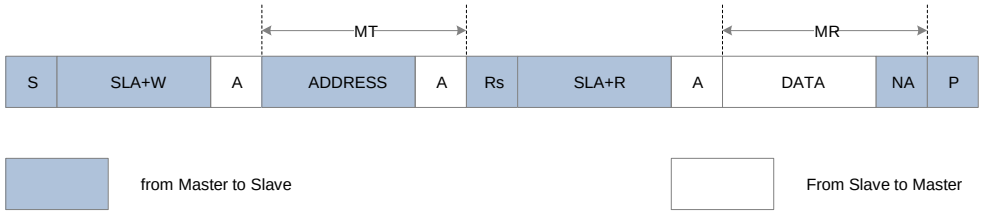
The status code 0xF8 means that there is no relevant information at present, because the TWINT flag is "0". This state may occur when the TWI interface is not involved in a serial transfer or the current transfer is not yet complete.

Status 0x00 indicates that a bus error occurred during a serial transfer. A bus error occurs when an illegal START or STOP occurs. For example, START or STOP appears between address and data, address and ACK. A bus error will set TWINT. To recover from an error, TWSTO must be set and TWINT must be cleared by writing "1". This will put the TWI interface into unaddressed slave mode without STOP, release SCL and SDA, and clear the TWSTO bit.
combination mode

In some cases, several TWI modes must be combined in order to accomplish the desired job. For example, to read data from a serial EEPROM, a typical transfer includes the following steps:
1. The transfer must be started;
2. You must tell the EEPROM where the data should be read;
3. The read operation must be completed;
4. The transfer must end.

Note that data can be transferred from master to slave and vice versa. The master tells the slave where to read the data, using the master send mode. Next, read data from the slave, using the master receive mode. The direction of transmission will change. The master must maintain control of the bus at all stages, and all steps are uninterrupted operations. If in a multi-master system, another master changes the position of reading data between steps 2 and 3, this principle is broken, and the position of the master reading data will be wrong. Changing the direction of data transfer is accomplished by sending a REPEATED START between sending the address byte and receiving the data. After sending REPEATED START, the master still has the bus control right.

The following figure describes this transfer process:



Combining multiple TWI modes to access serial EEPROM maps

Multi-master system and arbitration

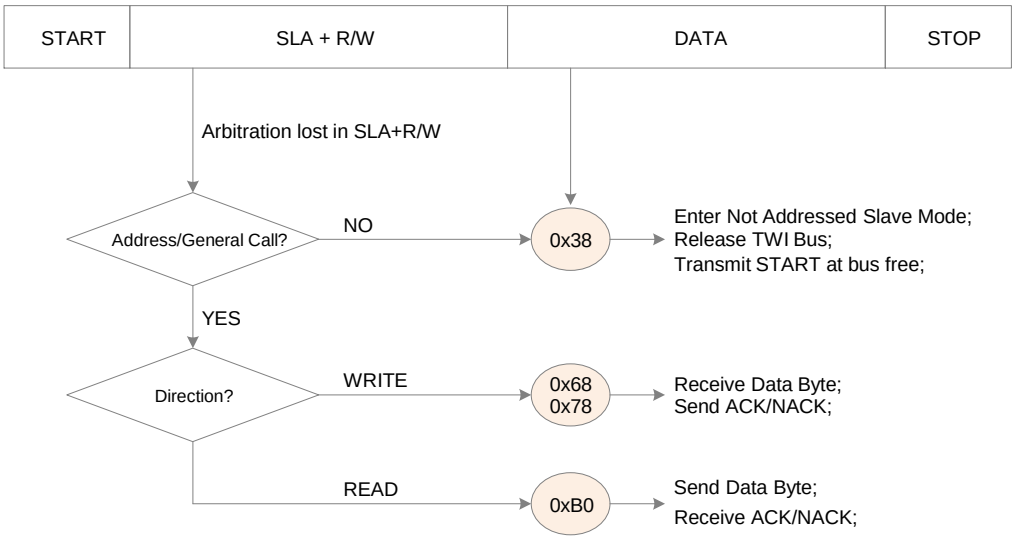
If there are multiple masters connected to the same TWI bus, one or more of them may start data transfer at the same time.
The TWI protocol ensures that in this case, through an arbitration process, allowing one of the masters to transmit will not lose data. The following describes the process of bus arbitration by taking two masters trying to send data to the slave as an example.

There are several different situations that generate the bus arbitration process:

- Two or more masters communicate with a slave simultaneously. In this case, neither the master nor the slave is aware of the contention on the bus;
- Two or more masters simultaneously access the same slave in different data or operation directions. Arbitration occurs in this case, either on the READ/WRITE bits or on the data bits. When other masters send "0" to the SDA line, the master that sends "1" to the SDA line will fail the arbitration. A failed master will either switch to unaddressed slave mode, or wait for the bus to be free and send a new START signal, depending on the operation of the application software.
- Two or more masters access different slaves. In this case, bus arbitration occurs during the SLA phase. When other masters send "0" to the SDA line, the master that sends "1" to the SDA line will fail the arbitration. A master that fails in SLA bus arbitration switches to slave mode and checks whether it is addressed by a master that has gained control of the bus. If addressed, it will enter SR or ST mode, depending on the READ/WRITE bits following the SLA. If it is not addressed, it will

switch to unaddressed slave mode, or wait for the bus to be free and send a new START signal, depending on the operation of the application software.

The following figure describes the process of bus arbitration:



Bus Arbitration Process Diagram

Registers definition

TWI register list

Register	Address	Default value	Description
TWBR	0x B8	0x00	TWI bit rate register
TWSR	0xB9	0x00	TWI status register
TWAR	0xBA	0x00	TWI address register
TWDR	0xBB	0x00	TWI data register
TWCR	0xBC	0x00	TWI control register
TWAMR	0xBD	0x00	TWI address mask register

TWBR – TWI bitrate register

TWBR – TWI bitrate register								
Address: 0xB8					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:0	TWBR[7:0]	TWI bit rate selection control bits. TWBR is the bit rate generator division factor. The Bit Rate Generator is a frequency divider used to generate the SCL clock in master mode. The calculation formula of the bit rate is as follows: $f_{scl} = f_{sys}/(16 + 2*TWBR*4^{TWPS})$

TWSR – TWI status register

TWSR – TWI status register								
Address: 0xB9					Default value: 0xF8			
Bit	7	6	5	4	3	2	1	0
Name	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:3	TWS[7:3]	TWI status flag. The 5-bit TWS reflects the state of the TWI logic and the bus. Different status values have different meanings, see the description of TWI working mode for details. The value read from TWSR includes 5-bit state value and 2-bit prescaler control bit, and the prescaler bit should be masked as "0" when detecting the state. This is state detection independent of the prescaler setting.
2	-	Reserved.
1	TWPS1	TWI prescaler control high bit. TWPS1 and TWPS0 together form TWPS[1:0], which is used to control the bit rate prescaler factor, and control the bit rate together with TWBR.
0	TWPS0	TWI prescaler control low bit. TWPS0 and TWPS1 together form TWPS[1:0], which is used to control the bit rate prescaler factor, and control the bit rate together with TWBR.

TWPS[1:0]	Prescaler factor
0	1
1	4
2	16
3	64

TWAR – TWI address register

TWAR – TWI address register								
Address: 0xBA					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TWAR6	TWAR5	TWAR4	TWAR3	TWAR2	TWAR1	TWAR0	TWGCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:1	TWA[6:0]	TWI slave address bits. TWA is the TWI slave address. When TWI works in slave mode, TWI will respond according to this address. master mode does not require this address. But in a multi-master system, it is also necessary to set the slave address for other masters to access.
0	TWGCE	TWI broadcast identification enable control bit. When the TWGCE bit is set to "1", TWI bus broadcast recognition is enabled. When the TWGCE bit is set to "0", TWI bus broadcast identification is prohibited. When TWGCE is set and the received address frame is 0x00, the TWI module will respond to this bus broadcast.

TWDR – TWI Data Register

TWDR – TWI Data Register								
Address: 0xBB						Default value: 0xFF		
Bit	7	6	5	4	3	2	1	0
Name	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	Description						
7:0	TWD[7:0]	TWI data registers. TWD is the next byte that will be transmitted on the bus, or the last byte that has just been received from the bus.						

TWCR – TWI control register

TWCR – TWI control register								
Address: 0xBC						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
R/W	R/W	R/W	R/W	R/W	R	R/W	-	R/W

Bit	Name	Description
7	TWINT	TWI interrupt flag bit. When TWI completes the current work and expects the application software to intervene, the hardware will set the TWINT bit. If the global interrupt is set and the TWIE bit is set, a TWI interrupt will be generated, and the MCU will execute the TWI interrupt service routine. When the TWINT flag is set, the low level of the SCL signal will be stretched. The TWINT flag can only be cleared by writing "1" to this bit. Hardware will not automatically clear this bit even if an interrupt service routine is executed. Also note that clearing this bit will immediately enable TWI operation. Therefore, accesses to the TWAR, TWAMR, TWSR, and TWDR registers are completed before clearing the TWINT bit.
6	TWEA	TWI enable acknowledge control bit. The TWEA bit controls the acknowledge pulse generation. When the TWEA bit is set to "1" and one of the following conditions is met, a response pulse will be generated on the TWI bus: 1) Receive the slave address of the device; 2) A general call is received when TWGCE is set; 3) A byte of data is received in master receive or slave receive mode. When setting the TWEA bit to "0", the device is temporarily disconnected from the TWI bus. When asserted, the device resumes address recognition again.
5	TWSTA	TWI start status control bit. When the CPU wants to become the master on the TWI bus, it needs to set the TWSTA bit. The hardware will detect if the bus is available and generate a start condition on the bus when the bus is free. When the bus is not idle, TWI will wait until it detects the stop state, and then generate a start state to declare that it wants to be the master. Software must clear the TWSTA bit after sending the start status.

4	TWSTO	TWI stop status control bit. When the TWSTO bit is "1" in master mode, TWI will generate a stop condition on the bus, and then automatically clear the TWSTO bit. In slave mode, setting the TWSTO bit enables the TWI to recover from an error state. At this time, no stop state will be generated, and it will only return TWI to a defined unaddressed slave mode, and release the SCL and SDA signal lines to a high-impedance state.
3	TWWC	TWI write conflict flag. If TWINT flag is low, writing the TWDR register will set the TWWC flag. If TWINT flag is high, writing the TWDR register will clear the TWWC flag.
2	TWEN	TWI enable control bit. The TWEN bit enables TWI operation and activates the TWI interface. When setting the TWEN bit to "1", SCL and SDA pins are connected to TWI control I/O. When setting the TWEN bit to "0", the TWI interface module is turned off, and all transmissions are terminated, including ongoing operations.
1	-	Reserved.
0	TWIE	TWI interrupt enable control bit. When the TWIE bit is set to "1" and the global interrupt is set, as long as the TWINT flag is high, the TWI interrupt request will be activated.

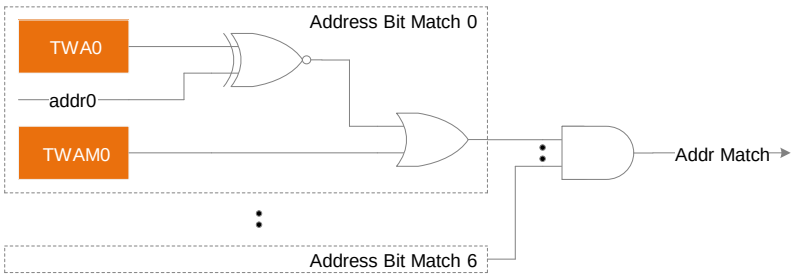
TWAMR – TWI address mask register

TWAMR – TWI address mask register								
Address: 0xBD						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	TWAR6	TWAR5	TWAR4	TWAR3	TWAR2	TWAR1	TWAR0	TWGCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7:1	TWAM[6:0]	TWI address mask control bit. TWAM is a 7-bit TWI slave address mask control. Each bit of TWAM is used to mask (disable) the corresponding address bit in TWAR. When the mask bit is set, the address match logic ignores the result of comparing the received address bits with the corresponding bits in TWA. The following figure shows the details of the address matching logic.
0	-	Reserved.

TWI address matching logic

The following figure shows the TWI address matching logic block diagram:



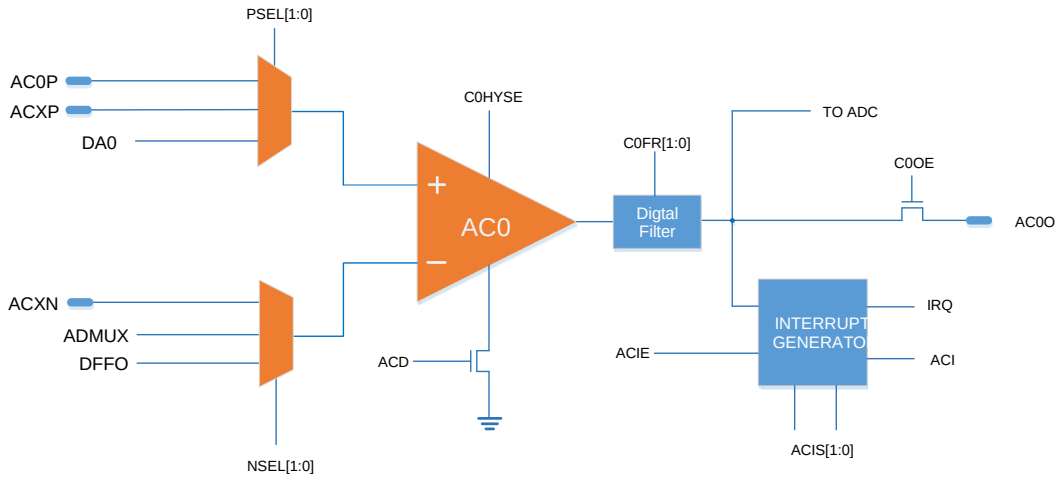
Analog Comparator 0 (AC0)

- ❑ 10mV comparison accuracy
- ❑ Factory Offset Calibration
- ❑ Support 3 off-chip analog inputs
- ❑ Support multiplexed input for ADC (ADMUX)
- ❑ Supports internal differential amplifier input (DFFO)
- ❑ Support internal 8-bit DAC input (DAO)
- ❑ Programmable output digital filter control

Overview

The analog comparator compares the levels of the positive terminal and the negative terminal of the input comparator. When the voltage of the positive terminal is higher than the voltage of the negative terminal, the output ACO of the analog comparator is set. When the level of ACO changes, the edge of the signal can be used to trigger an interrupt. The output signal ACO can also be used to trigger the input capture of timer counter 1 and control the PWM output generated by the timer.
LGT8FX8P integrates analog comparator AC0, including a multi-channel analog input selector. The input source of the positive and negative terminals of the comparator can be selected from an external port or from a variety of internally generated reference sources. The analog comparator itself supports offset calibration, which can ensure the consistency of the comparator's work. The comparator supports an optional hardware hysteresis function to improve the stability of the comparator output. At the same time, a hardware-programmable digital filter is integrated at the output of the comparator, and an appropriate filter setting can be selected according to the application requirements to obtain a more stable comparison output.
The output status of the comparator can be read directly through the register, and an interrupt request can also be generated to realize a more efficient real-time event capture function. The output of the comparator can also be directly output to the external IO port.

The block diagram of op amp/analog comparator 0 is shown in the figure below.



Functional diagram of analog comparator 0

Analog comparator input

Both inputs of the analog comparator support a variety of selectable input sources. There are three optional input channels for the positive terminal:

1. External independent analog input AC0P
2. Analog Comparator 0/1 Common Analog Input ACXP
3. Output DAO of internal 8-bit DAC

The selection of the input source is jointly controlled by the C0BG bit in the control status register C0SR and the C0PS0 bit in the C0XR register. For details, please refer to the register description in this chapter.
AC0P is a dedicated positive mode input channel for AC0. Note that the pin positions of AC0P are slightly different in different packages. The AC0P of the QFP48 package is an independent port. QFP32 encapsulates this AC0P port and PD6 in parallel to one port.
ACXP is the common positive input of comparator 0/1. LGT8FX8P has two analog comparators inside, and ACXP is connected to the positive terminal multiplexing selector of the two comparators at the same time, which is convenient for realizing the cooperative work of the two comparators.
DAO comes from the output of the internal 8-bit DAC. The reference source of the DAC can be selected from the system power supply, an internal reference or an input from an external reference. For the configuration of DAC, please refer to the relevant chapters of DAC.

C0BG	C0PS0	AC0 positive input source
0	0	AC0P
0	1	ACXP
1	0	DAO
1	1	Close the comparator positive input channel

The negative input can also choose from three different analog inputs:

1. Comparator 0/1 common analog input ACXN
2. ADC multiplexer output ADMUX
3. Internal differential amplifier output DFFO

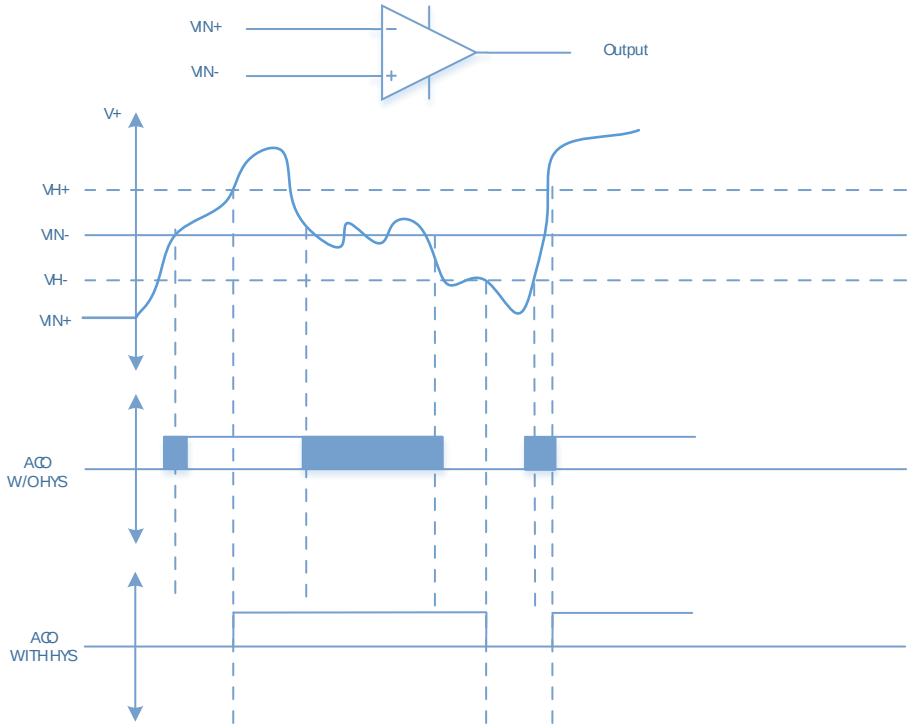
The comparator negative input channel selection is controlled by the CME00/01 bits in the ADCSRB register from the ADC module.
When the negative terminal input of the comparator is selected as ADMUX, the analog input channel needs to be selected through the CHMUX bit of the ADMUX register of the ADC module. In this mode, the input of the comparator can be expanded more flexibly.
ACXN is the common negative terminal input of comparator 0/1, which is convenient to realize the cooperative work of comparator 0/1; DFFO comes from the internal differential amplifier output. The differential amplifier has optional x1/x8/x16/x32 gain control, which can realize the detection and measurement of small signals.

CME01	CME00	AC0 negative input source
0	0	ACXN
0	1	ADMUX
1	0	DFFO
1	1	Close the comparator negative input channel

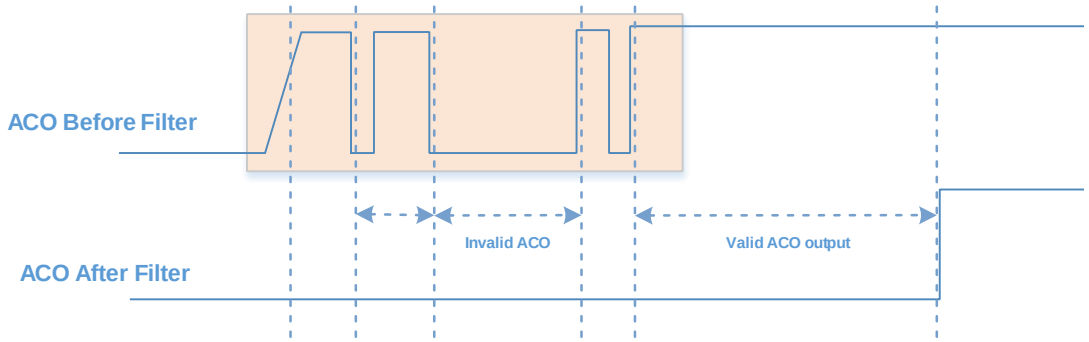
Comparator output filtering

A controllable hysteresis is internally supported on the comparator output. The user can enable the hysteresis circuit through the C0HYSE bit of the C0XR register. The hysteresis circuit can eliminate the unstable state during the state change process of the comparator to achieve the output filter function.
It is recommended that the user turn on the hysteresis circuit to obtain a stable comparator output when using the comparator.
As shown in the figure below, the hysteresis circuit is placed between the analog output and the digital output of the comparator. When the input voltage VIN+ of the positive terminal of the comparator is greater than (VIN- + VH+), the output of the comparator COUT is high; when the voltage of VIN+ is less than (VIN- - VH-), the output of the comparator is low.
The hysteresis circuit avoids the jitter caused by the circuit itself when the voltage at the positive terminal of the comparator is close to the voltage at the negative terminal.

Comparator hysteresis voltage and comparator output relationship diagram:



Although the hysteresis circuit is very effective in suppressing the voltage ripple close to the threshold of the comparator, in the actual application environment, the input signal will be interfered with varying degrees. Strong interference may cause the input level to rise instantaneously, which exceeds the threshold range of the hysteresis circuit and cannot be effectively suppressed. LGT8FX8P integrates a programmable digital filter at the output of the comparator, which can filter out the influence of transient interference on the output of the comparator. The digital filter can select an appropriate filter time width according to the application requirements. Only when the output of the comparator is stable and continues to meet the filter time limit, the filter circuit updates the output of the comparator.
So as to achieve a more stable output result.



Comparator Output Filter Timing

The digital filtering of AC0 is controlled by the C0FEN and C0FS bits of the C0XR register. For the specific setting method, please refer to the register definition part of this chapter.

Comparator output with PWM control

LGT8FX8P supports multi-channel PWM output, and the PWM signal can be used with the comparator module. The output of the comparator can be used to directly turn off the PWM signal, so as to realize a more flexible PWM protection scheme.
For the control related to PWM output, please refer to the relevant part of the timer chapter.

Registers definition

C0SR – AC0 control and status register

C0SR – AC0 control and status register								
Address: 0x50					Default value: 0x80			
Bit	7	6	5	4	3	2	1	0
Name	C0D	C0BG	C0O	C0I	C0IE	C0IC	C0IS1	C0IS0
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	C0D	Analog comparator disable bit. When setting the C0D bit to "1", the analog comparator is turned off. When setting the COD bit to "0", the analog comparator is enabled.
6	C0BG	Analog comparator 0 positive input source selection bit. C0BG and the C0PS0 bit of the C0XR register jointly set the positive input source of AC0, {C0BG, C0PS0} = 00 = AC0P as positive input 01 = ACXP as positive input 10 = Output of internal DAC as positive input 11 = Turn off the positive input source of AC0
5	C0O	Output status bit for the analog comparator. The output of the analog comparator is directly connected to the C0O bit after synchronization. Software can read the value of the COO bit to obtain the output value of the analog comparator.
4	C0I	Interrupt flag bit for the analog comparator. The C0I bit is set when an output event from the analog comparator triggers the interrupt mode defined by the C0IS bit. When the interrupt enable bit C0IE is "1" and the global interrupt is set, an interrupt is generated. When the analog comparator interrupt service routine is executed, C0I will be automatically cleared, or writing "1" to the C0I bit can also clear this bit.
3	C0IE	Interrupt enable bit for the analog comparator. When the C0IE bit is set to 1 and the global interrupt is enabled, the interrupt of AC0 is enabled. When setting the C0IE bit to 0, the interrupt of AC0 is disabled.
2	C0IC	Analog comparator input capture enable bit • C0IC = 1, the input capture source of timer counter 1 is from the output of analog comparator. • C0IC = 0, the input capture source of timer counter 1 comes from external pin ICP1.
1	C0IS1	Analog comparator interrupt mode control high.
0	C0IS0	Analog comparator interrupt mode control low. C0IS0 and C0IS1 together form C0IS[1:0], which is used to control the interrupt trigger mode of the analog comparator.

C0IS[1:0]	Interrupt mode
00	The rising or falling edge of the ACO is triggered
01	Reserved.
10	The falling edge of the ACO is triggered
11	The rising edge of the ACO is triggered

ADCSRB – ADC control and status register B

ADCSRB – ADC control and status register B								
Address: 0x7B							Default value: 0x00	
Bit	7	6	5	4	3	2	1	0
Name	CME01	CME00	CME11	CME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	CME01	AC0 negative input selection, CME0 = {CME01, CME00} = 00: The external port ACXN is used as the negative input of AC0 01: ADC multiplexing output as AC0 negative input 10: Differential amplifier output as AC0 negative input 11: Close the negative input source of AC0
6	CME00	
5	CME11	AC1 negative input selection, CME1 = {CME11, CME10} = 00: The external port ACXN is used as the negative input of AC1 01: The external port AC1N is used as the negative input of AC1 10: ADC internal 1/5 divided voltage as AC1 negative terminal input 11: The output of the differential op amp is used as the negative input of AC1
4	CME10	
3	ACHS	AC trigger source channel selection 0 – AC0 output as trigger source for ADC automatic conversion 1 – AC1 output as trigger source for ADC automatic conversion
2:0	ADTS	See ADC register description.

C0XR – AC0 auxiliary control register

C0XR – AC0 auxiliary control register								
Address: 0x51				Default value: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	-	C0OE	C0HYSE	C0PS0	C0WKE	C0FEN	C0FS1	C0FS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	-	Reserved.
6	C0OE	Enable control of AC0 comparator output to external port C0OE = 1, AC0 comparator output to external port PD2 C0OE = 0, disable comparator output to external port
5	C0HYSE	AC0 output hysteresis function enable control. 1 = Enable output hysteresis 0 = output hysteresis disabled
4	C0PS0	AC0 positive terminal input source selection low. C0PS0 and C0BG jointly control the positive input source of AC0, please refer to the C0SR register definition.
3	C0WKE	AC0 is used to enable control of sleep wake-up. 1 = Enables the wake-up function of the comparator output

		0 = Comparator output wake-up disabled
2	C0FEN	Comparator digital filter enable control. 1 = Enable digital filter 0 = disable digital filter
1:0	C0FS[1:0]	Comparator digital filter width setting 00 = Off 01 = 32us 10 = 64us 11 = 96us

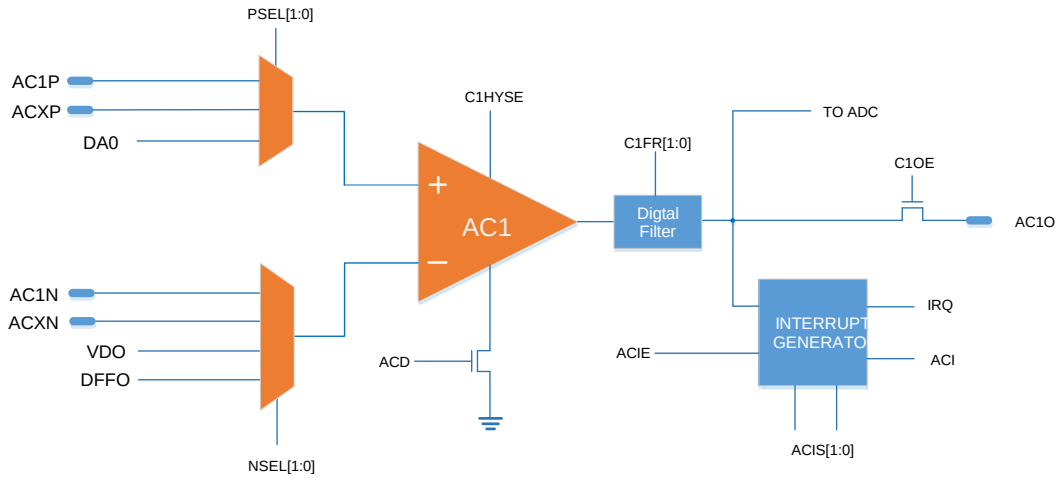
Analog Comparator 1 (AC1)

- ❑ 10mV comparison accuracy
- ❑ Factory Offset Calibration
- ❑ Support 4 off-chip analog inputs
- ❑ Support internal 1/5 voltage divider input (VDO)
- ❑ Supports internal differential amplifier input (DFFO)
- ❑ Support internal 8-bit DAC input (DAO)
- ❑ Programmable Output Filter Control

Overview

The analog comparator compares the levels of the positive terminal and the negative terminal of the input comparator. When the voltage of the positive terminal is higher than the voltage of the negative terminal, the output ACO of the analog comparator is set. When the level of ACO changes, the edge of the signal can be used to trigger an interrupt. The output signal ACO can also be used to trigger the input capture of timer counter 1 and control the PWM output generated by the timer.
LGT8FX8P integrates analog comparator AC1, including a multi-channel analog input selector. The input source of the positive and negative terminals of the comparator can be selected from an external port or from a variety of internally generated reference sources. The analog comparator itself supports offset calibration, which can ensure the consistency of the comparator's work. The comparator supports an optional hardware hysteresis function to improve the stability of the comparator output. At the same time, a hardware-programmable digital filter is integrated at the output of the comparator, and an appropriate filter setting can be selected according to the application requirements to obtain a more stable comparison output.
The output status of the comparator can be read directly through the register, and an interrupt request can also be generated to realize a more efficient real-time event capture function. The output of the comparator can also be directly output to the external IO port.

The block diagram of the analog comparator 1 is shown in the figure below.



Schematic diagram of analog comparator 1 module structure

Analog comparator input

Both inputs of the analog comparator support a variety of selectable input sources. There are three optional input channels for the positive terminal:

1. External independent analog input AC1P
2. Analog Comparator 0/1 Common Analog Input ACXP
3. Output DAO of internal 8-bit DAC

The selection of the input source is jointly controlled by the C1BG bit in the control status register C1SR and the C1PS0 bit in the C1XR register. For details, please refer to the register description in this chapter.
AC1P is the positive mode input channel dedicated to AC1.
ACXP is the common positive input of comparator 0/1. LGT8FX8P has two analog comparators inside, and ACXP is connected to the positive terminal multiplexing selector of the two comparators at the same time, which is convenient for realizing the cooperative work of the two comparators.
DAO comes from the output of the internal 8-bit DAC. The reference source of the DAC can be selected from the system power supply, an internal reference or an input from an external reference. For the configuration of DAC, please refer to the relevant chapters of DAC.

C1BG	C1PS0	AC1 positive input
0	0	AC1P
0	1	ACXP
1	0	DAO
1	1	Comparator positive input off

The negative input can also choose 4 different analog inputs:

1. The external analog input AC1N is used as the negative terminal input of AC1
2. Comparator 0/1 common negative terminal input ACXN
3. The output of the 1/5 voltage divider inside the ADC is used as the negative input of AC1
4. The internal differential amplifier outputs DFFO as the negative input of AC1

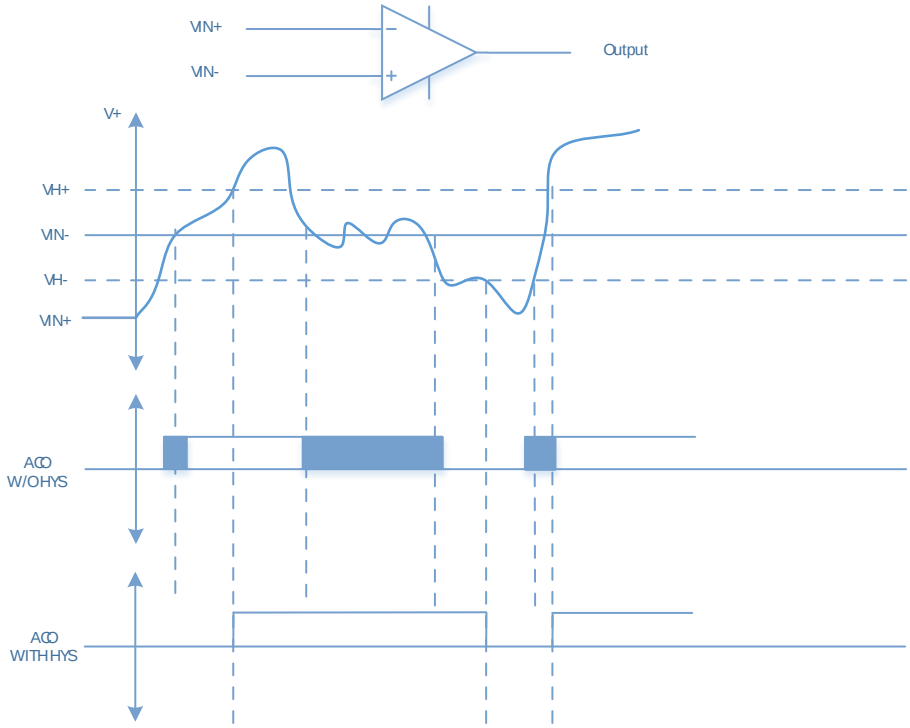
The comparator negative input channel selection is controlled by the CME11/10 bits in the ADCSRB register from the ADC module.
When the negative terminal input of the comparator is selected as the output of the multi-channel voltage divider inside the ADC, it is necessary to select the input reference source of the multi-channel voltage divider through the VDS bit of the ADCSRC register of the ADC module.
ACXN is the common negative terminal input of comparator 0/1, which is convenient to realize the cooperative work of comparator 0/1; DFFO comes from the internal differential amplifier output. The differential amplifier has optional x1/x8/x16/x32 gain control, which can realize the detection and measurement of small signals.

CME11	CME10	AC1 negative input
0	0	ACXN
0	1	AC1N
1	0	VDO
1	1	DFFO

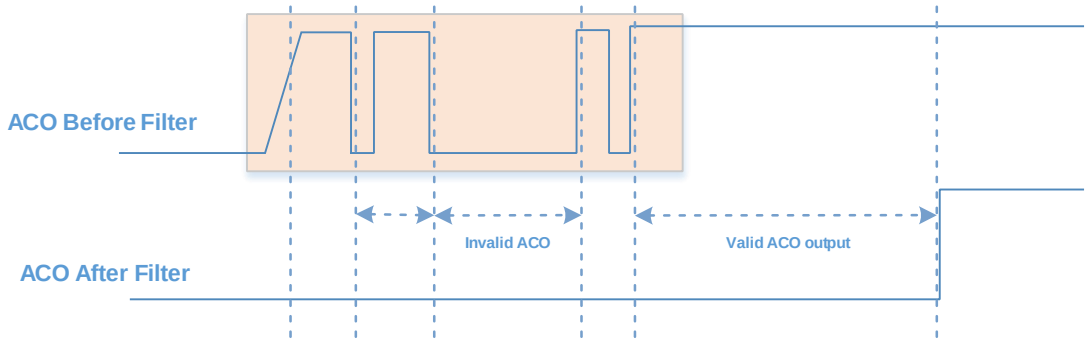
Comparator output filtering

A controllable hysteresis is internally supported on the comparator output. The hysteresis circuit can be enabled by the user through the C1HYSE bit of the C1XR register. The hysteresis circuit can eliminate the unstable state during the state change process of the comparator to achieve the output filter function.
It is recommended that the user turn on the hysteresis circuit to obtain a stable comparator output when using the comparator.
As shown in the figure below, the hysteresis circuit is placed between the analog output and the digital output of the comparator. When the input voltage VIN+ of the positive terminal of the comparator is greater than (VIN- + VH+), the output of the comparator COUT is high; when the voltage of VIN+ is less than (VIN- - VH-), the output of the comparator is low.
The hysteresis circuit avoids the jitter caused by the circuit itself when the voltage at the positive terminal of the comparator is close to the voltage at the negative terminal.

Comparator hysteresis voltage and comparator output relationship diagram:



Although the hysteresis circuit is very effective in suppressing the voltage ripple close to the threshold of the comparator, in the actual application environment, the input signal will be interfered with varying degrees. Strong interference may cause the input level to rise instantaneously, which exceeds the threshold range of the hysteresis circuit and cannot be effectively suppressed. LGT8FX8P integrates a programmable digital filter at the output of the comparator, which can filter out the influence of transient interference on the output of the comparator. The digital filter can select an appropriate filter time width according to the application requirements. Only when the output of the comparator is stable and continues to meet the filter time limit, the filter circuit updates the output of the comparator.
So as to achieve a more stable output result.



Comparator Output Filter Timing

The digital filtering of AC1 is controlled by the C0FEN and C1FS bits of the C1XR register. For the specific setting method, please refer to the register definition part of this chapter.

Comparator output with PWM control

LGT8FX8P supports multi-channel PWM output, and the PWM signal can be used with the comparator module. The output of the comparator can be used to directly turn off the PWM signal, so as to realize a more flexible PWM protection scheme.
For the control related to PWM output, please refer to the relevant part of the timer chapter.

Registers definition

C1SR – AC1 control and status register

C1SR – AC1 control and status register								
Address: 0x2F					Default value: 0x80			
Bit	7	6	5	4	3	2	1	0
Name	C1D	C1BG	C1O	C1I	C1IE	C1IC	C1IS1	C1IS0
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	C1D	Analog comparator disable bit. When setting the C1D bit to "1", the analog comparator is turned off. When setting the C1D bit to "0", the analog comparator is turned on.
6	C1BG	Analog comparator 1 positive input source selection bit. C1BG and the C1PS0 bit of the C1XR register jointly set the positive input source of AC1, {C1BG, C1PS0} = 00 = AC1P as positive input 01 = ACXP as positive input 10 = Output of internal DAC as positive input 11 = Turn off the positive input source of AC1
5	C1O	Output status bit for the analog comparator. The output of the analog comparator is directly connected to the C1O bit after synchronization. Software can read the value of the C1O bit to obtain the output value of the analog comparator.
4	C1I	Interrupt flag bit for the analog comparator. The C1I bit is set when an output event from the analog comparator triggers the interrupt mode defined by the C1IS bit. When the interrupt enable bit C1IE is "1" and the global interrupt is set, an interrupt is generated. When the analog comparator interrupt service routine is executed, C1I will be automatically cleared, or writing "1" to the C1I bit can also clear this bit.
3	C1IE	Interrupt enable bit for the analog comparator. When the C1IE bit is set to 1 and the global interrupt is enabled, the interrupt of AC1 is enabled. When setting the C1IE bit to 0, the interrupt of AC1 is disabled.
2	C1IC	Analog comparator input capture enable bit C1IC = 1, the input capture source of TC1 is the output of analog comparator. C1IC = 0, the input capture source of TC1 comes from the external pin ICP1.
1	C1IS1	Analog comparator interrupt mode control high.
0	C1IS0	Analog comparator interrupt mode control low. C1IS0 and C1IS1 together form C1PS[1:0], which is used to control the interrupt trigger mode of the analog comparator.

C1IS[1:0]	Interrupt mode
00	Triggered by rising or falling edge of AC1
01	Reserved.
10	Triggered by falling edge of AC1
11	Triggered by rising edge of AC1

ADCSRB – ADC control and status register B

ADCSRB – ADC control and status register B								
Address: 0x7B	Default value: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	CME01	CME00	CME11	CME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	CME01	AC0 negative input selection, CME0 = {CME01, CME00} 00: The external port ACXN is input as the negative terminal of AC0 01: ADC multiplexed output as AC0 negative input 10: Differential amplifier output as AC0 negative input 11: Turn off the negative input source of AC0
6	CME00	
5	CME11	AC1 negative input selection, CME1 = {CME11, CME10} 00: External port ACXN input as AC1 negative 01: External port AC1N is input as the negative terminal of AC1 10: The ADC internal 1/5 divider is input as the negative terminal of AC1 11: The output of the differential op amp is used as the negative input of AC1
4	CME10	
3	ACHS	AC triggers source channel selection 0 – AC0 output as trigger source for ADC automatic conversion 1 – AC1 output as trigger source for ADC automatic conversion
2:0	ADTS	See ADC register description.

C1XR – AC1 auxiliary control register

C1XR – AC1 auxiliary control register								
Address: 0x3A	Default value: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	-	C1OE	C1HYSE	C1PS0	C1WKE	C1FEN	C1FS1	C1FS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	-	Reserved.
6	C1OE	Enable control of AC1 comparator output to external port C1OE = 1, the comparator output of AC1 to the external port PE5 C1OE = 0, disable comparator output to external port
5	C1HYSE	AC1 output hysteresis enable. 1 = Enable output hysteresis 0 = Output hysteresis disabled
4	C1PS0	AC1 positive terminal input source selection low. C1PS0 and C1BG jointly control the positive input source of AC1, please refer to the C1SR register definition
3	C1WKE	AC1 is used for the enable control of sleep wake-up. 1 = Enables the wake-up function of the comparator output

		0 = Turn off wake-up on the comparator output
2	C1FEN	Comparator digital filtering enable control. 1 = Digital filter enabled 0 = Digital filter disabled
1:0	C1FS[1:0]	Comparator digital filter width setting 00 = Off 01 = 32us 10 = 64us 11 = 96us

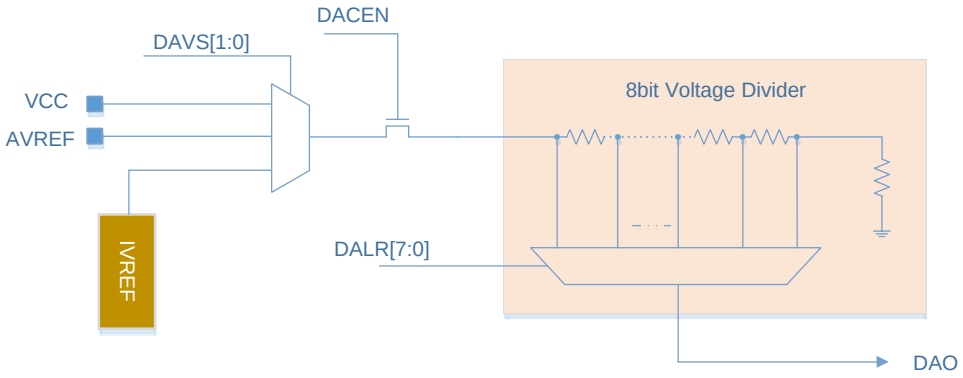
Digital to Analog Converter (DAC)

- ❑ 8-bit digital-to-analog output
- ❑ DAC output can be used as analog comparator reference input
- ❑ Support DAC output to external port (DAO)
- ❑ Optional VCC/AVREF/IVREF divider power supply

Overview

LGT8FX8P integrates an 8-bit programmable digital-to-analog converter (DAC) inside. The reference power input of the DAC can be selected from the system operating power supply, the internal reference voltage source or the AVREF input from the external port of the chip. The output of the DAC can be selected as the input source of the internal comparator AC0/1, and can also be directly output to the external pin of the chip as an external reference.

When the DAC outputs to an external pin, it cannot be used to drive the load directly, and a voltage follower or other similar driving circuit is required. The internal structure of the DAC is shown in the figure below:



Registers definitions

DACON – DAC control register

DACON – DAC control registers								
Address: 0xA0					Default value: 0000_0000			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	DACEN	DAOE	DAVS1	DAVS0
R/W	-	-	-	-	R/W	W/R	R/W	W/R

Bit	Name	Description
7:4	-	Reserved.
3	DACEN	DAC enable control bit 1 = Enable DAC module 0 = Disable DAC module
2	DAOE	DAC output to external port enable control 1 = Enable DAC output to external terminal PD4 0 = Disable DAC output to external port
1	DAVS1	DAC reference voltage source selection bit 1
0	DAVS0	DAC Reference Voltage Source Select Bit 0. [DVS1, DVS0] =

		00 : Voltage source selection system operating voltage VCC 01 : The voltage source is selected as external input AVREF 10 : The voltage source is selected as the internal reference voltage 11 : Turn off the DAC reference source, and also turn off the DAC module

DALR – DAC data register

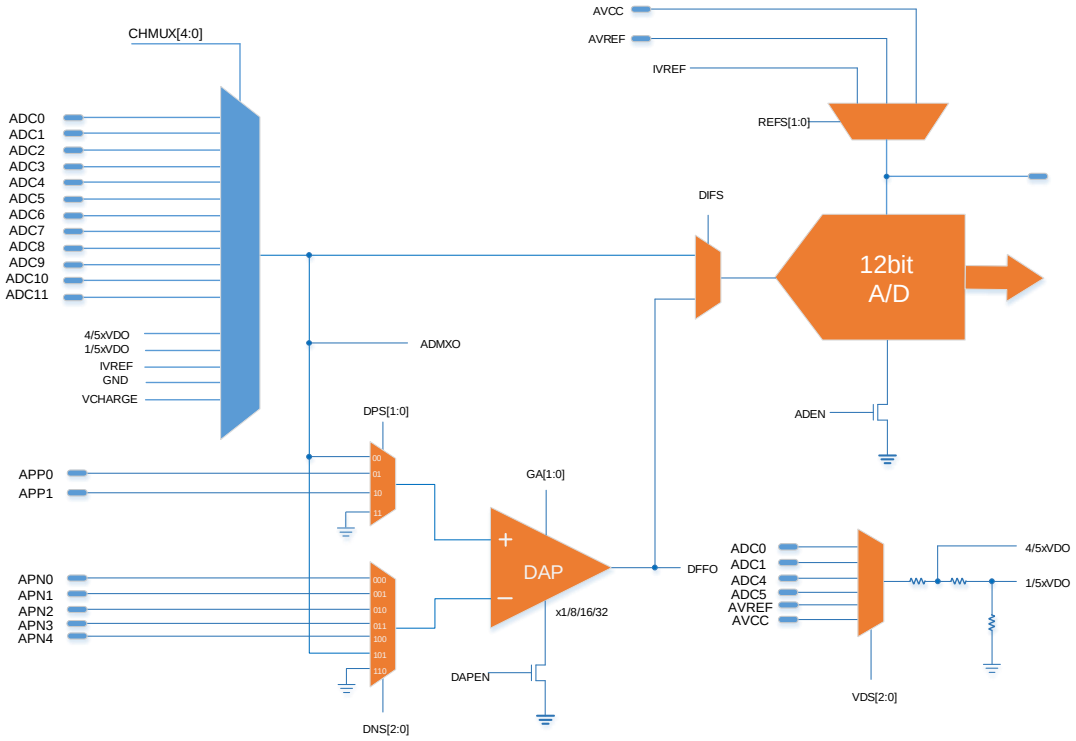
DALR – DAC data register								
Address: 0xA1					Default value: 0000_0000			
Bit	7	6	5	4	3	2	1	0
	DALR[7:0]							
R/W	W/R							

Bit	Name	Description
7:0	DALR	DAC data register, set the output voltage of DAC mode The relationship between DAC output voltage and DALR: ***VDAO = VREF*(DALR + 1)/256*** Where: • **VDAO** is the DAC output analog voltage • **VREF** is the DAC reference voltage source, selected by the DAVS bit of the DACON register

12-bit Analog-to-Digital Converter (ADC)

- ❑ 12-bit resolution, ±1LSB for DNL, ±1.5LSB for INL
- ❑ Sampling rate up to 500KSPS at highest resolution
- ❑ 12 multiplexed single-ended input channels
- ❑ Multiple Input Programmable Gain Differential Amplifier Channels
- ❑ Input voltage range is 0-VCC
- ❑ Internal 1.024V/2.048V/4.096V reference voltage
- ❑ Support AVCC and external reference voltage input
- ❑ Internal multi-input 1/5, 4/5 voltage divider circuit
- ❑ Support offset calibration in positive and negative directions
- ❑ Auto-start conversion trigger mode based on interrupt source
- ❑ Supports automatic channel detection for overflow/underflow
- ❑ Conversion results support optional alignment modes
- ❑ End of Conversion Interrupt Request

Overview



ADC block diagram

The analog-to-digital converter is a 12-bit successive approximation ADC. The ADC is connected with a 17-channel analog multiplexer, which can sample and convert 12 analog inputs from external ports of the chip and internal voltage sources of 5 channels. The ADC integrates a differential operational amplifier with a programmable gain of x1/x8/x16/x32. The amplifier input can come from an external port or the output of the ADC multiplexer. The result of the differential op amp can be used as an analog input to the ADC.

The internal analog input sources of the ADC include multi-channel input voltage dividers inside the ADC; internal reference voltage sources; internal analog reference grounds and analog outputs from the touch key module. The internal multiple input voltage divider outputs 4/5, 1/5 two voltages at the same time; the input of the voltage divider can be selected from the level of the external port or from the system power supply.

The ADC supports offset calibration. The process of offset calibration is controlled by software. Offset calibration includes positive and negative calibration quantities. After offset calibration is enabled, the ADC controller will automatically use the positive and negative calibration values to calibrate the ADC sampling results.
For the method of offset calibration, please refer to the relevant part of this chapter.

Operation of the ADC

The ADC converts the input analog voltage into a 12-bit digital quantity through successive approximation methods. The minimum value represents GND, and the maximum value represents the reference voltage minus 1LSB. The reference voltage source can be the power supply voltage AVCC of the ADC, the external reference voltage AVREF or the internal reference voltage of 1.024V/2.048V, which is selected by writing the REFS bit of the ADMUX register.

The analog input channel can be selected by writing the CHMUX bit of the ADMUX register. Any ADC input pin, external reference voltage pin, and internal reference voltage source can be used as ADC single-ended input. The ADC input channel can be switched to the internal differential amplifier by setting DIFS in the ADTMR register. The relative input source and gain of the differential amplifier can be set by the DAPCR register.

The ADC can be started by setting the ADEN bit of the ADCSRA register. When ADEN is cleared to zero, the ADC does not consume power, so it is recommended to turn off the ADC before entering sleep mode.

The ADC conversion result is 12 bits, stored in the ADC data registers ADCH and ADCL. The conversion result is right-aligned by default, but it can be changed to left-aligned by setting the ADLAR bit of the ADMUX register.

If the conversion result is set to be left-aligned, and only 8-bit conversion precision is required, it is enough to read ADCH. Otherwise, read ADCL first, and then read ADCH to ensure that the content in the data register is the result of the same conversion. Once ADCL is read, the data registers ADCL and ADCH are latched, and the conversion result can be updated to the data registers ADCL and ADCH after reading ADCH.

An interrupt can be triggered at the end of the ADC conversion. Even if the end of conversion occurs between reading ADCL and ADCH, the interrupt will still trigger.

Start a conversion

Writing "1" to the ADC start conversion bit ADSC bit can start a single conversion. This bit remains high during a conversion until cleared by hardware after the conversion is complete. If the channel is changed during conversion, the ADC will complete the conversion before changing the channel.

There are different trigger sources for ADC conversions. Setting the ADC automatic trigger enable bit ADATE in the ADCSRA register can enable automatic triggering. The trigger source can be selected by setting the ADC trigger selection bit ADTS in the ADCSRB register. When the selected trigger signal has a rising edge, the ADC prescaler is reset and conversion begins. This provides a way to initiate transitions at regular intervals. Even if the trigger signal still exists after the conversion, it will not start a new conversion. If the trigger signal generates another rising edge during the conversion, this rising edge will also be ignored. Even if the specific interrupt is disabled or the global interrupt enable bit is "0", its interrupt flag will still be set. This allows a conversion to be triggered without generating an interrupt. But in order to trigger a new conversion on the next interrupt event, the interrupt flag must be cleared.

Using the ADC interrupt flag as a trigger source, the next ADC conversion can be started immediately after the current conversion is completed. Afterwards, the ADC works in the continuous conversion mode, continuously sampling and updating the ADC data register. The first conversion is initiated by writing "1" to the ADSC bit in the ADCSRA register. In this mode, subsequent ADC conversions do not depend on whether the ADC interrupt flag ADIF is set.

The conversion is enabled by writing "1" to the ADSC bit of the ADCSRA register. In this mode, subsequent ADC conversions do not depend on whether the ADC interrupt flag ADIF is set.

If autotrigger is enabled, ADSC setting the ADCSRA register will initiate a single conversion. The ADSC flag can also be used to detect whether a conversion is in progress. Regardless of how the conversion is initiated, ADSC is always "1" during the conversion.

Prescaler and ADC conversion timing

Under default conditions, the successive approximation circuit requires an input clock from 300KHz to 3MHz for maximum accuracy. If the required conversion accuracy is lower than 12 bits, the frequency of the input clock can be higher than 3MHz to achieve a higher sampling rate.

The ADC module includes a prescaler that can generate an acceptable ADC input clock from the system clock. The prescaler is set by the ADPS bits of the ADCSRA register. Setting ADEN in the ADCSRA register will enable the ADC and the prescaler will start counting. As long as the ADEN bit is "1", the prescaler will continue to count until ADEN is cleared.

After the ADSC of the ADCSRA register is set, the single-ended conversion starts on the rising edge of the next ADC clock cycle. A normal conversion requires 15 ADC clock cycles. After the ADC is enabled (ADEN in the ADCSRA register is set), 50 ADC input clock cycles are required to initialize the analog circuit before the first conversion can be performed effectively.

During the ADC conversion, the sampling and holding starts 1.5 ADC input clocks after the start of the conversion, and the result output of the first ADC conversion occurs 14.5 ADC input clocks after the start. After the conversion, the ADC result is sent to the ADC data register, and the ADIF flag is set. ADSC is also cleared. The software can then set the ADSC flag again or trigger it automatically to start a new conversion.

Sampling channels and reference voltage

The MUX and REFS in the ADMUX register realize single buffering through the temporary register. The CPU can perform random access to temporary registers. Before the conversion starts, the CPU can configure the channel and reference source selection at any time. In order to ensure that the ADC has sufficient sampling time, once the conversion starts, the configuration of channel and reference source selection is not allowed. After the conversion is completed (ADIF of the ADCSRA register is set), the channel and reference source selection will be updated. The conversion starts at the rising edge of the next ADC input clock after ADSC is set. Therefore, it is recommended that users do not operate ADMUX to select a new channel and reference source within one ADC input clock cycle after ADSC is set.
When using automatic triggering, the time at which the triggering event occurs is indeterminate. To control the effect of new settings on conversions, special care must be taken when updating the ADMUX registers. If both ADATE and ADEN are set, the interrupt time can occur at any time, thus automatically triggering and starting the conversion of the ADC. If the contents of the ADMUX registers are changed during this time, the user cannot tell whether the next conversion is based on the old configuration or the new configuration. Users are advised to update ADMUX at the following safe times:

1) ADATE or ADEN bit is "0";
2) During conversion, but at least one ADC input clock cycle after the trigger event;
3) After the end of the conversion, but before the trigger source's interrupt flag is cleared.

If ADMUX is updated in any of the cases mentioned above, the new configuration will take effect until the next conversion.
When selecting the ADC input channel, it should be noted that the channel is selected before starting the conversion, and a new analog input channel can be selected after one ADC clock cycle after ADSC is set, but the easiest way is to wait until the conversion is completed before changing the channel. .

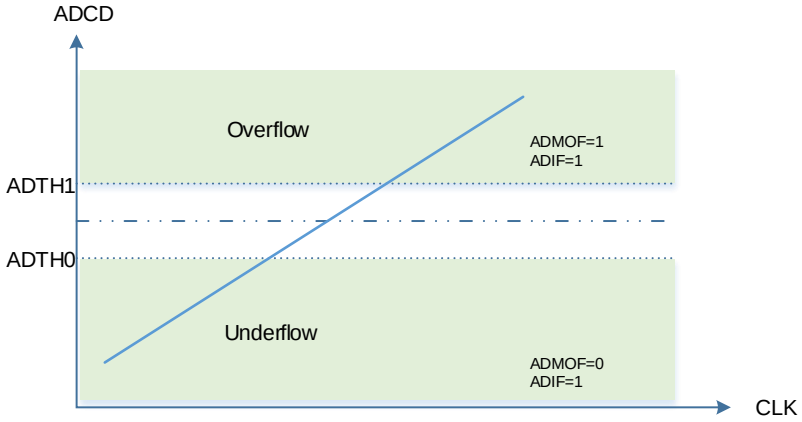
The reference voltage source Vref of the ADC reflects the conversion range of the ADC. If the single-ended channel level exceeds Vref, the conversion result will be close to the maximum value of 0xFFF. Vref can be AVCC, the voltage of the external AREF pin, and the internal reference voltage source.

Notes on using internal reference (1.024V/2.048V/4.096V):

After the chip is powered on, the internal reference is calibrated to 1.024V by default. If the user uses the internal reference of 1.024V, it can be used directly without other operations. But if you need to use the internal reference voltage of 2.048V or 4.096V, you need to update the calibration value of the internal reference by yourself. The calibration value of 2.048V/4.096V is loaded into the register VCAL2/3 (0xCE/0xCC) after power-on. When the program is initialized, the value of VCAL2/3 is read and written into the VCAL (0XC8) register to complete the calibration .

Automatic Channel Monitoring

The automatic channel monitoring mode is used to monitor the voltage change of the selected ADC input channel in real time. The software enables the automatic channel monitoring function by setting the AMEN bit of the ADCSRC register. The ADC automatically converts the voltage of the selected channel. When the conversion result is outside the given overflow range, the ADC interrupt flag (ADIF) will be set, and At the same time stop automatic monitoring. Software can respond to overflow events by means of interrupts or queries. The AMOF bit of the ADMSC register is used to indicate the type of overflow event. The ADIF flag bit is automatically cleared by hardware after responding to the interrupt reset; in the query mode, it can be cleared by software writing 1. Auto-monitor mode can only be re-enabled when ADIF is cleared and by setting the AMEN bit in the ADCSRC register.



To overcome the instability of a single ADC conversion result, auto-detection supports a configurable digital filtering function. Digital filtering detects the continuous conversion results, and only when a consistent result is obtained within a limited number of continuous conversions, an overflow event is triggered. The number of consecutive conversions can be set by the AMFC[3:0] bits of the ADMSC register.

The automatic channel monitoring function is controlled by the AMEN bit of the ADCSRC register. Register ADT0 is used to set the threshold of underflow; ADT1 is used to set the threshold of overflow. ADT0/1 are 16-bit registers. After the software sets the AMEN bit, it will immediately stop the current ADC conversion action, reset the ADC control status, and then enter the automatic conversion mode.

Before starting the automatic channel detection mode, it is necessary to set the detected channel and other related configurations. Software can disable auto-detect mode at any time by clearing the AMEN register.

Multiple input voltage divider circuit (VDS)

The ADC contains a multi-input voltage divider module. The divided voltage input voltage source can be selected from external ADC input channel (ADC0/1/4/5), external reference AVREF or analog working power supply. The voltage divider module simultaneously outputs two voltages of 4/5 and 1/5 to the internal 12 and 13 input channels of the ADC respectively. Among them, 4/5 is mostly used for ADC offset calibration; 1/5 is mostly used for power supply voltage detection and other similar applications in addition to internal offset calibration. The functions related to the voltage divider circuit are mainly controlled by the ADCSRD register.

ADC offset calibration

Due to the deviation of the manufacturing process and the inherent characteristics of the circuit structure, the internal comparator circuit of the ADC will generate different degrees of offset errors. Therefore, compensating the offset voltage is very critical for generating a high-precision ADC conversion structure.
The ADC inside the LGT8FX8P chip supports offset voltage test-related interfaces, and can complete offset measurement and calibration with the cooperation of software.

The principle of offset calibration:
Offset calibration is mainly to test the ADC conversion results in both positive and negative directions by changing the input polarity of the internal comparator. Since the offset voltage in the positive and negative directions also shows two polarities, an intermediate offset error value can be obtained by subtracting the two conversion results. In normal application, the conversion result can be adjusted accordingly according to the offset voltage.

Offset Calibration Process:

- Configure the VDS module and select the VDS input source as an analog power supply (AVCC)
- The reference voltage of the ADC is selected as the analog power supply (AVCC)
- ADCSRC[SPN] = 0, ADC reads 4/5VDO channel, and the conversion value is recorded as PVAL
- ADCSRC[SPN] = 1, ADC reads 4/5VDO channel, conversion value record bit NVAL
- Store the value (NVAL – PVAL) >>1 into the OFR0 register
- ADCSRC[SPN] = 1, ADC reads 1/5VDO channel, and the conversion result is recorded as NVAL
- ADCSRC[SPN] = 0, ADC reads 1/5VDO channel, and the conversion result is recorded in PVAL
- Store value (NVAL – PVAL) >> 1 into OFR1 register
- Set ADCSRC[OFEN]=1 to enable offset compensation function

Special Note: Since the offset error has positive and negative directions, the above data and operations are all signed operations.

It is necessary to change the ADC related configuration during the offset calibration process, so it is recommended that the offset calibration be completed before the normal configuration. In order to improve the calibration accuracy, it is recommended to sample multiple times for filtering when the ADC reads the channel conversion.
Offset Calibration After OFR0/1 configuration is complete, enable automatic offset compensation through the OFEN bit. After normal conversion in the future, the ADC control will automatically use OFR0/1 for compensation according to the ADC conversion result.

ADC Dynamic Calibration

The offset calibration method described above is based on the offset under a test environment and test input. When the system environment changes, the offset of the ADC will also change accordingly. Therefore, if real-time calibration compensation can be realized, it is very important to overcome the performance difference of the device caused by the change of the working environment and improve the measurement accuracy of the ADC.
A suggested algorithm is provided here. Based on the principle of the offset calibration algorithm, it can dynamically compensate the offset error caused by the working environment and obtain consistent and accurate test results.
This method eliminates the need to calculate the offset voltage and enable offset compensation (OFEN). The algorithm only needs to control the polarity of the ADC conversion through the SPN, and sample two measurement results under different SPNs. In the two results, the error introduced by the offset is shown in both positive and negative directions, so we can simply add the average The method offsets the error caused by detuning.
We assume that when the ADC is converting, the test error introduced by the offset is VOFS, so the SPN is controlled to perform two consecutive ADC conversions, and the obtained ADC conversion results can be expressed as:

SPN = 1 时, $V_{ADC1} = V_{REL} + V_{OFS1}$
SPN = 0 时, $V_{ADC0} = V_{REL} – V_{OFS0}$

The effect of VOFS on the actual sampled input VREL can be eliminated by adding the two measurements together. Due to the matching characteristics of the circuit, VOFS1 and VOFS0 may

not be exactly the same, but the effect of compensating the offset error can still be achieved overall.
Dynamic offset compensation algorithm flow:

1. Initialize ADC conversion parameters according to application needs
2. Set SPN=1, start ADC sampling, and record the ADC sampling result as VADC1
3. Set SPN=0, start ADC sampling, and record the ADC sampling result as VADC2
4. (VADC1 + VADC2) >> 1 is the conversion result of this ADC

In practical applications, this algorithm can be combined with the sampling average algorithm to obtain more ideal results.

Register definitions

List of ADC registers

Register	Address	Default value	Description
ADCL	0x78	0x00	ADC data low byte register
ADCH	0x79	0x00	ADC data high byte register
ADCSRA	0x7A	0x00	ADC control and status register A
ADCSRB	0x7B	0x00	ADC control and status register B
ADMUX	0x7C	0x00	ADC multiplexing control register
ADCSRC	0x7D	0x01	ADC control and status register C
DIDR0	0x7E	0x00	Digital Input Disable Control Reg. 0
DIDR1	0x7F	0x00	Digital Input Disable Control Reg. 1
DAPCR	0xDC	0x00	Differential amplifier control register
OFR0	0xA3	0x00	Offset compensation register 0
OFR1	0xA4	0x00	Offset compensation register 1
ADT0L	0xA5	0x00	Underflow threshold LSB register
ADT0H	0xA6	0x00	Underflow threshold MSB register
ADT1L	0xAA	0x00	Overflow threshold LSB register
ADT1H	0xAB	0x00	Overflow threshold MSB register
ADMSC	0xAC	0x01	Auto status monitoring and control registers
ADCSRD	0xAD	0x00	ADC control and status register D

ADCL – ADC data low byte register

ADCL – ADC data low byte register								
Address: 0x78					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name0	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
Name1	ADC3	ADC2	ADC1	ADC0	-	-	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	Description
7:0	ADC[7:0]/ ADC[3:0]	ADC data low byte register. When the ADLAR bit is "0", the register holds the LSB of the conversion, that is, the ADCL is ADC [7:0], as shown in Name0; When the ADLAR bit is "1", the register value is aligned by the high bits, that is, the upper 4 bits of the ADCL are ADC [3:0], and the lower 4 bits are meaningless, as shown in Name1.

ADCH – ADC data high byte register

ADCH – ADC data high byte register								
Address: 0x79					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name0	-	-	-	-	ADC11	ADC10	ADC9	ADC8
Name1	ADC11	ADC10	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	Description
7:0	ADC[11:8]/ ADC[11:4]	ADC data high byte register. When the ADLAR bit is "0", the storage of the ADC output data in the register is aligned with the lower bits, that is, the lower 4 bits of ADCH are ADC[11:8], and the upper 4 bits are meaningless, as shown in Name0; when the ADLAR bit is When "1", the storage of the ADC output data in the register is aligned with the high bit, that is, ADCH is ADC[11:4], as shown in Name1.

ADCSRA – ADC control and status register A

ADCSRA – ADC control and status register A								
Address: 0x7A					Default value: 0x05			
Bit	7	6	5	4	3	2	1	0
Name	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	1	0

Bit	Name	Description
7	ADEN	ADC enable control bit. When setting the ADEN bit to "1", the ADC is enabled. When setting the ADEN bit to "0", the ADC is disabled.
6	ADSC	ADC starts converting. In one-shot conversion mode, setting ADSC will initiate a conversion. In continuous conversion mode, setting ADSC will initiate the first conversion.
5	ADATE	ADC auto trigger enable control bit. When setting the ADATE bit to "1", the automatic trigger function is enabled. The rising edge of the selected trigger signal initiates a conversion. The selection of the trigger source is controlled by ADTS in the ADCSRB register. When setting the ADATE bit to "0", the automatic trigger function is disabled.
4	ADIF	ADC interrupt flag bit. ADIF is set when the ADC completes a conversion and updates the data register. If the ADC interrupt enable bit ADIE is "1" and the global interrupt is set, the ADC interrupt will be generated. Executing the ADC interrupt will clear the ADIF bit, and it can also be cleared by writing "1" to this bit.
3	ADIE	ADC interrupt enable control bit. When the ADIE bit is set to "1" and the global interrupt is set, the ADC interrupt is enabled. When setting the ADIE bit to "0", the ADC interrupt is disabled.

2:0	ADPS[2:0]	ADC prescaler selection control bits. ADPS selects the system clock to generate the prescaler factor of the ADC clock.

ADPS[2:0]	Prescaler factor
0	2
1	2
2	4
3	8
4	16
5	32 (default)
6	64
7	128

ADCSRB – ADC control and status register B

ADCSRB – ADC control and status register B								
Address: 0x7B						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ACME01	ACME00	ACME11	ACME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	W/O	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	Description
7 6	ACME01 ACME00	Comparator 0 negative input selection: 00: Negative terminal selects external input ACIN0 01: Negative terminal selects ADC multiplexing output 1X: The negative terminal selects the output of op amp 0
5 4	ACME11 ACME10	Comparator 1 negative input selection: 00: Negative terminal selects external input ACIN2 01: Negative terminal selects ADC multiplexing output 1X: The negative terminal selects the output of op amp 1
3	ACTS	AC trigger source channel selection: 0 – AC0 output as trigger source for ADC automatic conversion 1 – AC1 output as trigger source for ADC automatic conversion
2:0	ADTS[2:0]	ADC automatic trigger source selection control bit. When the ADATE bit is set to "1", the automatic trigger function is enabled, and the selection of the trigger source is controlled by ADTS. When setting the ADATE bit to "0", the setting of ADTS is invalid. The rising edge of the interrupt flag on the selected trigger signal initiates a conversion. When switching from a trigger source with the interrupt flag cleared to a trigger source with the interrupt flag set, the trigger signal will generate a rising edge. If ADEN is set at this time, the ADC will also start a conversion. When switching to continuous conversion mode (ADTS=0), the auto-trigger function is disabled.

ADTS[2:0]	Trigger source
0	Continuous conversion mode
1	Comparator 0/1

		2	External interrupt 0
		3	Timer counter 0 compare match
		4	Timer counter 0 overflow
		5	Timer counter 1 compare match B
		6	Timer counter 1 overflow
		7	Timer counter 1 input capture event

ADMUX – ADC multiselect control register

ADMUX – ADC multiselect control register								
Address: 0x7C					Default value: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	REFS1	REFS0	ADLAR	CHMUX4	CHMUX3	CHMUX2	CHMUX1	CHMUX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	Description	
7:6	REFS[1:0]	Cooperate with REFS2 of the ADCSRD register to select the reference voltage source of the ADC. The reference voltage is selected by setting the REFS control bit. If the REFS setting is changed during the conversion, the change will only take effect after the current conversion is completed.	
		REFS2, REFS[1:0]	Reference voltage selection
		0_00	AREF
		0_01	AVCC
		0_10	On-chip 2.048V reference
		0_11	On-chip 1.024V reference
		1_00	On-chip 4.096V reference
5	ADLAR	Conversion result left justification enable control bit. When setting the ADLAR bit to "1", the conversion result is left justified in the ADC data register. When setting the ADLAR bit to "0", the conversion result is right-justified in the ADC data register.	
4:0	CHMUX[4:0]	ADC input source selection control bit.	
		CHMUX[4:0]	single-ended input source / description
		0_0000	PC0
		0_0001	PC1
		0_0010	PC2
		0_0011	PC3
		0_0100	PC4
		0_0101	PC5
		0_0110	PE1
		0_0111	PE3
		0_1001	PC7

External port input

		0_1010	PF0	
		0_1011	PE6	
		0_1100	PE7	
		0_1110	4/5VDO	Internal voltage dividing circuit
		0_1000	1/5VDO	
		0_1101	IVREF	Internal reference
		0_1111	AGND	Simulated
		1_XXXX	DACO	Internal DAC output

ADCSRC – ADC control status register C

ADCSRC – ADC control status register C								
Address: 0x7D				Default value: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	OFEN	-	SPN	AMEN	-	SPD	DIFS	ADTM
R/W	R/W	-	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	Description
7	OFEN	1 = enable offset compensation; 0 = disable offset compensation
6	-	Unimplemented
5	SPN	ADC conversion input polarity control, used only for offset calibration process. Must be cleared normally
4	AMEN	Channel automatic monitoring enable; 1: Enable channel automatic monitoring function 0: Disable channel automatic monitoring function
3	-	Unimplemented
2	SPD	0=ADC low speed conversion mode 1=ADC high-speed conversion mode, only for low impedance analog input
1	DIFS	0 = ADC conversion comes from ADC multiplexer 1 = ADC conversion is from internal differential amplifier
0	ADTM	Test mode, output internal reference voltage from AVREF port

DIDR0 – Digital Input Disable Control Register 0

DIDR0 – Digital input disable control register 0								
Address: 0x7E						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	PE3D	PE1D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
7	PE3D	1=Turn off PE3 digital input
6	PE1D	1=Turn off PE1 digital input
5	PC5D	1=Turn off PC5 digital input
4	PC4D	1=Turn off PC4 digital input

3	PC3D	1=Turn off PC3 digital input
2	PC2D	1=Turn off PC2 digital input
1	PC1D	1=Turn off PC1 digital input
0	PC0D	1=Turn off PC0 digital input

DIDR1 – Digital input disable control register 1

DIDR1 – Digital input disable control register 1								
Address: 0x7F						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	PE7D	PE6D	PE0D	C0PD	PF0D	PC7D	PD7D	PD6D
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Description
0	PD6D	1=Turn off PD6 digital input
1	PD7D	1=Turn off PD7 digital input
2	PC7D	1=Turn off PC7 digital input
3	PF0D	1=Turn off PF0 digital input
4	C0PD	1=Turn off AC0P digital input (LQFP48)
5	PE0D	1=Turn off PE0 digital input
6	PE6D	1=Turn off PE6 digital input
7	PE7D	1=Turn off PE7 digital input

ADCSRD – ADC control register D

ADCSRD – ADC control register D								
Address: 0xAD						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	BGEN	REFS2	IVSEL1	IVSEL0	-	VDS2	VDS1	VDS0
R/W	R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	Description
7	BGEN	Internal reference global enable control, 1=enable
6	REFS2	Combined with REFS of the ADMUX register to select the reference voltage for ADC conversion. Refer to the definition of REFS in the ADMUX register
5:4	IVSEL	When the reference voltage of the ADC is selected as VCC or AVREF, IVSEL is used to control the output voltage of the internal reference: 00 = 1.024V 01 = 2.048V 1x = 4.096V
3	-	Reserved.
2:0	VDS[2:0]	Voltage divider circuit input source selection: 000/111 = Turn off the voltage divider circuit module 001 = ADC0 010 = ADC1 011 = ADC4

		100 = ADC5 101 = External reference input (AVREF) 110 = System Power

DAPCR – Differential Op Amp Control Register

DAPCR – Differential Op Amp Control Register								
Address: 0xDC						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	DAPEN	GA1	GA0	DNS2	DNS1	DNS0	DPS1	DPS0
R/W	W/R	W/R	W/R	W/R	W/R	R/W	R/W	R/W

Bit	Name	Description
7	DAPEN	1 = Differential amplifier enabled; 0 = Differential amplifier disabled
6:5	GA[1:0]	Differential Amplifier Gain Control: 00 = x1 01 = x8 10 = x16 11 = x32
4:2	DNS[2:0]	Input source selection bit for the inverting input of the differential amplifier: 000 = ADC2/APN0 001 = ADC3/APN1 010 = ADC8/APN2 011 = ADC9/APN3 100 = PE0/APN4 101 = ADC multiplexing 110 = AGND 111 = Disable differential amplifier inverting input
1:0	DPS[1:0]	Input source selection bit for the positive input of the differential amplifier: 00 = ADC multiplexing 01 = ADC0/APP0 10 = ADC1/APP1 11 = AGND

OFR0 – Offset compensation register 0

OFR0 – Offset compensation register 0								
Address: 0xA3						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	OFR0[7:0]							
R/W	W/R							

Bit	Name	Description
7:0	OFR0	Offset compensation register 0; OFR0 is a signed number. stored in two's complement format

OFR1 – Offset compensation register 1

OFR1 – Offset compensation register 1								
Address: 0xA4						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	OFR1[7:0]							
R/W	W/R							

Bit	Name	description
7:0	OFR1	Offset compensation register 1; OFR1 is a signed number. Stored in twos complement format

ADMSC – ADC Channel Monitoring Status Control Register

ADMSC – ADC channel monitoring status control register								
Address: 0xAC						Default value: 0x01		
Bit	7	6	5	4	3	2	1	0
Name	AMOF	-	-	-	AMFC3	AMFC2	AMFC1	AMFC0
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	Description
7	AMOF	Automatic monitoring overflow event type flag; 1=overflow, 0=underflow
6:4	-	Unimplemented
3:0	AMFC	Automatic monitoring of digital filter control bits: 0000 = disable configuration 0001 = One conversion, no filtering 0010 = two consecutive matches 0011 = three consecutive matches …. 1110 = 14 consecutive matches 1111 = 15 consecutive matches

ADT0L – Low 8 bits of automatic monitoring underflow threshold

ADT0L – Low 8 bits of automatic monitoring underflow threshold								
Address: 0xA5						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ADT0L[7:0]							
R/W	W/R							

Bit	Name	Description
7:0	ADT0L	Low 8 bits of automatic monitoring underflow threshold

ADT0H – High 8 bits of automatic detection underflow threshold

ADT0H – High 8 bits of automatic detection underflow threshold								
Address: 0xA6						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ADT0H[7:0]							
R/W	W/R							
Bit	Name	Description						
7:0	ADT0H	High 8 bits of automatic detection underflow threshold						

ADT1L – Low 8 bits of automatic detection overflow threshold

ADT1L – Low 8 bits of automatic detection overflow threshold								
Address: 0xAA						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ADT1L[7:0]							
R/W	W/R							
Bit	Name	Description						
7:0	ADT1L	Low 8 bits of automatic detection overflow threshold						

ADT1H – High 8 bits of automatic detection overflow threshold

ADT1H – High 8 bits of automatic detection overflow threshold								
Address: 0xAB						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ADT1H[7:0]							
R/W	W/R							
Bit	Name	Description						
7:0	ADT1H	High 8 bits of automatic detection overflow threshold						

VCAL – Internal Reference Calibration Register

VCAL – Internal Reference Calibration Register								
Address: 0xC8						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL[7:0]							
R/W	W/R							
Bit	Name	Description						
7:0	VCAL	Internal reference calibration register. The calibration value of 1.024V is loaded by default after power-on. Writing calibration values for other voltage references to this register enables calibration of the associated reference. For example, after the reference configuration is 2.048V, write VCAL2 into the modified register to complete the calibration of the 2.048V internal reference.						

VCAL1 – 1.024V reference calibration register

VCAL1 – 1.024V internal reference calibration register								
Address: 0xCD						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL1[7:0]							
R/W	R/O							

Bit	Name	Description
7:0	VCAL1	1.024V internal reference calibration factor

VCAL2 – 2.048V reference calibration register

VCAL2 – 2.048V internal reference calibration register								
Address: 0xCE						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL2[7:0]							
R/W	R/O							

Bit	Name	Description
7:0	VCAL2	2.048V internal reference calibration coefficient

VCAL3 – 4.096V reference calibration register

VCAL1 – 4.096V internal reference calibration register								
Address: 0xCC						Default value: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL3[7:0]							
R/W	R/O							

Bit	Name	Description
7:0	VCAL3	4.096V internal reference calibration coefficient

Register cheat sheet

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Extended IO Register									
$F6	GUID3	GUID Byte 3							
$F5	GUID2	GUID Byte 2							
$F4	GUID1	GUID Byte 1							
$F3	GUID0	GUID Byte 0							
$F2	PMCR	PMCE	CLKFS	CLKSS	WCLKS	OSCKEN	OSCMEN	RCKEN	RCMEN
$F0	PMX2	WCE	STOSC1	STOSC0	-	-	XIEN	E6EN	C6EN
$EE	PMX0	PMXCE	C1BF4	C1AF5	C0BF3	C0AC0	SSB1	TXD6	RXD5
$ED	PMX1	-	-	-	-	-	C3AC	C2BF7	C2AF6
$EC	TCKSR	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
$E2	PSSR	PSS1	PSS3	-	-	-	-	PSR3	PSR1
$E1	OCPUE	PUE7	PUE6	PUE5	PUE4	PUE3	PUE2	PUE1	PUE0
$E0	HDR	-	-	HDR5	HDR4	HDR3	HDR2	HDR1	HDR0
$DE	DAPTE	DAPTE	-	-	-	-	-	-	-
$DD	DAPTR	DAPTP	DAP Trimming						
$DC	DAPCR	DAPEN	GA1	GA0	DNS2	DNS1	DNS0	DPS1	DPS0
$D8									
$D7									
$D6									
$D5									
$D4									
$D2									
$D1									
$D0									
$CF	LDOCR	WCE				PDEN	VSEL2	VSEL1	VSEL0
$CE	VCAL2	Calibration value for 2.048V internal reference							
$CD	VCAL1	Calibration value for 1.024V internal reference							
$CC	VCAL3	Calibration value for 4.096V internal reference							
$C8	VCAL	Internal Voltage Reference calibration register							
$C6	UDR	USART Data Register							
$C5	UBRRH	-	-	-	-	USART Baud Rate Register High			
$C4	UBRRL	USART Baud Rate Register Low							
$C2	UCSRC	UMSEL1	UMSEL0	UPM1	UPM0	USBS0	UCSZ01	UCSZ00	UCPOL0
$C1	UCSRB	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80
$C0	UCSRA	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0
$BD	TWAMR	TWI Address Mask							-
$BC	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
$BB	TWDR	TWI Data							
$BA	TWAR	TWI Address							TWGCE

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$B9	TWSR	TWI Status bits					-	TWPS	
$B8	TWBR	TWI Bit Rate register							
$B6	ASSR	INTCK	-	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
$B4	OCR2B	Timer 2 Output Compare Register B							
$B3	OCR2A	Timer 2 Output Compare Register A							
$B2	TCNT2	Timer 2 Counter Register							
$B1	TCCR2B	FOC2A	FOC2B	-	-	WGM22	CS2		
$B0	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20
$AF	DPS2R	-	-	-	-	DPS2E	LPRCE	TOS1	TOS0
$AE	IOCWK	IOCD7	IOCD6	IOCD5	IOCD4	IOCD3	IOCD2	IOCD1	IOCD0
$AD	ADCSRD	BGEN	REFS2	IVSEL1	IVSEL0	-	VDS2	VDS1	VDS0
$AC	ADMSC	AMOF	-	-	-	AMFC3	AMFC2	AMFC1	AMFC0
$AB	ADT1H	ADC Auto-monitor Overflow threshold high byte							
$AA	ADT1L	ADC Auto-monitor Overflow threshold low byte							
$A9	PORTE	Port Output E (for compatible with LGT8FX8D)							
$A8	DDRE	Data Direction E (for compatible with LGT8FX8D)							
$A7	PINE	Port Input E (for compatible with LGT8FX8D)							
$A6	ADT0H	ADC Auto-monitor Underflow threshold high byte							
$A5	ADT0L	ADC Auto-monitor Underflow threshold low byte							
$A4	OFR1	ADC positive offset trimming							
$A3	OFR0	ADC negative offset trimming							
$A1	DALR	DAC data register							
$A0	DACON	-	-	-	-	DACEN	DAOE	DAVS1	DAVS0
$9F	OCR3CH	Compare output register high byte of Timer3 C channel							
$9E	OCR3CL	Compare output register low byte of Timer3 C channel							
$9D	DTR3H	Dead-band register high byte of Timer3							
$9C	DTR3L	Dead-band register low byte of Timer3							
$9B	OCR3BH	Compare output register high byte of Timer3 B channel							
$9A	OCR3BL	Compare output register low byte of Timer3 B channel							
$99	OCR3AH	Compare output register high byte of Timer3 A channel							
$98	OCR3AL	Compare output register low byte of Timer3 A channel							
$97	ICR3H	Input capture register high byte of Timer3							
$96	ICR3L	Input capture register low byte of Timer3							
$95	TCNT3H	Counter register high byte of Timer3							
$94	TCNT3L	Counter register low byte of Timer3							
$93	TCCR3D	Control register D of Timer3							
$92	TCCR3C	Control register C of Timer3							
$91	TCCR3B	Control register B of Timer3							
$90	TCCR3A	Control register A of Timer3							
$8D	DTR1H	Dead-band register high byte of Timer1							
$8C	DTR1L	Dead-band register low byte of Timer1							
$8B	OCR1BH	Timer 1 Output Compare B High							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$8A	OCR1BL	Timer 1 Output Compare B Low							
$89	OCR1AH	Timer 1 Output Compare A High							
$88	OCR1AL	Timer 1 Output Compare A Low							
$87	ICR1H	Timer 1 Input Capture High							
$86	ICR1L	Timer 1 Input Capture Low							
$85	TCNT1H	Timer 1 Counter High							
$84	TCNT1L	Timer 1 Counter Low							
$83	TCCR1D	DSX17	DSX16	DSX15	DAX14	-	-	DSX11	DSX10
$82	TCCR1C	FOC1A	FOC1B	DOC1B	DOC1A	DTEN1	-	-	-
$81	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS1		
$80	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10
$7F	DIDR1	PE7D	PE6D	PE0D	C0PD	PF0D	PC7D	PD7D	PD6D
$7E	DIDR0	PE3D	PE1D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
$7D	ADCSRC	OFEN	-	SPN	AMEN	-	SPD	DIFS	ADTM
$7C	ADMUX	REFS1	REFS0	ADLAR	CHMUX				
$7B	ADCSRB	CME01	CME00	CME11	CME10	-	ADTS		
$7A	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS		
$79	ADCH	ADC Data High							
$78	ADCL	ADC Data Low							
$76	DIDR2	-	PB5D	-	-	-	-	-	-
$75	IVBASE	Interrupt Vector Base Address							
$74	PCMSK4								
$73	PCMSK3	PCINT[39:32]							
$71	TIMSK3			ICIE3	-	OCIE3C	OCIE3B	OCIE3A	TOIE3
$70	TIMSK2	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2
$6F	TIMSK1	-	-	ICIE1	-	-	OCIE1B	OCIE1A	TOIE1
$6E	TIMSK0	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0
$6D	PCMSK2	PCINT[23:16]							
$6C	PCMSK1	PCINT[15:8]							
$6B	PCMSK0	PCINT[7:0]							
$69	EICRA	-	-	-	-	ISC11	ISC10	ISC01	ISC00
$68	PCICR	-	-	-	PCIE4	PCIE3	PCIE2	PCIE1	PCIE0
$67	RCKCAL	RC32K Calibration							
$66	RCMCAL	RC32M Calibration							
$65	PRR1	-	-	PRWDT	-	PRTIM3	PREFL	PRPCI	-
$64	PRR/0	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUART0	PRADC
$62	VDTCR	WCE	SWR	-	VDTS			VDREN	VDTEN
$61	CLKPR	WCE	CKOE1	CKOE0	-	CLKPS			
$60	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0
DirectIO Register									
$5F	SREG	I	T	H	S	V	N	Z	C
$5E	SPH	Stack Point High							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$5D	SPL	Stack Point Low							
$5C	E2PD3	E2PCTL Data register byte 3							
$5B	C1TR	AC1 trimming data							
$5A	E2PD1	E2PCTL Data register byte1							
$59	DSAH	DSA[31:16] access port of uDSC							
$58	DSAL	DSA[15:0] access port of uDSC							
$57	E2PD2	E2PCTL Data register byte 2							
$56	ECCR	WEN	EEN	ERN	SWM	CP1	CP0	ECS1	ECS0
$55	MCUCR	FWKEN	FPDEN	SWR	PUD	IRLD	IFAIL	IVSEL	WCE
$54	MCUSR	SWDD	-	-	OCDRF	WDRF	BORF	EXTRF	PORF
$53	SMCR	-	-	-	-	SM			SE
$52	C0TR	AC0 Trimming register							
$51	C0XR	-	C0OE	C0HYSE	C0PS0	C0WKE	C0FEN	C0FS1	C0FS0
$50	C0SR	C0D	C0BG	C0O	C0I	C0IE	C0IC	C0IS	
$4F	DTR0	TC0 Dead-band timing control register							
$4E	SPDR	SPI Data register							
$4D	SPSR	SPIF	WCOL	-	-	-	DUAL	-	SPI2X
$4C	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR	
$4B	GPIOR2	General Purpose Register 2							
$4A	GPIOR1	General Purpose Register 1							
$49	TCCR0C	DSX07	DSX06	DSX05	DSX04	-	-	DSX01	DSX00
$48	OCR0B	Timer 0 Output Compare Register B							
$47	OCR0A	Timer 0 Output Compare Register A							
$46	TCNT0	Timer 0 Counter							
$45	TCCR0B	FOC0A	FOC0B	OC0AS	DTEN0	WGM02	CS02	CS01	CS00
$44	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	DOC0B	DOC0A	WGM01	WGM00
$43	GTCCR	TSM	-	-	-	-	-	PSRASY	PSRSYNC
$42	EEARH	E2PCTL Address High							
$41	EEARL	E2PCTL Address Low							
$40	E2PD0	E2PCTL Data byte 0							
$3F	EECR	EEPM2	EEPM2	EEPM1	EEPM0	EERIE	EEMWE	EEWE	EERE
$3E	GPIOR0	General Purpose Register 0							
$3D	EIMSK	-	-	-	-	-	-	INT1	INT0
$3C	EIFR	-	-	-	-	-	-	INTF1	INTF0
$3B	PCIFR	-	-	-	-	PCIF3	PCIF2	PCIF1	PCIF0
$3A	C1XR	-	C1OE	C1HYSE	C1PS0	C1WKE	C1FEN	C1FS1	C1FS0
$39	SPFR	RDFULL	RDEMPT	RDPTR1	RDPTR0	WRFULL	WREMPT	WRPTR1	WRPTR0
$38	TIFR3	-	-	ICF3	-	-	OCF3B	OCF3A	TOV3
$37	TIFR2	-	-	-	-	-	OCF2B	OCF2A	TOV2
$36	TIFR1	-	-	ICF1	-	-	OCF1B	OCF1A	TOV1
$35	TIFR0	-	-	-	-	-	OCF0B	OCF0A	TOV0
$34	PORTF	Port Output of Group F							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$33	DDRF	Data Direction of Group F							
$32	PINF	Port Input of Group F							
$31	DSDY	DSDY access port of uDSC							
$30	DSDX	DSDX access port of uDSC							
$2F	C1SR	C1D	C1BG	C1O	C1I	C1IE	C1IC	C1IS	
$2E	PORTE	Port Output of Group E							
$2D	DDRE	Data Direction of Group E							
$2C	PINE	Port Input of Group E							
$2B	PORTD	Port Output of Group D							
$2A	DDRD	Data Direction of Group D							
$29	PIND	Port Input of Group D							
$28	PORTC	Port Output of Group C							
$27	DDRC	Data Direction of Group C							
$26	PINC	Port Input of Group C							
$25	PORTB	Port Output of Group B							
$24	DDRB	Data Direction of Group B							
$23	PINB	Port Input of Group B							
$22	DSSD	DSSD access port of uDSC							
$21	DSIR	Instruction regiter of uDSC							
$20	DSCR	DSUEN	MM	D1	D0	-	DSN	DSZ	DSC

Instruction set cheat sheet

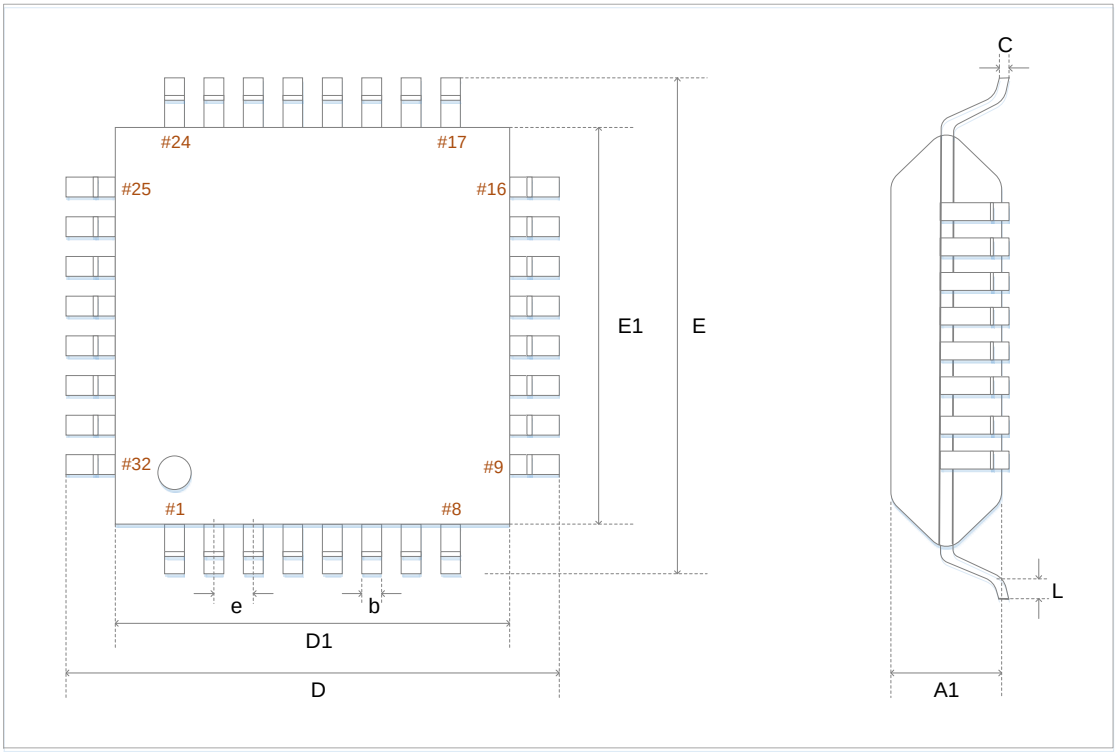
Instruction	operands	description	operation	flag bits	cycles
Arithmetic logic operation instructions					
ADD	R_d, R_r	Addition	$R_d \leftarrow R_d + R_r$	Z,C,N,V,H	1
ADC	R_d, R_r	Addition with carry	$R_d \leftarrow R_d + R_r + C$	Z,C,N,V,H	1
ADIW	R_{dl}, K	add immediate value	$R_{dh}:R_{dl} \leftarrow R_{dh}:R_{dl} + K$	Z,C,N,V,S	1
SUB	R_d, R_r	Subtraction	$R_d \leftarrow R_d - R_r$	Z,C,N,V,H	1
SUBI	R_d, K	subtract immediate value	$R_d \leftarrow R_d - K$	Z,C,N,V,H	1
SBC	R_d, R_r	Subtraction with borrow	$R_d \leftarrow R_d - R_r - C$	Z,C,N,V,H	1
SBCI	R_d, K	Subtract immediate with borrow	$R_d \leftarrow R_d – K - C$	Z,C,N,V,H	1
SBIW	R_{dl}, K	16 bits subtract	$R_{dh}:R_{dl} \leftarrow R_{dh}:R_{dl} - K$	Z,C,N,V,S	1
AND	R_d, R_r	Logic AND	$R_d \leftarrow R_d \& R_r$	Z,N,V	1
ANDI	R_d, K	Logic AND immediate	$R_d \leftarrow R_d \& K$	Z,N,V	1
OR	R_d, R_r	Logic OR	$R_d \leftarrow R_d \mid R_r$	Z,N,V	1
ORI	R_d, K	Logic OR immediate	$R_d \leftarrow R_d \mid K$	Z,N,V	1
EOR	R_d, R_r	Logic EXOR	$R_d \leftarrow R_d \oplus R_r$	Z,N,V	1
COM	R_d	Negate	$R_d \leftarrow \$FF - R_d$	Z,C,N,V	1
NEG	R_d	2’s complement	$R_d \leftarrow \$00 - R_d$	Z,C,N,V,H	1
SBR	R_d, K	Set bits	$R_d \leftarrow R_d$ v K	Z,N,V	1
CBR	R_d, K	Clear bits	$R_d \leftarrow R_d$ v ($FF – K)	Z,N,V	1
INC	R_d	Increment	$R_d \leftarrow R_d + 1$	Z,N,V	1
DEC	R_d	Decrement	$R_d \leftarrow R_d - 1$	Z,N,V	1
TST	R_d	Test if 0	$R_d \leftarrow R_d \& R_d$	Z,N,V	1
CLR	R_d	Clear register	$R_d \leftarrow R_d \oplus R_d$	Z,N,V	1
SER	R_d	set register all 1	$R_d \leftarrow \$FF$	None	1
MUL	R_d, R_r	Unsigned multiplication	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
MULS	R_d, R_r	Signed multiplication	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
MULSU	R_d, R_r	signed number times unsigned number	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
FMUL	R_d, R_r	Unsigned multiplication, shifted	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
FMULS	R_d, R_r	Signed multiplication, shifted	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
FMULSU	R_d, R_r	signed number times unsigned number, shifted	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
Jump instructions					
RJMP	K	Relative jump	PC ← PC + K + 1	None	1
IJMP		Indirect jump (to Z-pointing address)	PC ← Z	None	2
JMP	K	Jump directly	PC ← K	None	2
RCALL	K	Relative address subroutine call	PC ← PC + K + 1	None	1
ICALL		Indirect subroutine calls (Z-pointing addresses)	PC ← Z	None	2
CALL	K	Direct subroutine call	PC ← K	None	2
RET		Subroutine return	PC ← Stack	None	2
RETI		Interrupt returns	PC ← Stack	I	2

Instruction	operands	description	operation	flag bits	cycles
Jump Instructions (continued)					
CPSE	R_d, R_r	Jump if equal	If(R_d=R_r) PC ← PC + 2 or 3	None	1/2
CP	R_d, R_r	Compare	R_d - R_r	Z,N,V,C,H	1
CPC	R_d, R_r	Compare with carry	R_d - R_r – C	Z,N,V,C,H	1
CPI	R_d, K	Compare immediate	R_d - K	Z,N,V,C,H	1
SBRC	R_r, b	Skip next if bit clear	If(R_r(b)=0) PC ← PC + 2 or 3	None	1/2
SBRS	R_r, b	Skip next if bit set	If(R_r(b)=1) PC ← PC + 2 or 3	None	1/2
SBIC	P, b	Skip next if I/O bit clear	If(P(b)=0) PC ← PC + 2 or 3	None	1/2
SBIS	P, b	Skip next if I/O bit set	If(P(b)=1) PC ← PC + 2 or 3	None	1/2
BRBS	s, k	Branch if status bit set	If(SREG(S)=1) PC ← PC + K + 1	None	1/2
BRBC	s, k	Branch if status bit clear	If(SREG(S)=0) PC ← PC + K + 1	None	1/2
BREQ	k	Branch if EQ	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if NEQ	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if carry set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if carry clear	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if NLT	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if LT	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if negative	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if positive	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if signed NLT	if (N⊕V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if signed < 0	if (N⊕V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if half carry set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if half carry clear	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T clear	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if overflow set	f (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if overflow clear	f (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if global interrupt set	f (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if global interrupt clear	f (I = 0) then PC ← PC + k + 1	None	1/2
Data transfer instructions					
MOV	Rd, Rr	Move data between registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Move one word of data	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load the number immediately	Rd ← K	None	1
LD	Rd, X	Indirect loading	Rd ← (X)	None	1/2
LD	Rd, X+	Indirect loading, address increment	Rd ← (X), X ← X + 1	None	1/2
LD	Rd, -X	Address decrement, indirect loading	X ← X - 1, Rd ← (X)	None	1/2
LD	Rd, Y	Indirect loading	Rd ← (Y)	None	1/2
LD	Rd, Y+	Indirect loading, address increment	Rd ← (Y), Y ← Y + 1	None	1/2
LD	Rd, -Y	Address decrement, indirect loading	Y ← Y - 1, Rd ← (Y)	None	1/2
LDD	Rd, Y+q	Indirect loading with offset	Rd ← (Y + q)	None	1/2
LD	Rd, Z	Indirect loading	Rd ← (Z)	None	1/2

LD	Rd, Z+	Indirect loading, address increment	Rd ← (Z), Z ← Z+1	None	1/2
LD	Rd, -Z	Address decrement, indirect loading	Z ← Z - 1, Rd ← (Z)	None	1/2
LDD	Rd, Z+q	Indirect loading with offset	Rd ← (Z + q)	None	1/2
LDS	Rd, k	Load directly from SRAM	Rd ← (k)	None	2
ST	X, Rr	Indirect storage	(X) ← Rr	None	1
ST	X+, Rr	Indirect store, address increment	(X) ← Rr, X ← X + 1	None	1
ST	-X, Rr	Address decrement, indirect store	X ← X - 1, (X) ← Rr	None	1
ST	Y, Rr	Indirect store	(Y) ← Rr	None	1
ST	Y+, Rr	Indirect store, address increment	(Y) ← Rr, Y ← Y + 1	None	1
ST	-Y, Rr	Address decrement, indirect store	Y ← Y - 1, (Y) ← Rr	None	1
STD	Y+q, Rr	Indirect store with offset	(Y + q) ← Rr	None	1
ST	Z, Rr	Indirect store	(Z) ← Rr	None	1
ST	Z+, Rr	Indirect store, address increment	(Z) ← Rr, Z ← Z + 1	None	1
ST	-Z, Rr	Address decrement, indirect store	Z ← Z - 1, (Z) ← Rr	None	1
STD	Z+q, Rr	Indirect store with offset	(Z + q) ← Rr	None	1
STS	k, Rr	Store directly into SRAM	(k) ← Rr	None	2
LPM		Load program space data	R0 ← (Z)	None	2
LPM	Rd, Z	Load program space data	Rd ← (Z)	None	2
LPM	Rd, Z+	Load program data, address increment	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, Z+	Indirect loading, address increment	Rd ← (Z), Z ← Z+1	None	1
LD	Rd, -Z	Address decrement, indirect loading	Z ← Z - 1, Rd ← (Z)	None	1
LDD	Rd, Z+q	Indirect loading with offset	Rd ← (Z + q)	None	1
LDS	Rd, k	Load directly from SRAM	Rd ← (k)	None	2
IN	Rd, P	Read port	Rd ← P	None	1
OUT	P, Rr	Write port	P ← Rr	None	1
PUSH	Rr	Push stack	STACK ← Rr	None	1
POP	Rd	Pop stack	Rd ← STACK	None	1/2
SBI	P, b	Set IO register	I/O(P, b) ← 1	None	1
CBI	P, b	Clear IO register	I/O(P, b) ← 0	None	1
LSL	Rd	Logic shifts left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical shift right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z	1
ROL	Rd	Rotate left with carry	Rd(0)←C, Rd(n+1) ← Rd(n), C←Rd(7)	Z	1
ROR	Rd	Rotate right with carry	Rd(7)←C, Rd(n) ← Rd(n+1), C←Rd(0)	Z	1
ASR	Rd	Arithmetic shifted right	Rd(n) ← Rd(n+1), n=0:6	Z	1
SWAP	Rd	Bit swapping	Rd(3:0) ← Rd(7:4), Rd(7:4) ← Rd(3:0)	None	1
BSET	s	Set status bit	SREG(s) ← 1	SREG(s)	1
BCLR	s	Clear status bit	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Store to the T bit	T ← Rr(b)	T	1
BLD	Rd, b	Read the T bit into register	Rd(b) ← T	None	1
SEC		Set carry flag	C ← 1	C	1
CLC		Clear carry flags	C ← 0	C	1

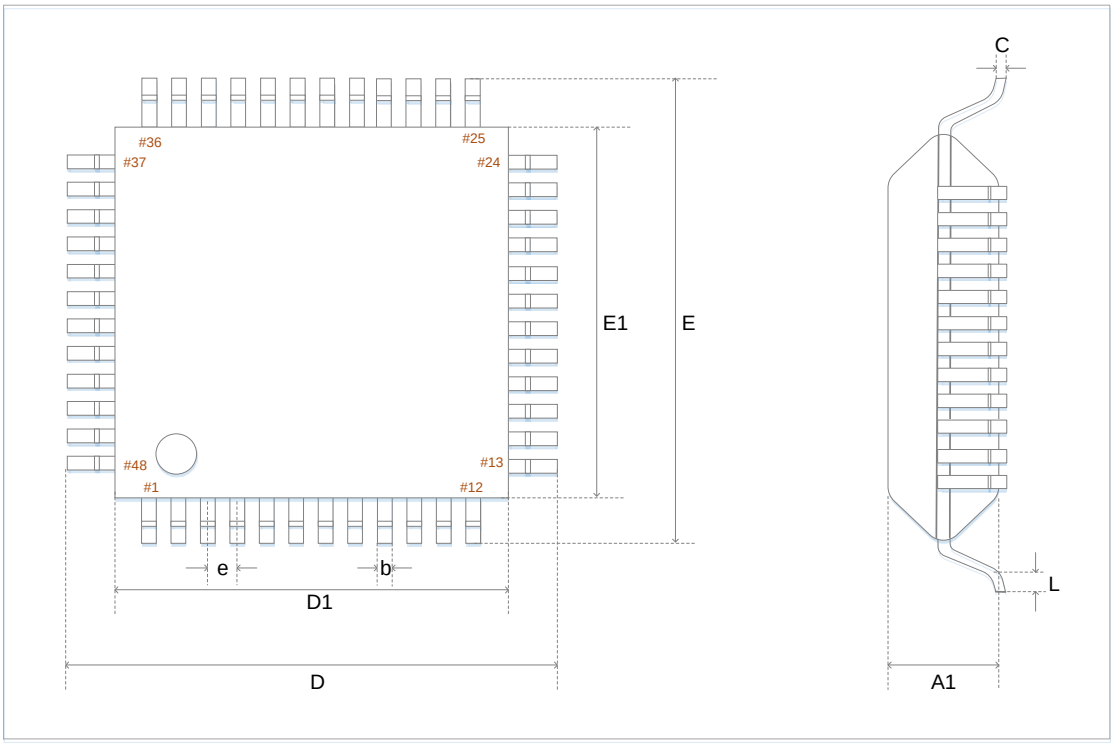
SEN		Set negative flag	N ← 1	N	1
CLN		Clear negative flag	N ← 0	N	1
SEZ		Set zero flag	Z ← 1	Z	1
CLZ		Clear zero flag	Z ← 0	Z	1
SEI		Enable global interrupts	I ← 1	I	1
CLI		Clear global interrupts	I ← 0	I	1
SES		Set the symbolic test flag	S ← 1	S	1
CLS		Clear the symbol test flag	S ← 0	S	1
SEV		Set 2's complement overflow flag	V ← 1	V	1
CLV		Clear 2's complement overflow flag	V ← 0	V	1
SET		Set the T bit (SREG)	T ← 1	T	1
CLT		Clear T bit (SREG)	T ← 0	T	1
MCU control instructions					
NOP		Null instruction		None	1
SLEEP		Enter sleep mode		None	1
WDR		Watchdog reset		None	1
BREAK		Soft breakpoints	For debugging purposes only	None	N/A
NOP		Null instruction		None	1
SLEEP		Enter sleep mode		None	1

Encapsulation parameters



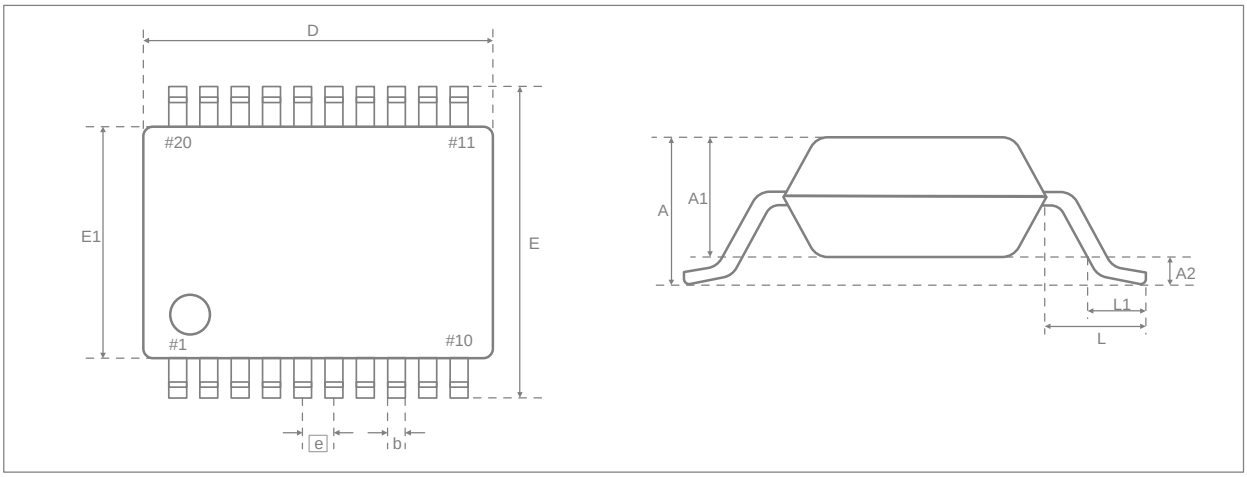
LQFP32 general size definition

Char code	minimum	Typical value	maximum	unit
D	8.90	9.00	9.10	mm
D1	6.90	7.00	7.10	mm
b	0.2	0.30	0.4	mm
e	0.75	0.80	0.85	mm
E	8.90	9.00	9.10	mm
E1	6.90	7.00	7.10	mm
C	-	0.10	-	mm
L	0.55	0.60	0.65	mm
A1	-	1.40	-	mm



LQFP48 Universal Size Definition

Char Code	minimum	Typical value	maximum	unit
D	8.80	9.00	9.20	mm
D1	6.80	7.00	7.20	mm
b	0.17	0.22	0.27	mm
e	-	0.50BSC	-	mm
E	8.80	9.00	9.20	mm
E1	6.80	7.00	7.20	mm
C	0.09	-	0.2	mm
L	0.45	0.60	0.75	mm
A1	1.35	1.40	1.45	mm



SSOP20L universal size definition

Char Code	minimum	Typical value	maximum	unit
D	6.90	7.20	7.50	mm
A2	0.03	0.05	0.07	mm
b	0.22	0.30	0.38	mm
e	-	0.65	-	mm
E	7.40	7.80	8.20	mm
E1	5.00	5.30	5.60	mm
L1	0.55	-	0.95	mm
L	-	-	-	mm
A1	-	2.0	-	mm

Version history

R.01	English translation by Frankie. Tools used: Google/Bing
V1.0.4 2017/11/15	Correct the definition of SSOP20 PIN8/11
V1.0.3 2017/6/23	Add SSOP20 Package Definition Updated the instructions for the TMR3 interrupt flag bit
V1.0.2 2017/5/15	Update the instructions for automatic PWM shutdown and restart in TMR0/TRM1/TMR3 Updated the instructions for SPI interrupt handling in the SPI section and updated the SPFR registers
V1.0.1 2017/2/13	Removed the I2C1 section, this function is not available to improve the definition of some registers
V1.0.0 2016/12/29	Initial version